Федеральное государственное бюджетное образовательное учреждение
высшего профессионального образования
«Московский государственный университет дизайна и технологии»

На правах рукописи

Ланшаков Денис Евгеньевич

Разработка технологий и конструкций сложных цельновязаных изделий на базе комплексной автоматизированной системы

Специальность 05.19.02 – Технология и первичная обработка текстильных материалов и сырья

Диссертация на соискание ученой степени
кандидата технических наук

Научный руководитель –
доктор технических наук
профессор Колесникова Елена Николаевна

Москва – 2014 г.

ОГЛАВЛЕНИЕ

ВВЕДЕНИЕ

В настоящее время производство цельновязаных изделий, вырабатываемых по сложным ресурсосберегающим технологиям, зависит от используемых технологий и методов их проектирования.

Внедрение в производство таких технологий связано с необходимостью увеличения прибыли за счет:

- сокращения затрат сырьевых ресурсов;
- сокращения арендуемых производственных площадей;
- полного или частичного исключения из цепочки технологической обработки изделия, некоторых этапов обработки, например, при производстве цельновязаных изделий частично или полностью исключается швейная обработка;
- повышения качества, выпускаемой продукции;
- расширения ассортимента.

Однако эффективность работы инженеров при проектировании технологически сложных изделий снижается, что связано с необходимостью обработки больших объемов данных проектирования, при этом в некоторых случаях инструменты, позволяющие автоматизировать рутинные ручные операции, отсутствуют.

Кроме того может отсутствовать связь между ключевыми специалистами, позволяющая выполнять динамическое отслеживание изменений результатов их работы. Например, при проектировании сложного цельновязаного изделия геометрические изменения кромок соединения, получаемые дессинатором в момент проектирования сбавок (прибавок), разработки технологии узла соединения и т.д., не учитываются конструктором при выполнения градации пакета лекал по необходимым размеро-ростам. Отсутствие такой связи между ключевыми специалистами может привести к увеличению объема времени,

требуемого для проектирования изделия и к увеличению расхода сырья, необходимого на отработку образцов.

С целью повышения качества цельновязаных изделий, расширения ассортиментного ряда, выпускаемых изделий, сокращения времени и сырья, затрачиваемого на проектирование трикотажных изделий, а также повышении эффективности работы конструктора и дессинатора в данной диссертационной работе предполагается провести исследование этапа проектирования трикотажных изделий и усовершенствовать технологии вязания узлов соединения.

Поэтому **цель работы** заключается в разработке комплексной системы проектирования, обеспечивающей увеличение эффективности работы специалистов отдела проектирования, повышения качества, проектируемых трикотажных изделий и расширении ассортимента цельновязаных изделий.

Для реализации поставленной цели в процессе диссертационных исследований необходимо было выполнить следующее:

- выполнить анализ современных методик конструирования и систем автоматизированного проектирования, используемые на швейном и трикотажном производствах;
- разработать технологический модуль перехода от конструирования к разработке технологии вязания, с учетом возможности использования при конструировании любых методик конструирования;
- разработать технологию получения сложных узлов соединения в цельновязаных изделиях с учетом технологических возможностей вязальной машины и параметров полотна;
- разработать технологию получения качественной посадки цельновязаного изделия на теле человека за счет формирования необходимого рельефа;
- разработать программное обеспечение, позволяющее выполнять конструкторскую и технологическую подготовку производства

трикотажных изделий, в том числе сложных цельновязаных изделий, а также позволяющее передать результаты проектирования в САПР, используемую для разработки программы вязания.

В процессе выполнения данной диссертационной работы использовались теоретические и экспериментальные методы исследования. К ним можно отнести:

- методы системного анализа и синтеза;
- теорию вязания и строения трикотажа;
- методы конструкторской подготовки трикотажных и швейных изделий;
- математическое моделирование;
- объектно-ориентированное программирование;
- аппроксимацию и интерполирование объектов.

Обработка данных, получаемых в результате экспериментальных исследований, а также математическое моделирование выполнялось при помощи языка программирования C++ в среде разработки MS Visual Studio 2010. Разработка программ вязания выполнялось в САПР M1 3.15 фирмы STOLL.

Экспериментальные исследования осуществлялись на производственной базе ООО «Пафос» в городе Москве на плосковязальных машинах CMS 340 TC-L класса 6.2, а также CMS 340 TC-KW класса 7.2 фирмы STOLL.

Научная новизна диссертационных исследований заключается в том, что автором впервые получены следующие результаты:

- разработан алгоритм расчета сбавок (прибавок) по контуру развертки детали с учетом задаваемых технологических ограничений;
- разработан метод проектирования узла соединения цельновязаного изделия за счет балансирования-выравнивания элементов соединения на соединяемых кромках;

- разработан алгоритм определения положения линии в узле соединения цельновязаного изделия, после которой соединяемые кромки будут иметь разное количество петельных рядов;
- разработан метод моделирования выпуклого участка трикотажа, получаемого за счет провязывания дополнительных петельных рядов;
- разработан алгоритм расчета выпуклого участка трикотажа с учетом провязывания дополнительных петельных рядов переменной ширины;
- разработан способ формирования рельефа за счет замены выпуклости, формируемой при закрытии швейной вытачки, на геометрически аналогичную выпуклость, формируемую за счет провязывания дополнительных петельных рядов;
- предложена технология вязания выпуклого участка трикотажа, использующая в вязании один или два нитеводителя;
- разработано специальное программное обеспечение “DESIGNER K-WEAR”, а также получено свидетельство о государственной регистрации программ для ЭВМ № 2013618038.

Практическая значимость, полученных результатов диссертационного исследования, заключается в сокращении времени (в среднем на 48,8%), затрачиваемого на проектирование цельновязаных изделий, повышении качества цельновязаных изделий, за счет улучшения качества исполнения узлов соединения деталей и улучшении посадки готового изделия на теле человека, расширении ассортиментного ряда, а также увеличении эффективности работы (в среднем на 36,8%) специалистов отдела проектирования.

По материалам диссертационной работы опубликовано две статьи, тезисы двух докладов, а также получено свидетельство о государственной регистрации программы для ЭВМ. Кроме того на производственной базе ООО «Пафос» выполнено внедрение разработанного специального программного обеспечения “DESIGNER K-WEAR”, что подтверждено актом о внедрении.

Диссертационная работа состоит из введения, четырех глав, общих выводов, списка литературы, состоящего из 34 источников литературы, и двух приложений. Диссертационная работа содержит 187 страниц машинного текста, 76 рисунков, 13 таблиц и 91 формулу.

Приложение 1 содержит 416 страниц машинного текста. Приложение 1 включает в себя исходные тексты, выполненные на языке программирования C++, заголовочных файлов и файлов реализации, разработанного специального программного обеспечения “DESIGNER K-WEAR”.

Приложение 2 содержит 69 страниц машинного текста, 36 рисунков и 14 таблиц. Приложение 2 является учебным пособием по работе в специальном программном обеспечении “DESIGNER K-WEAR”.

Положения, выносимые на защиту:

- технологию получения качественного узла соединения в сложном цельновязаном изделии с учетом использования кривых Безье на этапе конструкторской подготовки;
- технологию и метод моделирования выпуклого участка трикотажа, формируемого за счет провязывания дополнительных петельных рядов;
- новый способ получения выпуклого участка трикотажа, формируемого за счет замены выпуклости, получаемой за счет закрытия швейной вытачки, на геометрически аналогичную выпуклость, получаемую за счет провязывания дополнительных петельных рядов;
- метод конструкторской подготовки сложного цельновязаного изделия, учитывающий использование любой методики конструирования.

1 АНАЛИЗ АСПЕКТОВ ЭТАПА ПРОЕКТИРОВАНИЯ СЛОЖНЫХ ЦЕЛЬНОВЯЗАНЫХ ИЗДЕЛИЙ В СОВРЕМЕННЫХ ПРОМЫШЛЕННЫХ УСЛОВИЯХ

Цельновязаными называются трикотажные изделия, форма которых достигается при вязании в автоматическом режиме [1.2]. Например, при вязании джемпера на двухфонтурной плосковязальной машине происходит одновременное вязание трех трубок, две из которых являются рукавами, а одна станом, и на уровне проймы выполняется их соединение.

К ассортименту цельновязаных изделий можно отнести трикотажные изделия чулочно-носочной группы, бельевой группы, группы верхнетрикотажных изделий (плечевые, поясные, головные уборы), а также трикотажные изделия спортивного ассортимента (например, женский купальный костюм). К цельновязаным изделиям, получаемым на плосковязальных машинах можно отнести: шарфы, шапки, джемпера, свитера, рейтузы и т.д.

В настоящее время ассортимент цельновязаных изделий постоянно расширяется за счет ввода в производство новых технологий вязания, повышения уровня технологического оборудования и т.д. Однако предлагаемые производителями вязального оборудования программные решения, необходимые для проектирования цельновязаных изделий имеют ограниченный набор готовых шаблонов (модулей), например, предлагается вязание джемперов только с определенными видами рукавов и использованием определенных технологий вязания узлов соединений. Причем конструктивное моделирование, например, перевод вытачки, изменение формы горловины и т.д., или ввод в производство новой ассортиментной группы, например, сувенирной продукции, доступно дессинатору только на базе имеющихся в системе проектирования шаблонов. Изменение конструкции изделия, а также проектирование технологии вязания, в том числе на участке соединения деталей, дессинатор вынужден выполнять в ручном режиме.

Цельновязаные изделия конструктивно и технологически могут быть сложными, состоять из двух и более одновременно вырабатываемых деталей, которые на определенном участке должны автоматически соединиться. Линии, по которым выполняется соединение деталей, на этапе аппроксимации могут поменять свою форму [1.2].

С целью усовершенствования конструкторской подготовки сложных цельновязаных изделий проведем анализ этапа конструкторской подготовки сложных цельновязаных изделий и рассмотрим известные методы конструирования и системы автоматизированного проектирования, применяемые в швейной и трикотажной промышленности.

1.1 Анализ этапа конструкторской подготовки сложных цельновязаных изделий

Конструкторская подготовка проектируемого изделия в трикотажном производстве на большинстве предприятий носит более консервативный характер в отличие от швейного производства. Это сказывается на значительных различиях между используемыми системами проектирования определенными выше производствами в современных промышленных условиях, это в свою очередь, оказывает влияние на время необходимое для формирования пакета лекал проектируемого изделия.

Зачастую в качестве формирующих факторов в конструировании на трикотажном производстве используются ручные измерительно-начертательные инструменты, а также аналогичные средства нанесения линий, при этом чертеж выполняется на масштабируемой бумаге.

Наиболее распространенными методами построения лекал для трикотажных изделий является формирование двумерных разверток деталей одежды с использованием значений типовых размерных признаков человека.

Практикуемые в производстве методики построения разверток для трикотажных изделий незначительно отличаются от методик, использующихся в конструировании изделий на швейном производстве [1.5].

Различия между методиками состоят в расхождении значений применяемых прибавок, припусков и т.п., что обусловлено особенностями производства полотен, используемых для дальнейшей выработки изделий в потоке, и физико-механическими характеристиками этих полотен. Кроме того при конструировании трикотажного изделия, например, вырабатываемого на плосковязальной машине, зачастую необходимо учитывать сбавки (прибавки) петельных столбиков, что в свою очередь сказывается на форме некоторых графических объектов, используемых для конструирования. Например, при конструировании проймы в кроеном изделии её низ оформляют в виде дуги, что придает изделию красивый внешний вид и обеспечивает человеку необходимую степень свободы движения при эксплуатации этого изделия. При конструировании проймы в регулярном трикотажном изделии часто её низ оформляют в виде отрезка прямой, который необходим для расчета и равномерного распределения сбавок, что, в дальнейшем, придаст изделию аккуратный и эстетичный вид. Формирование необходимой степени свободы движения у трикотажного изделия, произведенного регулярным способом, будет осуществляться на этапе влажно-тепловой обработки.

Любой метод конструирования содержит в себе ряд факторов, определяющих форму готового изделия. При этом основную долю этих факторов составляют особенности телосложения человека, покрой, а также способы технологической обработки в процессе формирования изделия.

Рассмотрим известные методы разработки разверток деталей проектируемого изделия на трикотажном производстве [1.4].

1.1.1 Известные методы конструирования

Существующие методы конструирования, в зависимости от результирующей точности, можно разделить на две категории (рисунок 1.1).

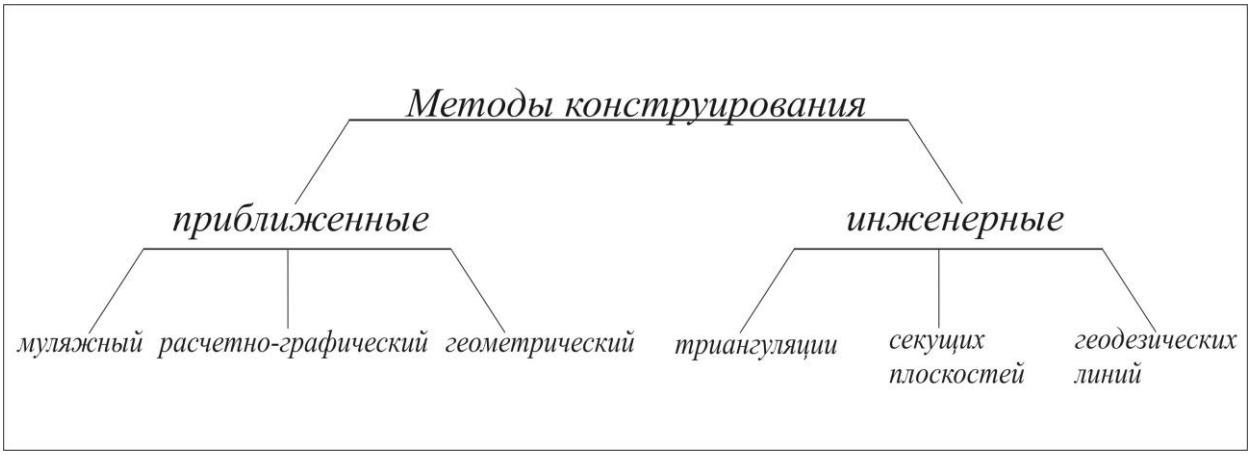


Рисунок 1.1 – Известные методы конструирования

Рассмотрим известные методы конструирования в соответствии с рисунком 1.1.

Муляжный метод.

Создание образца и дальнейшее формирование разверток деталей, проектируемого изделия, выполняется путем макетирования изделия на формирующем объемном теле. В этом случае использование эмпирического способа проектирования изделия позволит в полной мере учесть антропоморфные характеристики тела человека и физико-механические свойства, используемого полотна, в процессе формообразования [1.3].

Однако данный метод формирования лекал экономически неэффективен, так как допускает большой расход перерабатываемого сырья из-за отсутствия аналитической составляющей, что в дальнейшем часто потребует изменения лекал изделия при смене типа сырья или переплетения. Этот метод также требует большого количества времени на проектирование лекал, что следует из-за большого количества примерок, необходимых для формирования качественной посадки.

Расчетно-графический метод.

В настоящее время существует огромное разнообразие способов проектирования лекал, относящихся к расчетно-графическим методам (РГМ). Однако, последние варианты РГМ, например, ЕМКО СЭВ, единая методика конструирования мужской, женской и детской одежды ЦНИИШП, имеют особую популярность в производстве за счет универсальности использования, научной обоснованности, элементарности эмпирических расчетов и простоты графических построений [1.3].

Универсальность РГМ определена возможностью проектирования лекал изделия не только с учетом физико-механических свойств полотна, но и различных вариантов покроев, силуэтных форм и т.п., формируемых за счет модных тенденций в одежде.

Научная обоснованность РГМ определяется использованием антропометрических характеристик населения, а также применением обоснованных технологических припусков и прибавок.

Использование РГМ совместно с ЭВМ позволяет проектировать лекала с учетом прогнозирования изменений, получаемых в процессе производства и эксплуатации готовых изделий. Это возможно при использовании РГМ в параметрической форме.

Данная форма подразумевает создание алгоритма с поддержкой математического аппарата при помощи заранее известных текстовых команд, вводимых с клавиатуры. Согласно этому алгоритму будет выполняться построение и последующее преобразование графических объектов с целью создания конструкции проектируемого изделия. Затем на основе конструкции будет формироваться пакет готовых лекал.

Такой подход к формированию разверток деталей проектируемого изделия позволит, пользуясь принципами математического программирования, разрабатывать алгоритм построения с использованием различных расчетных формул, характеризующих свойства используемых графических объектов. При

этом расчетные формулы могут быть получены в результате теоретических или эмпирических исследований.

Таким образом, получение разверток деталей проектируемого изделия с использованием РГМ в параметрической форме, где алгоритм построения, дополнен расчетными формулами, полученными в результате теоретических и эмпирических исследований факторов, формирующих изделие, позволит в результате получить готовое изделие, имеющее качественную посадку и отвечающее последним тенденциям моды.

Геометрический метод.

Данный метод основан на конструктивном построении разверток основных деталей, проектируемого изделия, с использованием шаблонов приближенных разверток, заданных поверхностью фигуры человека. Такие шаблоны позволяют выполнять построение основных деталей, проектируемого изделия и формировать базу для типового проектирования одежды [1.3].

Геометрический метод в сравнении с расчетно-графическими методами менее трудоемок, однако, данный метод не позволяет максимально учитывать физико-механические свойства используемого полотна и изменение геометрических характеристик деталей (цельновязаного изделия), получаемых в процессе технологической обработки, так как не имеет аналитической основы.

Метод триангуляции.

Данный метод основан на построении приближенной технической поверхности за счет аппроксимации её геометрическими элементами условно-развертывающихся поверхностей, причем точность построения зависит от количества элементов аппроксимации [1.3].

Использование метода триангуляции для ручного построения не представляется возможным, так как для формирования развертки, проектируемого изделия, необходимо большое количество входных данных, характеризующих пространственное расположение элементов аппроксимации. В настоящее время этот метод не учитывает свойства трикотажных полотен и их усадку при отделке.

Однако данный метод с использованием ЭВМ может позволить формирование лекала, проектируемого изделия, с учетом эмпирических значений физико-механических свойств полотна, геометрических изменений, получаемых в процессе технологической обработки и последующей эксплуатации изделия. Совокупность полученных и заложенных данных в конструкцию позволит получить заведомо качественную посадку готового изделия.

Однако, для получения требуемого количества наиболее точных входных данных (массив точек аппроксимируемой поверхности или координаты пространственного расположения элементов аппроксимации) необходимо использование дополнительных дорогостоящих сканирующих технических средств или аппаратов и дополнительного программного обеспечения для обработки данных.

Метод секущих плоскостей.

Данный метод предложен в 1954 году А. И. Ивановой и является первой попыткой в формировании развертки одежды с использованием начертательной геометрии и черчения.

Основа метода секущих плоскостей состоит в условном приравнивании каждого участка определенной детали к развертываемой геометрической поверхности и дальнейшем последовательном развертывании и проецировании на одну плоскость [1.4].

Однако трудоемкость и сложность совмещения внешних границ отдельных участков друг относительно друга не позволяет данный метод использовать на практике.

Метод геодезических линий.

Суть метода состоит в моделировании на поверхности тела ряда геодезических линий с заданным шагом и последовательном построении разверток выделенных участков поверхности в одной плоскости, ограниченных геодезическими линиями.

В настоящее время данный метод используется в сканировании фигуры человека. Метод геодезических линий позволяет в дальнейшем сформировать

пакет лекал с учетом физико-механических свойств полотна и особенностей технологической обработки (ВТО, швейная обработка и т.д.), а также предусмотреть геометрические изменения, появляющиеся в процессе эксплуатации изделия [1.4].

Таким образом, от выбора метода конструирования напрямую зависит не только качество посадки готового изделия и его эстетические характеристики, но также время, затраченное на конструкторскую подготовку и расход сырья.

При использовании одного из методов конструирования совместно с ЭВМ становится возможным проектирование разверток с учетом геометрических преобразований, получаемых цельновязаными изделиями в процессе технологической обработки и дальнейшей эксплуатации готового изделия. Однако такой способ построения развертки осуществим только при наличии в методе конструирования аналитической основы (например, расчетно-графический метод, метод геодезических линий, метод триангуляции и т.д.).

Наиболее популярным на производстве и простым в реализации методом конструирования является расчетно-графический (или его различные вариации). При использовании этого метода с применением ЭВМ возможно построение лекал деталей, проектируемого изделия, с меньшим пакетом требуемых входных данных, например, в сравнении с методом триангуляции или методом геодезических линий. Также этот метод не требует использование дополнительных сканирующих технических средств, аппаратов и соответствующего программного обеспечения.

Применение РГМ в конструировании сложного цельновязаного изделия позволит использовать различные методы конструирования, применяемые в швейном производстве. Такой подход к формированию разверток деталей сложного цельновязаного изделия позволит проектировать изделия с улучшенной посадкой, а также перейти к низкоуровневому проектированию узла соединения.

Низкоуровневым проектированием узла соединения назовем проектирование, которое заключается в размещении соединяемых петель в плоскости проектирования с наименьшим отклонением от проектируемой линии

соединения деталей. Например, отклонение линии соединения от проектируемой в кроеном изделии будет зависеть в большей степени от прибавки на посадку детали, навыков стачивания у швеи и настройки швейной машины, в меньшей степени – от типа полотна. При проектировании сложного цельновязаного изделия, аппроксимированные кромки соединяемых изделий, могут иметь большое отклонение от проектируемой линии соединения, что в дальнейшем может привести к ухудшению посадки изделия и его внешнего вида. Такое отклонение появляется в результате аппроксимации и балансирования элементов соединения, что вызвано ограничениями технологических возможностей вязальной машины и величиной относительного удлинения перерабатываемой пряжи.

1.1.2 Программные решения по автоматизации проектирования изделий

Внедрение систем автоматизированного проектирования в производство позволяет сократить время, необходимое для разработки пакета лекал. Данный показатель достигается, прежде всего, за счет автоматизации инструментов построения графических объектов, а также инструментов их редактирования, причем редактирование графических объектов может выполняться как индивидуально, так и в комплексе с другими графическими объектами.

Например, в кривую могут быть внесены изменения, касающиеся её взаимного расположения на плоскости относительно других графических объектов или изменения, касающиеся определенной группы её точек. В результате это может привести к взаимному изменению координат всех точек, принадлежащих кривой, или группы точек. То есть в первом случае будет выполнено перемещение кривой, а во втором – изменение её формы. Однако такие вносимые изменения можно отнести только к форме индивидуального редактирования графического объекта. Примером редактирования комплекса графических объектов может являться операция добавления технологических

припусков или изменения линейных размеров развертки, вызванных усадкой (присадкой) полотна и т.д.

Применение инструментов автоматизации построения и редактирования в системе проектирования подразумевает выполнение множества математических вычислений. Современный уровень компьютерной техники позволяет это выполнять, что в свою очередь создает тенденции в развитии систем автоматизированного проектирования.

Результатом развития является разработка новых модулей обработки информации, позволяющих внедрять в производство более сложные технологии, а также способные перенести проектирование на новый качественный уровень. Совершенствование инструментов построения и редактирования графических объектов, используемых в конструировании, дает возможность построения разверток не только с учетом физико-механических свойств перерабатываемого сырья, но также с учетом изменений линейных размеров деталей, получаемых в процессе технологической обработки и последующей эксплуатации готового изделия.

Преимущества использования системы автоматизированного проектирования при конструировании и разработке технологии вязания изделия заключаются в возможности обработки в малые промежутки времени огромного массива данных. Что в свою очередь позволяет выполнять конструкторскую и технологическую подготовку с более точными параметрами и корректировками в сравнении с ручной формой проектирования.

Рассмотрим системы автоматизированного проектирования, используемые в швейной и трикотажной промышленности России. Известно несколько уникальных систем автоматизированного проектирования, применяемых в швейном производстве: САПР Ассоль, СПО Леко.

САПР Ассоль.

Центр Ассоль занимается разработкой систем автоматизированного проектирования с 1999 года. С 2007 года данный центр является партнером и авторизированным разработчиком компании AutoDesk.

Конструирование в САПР Ассоль основано на параметрическом построении конструкции проектируемого изделия. Данный способ конструирования позволяет выполнять построение разверток деталей, используя известные и собственные методики конструирования. Также в данной системе реализован новый метод визуального параметрического конструирования изделия (рисунок 1.2). Этот метод состоит в построении развертки изделия за счет набора автоматизированных инструментов проектирования, реализованных в системе Ассоль. Управление инструментами выполняется при помощи комбинированного использования клавиатуры и мыши.

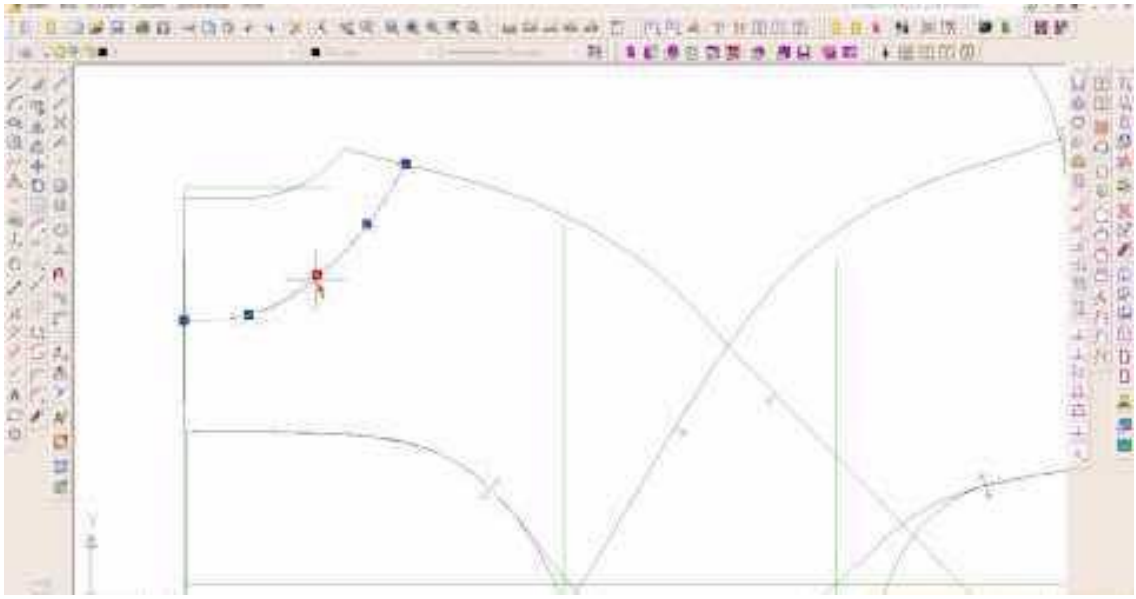


Рисунок 1.2 – Окно конструирования САПР Ассоль

Ввод аналоговых лекал в САПР Ассоль реализован в виде фотодигитайзера. Суть данного способа ввода лекал в систему состоит в распознавании сделанной фотографии аналоговой развертки или группы разверток, непосредственно в самой системе.

Подсистема вывода в САПР Ассоль способна выполнять печать на графопостроителе или раскрой полотна с использованием автоматических настилочно-раскройных комплексов.

Однако использование САПР Ассоль в проектировании изделий, выпускаемых на трикотажном производстве, не рационально. Имея достаточно мощную базу в конструировании, данная система не позволяет выполнить передачу пакета лекал в среду разработки программы вязания, поставляемую производителем вязального оборудования. Результатом использования системы Ассоль будет являться сокращение времени, необходимого на проектирование изделия, только за счет автоматизации этапа конструкторской подготовки изделия.

САПР Леко.

Система проектирования Леко разрабатывается фирмой Вилар, основанной в 1989 году и являющейся разработчиком программного обеспечения для легкой промышленности.

Способ конструирования в САПР Леко, также как и в САПР Ассоль, основан на параметрическом построении развертки. Уникальность системы проектирования Леко заключается в построении графических объектов при помощи формализованного текстового представления (алгоритма) методики конструирования (рисунок 1.3). Алгоритм построения разрабатывается при помощи собственного языка программирования с реализованным мощным математическим аппаратом, заложенным в системе проектирования. При таком способе построения разверток возможно выполнять разработку собственных методик конструирования с использованием дополнительных алгоритмов расчета, интегрированных в алгоритм построения конструкции. Это позволяет разрабатывать конструкцию изделия с учетом физико-механических свойств перерабатываемого сырья, а также геометрических изменений и деформаций, получаемых изделием в процессе технологической обработки и последующей эксплуатации.

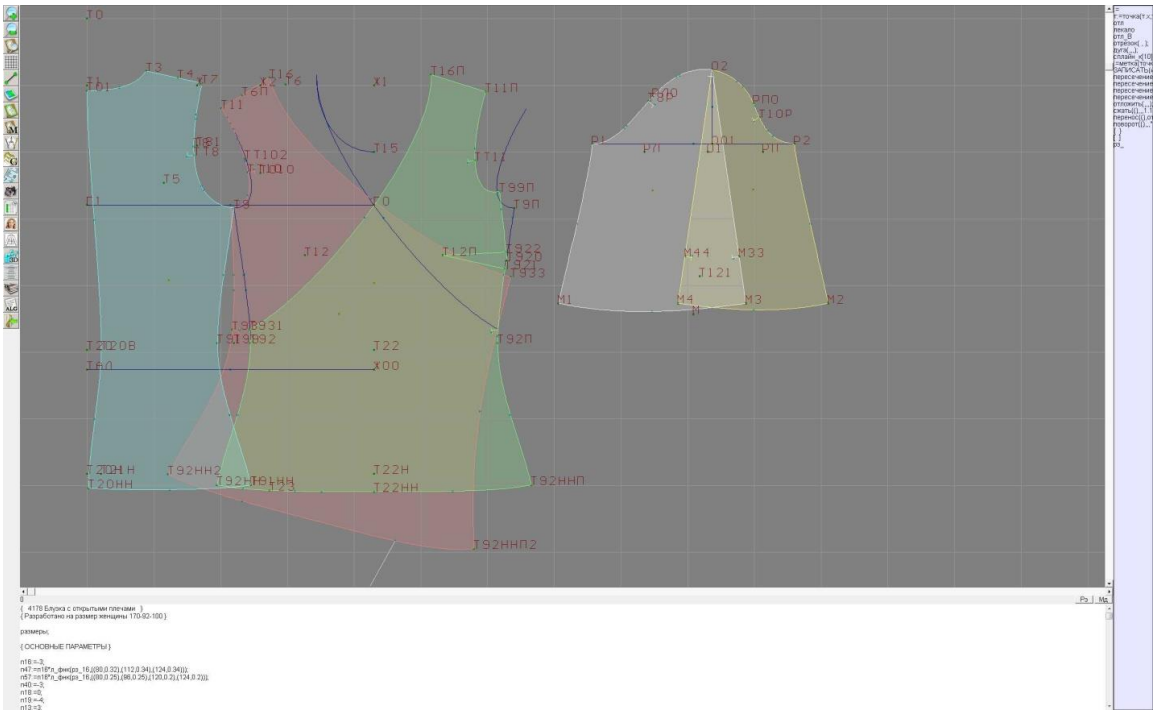


Рисунок 1.3 – Окно конструирования САПР Леко

Вывод результатов конструирования в системе Леко реализован только в виде печати пакета лекал на графопостроителе.

Таким образом, использование системы проектирования Леко на трикотажном производстве позволит сократить время необходимое на конструкторскую подготовку изделия. Также это позволит проектировать лекала с более качественной посадкой, что обусловлено возможностью ввода в алгоритм построения методов расчета корректирующих коэффициентов, полученных в результате исследования физико-механических характеристик перерабатываемого сырья, типа переплетения и его свойств и т.д. Однако, интегрирование готовых лекал в систему разработки программ вязания невозможно, так как в системе Леко отсутствует модуль аппроксимации.

В трикотажном производстве наиболее часто используются системы автоматизированного проектирования Stoll M1 3.15 и SDS One Apex. Рассмотрим принципы работы этих систем.

САПР STOLL M1 3.15.

Данная система автоматизированного проектирования выпускается фирмой STOLL, которая также является производителем плосковязальных машин. Разработка программ вязания в данной системе проектирования возможна только для вязальных машин, выпускаемых данной фирмой. Такое ограничение связано, прежде всего, с использованием в системе управления вязальной машиной собственного языка программирования.

Основу системы M1 составляют модули, служащие для разработки программы вязания. Конструкторская подготовка в данной системе проектирования (подсистема Shape Editor) представляет собой последовательный ввод размеров описанного прямоугольника вокруг графического объекта (рисунок 1.4). Причем начальные координаты построенного графического объекта будут равны координатам предыдущего графического объекта.

При проектировании можно вводить различные параметры расчета сбавок прибавок, например, расчет сбавки через два петельных ряда, однако, это можно выполнять только в ручном режиме по заранее рассчитанным данным.

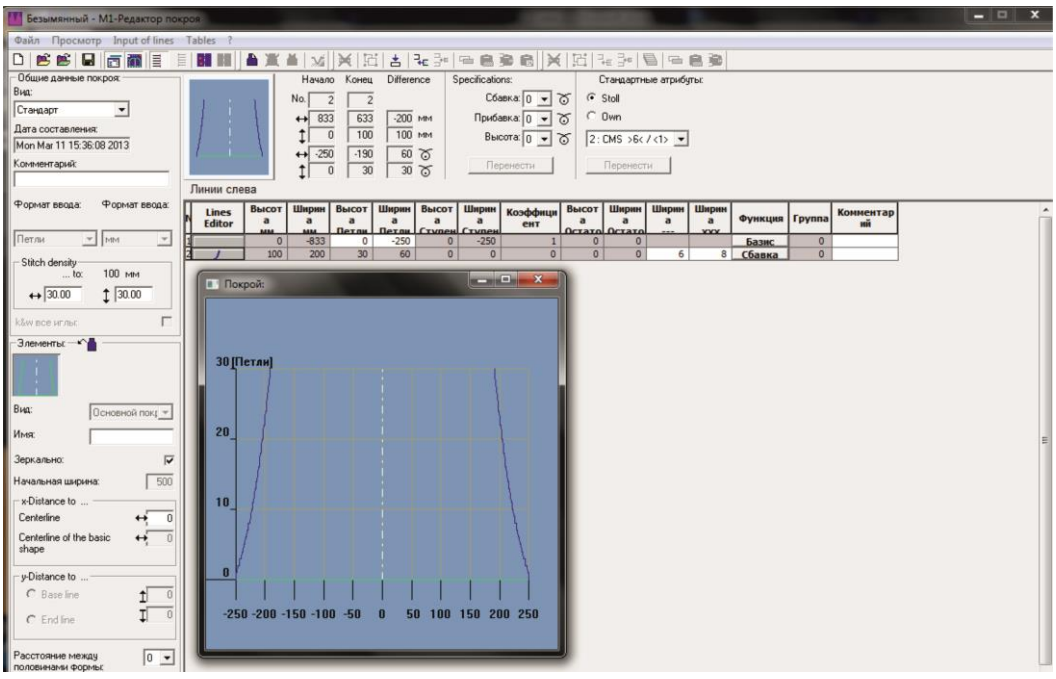


Рисунок 1.4 – Окно подсистемы Shape Editor в системе M1

В системе STOLL M1 имеется подсистема Shape Editor, предназначенная для конструкторской подготовки цельновязаных изделий. Принцип работы данной подсистемы заключается в проектировании цельновязаного изделия, находящегося в одной плоскости, на основе готового шаблона, имеющегося в базе данных M1 (рисунки 1.5, 1.6).

Подсистема Shape Editor не имеет аналогичных инструментов конструирования, реализованных в вышеописанных системах, например, отсутствует возможность параметрического конструирования, что в свою очередь не позволяет выполнить качественную конструкторскую подготовку без использования каких-либо дополнительных технических средств или программного обеспечения.

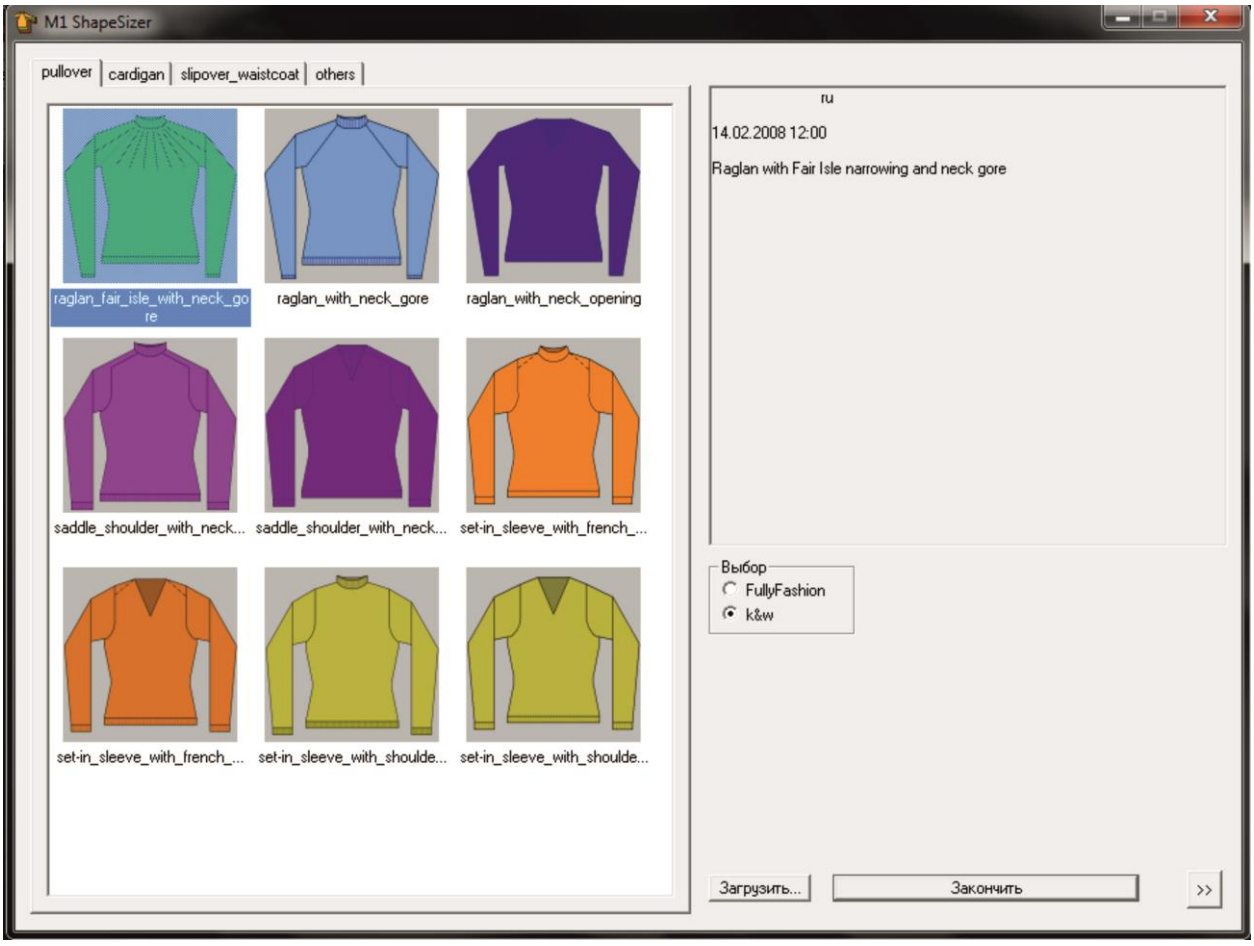
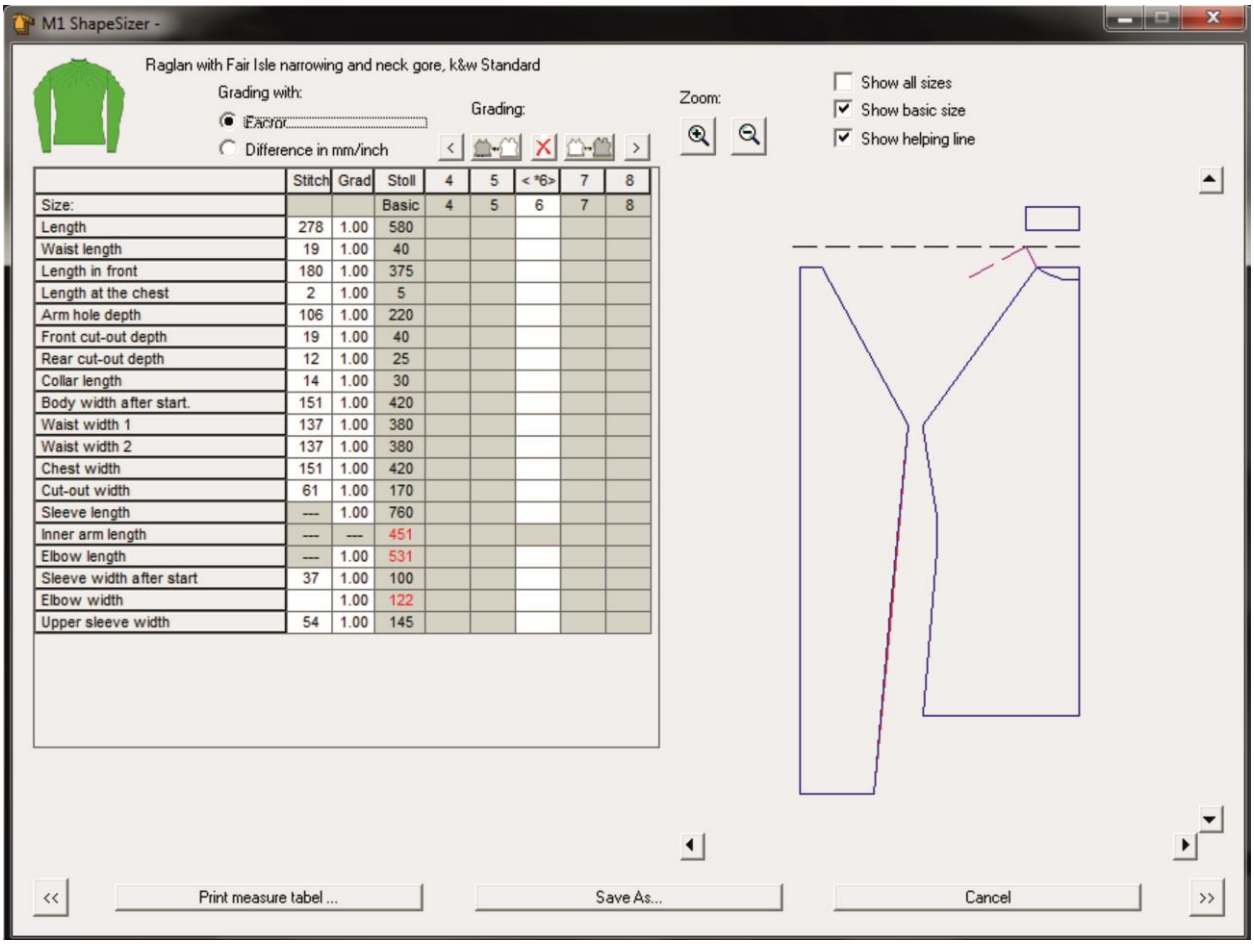


Рисунок 1.5 – Окно с шаблонами цельновязаных изделий в системе проектирования STOLL M1



	Stitch	Grad	Stoll	4	5	< *6>	7	8
Size:			Basic	4	5	6	7	8
Length	278	1.00	580					
Waist length	19	1.00	40					
Length in front	180	1.00	375					
Length at the chest	2	1.00	5					
Arm hole depth	106	1.00	220					
Front cut-out depth	19	1.00	40					
Rear cut-out depth	12	1.00	25					
Collar length	14	1.00	30					
Body width after start.	151	1.00	420					
Waist width 1	137	1.00	380					
Waist width 2	137	1.00	380					
Chest width	151	1.00	420					
Cut-out width	61	1.00	170					
Sleeve length	---	1.00	760					
Inner arm length	---	---	451					
Elbow length	---	1.00	531					
Sleeve width after start	37	1.00	100					
Elbow width		1.00	122					
Upper sleeve width	54	1.00	145					

Рисунок 1.6 – Окно установки линейных размеров проектируемого цельновязаного изделия

Таким образом, система проектирования STOLL M1 3.15 позволяет осуществлять конструкторскую подготовку трикотажных изделий на низком уровне, следствием чего, будет являться некачественная посадка готового изделия, а также большой объем сырья, затраченного на производство образцов. На этапе конструкторской подготовки цельновязаных изделий система проектирования STOLL M1 не позволяет выполнять конструирование сложных цельновязаных изделий произвольной формы, а ограничивает программиста набором собственных шаблонов.

САПР SDS ONE APEX.

Графическая станция SDS ONE выпускается производителем вязального оборудования Shima Seiki и предназначена для работы только с оборудованием данной фирмы (рисунок 1.7).

Система SDS ONE APEX объединяет различные стадии дизайна, программирования, моделирования, производства и др. в единый согласованный рабочий процесс, поэтому в данной системе можно выполнять проектирование как регулярных и цельновязаных изделий, так и создавать образы раскладок для автоматических настилочно-раскройных комплексов, разрабатывать программы для вышивальных машин и др.

Данная система обладает собственной программой конструирования, позволяющей проектировать развертку детали более точно, чем в STOLL M1. Это достигается за счет использования комплексных инструментов проектирования и редактирования при создании развертки детали в ручном режиме. Например, для создания развертки жилета, первоначально необходимо составить технический эскиз изделия. Затем программа конструирования распознает эскиз и создает сводную таблицу, содержащую параметры графических объектов, входящих в конструкцию жилета. Однако, формирование разверток деталей проектируемого изделия, при использовании различных методик конструирования, а также внедрения в их структуру собственных методов расчета графических объектов невозможно.

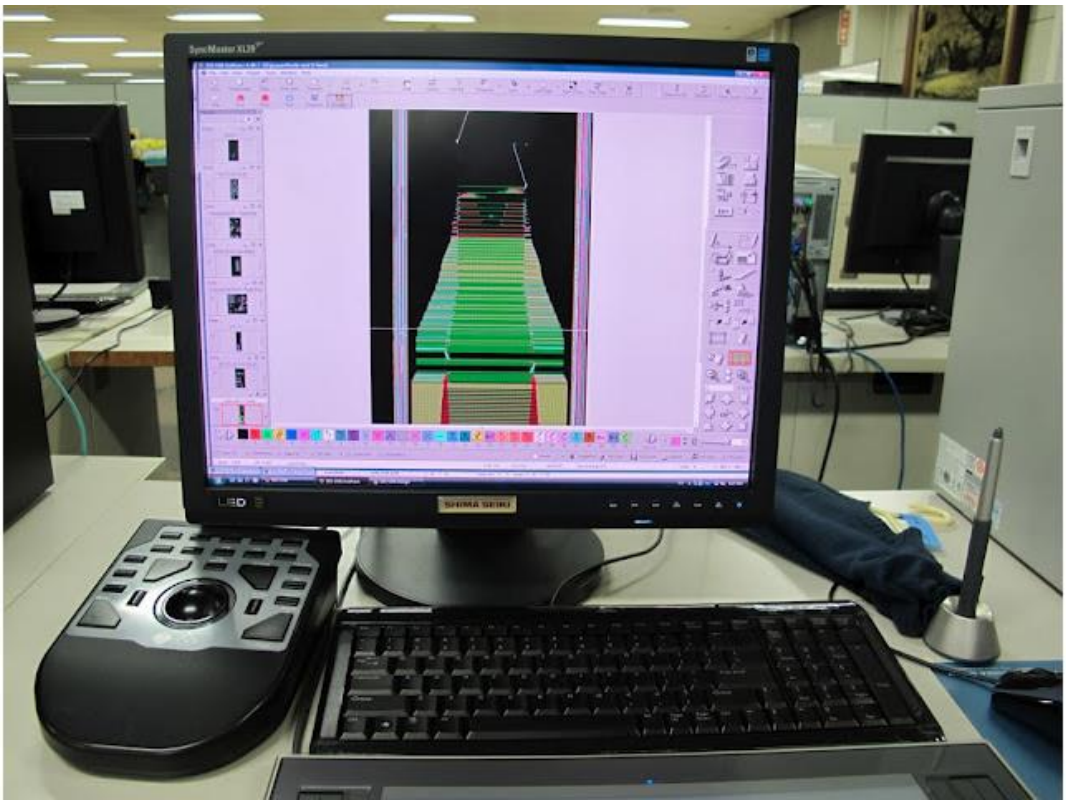


Рисунок 1.7 – Графическая станция SDS ONE

Проектирование цельновязаных изделий выполняется в автоматическом режиме по предварительно сформированному пакету лекал.

Таким образом, система SDS ONE APEX позволяет выполнять конструкторскую подготовку на более высоком уровне, чем STOLL M1. Это возможно за счет реализованного набора инструментов проектирования и редактирования графических объектов, используемых в конструкции изделия. Однако данная система предназначена только для работы с оборудованием фирмы Shima Seiki.

Рассмотренные специализированные системы автоматизированного проектирования позволяют формировать развертки деталей одежды, разрабатывать программы вязания, в том числе программы для вязания сложных цельновязаных изделий. Однако их комплексное применение в проектировании цельновязаных изделий со сложными линиями соединений невозможно, так как:

- специализация отдельных систем проектирования направлена только на область применения в определенном виде производства (в швейном или трикотажном производстве);
- не имеют возможности преобразования сложных линий соединения деталей цельновязаных изделий в технологию вязания.

Данный недостаток работы различных систем автоматизированного проектирования можно исключить за счет создания сложных модулей перехода (конвертеров).

Для предприятия разработка конвертера экономически не эффективна, так как кроме покупки двух различных систем проектирования предприятию необходимо будет произвести финансовые затраты на оплату работы по созданию конвертера высококвалифицированному программисту. Однако технологически такой конвертер позволит не только сократить время необходимое на разработку конструкции и проектирование технологии вязания трикотажных изделий, а также расширить ассортимент выпускаемых изделий, улучшить посадку и уменьшить себестоимость выпускаемой продукции.

С целью разработки технологических функций модуля перехода, способов проектирования рельефа и методов автоматизации проектирования узлов соединения в сложных цельновязаных изделиях проведем анализ этапа технологической подготовки сложного цельновязаного изделия.

1.2 Анализ этапа технологической подготовки сложных цельновязаных изделий

На этапе технологической подготовки спроектированные развертки деталей изделия являются факторами определяющими технологию вязания. Причем зависимость взаимного расположения и формы графических объектов, оформляющих кромки соединения, где совокупность этих кромок формирует узел соединения деталей в цельновязаном изделии, будет являться определяющей составляющей в расчете машинного времени и в проектировании рационального

процесса вязания, а также в формировании внешнего вида изделия. Оптимизация совокупности вышеперечисленных показателей позволит снизить себестоимость изделия, что, в свою очередь, приведет к увеличению чистой прибыли у предприятия.

На этапе технологической подготовки выполняется аппроксимация графических объектов, входящих в состав контура, сформированной развертки детали изделия. Аппроксимацию необходимо выполнять по высоте петельного ряда (вертикальная ось) и ширине петельного столбика (горизонтальная ось). В результате аппроксимации кромка развертки детали примет ступенчатый вид (рисунок 1.8).

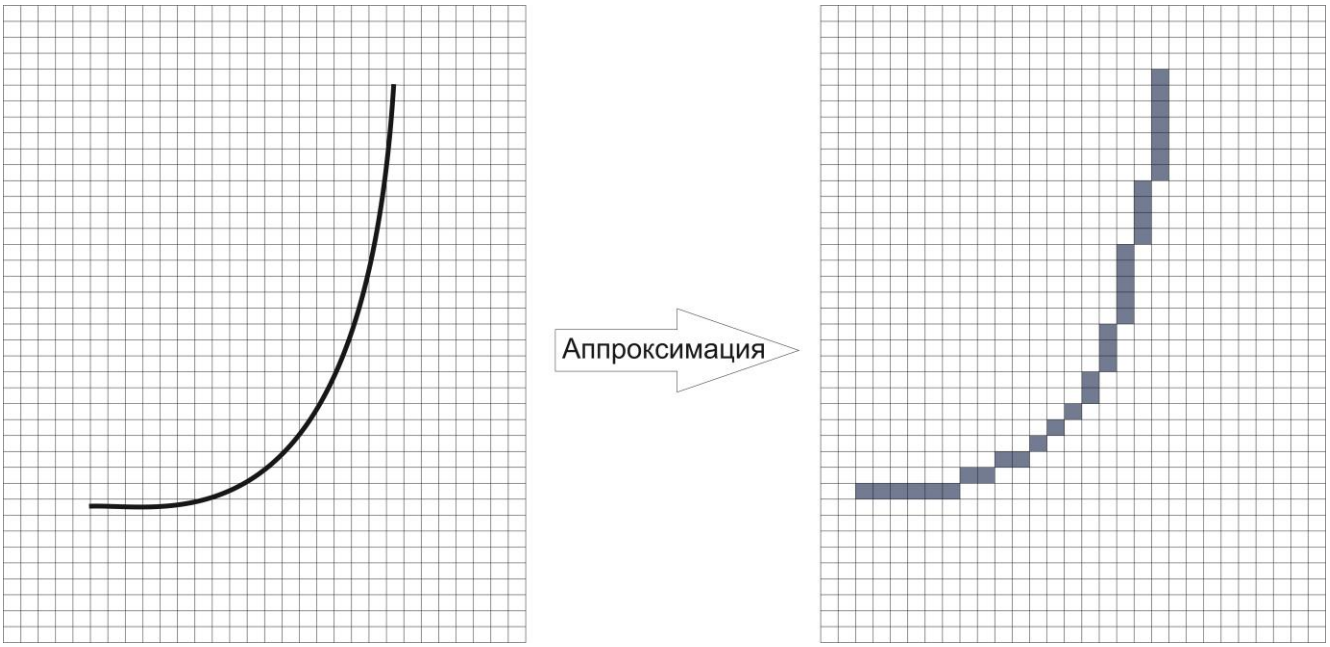


Рисунок 1.8 – Аппроксимация графического объекта

В современных системах автоматизированного проектирования операция аппроксимации контура развертки выполняется по шаблону, заложенному в систему. Например, в системе проектирования STOLL M1 3.15 операция аппроксимации может быть выполнена только по трем графическим объектам: отрезку прямой линии, а также отрезкам квадратной и кубической парабол (рисунки 1.9, 1.10, 1.11). Причем параметры аппроксимации, такие как высота

петельного ряда, ширина петельного столбика, максимальная сбавка (прибавка) в петельном ряду и т.д. могут быть применены только к целому графическому объекту.

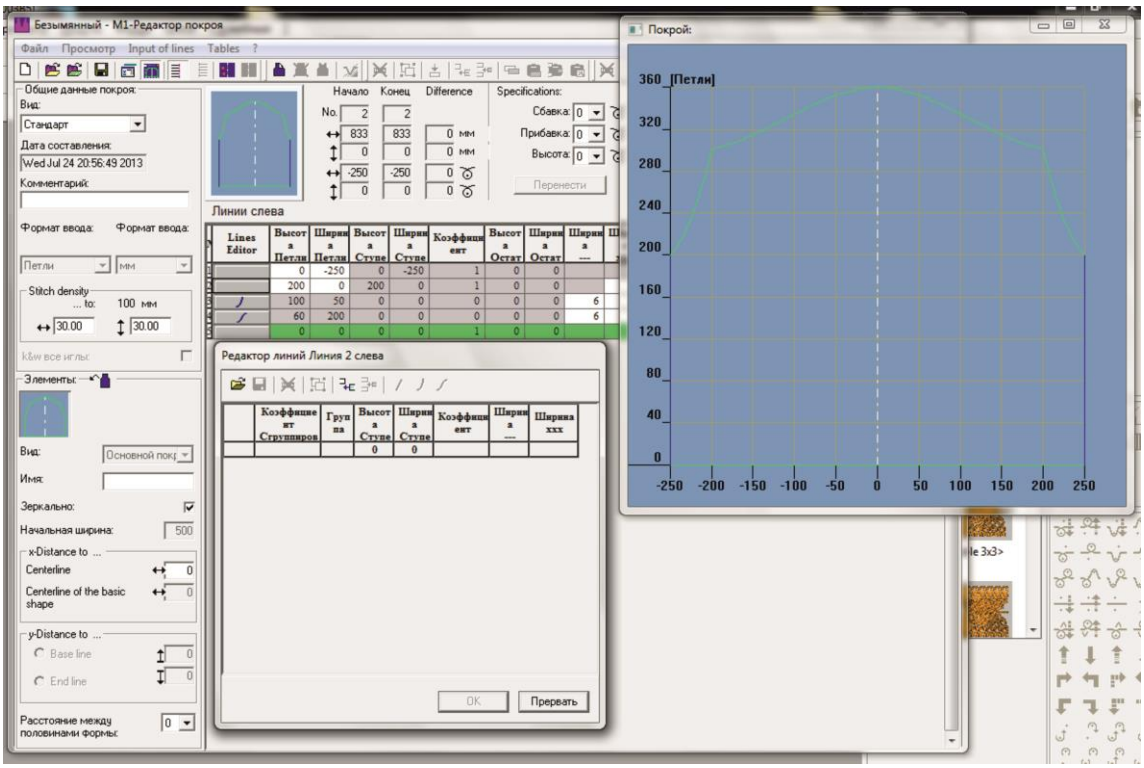


Рисунок 1.9 – Аппроксимация отрезка прямой линии в системе проектирования STOLL M1 3.15

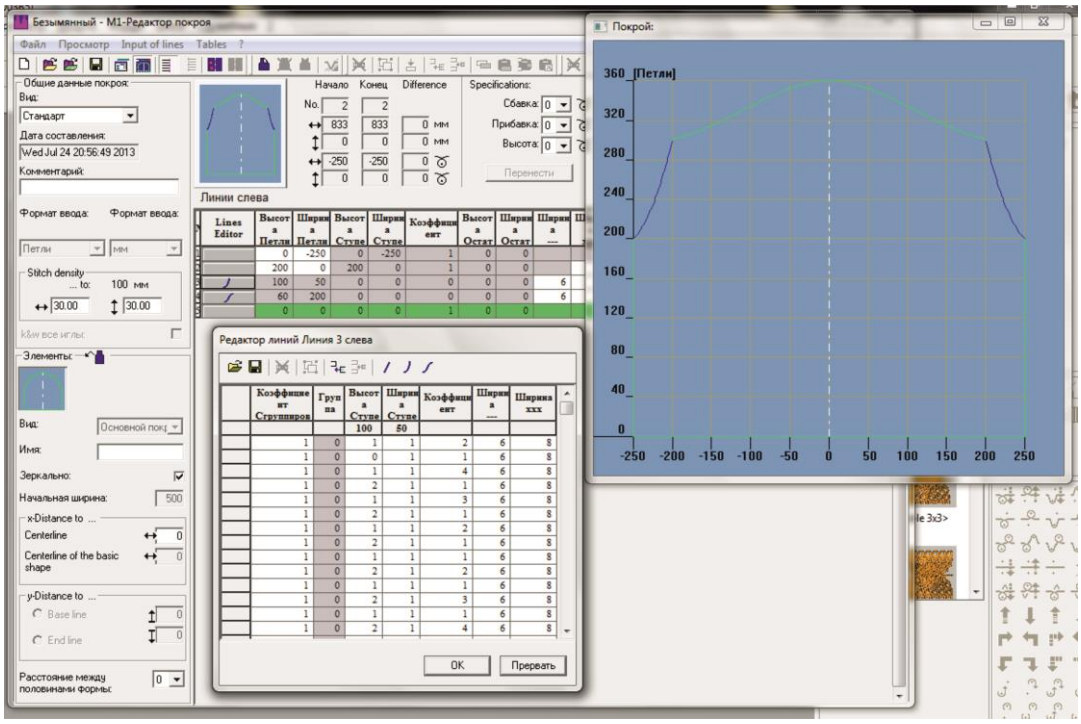


Рисунок 1.10 – Аппроксимация квадратной параболы в системе проектирования STOLL M1 3.15

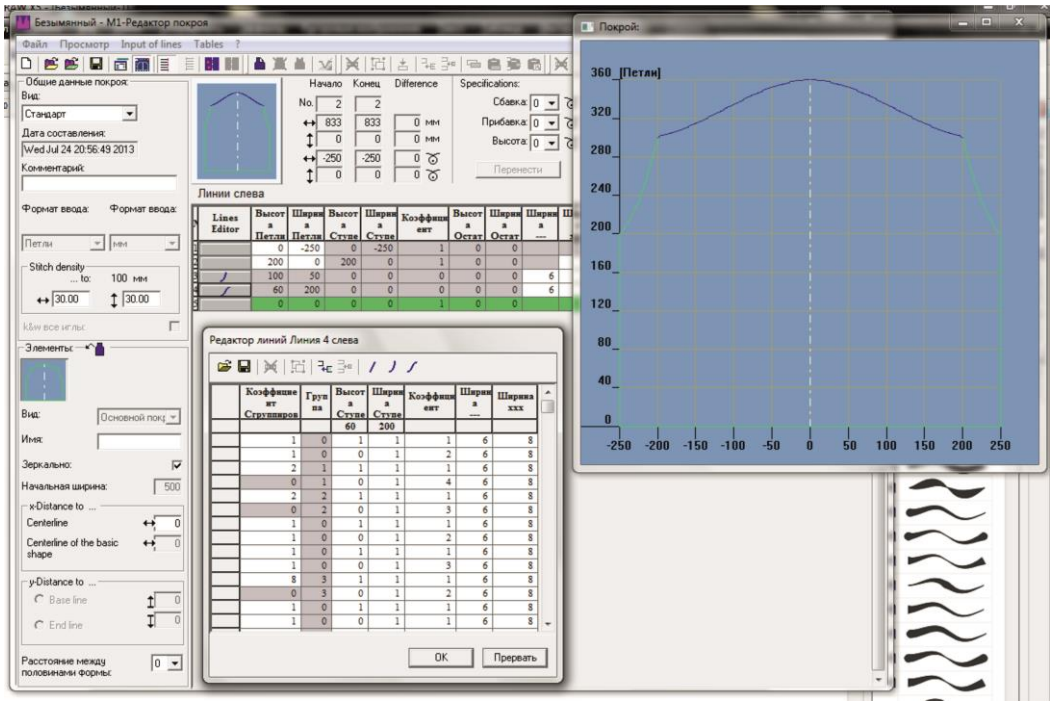


Рисунок 1.11 – Аппроксимация кубической параболы в системе проектирования STOLL M1 3.15

Отсутствие возможности задания параметров аппроксимации для различных участков аппроксимируемого графического объекта не позволяет проектировать сбавки (прибавки) в узле соединения цельновязаного изделия с учетом художественного замысла, конструктивных особенностей проектируемого изделия и т.д.

Узел соединения деталей в сложном цельновязаном изделии является сложным участком. Это связано, прежде всего, с необходимостью выполнения перерасчета сбавок (прибавок) на участках соединения деталей после процесса аппроксимации. Такой перерасчет сбавок (прибавок) необходим для снижения напряжения в петлеобразующих органах, получаемых во время процесса переноса петель, а также для получения качественного соединения деталей в сложном цельновязаном изделии.

Проблемой проектирования узлов соединения в сложных цельновязаных изделиях занималась Сичкарь Т.В. [3.1].

В своей работе автором установлено, что соединение деталей в цельновязаном изделии может осуществляться не только вдоль петельного столбика и ряда, а также по наклонному участку. Автором рассмотрены некоторые виды трикотажных изделий и определены участки соединения основных деталей в плечевых, поясных изделиях, а также головных уборах. Однако Сичкарь Т.В. в своей работе не приводит технологии соединения деталей в сложных цельновязаных изделиях с произвольной формой линии соединения, например, к таким изделиям можно отнести сувенирный ассортимент, трикотаж технического назначения и т.п.

Скопинцева Е.А. проблему соединения деталей в цельновязаном изделии рассматривала на примере плечевого изделия [3.2]. В своей работе автор делает акцент на разбивку узла соединения деталей (стана и рукава) на две части.

В нижней части проймы сбавки (прибавки) рассчитываются по аппроксимированной прямыми отрезками кромке, причем соединяемые участки имеют одинаковое число петельных рядов. Во второй части узла соединения, выполняется расчет сбавок (прибавок) по отдельным участкам, имеющим

различное количество петельных рядов. В этом случае дополнительно осуществляется расчет дополнительных петельных рядов необходимых для посадки рукава в пройму стана.

Автор в своей работе не указывает расположение границы между двумя частями узла соединения, а именно линию, после которой участки соединения будут иметь различное количество петельных рядов. Кроме того, разработанная методика проектирования узла соединения деталей в сложном цельновязаном изделии за счет расчета дополнительных петельных рядов, необходимых для посадки рукава в пройму стана, конструктивно упрощает узел соединения, что, в свою очередь, приведет к ухудшению посадки изделия на теле человека. Также автором не рассмотрена возможность применения методики проектирования узла соединения в сложном цельновязаном изделии в случае, когда участки соединения состоят из нескольких сложных графических объектов, тем самым создавая сложный профиль (рисунок 1.12).

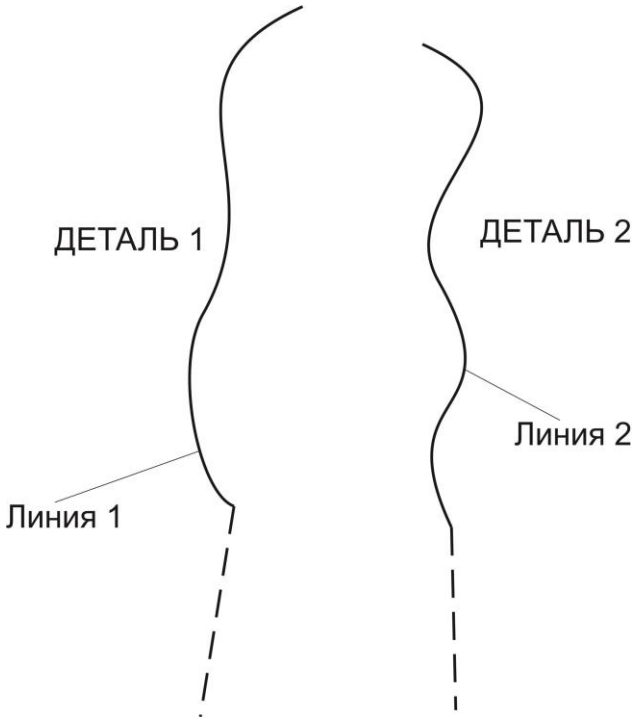


Рисунок 1.12 – Абстрактный узел соединения в сложном цельновязаном изделии

На рисунке 1.12 сплошные линии «Линия1» и «Линия2» обозначают участки по которым необходимо выполнить соединение деталей в сложном цельновязаном изделии, которое по художественному замыслу имеет очень сложный узел соединения. Соответственно «ДЕТАЛЬ1» и «ДЕТАЛЬ2» обозначают взаимное расположения соединяемых деталей.

Использование методики проектирования узла соединения разработанной Скопинцевой Е.А., не позволит выполнить этот расчет на приведенном примере. Это, в первую очередь, связано с отсутствием в методике алгоритма нахождения границ, после которых участки соединения будут иметь одинаковое и различное количество петельных рядов. Кроме того способ аппроксимации профилей участков соединения прямыми отрезками приведет к упрощению узла соединения, следовательно, к ухудшению художественного оформления изделия.

Проектирование узла соединения в сложном цельновязаном изделии является одной из самых сложных задач на этапе технологической подготовки изделия. Однако, правильно спроектированный узел, максимально приближенный к исходной конструкции не позволит сформировать у изделия хорошую посадку на теле человека.

Посадка готового изделия на теле человека в первую очередь будет зависеть от факторов, формирующих рельеф. В процессе технологической обработки сложного цельновязаного изделия такими факторами будут являться операции по формированию рельефа в процессе влажно-тепловой и швейной обработки (обработка вытачек).

С целью улучшения качества посадки на теле человека, а также расширения ассортимента за счет формирования внешнего вида сложного цельновязаного изделия рельефными участками, рассмотрим способы формирования рельефа.

1.2.1 Способы формирования рельефа на сложных цельновязаных изделиях

Наиболее простым способом получения выпуклого участка трикотажа в производстве является процесс его формирования за счет влажно-тепловой обработки по заранее известному контуру. Суть данного способа заключается в вытягивании участка трикотажа и его закреплении при помощи горячего пара. При таком формировании выпуклого участка возникает принудительное изменение линейных параметров петель. Следствием этого послужит ухудшение внешнего вида изделия за счет появления участка с вытянутыми петлями, а также появление локальных разрушений пряжи в процессе эксплуатации.

Одним из традиционных и популярных способов формирования рельефа в одежде является обработка участка полотна вытачкой швейного типа (рисунок 1.13).

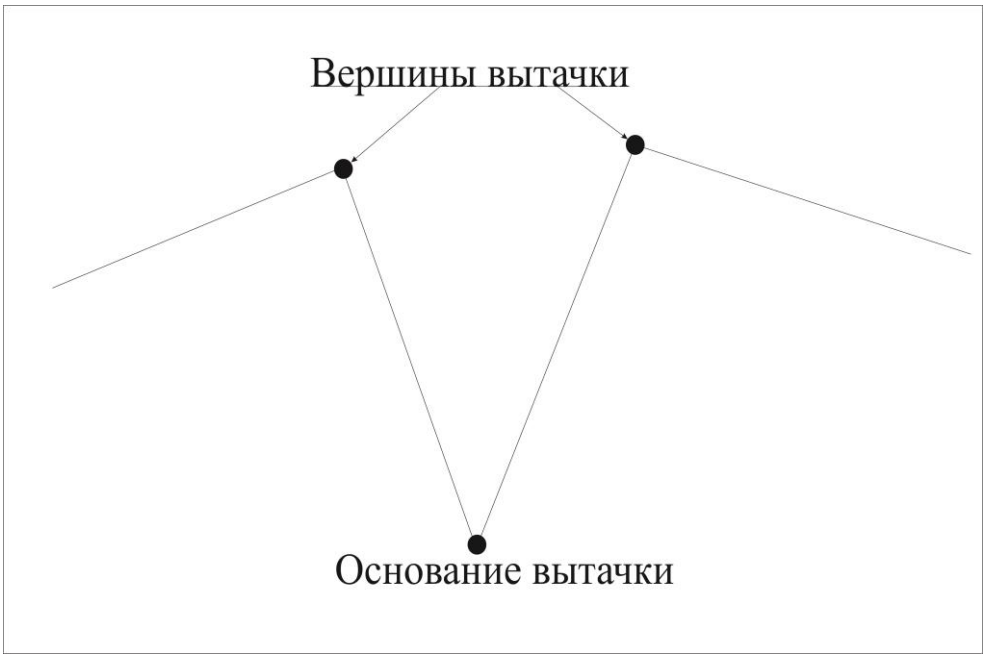


Рисунок 1.13 - Вытачка швейного типа

Суть данного способа заключается в прокладывании строчки от основания вытачки к её вершинам. Причем тип швейной машины, а также вид строчки

подбирается исключительно из физико-механических свойств швейной нитки, типа и толщины полотна.

Вытачка может быть разрезного и неразрезного типа. Технологическая обработка для неразрезного типа вытачки заключается в прокладывании строчки по краю вытачки и дальнейшем заутюживании её остатка. При разрезном типе вытачки кроме швейной обработки выполняется подкрой остатка.

Применение швейных вытачек в конструкции сложного цельновязаного изделия нерационально. Это обусловлено, прежде всего, включением в карту технологической обработки операций швейного типа. Кроме того швейная вытачка приводит к локальному разрушению участка трикотажа (разрезной тип вытачки), а также ухудшению внешнего вида готового изделия за счет нарушения рисунка переплетения.

Кроме вышеописанных способов в трикотажном производстве формирование рельефа возможно в процессе вязания. В настоящее время наиболее популярными способами формирования рельефа в процессе вязания являются:

- способ формирования рельефа на трикотажном полотне за счет группового переноса петель;
- способ формирования рельефа на трикотажном полотне за счет провязывания дополнительных петельных рядов между петельными рядами основного полотна, то есть использования частичного вязания.

Суть первого способа состоит в вертикальном разделении полотна на несколько зон вязания, где каждую зону вяжет отдельный нитеводитель .

Рассмотрим пример формирования рельефа на трикотажном полотне при помощи первого способа (рисунок 1.14).

На рисунке 1.14 изображена деталь стана с плечевыми вытачками, причем характер и форма данных вытачек напрямую зависит от эскиза художника, то есть в данном примере первоочередная задача конструктора – формирование внешнего

вида изделия, а не его практичной значимости (посадки и т.д.). Переплетение основного полотна – кулирная гладь.

Допустим при вязании детали до начала вытачек в вязании задействовано на двухсистемной машине два нитеводителя. Однако при начале вязания участка с вытачками необходимо выполнить ввод в вязание дополнительного нитеводителя при этом каждый нитеводитель будет работать на разных участках, а при начале вязания участка с горловиной дополнительно потребуется еще один нитеводитель (рисунок 1.15). В этом случае имеем четыре разных участка.

Таким образом, при вывязывании данной детали потребуется минимум 4 нитеводителя.

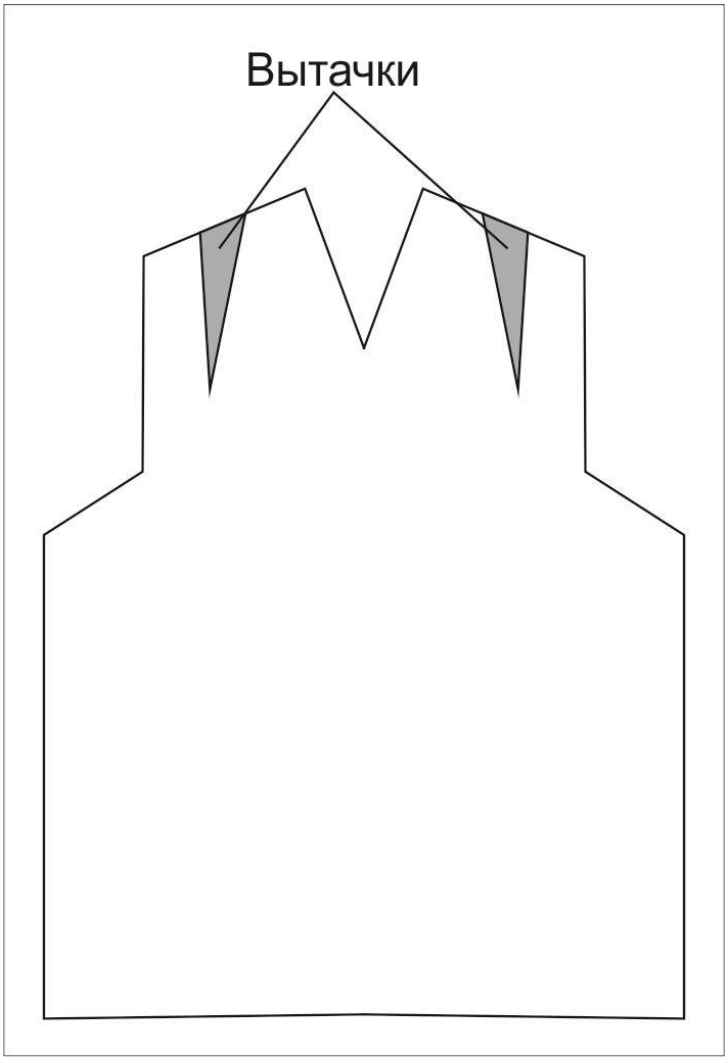


Рисунок 1.14 – Деталь стана с плечевыми вытачками

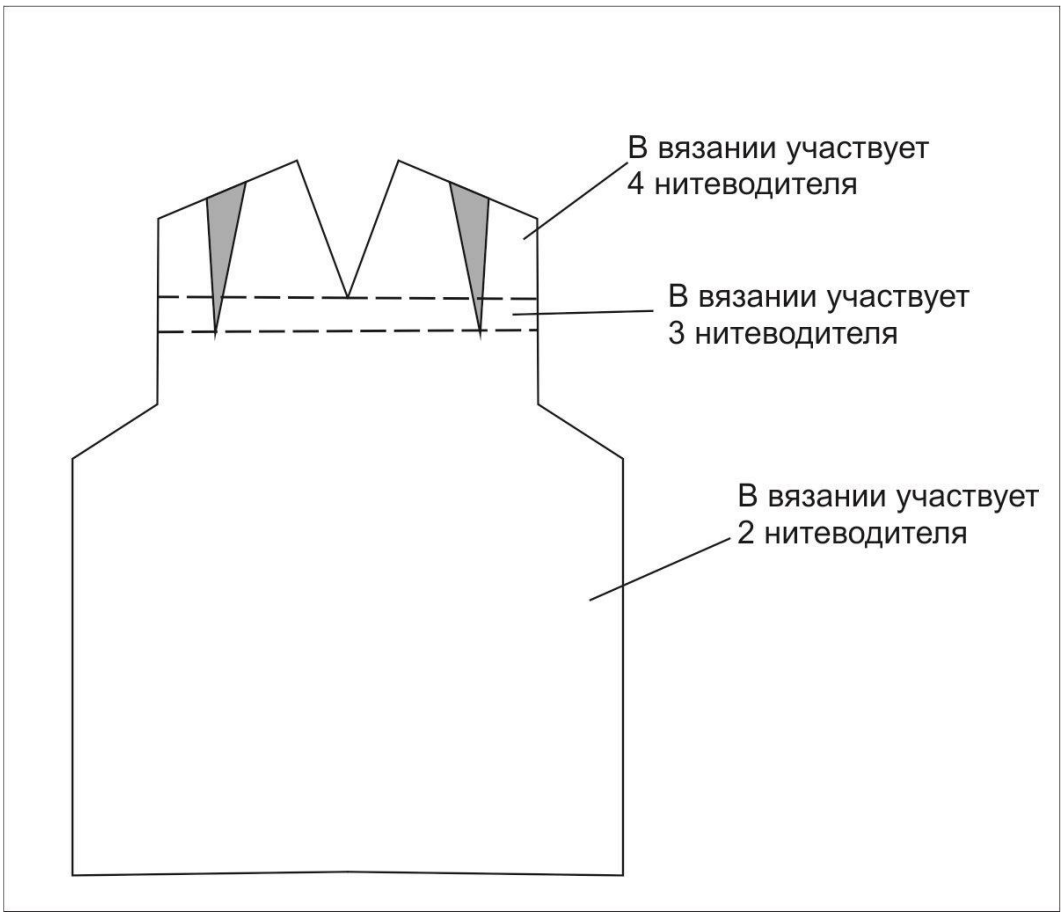


Рисунок 1.15 – Разделение детали на участки по количеству нитеводителей, участвующих в вязании

При таком способе формирования выпуклого участка трикотажа будет иметь место петельный столбик со сдвоенными петлями, что может изменить структуру базового переплетения и явиться фактором, ухудшающим внешний вид изделия. Кроме того при переносе петель возможно возникновение дефектов, так как сама операция переноса является сложным процессом. Также данная операция может явиться ограничением в формировании выпуклого участка при использовании двойных и одинарных сложных переплетений на базе ажурных, футерованных, прессовых и т.д. переплетений.

Суть второго способа заключается в формировании выпуклого участка трикотажа за счет провязывания дополнительных петельных рядов между петельными рядами основного полотна (рисунок 1.16).

Данный способ позволяет осуществлять формирование выпуклого участка трикотажа без ухудшения эстетических характеристик внешнего вида изделия, а также локальных нарушений структуры трикотажа, если дополнительные ряды провязываются основным переплетением. Однако число дополнительных рядов должно быть кратно раппорту переплетения.

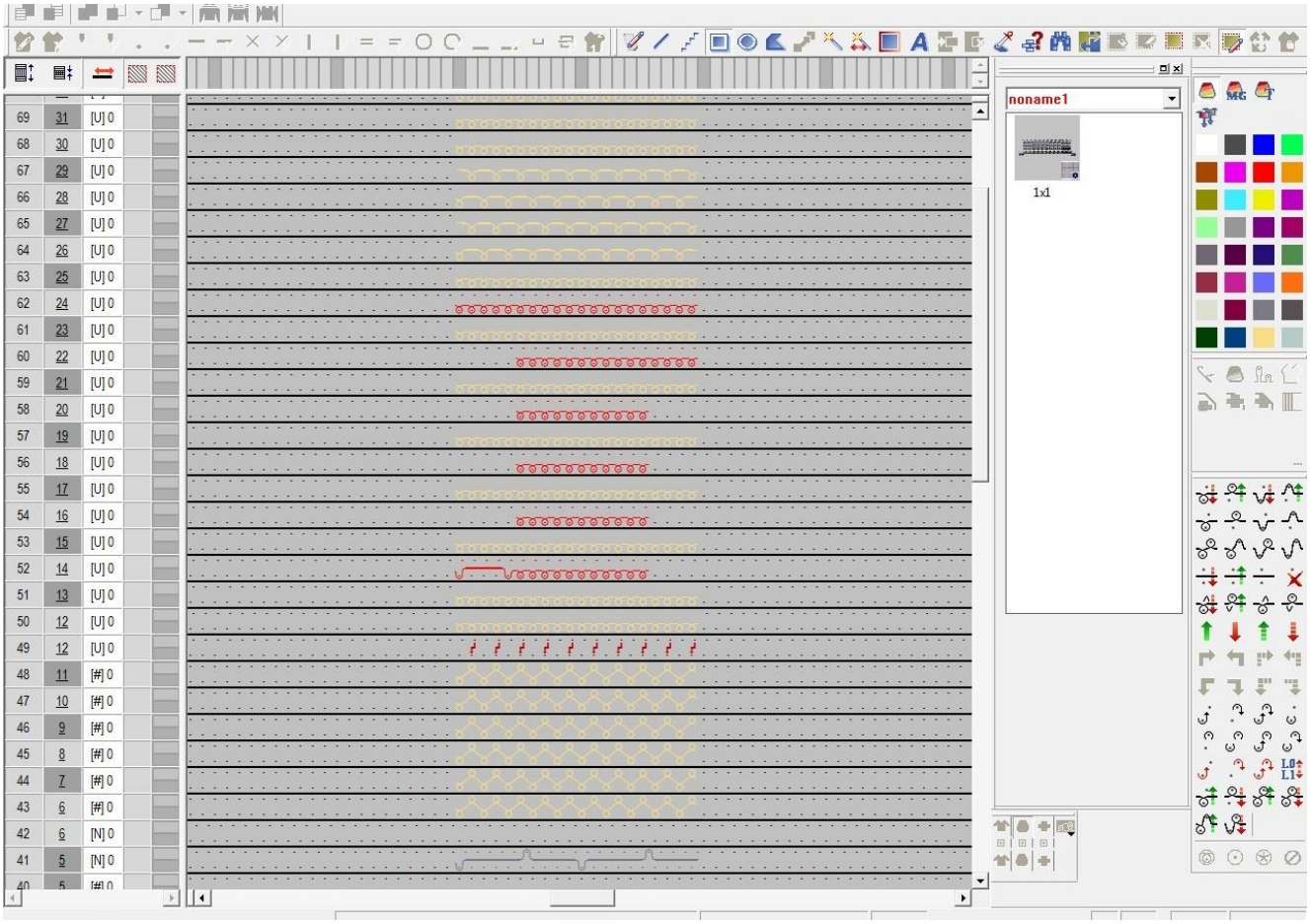


Рисунок 1.16 – Провязывание дополнительных петельных рядов между петельными рядами основного полотна в STOLL M1 3.15

На рисунке 1.16 показано, что ввод нитеводителя, провязывающего дополнительные петельные ряды, выполняется в 52 технологическом ряду.

Анализ патентной и технической литературы в области проектирования выпуклых участков трикотажа позволил выявить отсутствие методики проектирования формы выпуклости при провязывании дополнительных петельных рядов. Также данный анализ позволил выявить отсутствие методики,

позволяющей геометрически заменить вытачки способом, предусматривающим провязывание дополнительных петельных рядов. Методы проектирования выпуклых участков трикотажа другими вышеописанными способами заложены в методики конструирования и являются их неотъемлемыми частями.

Таким образом, проведенный анализ критериев проектирования сложных цельновязаных изделий позволил определить задачи, которые должны быть решены при выполнении данной работы.

1.3 Постановка задач исследования

1. Определить графические объекты, позволяющие перейти от конструирования к проектированию технологии вязания сложных цельновязаных изделий.
2. Разработать технологию проектирования сбавок (прибавок) по профилю графического объекта.
3. Разработать технологический модуль автоматического расчета сбавок (прибавок) по профилю графического объекта с учетом различных параметров аппроксимации.
4. Разработать алгоритм расчета высоты участка профиля соединения деталей в сложном цельновязаном изделии, содержащем в себе одинаковое и разное количество петельных рядов.
5. Выполнить исследование наиболее рационального способа соединения деталей в сложном цельновязаном изделии.
6. Разработать метод соединения деталей сложного цельновязаного изделия по сложному профилю, включающего множество графических объектов.
7. Разработать алгоритм компенсирования петельными рядами и столбиками кромок соединения деталей сложного цельновязаного изделия при использовании сложного профиля соединения.
8. Разработать технологический модуль автоматического проектирования соединения деталей в сложном цельновязаном изделии по сложному профилю с учетом различных параметров аппроксимации.

9. Разработать технологию и способ проектирования профиля кривой, оформляющей выпуклый участок на цельновязаном изделии за счет провязывания дополнительных петельных рядов с использованием выбранных графических объектов.
10. Разработать способ получения выпуклости на сложном цельновязаном изделии, геометрически заменяющий вытачку, провязыванием дополнительных петельных рядов.
11. Разработать технологию провязывания дополнительных петельных рядов на выпуклых участках сложных цельновязаных изделиях.
12. Разработать модуль автоматического проектирования выпуклого участка произвольной формы за счет провязывания дополнительных петельных рядов на сложном цельновязаном изделии.
13. Разработать программу для конструирования сложных цельновязаных изделий произвольной формы с возможностью получения выпуклых участков, при использовании любой расчетно-графической методики конструирования, а также позволяющей выполнить переход к разработке технологии вязания.

ВЫВОДЫ ПО ГЛАВЕ 1

1. Установлено, что поиск технических решений, связанных с автоматизацией процесса проектирования сложных цельновязаных изделий является наиболее перспективным направлением.
2. Проведенный анализ методов конструирования показал, что наиболее рациональным и простым в реализации методом для конструирования сложных цельновязаных изделий является расчетно-графический.
3. На основании анализа известных систем автоматизированного проектирования, используемых в России, установлено отсутствие программного решения, позволяющего выполнить переход от конструкторской подготовки сложного цельновязаного изделия к разработке программы его вязания.
4. Определен наиболее рациональный способ формирования рельефа с помощью провязывания дополнительных, позволяющий получить выпуклую поверхность трикотажа без изменения структуры переплетения.
5. Поставлены задачи исследования, решение которых позволит создать новый подход к проектированию сложных цельновязаных изделий, а также разработать экономически эффективное программное решение, способное на синхронизированную работу с системами автоматизированного проектирования других разработчиков, используемых в трикотажном производстве.

2 РАЗРАБОТКА КОНЦЕПЦИИ ПРОЕКТИРОВАНИЯ СЛОЖНЫХ ЦЕЛЬНОВЯЗАНЫХ ИЗДЕЛИЙ

Разработка технологии вязания сложных цельновязаных изделий является сложным процессом, включающим в себя разработку пакета лекал, разработку технологии вязания и разработку цепочки последующей технологической обработки.

Наиболее сложным этапом в проектировании цельновязаного изделия является этап по разработке технологии вязания. Так как формирование сложного цельновязаного изделия, а именно соединение между собой его основных и дополнительных деталей, в большинстве случаев, полностью осуществляется на вязальной машине непосредственно в процессе вязания, то наиболее технологически сложными участками являются узлы соединения. Это связано с тем, что в момент расчёта необходимых сбавок (прибавок) в узле соединения деталей необходимо выполнять параллельное балансирование петельных рядов и столбиков по кромкам соединения деталей, что необходимо для формирования качественной посадки изделия, эстетичного внешнего вида и т.д. Также сложность разработки технологии вязания на участке соединения деталей связано с технологическими ограничениями вязальной машины, например, с ограничением по сдвигу [3.2].

В настоящее время существует множество систем автоматизированного проектирования, позволяющих осуществлять разработку технологии вязания сложных цельновязаных изделий на базе вязальных машин различных марок. Однако возможности проектирования таких систем ограничены относительно малым набором модулей цельновязаных изделий, причем разработка нового вида сложного цельновязаного изделия возможна только на базе нового шаблона. Следствием технологических ограничений системы проектирования является повышение трудоемкости этапа проектирования сложного цельновязаного изделия, и увеличение объема времени, затрачиваемого на этап проектирования. Кроме этого шаблонное ограничение системы проектирования снижает

эффективность использования вязальной машины на производстве, что непосредственно оказывает влияние на размер чистой прибыли, получаемой предприятием. Такое неэффективное использование вязального оборудования будет иметь место на предприятиях, выпускающих трикотажные изделия потребительского назначения: верхнетрикотажные изделия, изделия бельевой группы и т.п., так как таким предприятиям характерна частая смена ассортимента, что связано с необходимостью следования тенденциям моды. Причем смена ассортимента должна проходить в максимально сжатые сроки.

Большинство современных систем проектирования, используемых на трикотажном производстве, не позволяют выполнять частую и оперативную смену ассортимента сложных цельновязаных изделий, особенно, если эти изделия имеют нетрадиционные узлы соединения, характерные изделиям верхнего трикотажа, сувенирного ассортимента, трикотажных изделий специального или технического назначения.

Сложности, возникают на этапе проектирования сложных цельновязаных изделий из-за отсутствия возможности параметрического построения конструкции изделия внутри системы проектирования, используемой на трикотажном производстве.

Отсутствие такой возможности не позволяет осуществлять автоматическую градацию по размеро-росту, использовать различные методики конструирования, отслеживать динамическое изменение контура лекала в процессе аппроксимации и т.д.

Параллельное использование на трикотажном производстве нескольких систем автоматизированного проектирования, например, использование системы, необходимой для разработки технологии вязания и системы конструирования, нерационально (см. главу 1). Однако эффективность использования нескольких систем проектирования будет более высокой, чем при использовании только одной системы, что связано с автоматизацией некоторых ручных операций: конструирование, аппроксимация и т.п.

2.1 Ввод понятия «комплексная система проектирования». Разработка технических аспектов комплексной системы проектирования

Несомненно, внедрение системы конструирования в трикотажное производство позволит сократить время, необходимое на разработку пакета лекал изделия с учетом параметров трикотажа. Причем сокращение времени, необходимого для разработки пакета лекал, прежде всего, будет выполняться за счет автоматизации инструментов построения графических объектов, а также инструментов их редактирования, причем редактирование графических объектов может осуществляться как индивидуально, так и в комплексе с другими графическими объектами.

Например, в кривую могут быть внесены изменения, касающиеся её взаимного расположения на плоскости относительно других графических объектов или изменения, касающиеся определенной группы её точек. В результате это приведет к взаимному изменению координат всех точек, принадлежащих кривой, или только группы точек. То есть в первом случае будет выполнено перемещение кривой, а во втором – изменение её формы. Однако такие вносимые изменения можно отнести только к форме индивидуального редактирования графического объекта. Примером редактирования комплекса графических объектов может являться операция добавления технологических припусков или изменения линейных размеров развертки, вызванных усадкой (присадкой) полотна и т.д.

Применение инструментов автоматизации построения и редактирования в системе проектирования подразумевает выполнение множества математических вычислений. Современный уровень компьютерной техники позволяет это выполнять, что в свою очередь создает тенденции в развитии систем автоматизированного проектирования.

Результатом модернизации является разработка новых модулей обработки информации, позволяющих внедрять в производство более сложные технологии, а также способные перенести проектирование на новый качественный уровень.

Совершенствование инструментов построения и редактирования графических объектов, используемых в конструировании, дает возможность построения разверток не только с учетом физико-механических свойств перерабатываемого сырья, но также с учетом изменений линейных размеров деталей, получаемых в процессе технологической обработки и последующей эксплуатации готового изделия.

Преимущества использования системы автоматизированного проектирования при конструировании и разработке технологии вязания изделия заключаются в возможности обработки в малые промежутки времени огромного массива данных. Что в свою очередь позволяет выполнять конструкторскую и технологическую подготовку с более точными параметрами и корректировками в сравнении с ручной формой проектирования.

Автоматизация конструкторской подготовки сложного цельновязаного изделия заключается в формировании пакета лекал с указанием на них узлов соединения деталей, а также других конструктивных элементов (вытачек, горловин, карманов и т.д.) и последующей передаче этого пакета лекал в виде массива данных в систему проектирования технологии вязания.

В системе проектирования технологии вязания, при формировании контура развертки, простейшей единицей измерения является петля, где взаимное расположение множества петель в одной плоскости определяет контур вывязываемой детали и дальнейшую технологию вязания. Исходя из этого, массив данных, передаваемый в систему проектирования технологии вязания, должен содержать в себе информацию о количестве петельных столбиков, рядов, исходных параметров трикотажа, а также информацию по их взаимному расположению.

Поэтому взаимодействие системы конструирования и системы, предназначенной для разработки программы вязания, должно выполняться через общий (промежуточный) тип данных.

Использование общего типа данных позволит:

- оптимизировать взаимодействие конструктора с дессинатором, тем самым повысив эффективность их работы;
- использовать для разработки конструкции цельновязаного изделия методики конструирования для швейных изделий;
- автоматизировать рутинные ручные операции, выполняемые конструктором (корректировка лекал после аппроксимации, градация и т.д.) и дессинатором (проектирование сбавок (прибавок) с учетом различных технологических ограничений, проектирование узла соединения и т.д.);
- сократить время, необходимое на проектирование изделия;
- сократить объем, затрачиваемого на проектирование сырья;
- улучшить оперативность смены ассортимента;
- расширить ассортимент.

Использование двух различных систем проектирования подразумевает переход на комплексное проектирование изделия, где взаимодействие системы конструирования и системы разработки программы вязания выполняется с использованием промежуточного технического решения. Причем данные, которыми оперируют при конструировании и разработке технологии вязания должны быть преобразованы в универсальный тип данных.

За счет использования универсального типа данных станет возможным выполнение автоматических корректировок лекал с учетом технологических особенностей вязания (динамическое отслеживание изменения связей и форм графических объектов и на их основе перестроения контура развертки), оптимизации технологии вязания согласно задаваемым требованиям, оптимизация технологии узлов соединения и т.д.

Таким образом, комплексную систему проектирования, используемую на трикотажном производстве, для разработки сложного цельновязаного изделия можно разделить на несколько подсистем:

- подсистема 1. Этой подсистемой может являться полноценная система конструирования, используемая на швейном производстве, векторный графический редактор или любое другое программное решение, где графический объект имеет аналитическую форму записи;
- подсистема 2. Суть работы этой подсистемы заключается в преобразовании, сформированного в подсистеме 1 пакета лекал, в массив данных, необходимых для разработки программы вязания;
- подсистема 3. На месте подсистемы 3 выступает система автоматизированного проектирования, используемая для разработки технологии вязания на трикотажном производстве. Результатом работы подсистемы 3 будет являться программа вязания.

Комплексное использование подсистем 1, 2, 3 позволит автоматизировать этап проектирования сложного цельновязаного изделия, упростить этапы конструкторской и технологической подготовки изделия, расширить и улучшить ассортиментный ряд, выпускаемой предприятием продукции, а также увеличить эффективность эксплуатации вязального оборудования.

Таким образом, ***комплексная система проектирования*** – это совокупность взаимодополняющих программных и технических решений, используемых для частичной или полной автоматизации этапа проектирования. Схема работы комплексной системы проектирования представлена на рисунке 2.1.

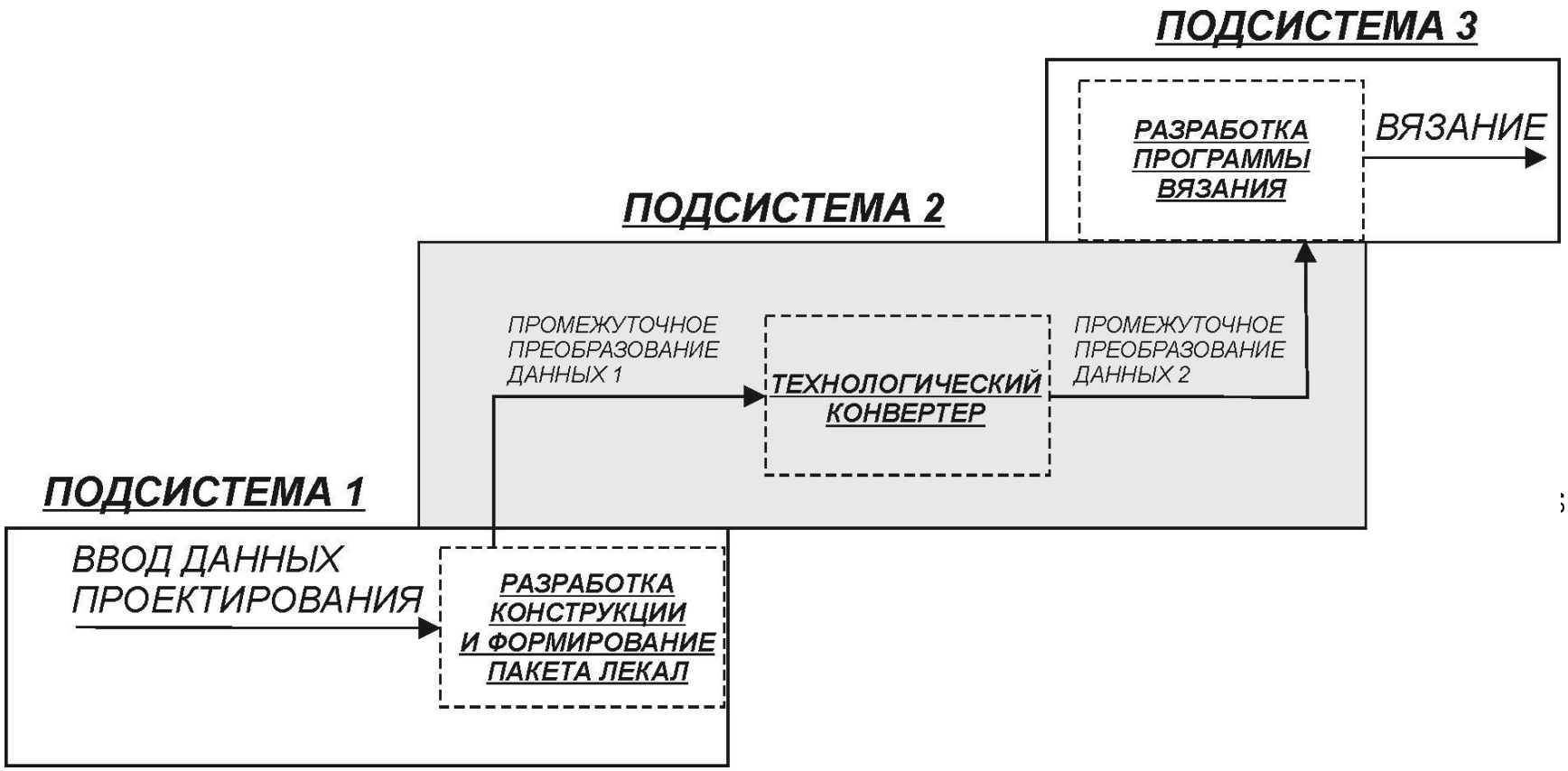


Рисунок 2.1 – Схема комплексной системы проектирования

Согласно рисунку 2.1 на начальном этапе обработки данных в подсистеме 1 выполняется ввод основных и дополнительных параметров проектирования. Причем основными параметрами проектирования на данном этапе являются параметры, используемые в конструировании (обхваты, ширины, припуски, прибавки и т.д.).

В подсистеме 2 данные, полученные из подсистемы 1, преобразуются в универсальный тип данных, с которыми также может работать и подсистема 3.

Суть универсального типа данных заключается в объединении подсистемы 1 и подсистемы 3 в одну комплексную систему проектирования, где подсистема 2 должна являться инструментом, при помощи которого подсистема 1 и подсистема 3 смогут взаимодействовать между собой.

Сформированный пакет лекал, передаваемый в подсистему 2 из подсистемы 1 должен удовлетворять следующие требования:

- развертки должны иметь аналитическую форму записи;
- на развертках должны быть дополнительно указаны кромки соединения;
- карманы, воротники, горловины, рельеф должны быть аналитически прикреплены к необходимым разверткам.

Элементарным объектом в подсистеме 1 будет являться точка. Множества точек составляют графические объекты, используемые непосредственно в конструкции проектируемого изделия. Совокупностью графических объектов, формирующих развертку детали проектируемого изделия, будет являться более сложный графический объект (д-объект). Отличие д-объекта от развертки детали проектируемого изделия заключается в том, что в состав развертки входят непосредственно графические объекты, а в состав д-объекта только их аналитические записи.

Таким образом, ***д-объектом*** является массив аналитических форм записей графических объектов, входящих в структуру развертки детали проектируемого изделия.

Следовательно, результатом работы подсистемы 1 будет являться *массив д-объектов*, полученных в результате формирования пакета лекал. Схема работы подсистемы 1 представлена на рисунке 2.2.

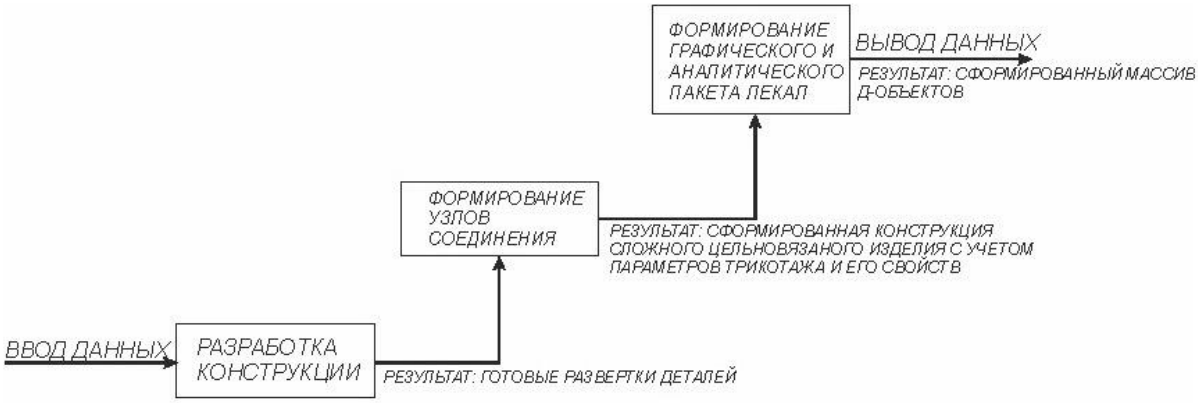


Рисунок 2.2 – Схема работы подсистемы 1

Далее в подсистеме 2 выполняется преобразование массива д-объектов в массив данных, содержащего в себе результаты проектирования сбавок (прибавок). На этапе обработки данных в подсистеме 2 выполняется проектирование сбавок (прибавок), формирование узлов соединения для цельновязаного изделия и т.д. Схема работы подсистемы 2 представлена на рисунке 2.3.

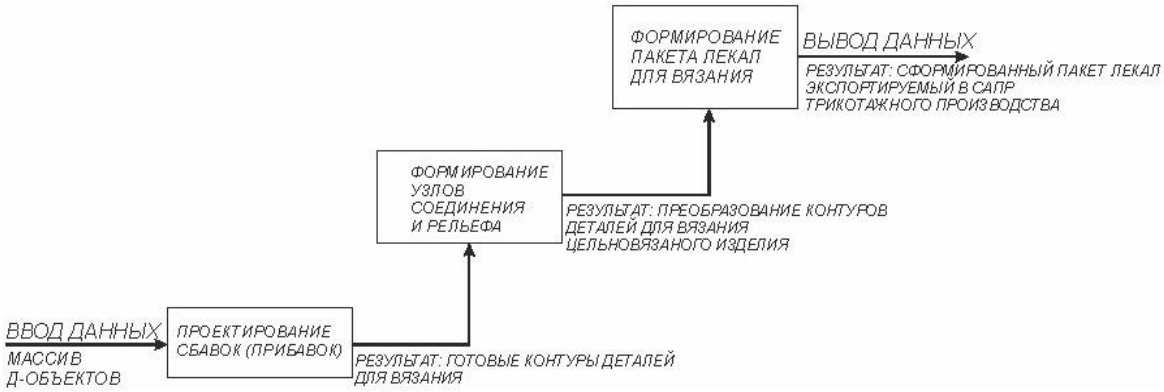


Рисунок 2.3 – Схема работы подсистемы 2

При импорте результатов работы подсистемы 2 в САПР, являющейся средой разработки программы вязания, необходимо учитывать, что развертки следует представлять в виде двумерного массива. Причем одна размерность массива должна представлять собой петельные ряды, а другая, в соответствии с петельным рядом – количество сбавок (прибавок) в нем.

Использование комплексной системы проектирования в разработке сложного цельновязаного изделия позволит сократить время, затрачиваемое на проектирование, улучшить оперативность смены ассортимента, а также улучшить эффективность взаимодействия между конструктором и программистом.

Взаимодействие подсистем 1, 2, 3 между собой имеет взаимодополняющий характер. Причем взаимодействие подсистем может осуществляться только следующим образом:

- подсистема 1 может взаимодействовать только с подсистемой 2;
- подсистема 2 может взаимодействовать только с подсистемой 3.

Такое взаимодействие подсистем между собой подразумевает использование общих промежуточных типов данных. Например, при взаимодействии подсистемы 1 с подсистемой 2 их общим типом данных является массив д-объектов, который является результатом работы подсистемы 1 и в то же время входными данными для проектирования подсистемы 2.

Последовательное использование подсистем 1, 2, 3 в проектировании сложного цельновязаного изделия позволит осуществлять разработку конструкции и технологии вязания. Это возможно за счет изменения параметров элементарных объектов, используемых в каждой подсистеме, где в подсистеме 1 элементарным объектом является точка, в подсистеме 2 – трикотажная петля. Также это позволит выполнять более точное прогнозирование свойств изделия (усадка, присадка и т.д.), изменения его конструктивных характеристик при эксплуатации и т.д.

2.2 Разработка основ конструкторской подготовки сложных цельновязаных изделий

В настоящее время конструкторская подготовка цельновязаного изделия является простой операцией [3.5]. Упрощение конструирования при проектировании сложного цельновязаного изделия достигается за счет использования встроенных модулей конструирования совместно с технологическими модулями в САПР трикотажного производства. Количество входных данных, необходимых для конструирования сложного цельновязаного изделия, минимально, например, при конструировании шапки необходимо задать только три параметра: ширина шапки, высота шапки и высота участка заработка шапки. Поэтому этими данными будут являться значения линейных размеров деталей или участков цельновязаного изделия, находящегося в одной плоскости, например, при проектировании готового плечевого цельновязаного изделия (рисунок 2.4).

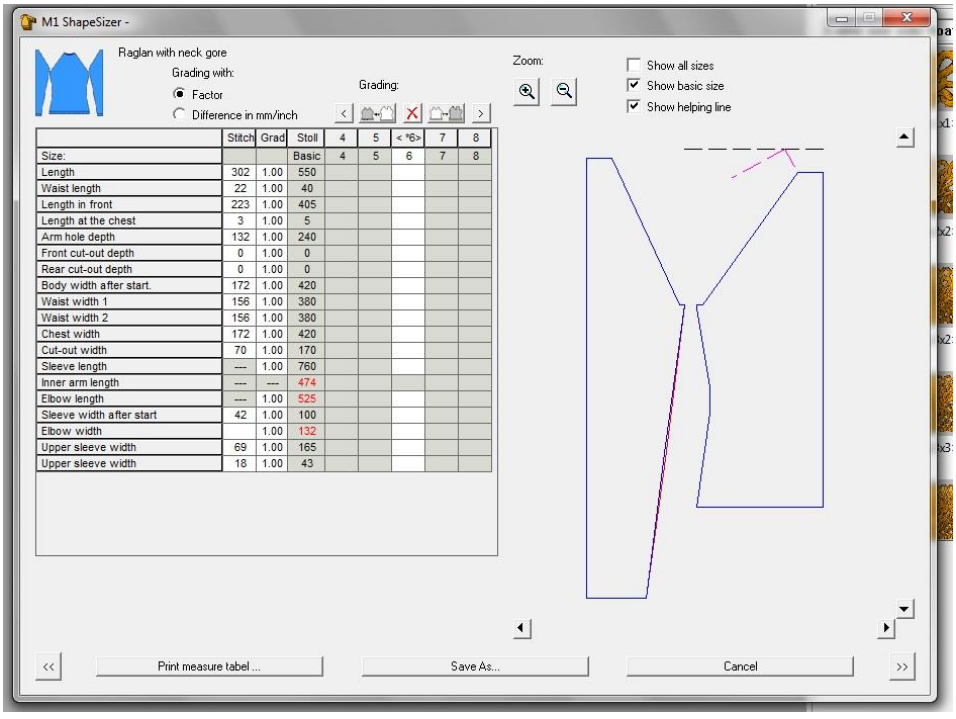


Рисунок 2.4 – Конструирование сложного цельновязаного изделия в STOLL M1

Однако использование средств конструирования, встроенных в систему проектирования трикотажного производства, не позволяет:

- осуществлять конструкторскую подготовку сложного цельновязаного изделия нетрадиционной формы, например с ассиметричным силуэтом, с использованием технологических средств автоматизации, а также выполнять проектирование рельефных участков заданной формы. Например, к изделиям нетрадиционной формы можно отнести изделия с ассиметричным силуэтом, сувенирные изделия и т.д.;
- осуществлять разработку конструкции сложного цельновязаного изделия, используя известные методы конструирования;
- проектировать сложные узлы соединения деталей, где соединяющиеся кромки деталей состоят из нескольких кривых второго или третьего порядка;
- выполнять конструкторскую подготовку сложного цельновязаного изделия с учетом прогнозируемых линейных изменений формы цельновязаного изделия, его деталей или локальных участков в процессе эксплуатации;
- выполнять динамическое отслеживание связей конструктивных элементов в развертках деталей сложного цельновязаного изделия на этапе проектирования технологии вязания.

Решение проблем, связанных с конструкторской подготовкой сложных цельновязаных изделий, позволит:

- проектировать развертки сложных цельновязаных изделий с использованием различных методов конструирования;
- улучшить посадку готового изделия за счет динамического отслеживания связей конструктивных элементов в развертках деталей сложного цельновязаного изделия;

- улучшить посадку готового изделия за счет проектирования технологически сложных узлов соединения;
- улучшить посадку сложного цельновязаного изделия за счет формирования на нем рельефа (выпуклых участков) в процессе вязания, где форма и размеры рельефа (выпуклого участка) спроектированы по какому-либо методу конструирования швейного изделия;
- расширить ассортимент, выпускаемых изделий, а также сократить время разработки за счет проектирования сложных конструкций и технологий цельновязаных изделий при помощи автоматизированных инструментов проектирования.

Решение данных проблем возможно за счет:

- использования в конструировании графических объектов, имеющих простую аналитическую форму записи;
- использования параметрической формы расчетно-графических методов конструирования;
- аппроксимации разверток деталей сложного цельновязаного изделия в зависимости от высоты петельного ряда и ширины петельного столбика;
- аппроксимации кромок в узле соединения с разделением на участки, где участки могут иметь различные параметры аппроксимации;
- применения в формировании выпуклых участков трикотажа технологически рациональных способов вязания, а также замене на них конструктивных элементов формирования рельефа развертки, например, геометрическая замена закрытой швейной вытачки провязыванием дополнительных рядов.

С целью оптимизации этапа конструкторской подготовки сложного цельновязаного изделия выберем графические объекты, используемые в конструировании.

2.3 Выбор и обоснование графических объектов, используемых для разработки конструкции сложного цельновязаного изделия

Часто используемые при конструировании графические объекты можно разделить на три типа: отрезок, ограниченная квадратная парабола и ограниченная кубическая парабола. Например, ограниченная квадратная парабола используется в конструкции плечевого изделия для оформления проймы или горловины, а ограниченная кубическая парабола – для оформления оката половины рукава. Однако некоторые методики конструирования в проектировании нижней части проймы стана подразумевают использование метода радиусографии [1.3].

Применение метода радиусографии позволяет оформлять нижнюю часть проймы стана в виде сегмента окружности. В некоторых случаях метод радиусографии позволяет формировать хорошую посадку изделия на теле человека, но в то же время применение данного метода построения нижней части проймы стана может привести к появлению лишних складок трикотажа в районе узла соединения, что может ухудшить внешний вид изделия.

Таким образом, графические объекты, используемые при конструировании сложного цельновязаного изделия, должны удовлетворять следующим требованиям:

- должны иметь простую форму аналитической записи;
- должны иметь ограничения по длине, интегрированные в аналитическую форму записи;
- должны иметь минимально возможное количество вариантов построения в момент интерполирования с учетом нескольких ограничений;

- должны удовлетворять формами графического построения часто используемым в конструировании графическим объектам;
- должны иметь возможность рекурсивного построения без значительных изменений в аналитической форме записи.

В настоящее время кривой, имеющей простую аналитическую форму записи с интегрированными ограничениями длины и выпуклости, а также полностью удовлетворяющей выше заявленным требованиям является кривая Безье [3.3, 1.10].

Как известно, кривая Безье – это гладкая кривая с параметрической формой записи, задаваемая выражением:

$$B(t) = \sum_{i=0}^{n} P_i * b_{i,n}(t), \qquad (2.1)$$

где $t \in [0; 1]$;

P_i – опорная вершина кривой Безье.

$$b_{i,n} = \binom{n}{i} * t^i * (1-t)^{n-i}, \qquad (2.2)$$

$$\binom{n}{i} = \frac{n!}{i! * (n-i)!} \qquad (2.3)$$

Подставив формулу (2.3) в формулу (2.2), получим:

$$b_{i,n} = \frac{n!}{i! * (n-i)!} * t^i * (1-t)^{n-i}, \qquad (2.4)$$

где $b_{i,n(t)}$ – базисная функция кривой Безье (полином Бернштейна);

n – степень кривой Безье;

i – порядковый номер опорной вершины кривой Безье.

Кривая Безье графически представляет собой полином, ограниченный выпуклой ломанной, причем данная кривая, всегда будет проходить через начальную и конечную опорные вершины P_0 и P_2 и никогда не выйдет за границы промежуточных опорных вершин P_1 (рисунок 2.6).

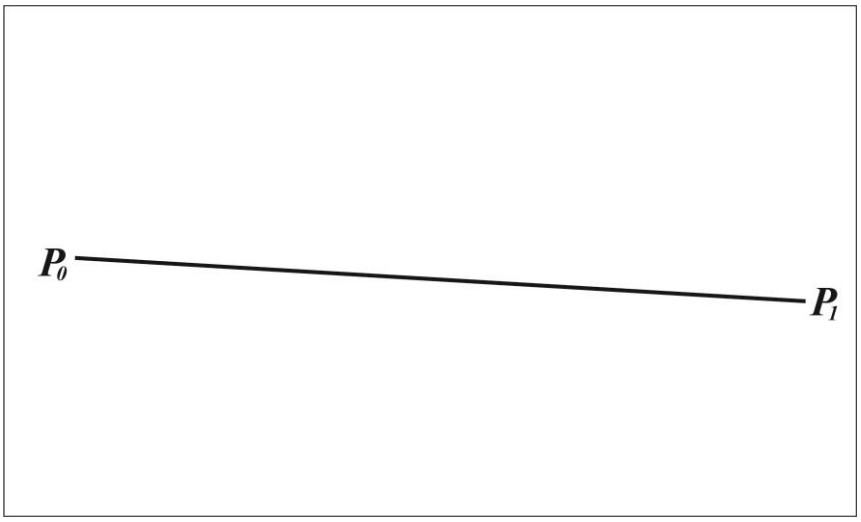


Рисунок 2.5 – Кривая Безье 1-ой степени

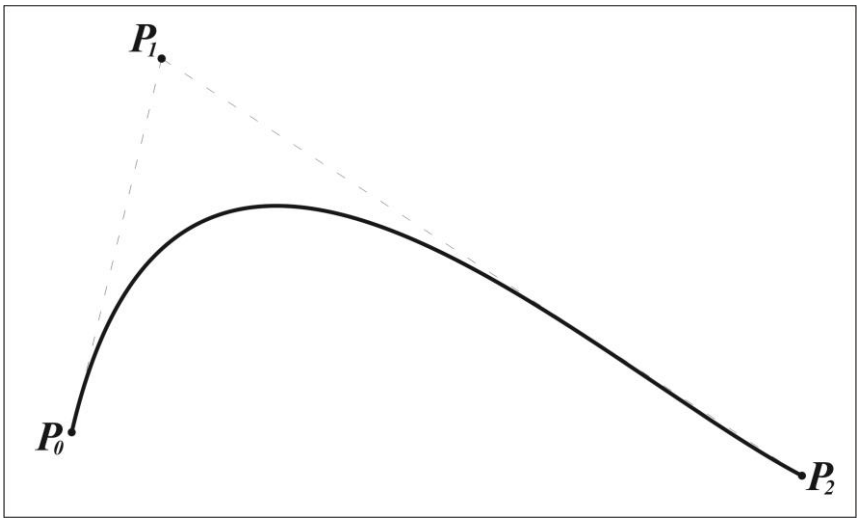


Рисунок 2.6 – Кривая Безье 2-ой степени

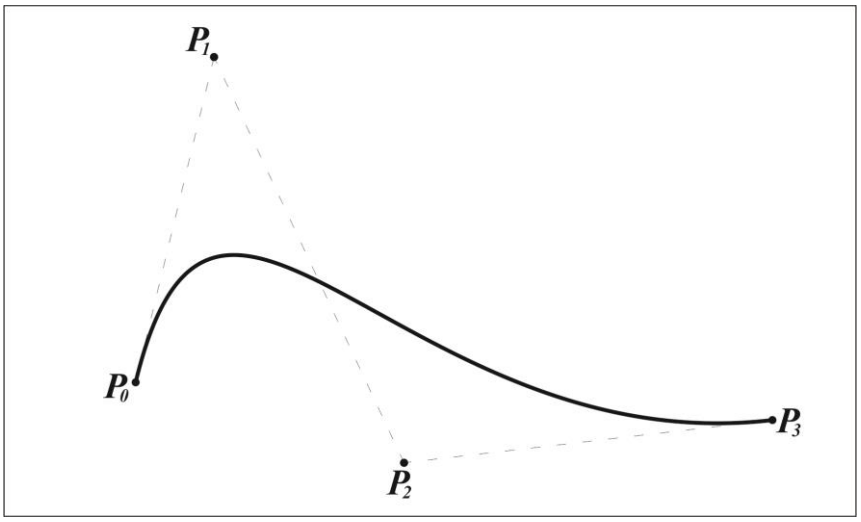


Рисунок 2.7 – Кривая Безье 3-ей степени

Представленные на рисунках 2.5, 2.6, 2.7 кривые Безье будут задаваться следующими выражениями:

кривая Безье 1-ой степени

$$B(t) = \sum_{i=0}^{1} P_i * b_{i,n}(t) = P_0 * b_{0,1}(t) + P_1 * b_{1,1}(t), \quad (2.5)$$

$$b_{0,1} = \frac{1!}{0! * (1-0)!} * t^0 * (1-t)^{1-0} = 1 - t\,, \quad (2.6)$$

$$b_{1,1} = \frac{1!}{1! * (1-1)!} * t^1 * (1-t)^{1-1} = t \quad (2.7)$$

Подставив формулы (2.6) и (2.7) в формулу (2.5), получим:

$$B(t) = (1-t) * P_0 + t * P_1 \quad (2.8)$$

кривая Безье 2-ой степени

$$B(t) = (1-t)^2 * P_0 + 2 * t * (1-t) * P_1 + t^2 * P_2 \quad (2.9)$$

кривая Безье 3-ей степени

$$B(t) = (1-t)^3 * P_0 + 3 * t * (1-t)^2 * P_1 + 3 * t^2 * (1-t) * P_2 + t^3 * * P_3 \quad (2.10)$$

Так как кривая Безье имеет параметрическую форму записи, то параметр t, входящий в структуру аналитической записи, будет являться фактором, ограничивающим длину кривой, выпуклость кривой Безье ограничивают

промежуточные опорные вершины. Кроме того параметрическая форма записи позволяет выполнять рекурсивное построение кривой без каких-либо изменений в аналитической форме записи.

Также кривой Безье можно с большой точностью интерполировать другие графические объекты. Например, кривая Безье может иметь форму кривой, оформляющей пройму стана, построенной методом радиусографии (рисунок 2.8). Кроме того использование интерполирования в конструкторской подготовке трикотажного изделия позволит осуществить ввод данных, необходимых для проектирования развертки, при помощи различных технических средств сканирования, например, при помощи трехмерного сканера, дигитайзера и т.п., или специальных программных решений, например, при помощи программы распознавания образов.

На рисунке 2.8 показано, что для интерполирования кривой проймы стана, построенной методом радиусографии, слева на рисунке 2.8, использовалась кривая Безье третьей степени, справа на рисунке 2.8.

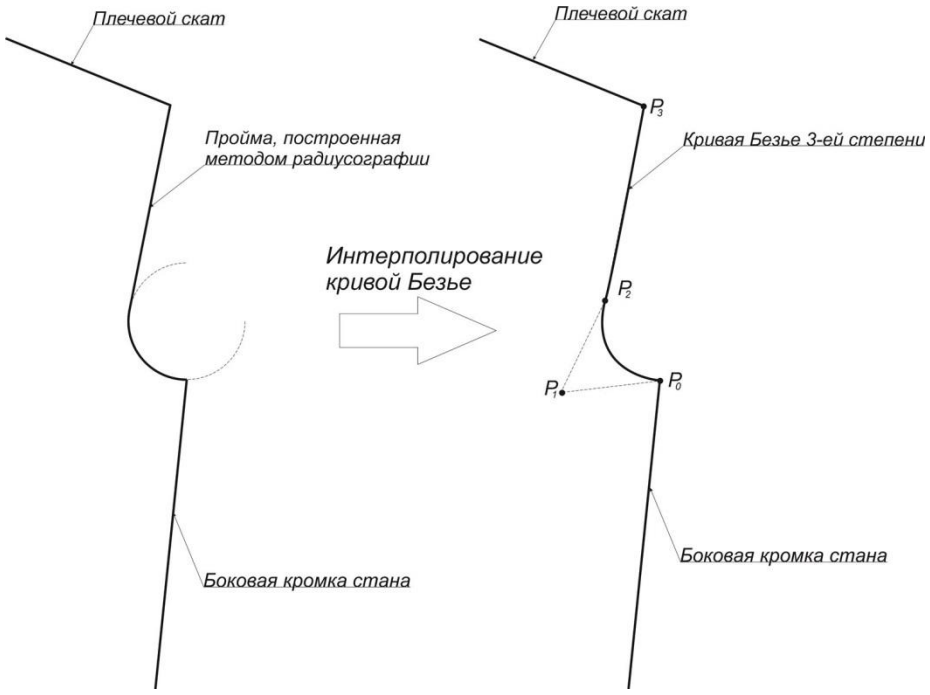


Рисунок 2.8 – Сравнение пройм, построенных разными способами

Также сравним, полученные кривые при помощи графического наложения (рисунок 2.9).

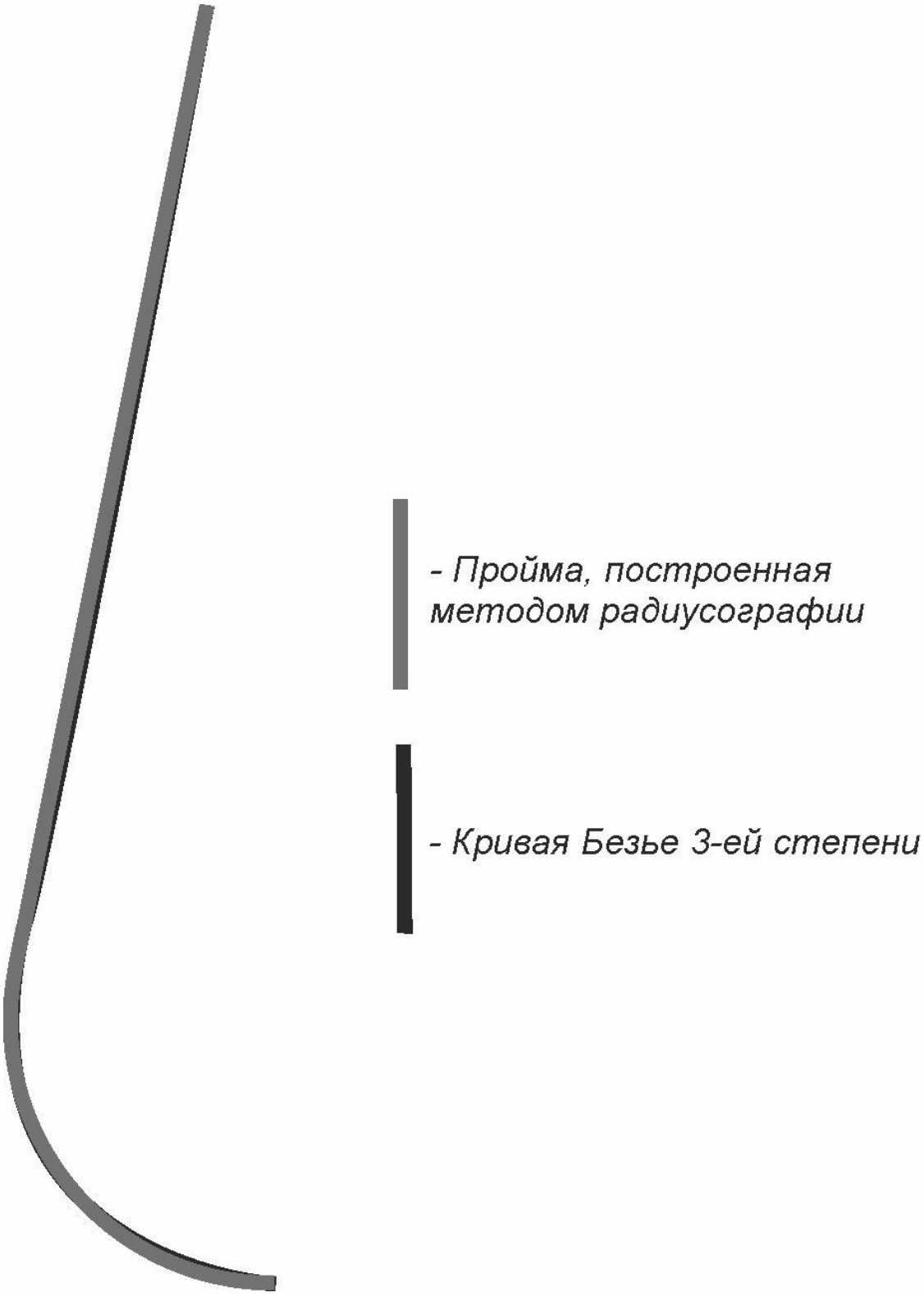


Рисунок 2.9 – Графической наложение кривых

Как видим пройма, построенная методом радиусографии, совпала с кривой Безье третьей степени.

Таким образом, применение кривых Безье в конструировании сложных цельновязаных изделий позволит:

- выполнить конструкторскую подготовку сложных цельновязаных изделий с применением графических объектов, имеющих простую форму аналитической записи;
- сформировать пакет лекал, необходимых для печати на плоттере, используемых на этапе влажно-тепловой обработки;
- сформировать массив д-объектов, необходимый для дальнейшего расчета сбавок (прибавок);
- интерполировать любой графический объект, используемый в конструировании текстильных изделий, в том числе и трикотажных, а также осуществлять построение разверток деталей по готовому изделию;
- выполнить расчет необходимых для вязания сбавок (прибавок) конструктивно и технологически сложного узла соединения.

2.4 Разработка технологии проектирования сбавок (прибавок) по профилю графического объекта

В настоящее время системы проектирования, используемые на трикотажном производстве, выполняют расчет сбавок (прибавок) после процесса аппроксимации.

Аппроксимация – это метод, позволяющий выполнить замену сложного объекта множеством простых объектов [1.10].

Аппроксимация профилей кривых второго и третьего порядка выполняется отрезками прямых линий. После процесса аппроксимации рассчитывается ширина и высота ступени, а также необходимое количество сбавок (прибавок) для вывязывания этой ступени (рисунок 2.10).

Такой способ расчета сбавок не эффективен для проектирования сложного цельновязаного изделия, так как:

- данный способ не позволяет рассматривать каждую ступень, полученную после аппроксимации, как отдельный участок, что необходимо для оптимизации кромки соединения;
- данный способ не позволяет указывать дополнительные ограничения аппроксимации, необходимые для различных участков кривой, заданной высоты;
- данный способ не дает возможности для определения положения линии в узле соединения, после которой соединяемые детали будут иметь разное количество петельных рядов;
- данный способ не позволяет выполнять динамическое отслеживание изменений формы кромки в процессе аппроксимации, например, при вводе дополнительных параметров аппроксимации аппроксимированная кромка изменится вместе с аппроксимируемой кривой, что не позволит определить степень её отклонения от начального состояния;
- данный способ не позволяет выполнять в автоматическом режиме аппроксимацию кривой прямыми отрезками различной длины (длина прямых отрезков изменяется для всей кривой).

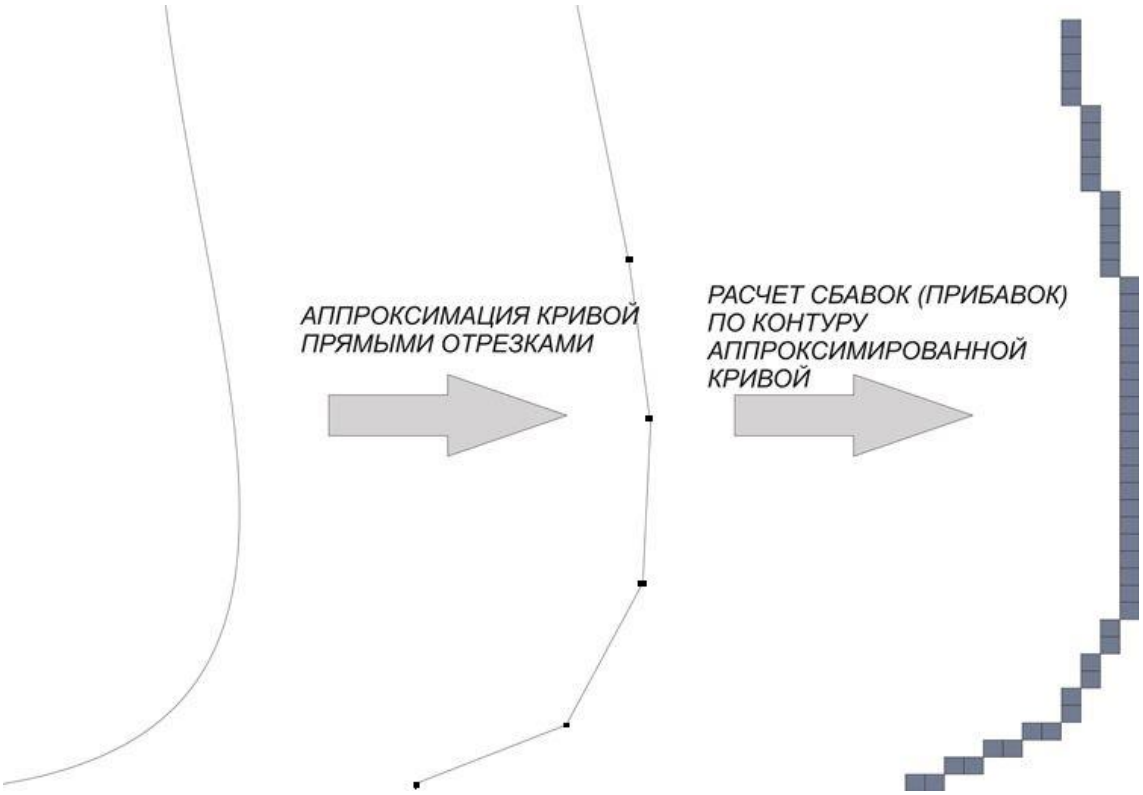


Рисунок 2.10 – Расчет сбавок (прибавок) по контуру кривой

Наиболее эффективным способом проектирования сбавок (прибавок) по профилю графического объекта будет являться аппроксимация, направленная вдоль петельного столбика, с шагом равным высоте петельного ряда.

Суть этого способа заключается в нахождении на каждом шаге аппроксимации крайней верхней внешней точки ячейки (где одна ячейка представляет собой одну петлю), являющейся наиболее близкой, по своим координатам, к точке, принадлежащей кривой. Причем высота и ширина ячейки будут соответственно равны высоте петельного ряда и ширине петельного столбика (рисунок 2.11).

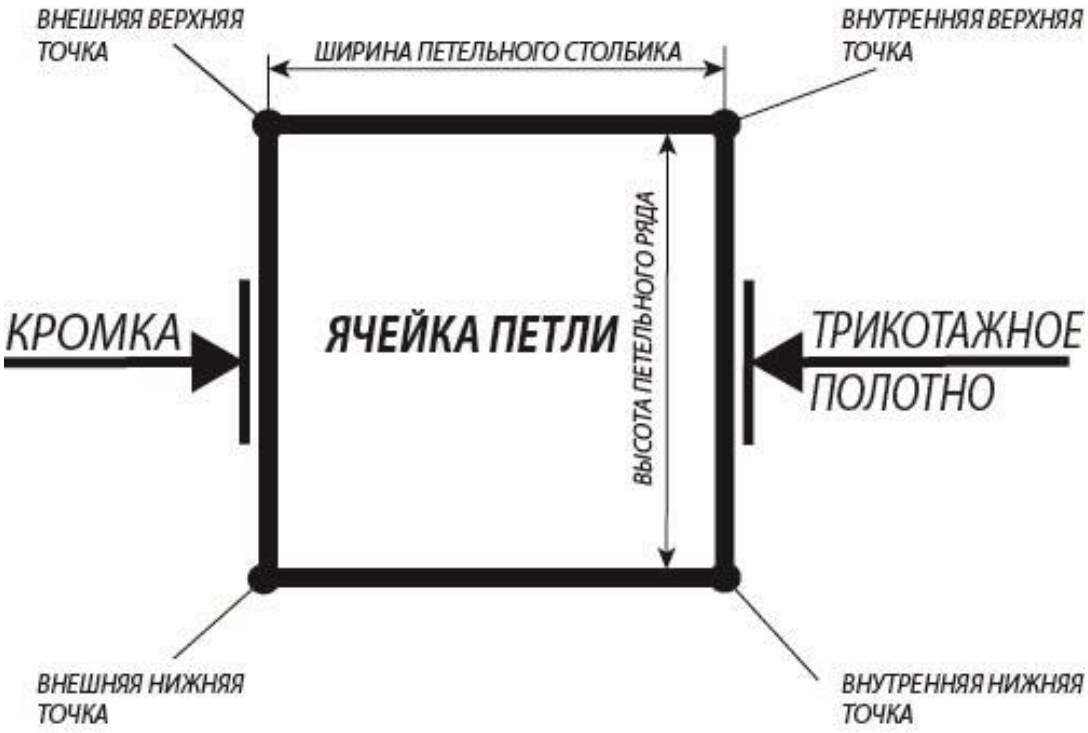


Рисунок 2.11 – Ячейка петли левой кромки купона

Согласно рисунку 2.11, изображенная на нем ячейка представляет собой крайнюю петлю левой кромки детали купона. Ячейка петли правой кромки с указанными параметрами представлена на рисунке 2.12.

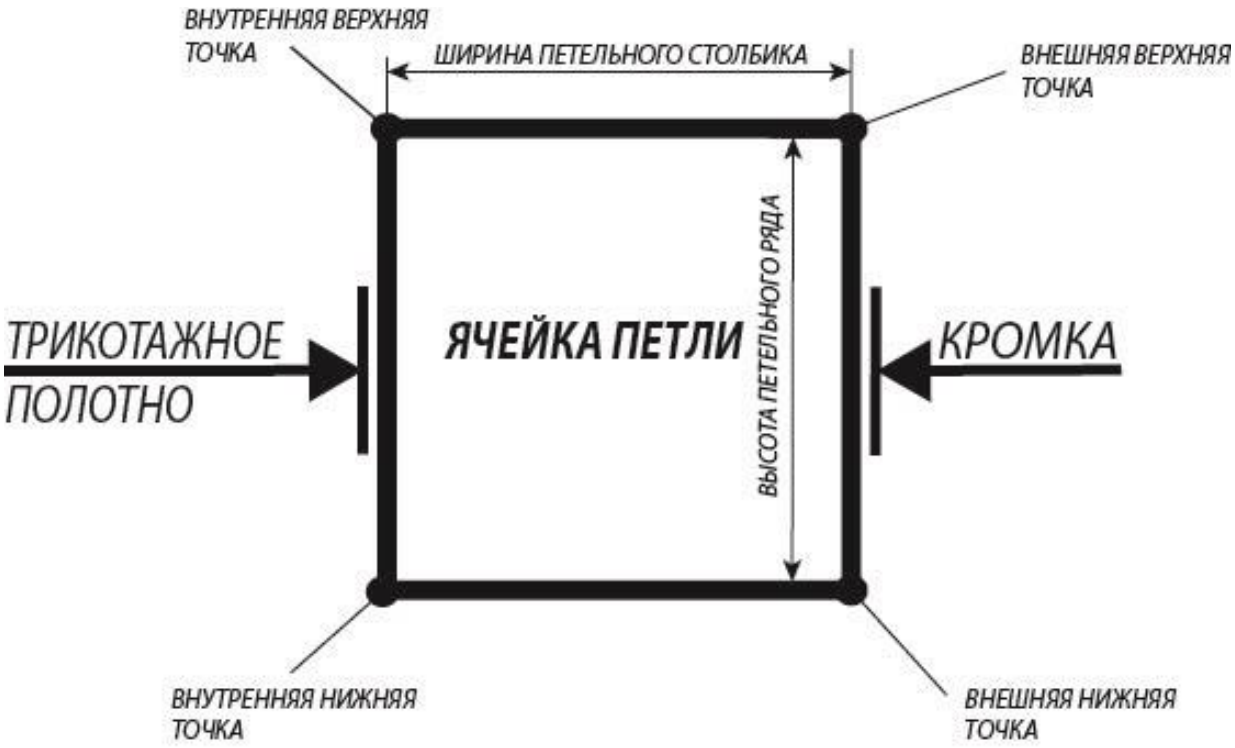


Рисунок 2.12 – Ячейка петли правой кромки купона

Приведение петли к упрощенной форме в виде ячейки позволит:

- выполнять точную аппроксимацию графического объекта;
- выполнять расчет сбавок (прибавок) по профилю графического объекта, исключив этап его аппроксимации прямыми отрезками;
- разделять графический объект на любое количество участков, задаваемой высоты, где высота участка может быть задана количеством петельных рядов;
- устанавливать для каждого участка дополнительные ограничения (максимальная сбавка в ряду или сбавка через ряд и т.д.) при аппроксимации в автоматическом режиме;
- перейти к низкоуровневому проектированию узла соединения в сложном цельновязаном изделии.

Поиск положения ячейки относительно кривой представлен на рисунке 2.13.

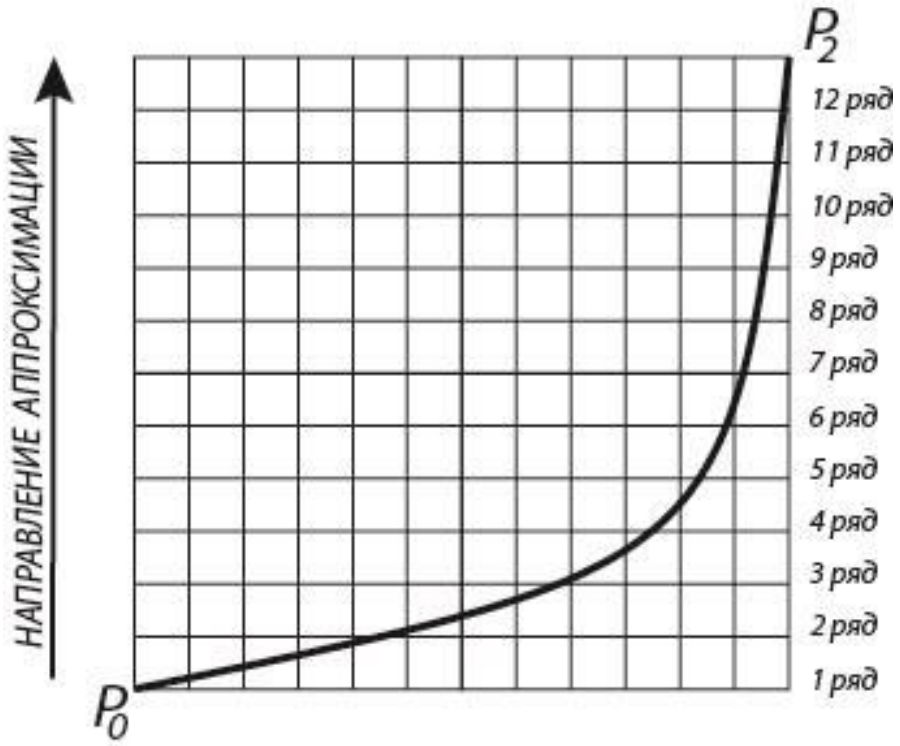


Рисунок 2.13 – Аппроксимация ячейками кривой Безье второй степени

Согласно рисунку 2.13 аппроксимация кривой Безье второй степени начинается из опорной вершины P_0 и выполняется по вертикальной направляющей, то есть вдоль петельного столбика, с шагом равным высоте петельного ряда.

Поиск оптимальных координат расположения ячейки относительно кривой Безье, выполняется по верхней внешней точки ячейки. Алгоритм поиска имеет следующий вид:

ПОЛУЧЕНИЕ КООРДИНАТ ТОЧКИ СТАРТА АППРОКСИМАЦИИ В ПЕТЕЛЬНОМ РЯДУ (ШАГ 1) >>> ШАГ АППРОКСИМАЦИИ (ШАГ 2) >>> ПОЛУЧЕНИЕ КООРДИНАТ НАЧАЛА РАСЧЕТА СБАВОК (ПРИБАВОК) В СЛЕДУЮЩЕМ ПЕТЕЛЬНОМ РЯДУ (ШАГ 3) >>> ПОЛУЧЕНИЕ КООРДИНАТ ОКОНЧАНИЯ РАСЧЕТА СБАВОК (ПРИБАВОК) (ШАГ 4) >>> ПОЛУЧЕНИЕ КООРДИНАТ ТОЧКИ ФИНИША АППРОКСИМАЦИИ В ПЕТЕЛЬНОМ РЯДУ (ШАГ 5)

Графически данный алгоритм на примере аппроксимации первых двух петельных рядов кривой Безье из рисунка 2.13 представлен на рисунке 2.14.

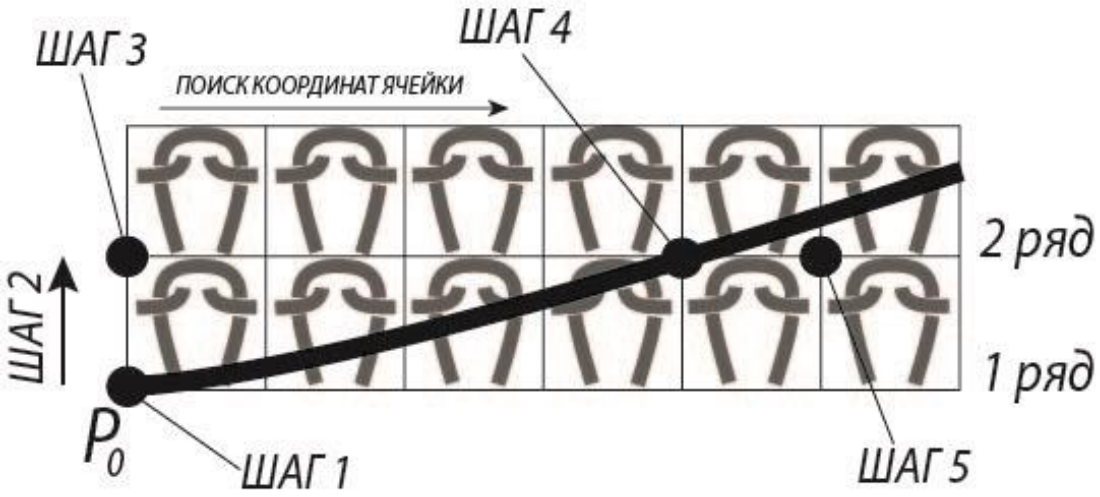


Рисунок 2.14 – Расчет сбавок в первом петельном ряду

Согласно рисунку 2.14 на шаге 1 координаты точки старта аппроксимации в первом петельном ряду будут равны начальной опорной вершине кривой Безье (P_0).

Примем, что кривая Безье, изображенная на рисунке 2.13, является левой кромкой купона, тогда ячейка будет соответствовать рисунку 2.11. Если кривая Безье будет являться правой кромкой купона, то ячейка будет соответствовать рисунку 2.12. Тогда координатам точки старта аппроксимации будут соответствовать координаты внешней нижней точки ячейки. Это связано с тем, что в первом петельном ряду нецелесообразно выполнять сбавку (прибавку). В противном случае сбавка (прибавка) в первом петельном ряду может привести к локальному изменению ширины купона, а также значительному отклонению аппроксимированной кромки от спроектированной кромки на развертке.

На шаге 2 выполняется изменение ординаты точки старта аппроксимации на величину равной высоте петельного ряда. Измененные координаты будут

являться координатами точки начала расчета сбавок (прибавок) во втором петельном ряду.

При переходе с шага 3 на шаг 4 выполняется поиск оптимальных координат внешней верхней точки ячейки, а также осуществляется расчет необходимого количества сбавок, выполняемых во втором петельном ряду. Начальные координаты внешней верхней точки ячейки будут равны координатам точки начала расчета сбавок во втором петельном ряду.

На шаге 5 осуществляется получение координат точки финиша аппроксимации первого петельного ряда. Абсцисса точки финиша аппроксимации должна отличаться от абсциссы точки окончания расчета сбавок минимум на одну ширину петельного столбика. Это условие необходимо для исключения возможности возникновения ошибки при расчете сбавок (прибавок).

Полученные координаты на шаге 5 будут являться координатами точки старта аппроксимации для следующего петельного ряда (согласно рисунку 2.14 – это второй петельный ряд).

Используя графический метод, аппроксимируем ячейками кривую Безье, изображенную на рисунке 2.10. Сравним результаты аппроксимации (рисунки 2.15, 2.16).

Из рисунков 2.15 и 2.16 видно, что более близкой к заданной кривой является кривая, полученная в результате аппроксимации ячейками.

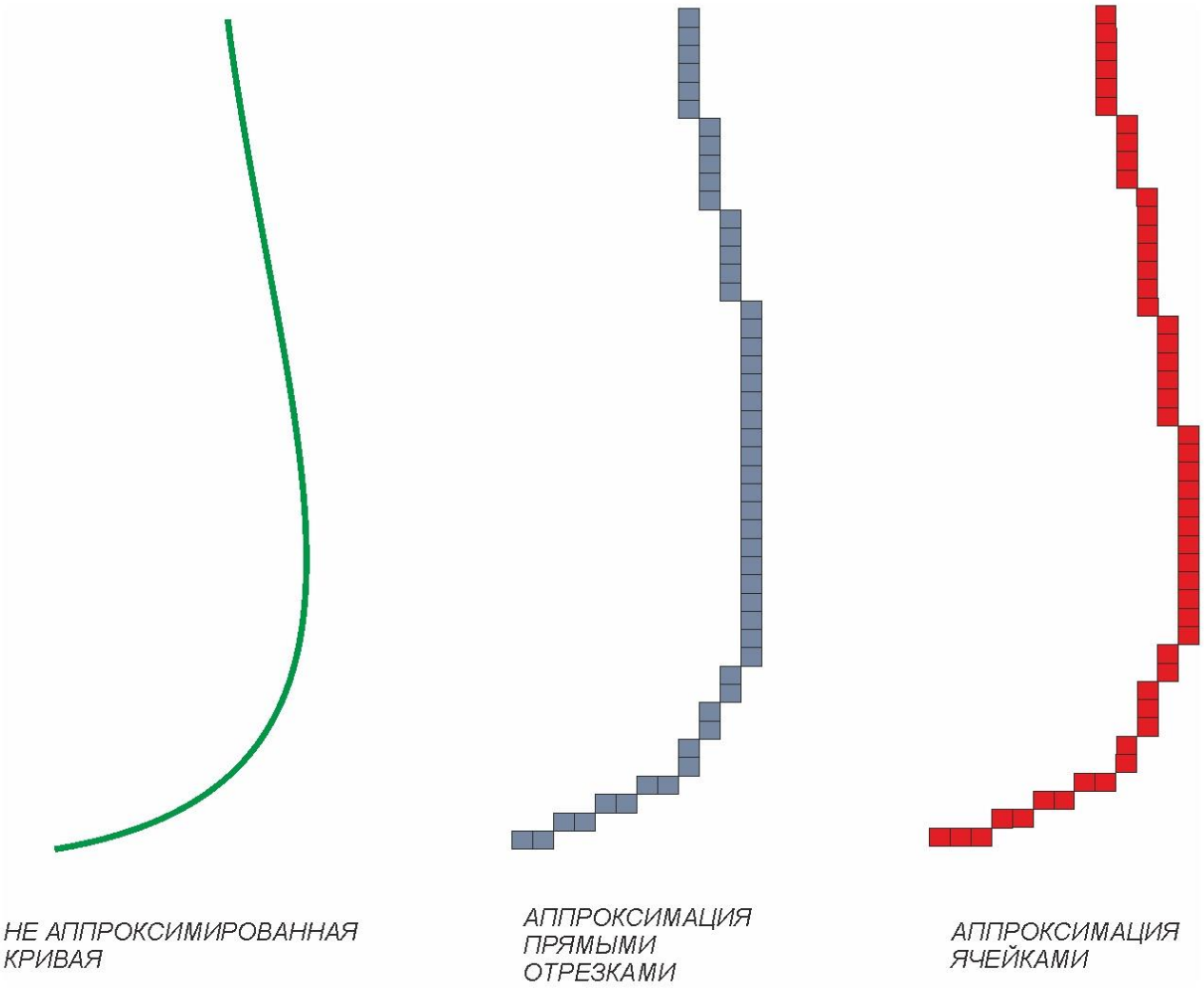


Рисунок 2.15 – Сравнение методов аппроксимации

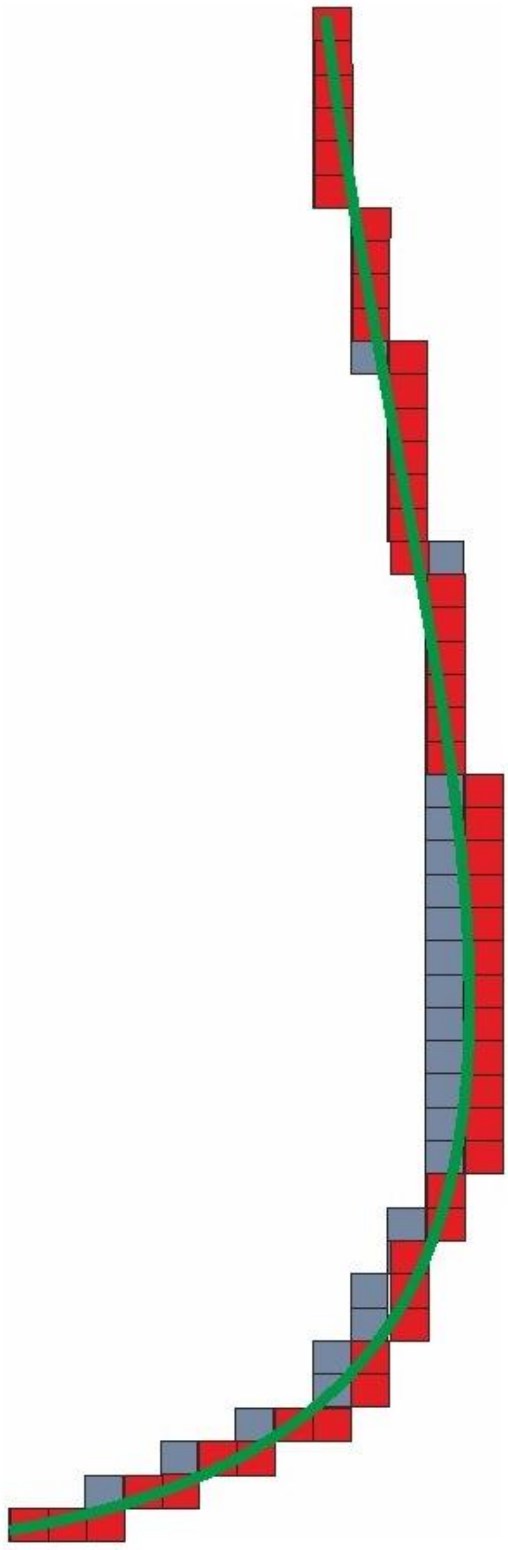


Рисунок 2.16 – Сравнение аппроксимированных кривых из рисунка 2.15 посредством графического наложения

Математически алгоритм аппроксимации кривой Безье будет иметь следующий вид:

Пусть St_{1i} (x_{St1i}, y_{St1i}) – точка старта аппроксимации i-го петельного ряда;

St_{2i} (x_{St2i}, y_{St2i}) – точка начала расчета сбавок (прибавок) i-го петельного ряда;

St_{3i} (x_{St3i}, y_{St3i}) – точка окончания расчета сбавок (прибавок) i-го петельного ряда;

St_{fi} (x_{Stfi}, y_{Stfi}) – точка финиша аппроксимации i-го петельного ряда;

B_{zi} (x_{Bzi}, y_{Bzi}) – точка кривой Безье с ординатой i-го петельного ряда;

A – ширина петельного столбика;

B – высота петельного ряда.

Для первого петельного ряда

ШАГ 1

$$x_{St11} = x_{P0}, \quad (2.1)$$

$$y_{St11} = y_{P0}, \quad (2.2)$$

где (x_{P0}, y_{P0}) – координаты начальной опорной вершины кривой Безье.

ШАГ 2

Увеличиваем значение ординаты точки старта аппроксимации на высоту петельного ряда, то есть $y_{St11} + B$, тогда

ШАГ 3

$$y_{St21} = y_{St11} + B, \quad (2.3)$$

$$x_{St21} = x_{St11} \quad (2.4)$$

ПОИСК ОПТИМАЛЬНЫХ КООРДИНАТ ВНЕШНЕЙ ВЕРХНЕЙ ТОЧКИ ЯЧЕЙКИ

Пусть шаг поиска равен величине ширины петельного столбика, тогда

$$S_{hm} = x_{St21} + m*A, \quad (2.5)$$

где S_{hm} – промежуточное значение внешней верхней точки ячейки;

$m = 1, 2, 3 \ldots n$ – количество пройденных петельных столбиков при поиске внешней верхней точки ячейки в положительном направлении оси абсцисс;

$m = -1, -2, -3 \ldots -n$ – количество пройденных петельных столбиков при поиске внешней верхней точки ячейки в отрицательном направлении оси абсцисс.

Условие для останова поиска:

$$dx = |x_{Bz1} - x_{St21} - m*A|, \qquad (2.6)$$

где $dx \rightarrow min.$

То есть условием останова поиска будет являться минимальное значение dx на заданном интервале поиска, тогда $m_{min2} = m$, где m имеет значение в момент останова поиска.

ШАГ 4

При рассчитанном минимальном значении dx в заданном интервале поиска, получим:

$|m_{min2}|$ - количество сбавок (прибавок) для следующего петельного ряда, то есть для второго петельного ряда.

Координаты точки окончания расчета сбавок будут задаваться следующим выражением:

$$x_{St31} = x_{St21} + m_{min2}*A, \qquad (2.7)$$

$$y_{St31} = y_{St21} \qquad (2.8)$$

ШАГ 5

$$x_{Sf1} = x_{St31} + k_1*A, \qquad (2.9)$$

где $k_1 = 1$ – поиск внешней верхней точки ячейки осуществлялся в положительном направлении оси абсцисс;

где $k_1 = -1$ – поиск внешней верхней точки ячейки осуществлялся в отрицательном направлении оси абсцисс.

$$y_{Sf1} = y_{St31} \qquad (2.10)$$

Приведем алгоритм аппроксимации и расчета сбавок (прибавок) для i-го петельного ряда по профилю кривой Безье с учетом дополнительных ограничений аппроксимации. В качестве ограничений аппроксимации введем максимальную сбавку (прибавку) в петельном ряду и количество петельных рядов, через которое выполняется сбавка (прибавка).

Пусть SB_{maxi} – величина максимальной сбавки (прибавки) в i-ом петельном ряду;

H_{rap} – количество петельных рядов, через которое будет выполняться сбавка, где $H_{rap} > 0$.

Рекомендуемое значение SB_{maxi} для узла соединения в цельновязаном изделии равно двум сбавкам (прибавкам). Это связано, прежде всего, с особенностями технологии цельновязаного изделия, а также физико-механическими свойствами, перерабатываемого сырья.

ШАГ 1

$$x_{St1i} = x_{Sf(i-1)}, \quad (2.11)$$

$$y_{St1i} = y_{Sf(i-1)}, \quad (2.12)$$

где $(x_{Sf(i-1)}, y_{Sf(i-1)})$ – координаты точки финиша аппроксимации предыдущего петельного ряда, то есть ряда $i - 1$.

ШАГ 2

Увеличиваем значение ординаты точки старта аппроксимации на высоту i-го петельного ряда, то есть $y_{St1i} + H_{rap} * B_i$, тогда

ШАГ 3

$$y_{St2i} = y_{St1i} + H_{rap} * B_i , \quad (2.13)$$

$$x_{St2i} = x_{St1i} \quad (2.14)$$

ПОИСК ОПТИМАЛЬНЫХ КООРДИНАТ ВНЕШНЕЙ ВЕРХНЕЙ ТОЧКИ ЯЧЕЙКИ ДЛЯ i-го ПЕТЕЛЬНОГО РЯДА

Пусть шаг поиска равен величине ширины петельного столбика, тогда

$$S_{hm} = x_{St2i} + m*A, \quad (2.15)$$

где S_{hm} – промежуточное значение внешней верхней точки ячейки;

$m = 1, 2, 3 \ldots n$ – количество пройденных петельных столбиков при поиске внешней верхней точки ячейки в положительном направлении оси абсцисс;

$m = -1, -2, -3 \ldots -n$ – количество пройденных петельных столбиков при поиске внешней верхней точки ячейки в отрицательном направлении оси абсцисс.

Условие для останова поиска:

$$dx = |x_{Bz1} - x_{St21} - m*A|, \quad (2.16)$$

где $dx \to min;$

$0 \le m \le SB_{maxi}$.

То есть условием останова поиска будет являться минимальное значение *dx* на заданном интервале поиска, тогда

$m_{min(i+1)} = m$, где *m* имеет значение в момент останова поиска.

ШАГ 4

При рассчитанном минимальном значении *dx* в заданном интервале поиска, получим:

$|m_{min(i+1)}|$ - количество сбавок (прибавок) для следующего петельного ряда, то есть для ряда $i + 1$.

Координаты точки окончания расчета сбавок будут задаваться следующим выражением:

$$x_{St3i} = x_{St2i} + m_{min(i+1)} * A, \quad (2.17)$$

$$y_{St3i} = y_{St2i} \quad (2.18)$$

ШАГ 5

$$x_{Sfi} = x_{St3i} + k_i * A, \quad (2.19)$$

где $k_i = 1$ – поиск внешней верхней точки ячейки осуществлялся в положительном направлении оси абсцисс;

где $k_i = -1$ – поиск внешней верхней точки ячейки осуществлялся в отрицательном направлении оси абсцисс;

$$y_{Sfi} = y_{St3i} \quad (2.20)$$

Рассмотрим пример аппроксимации ячейками кривой Безье второй степени (рисунок 2.17).

Пусть данная кривая является левой кромкой купона и имеет

следующую аналитическую запись:

$$\begin{cases} X(t) = 3{,}30 * (1{,}00 - t)^2 + 19{,}20 * t * (1{,}00 - t) + 4{,}74 * t^2, \\ Y(t) = 9{,}84 * (1{,}00 - t)^2 + 17{,}52 * t * (1{,}00 - t) + 0{,}36 * t^2; \end{cases}$$

Тогда начальная опорная вершина кривой Безье будет иметь следующие координаты:

$$x_0 = 3{,}30;$$

$$y_0 = 9{,}84;$$

За размерность примем условные единицы (усл. ед.), тогда

длина данной кривой будет составлять 11,9 усл. ед.

A = 0,04 усл. ед. – ширина петельного столбика.

B = 0,05 усл. ед. – высота петельного ряда.

Примем положительное направление оси ординат сверху вниз, оси абсцисс – слева направо.

Результаты расчетов ключевых значений приведем в таблице 1, где:

ax – абсцисса точки начала расчета сбавок (прибавок);

ay – ордината точки старта аппроксимации;

xl – абсцисса точки, принадлежащей кривой Безье.

Данные значения будут являться ключевыми, так как необходимы для контроля расчета сбавок (прибавок) в петельном ряду.

Исходя из ключевых значений, сбавка будет рассчитываться по следующей формуле:

$$\text{сбавка} = \frac{ax_i - ax_{i-1}}{A} \qquad (2.21)$$

Номер петельного ряда:

$$\text{ряд} = \frac{ay_0 - ay_i}{B} + 1, \qquad (2.22)$$

где i – номер шага аппроксимации;

ay_0 – ордината начальной опорной вершины кривой Безье.

$ay_0 = y_0 = 9{,}84$;

Так как кривая является левой кромкой купона, то отрицательное значение сбавки будет являться прибавкой.

Первый шаг аппроксимации:

$$i = 1;$$

$$ay_1 = 9{,}84 - 0{,}05 = 9{,}79;$$

Согласно формуле (2.4):

$$x_{St21} = x_0 = 3{,}30;$$

Рассчитаем значение абсциссы точки, лежащей на кривой Безье на уровне ay_1. То есть значение ординаты в данной точке будет равно 9.79.

Рассчитаем значение параметра *t* для этой точки:

$$9{,}79 = 9{,}84 * (1{,}00 - t)^2 + 17{,}52 * t * (1{,}00 - t) + 0{,}36 * t^2,$$

Решая данное уравнение, получим:

$$t_1 = -0{,}31001;$$

$$t_2 = 0{,}02118;$$

Так как $t \in [0; 1]$, то примем

$$t = t_2 = 0{,}02118;$$

Вычислим значение абсциссы точки, лежащей на кривой Безье:

$$xl = 3{,}30000 * (1{,}00000 - 0{,}02118)^2 + 19{,}20000 * 0{,}02118$$
$$* (1{,}00000 - 0{,}02118) + 0{,}36000 * 0{,}02118^2 = 3{,}56186;$$

Согласно формуле (2.5), рассчитаем все временные значения верхней внешней точки ячейки для первого шага аппроксимации.

$$m = 1{,}00;$$

$$S_{h1} = 3{,}30 + 1{,}00 * 0{,}04 = 3{,}34;$$

Согласно формуле (2.6), рассчитаем значение для останова поиска:

$$dx = |3{,}56186 - 3{,}34000| = 0{,}22186;$$

$$m = 2{,}00;$$

$$S_{h1} = 3{,}30 + 2{,}00 * 0{,}04 = 3{,}38;$$

$$dx = |3{,}56186 - 3{,}38000| = 0{,}18186;$$

Так как $dx \to 0$, то продолжаем поиск:

$$m = 3{,}00;$$

$$S_{h1} = 3{,}30 + 3{,}00 * 0{,}04 = 3{,}42;$$

$$dx = |3{,}56186 - 3{,}42000| = 0{,}14186;$$

$$m = 4{,}00;$$

$$S_{h1} = 3{,}30 + 4{,}00 * 0{,}04 = 3{,}46;$$

$$dx = |3{,}56186 - 3{,}46000| = 0{,}10186;$$

$$m = 5{,}00;$$

$$S_{h1} = 3{,}30 + 5{,}00 * 0{,}04 = 3{,}50;$$

$$dx = |3{,}56186 - 3{,}50000| = 0{,}06186;$$

$$m = 6{,}00;$$

$$S_{h1} = 3{,}30 + 6{,}00 * 0{,}04 = 3{,}54;$$

$$dx = |3{,}56186 - 3{,}54000| = 0{,}02186;$$

$$m = 7{,}00;$$

$$S_{h1} = 3{,}30 + 7{,}00 * 0{,}04 = 3{,}58;$$

$$dx = |3{,}56186 - 3{,}58000| = 0{,}02186;$$

$$m = 8{,}00;$$

$$S_{h1} = 3{,}30 + 8{,}00 * 0{,}04 = 3{,}62;$$

$$dx = |3{,}56186 - 3{,}62000| = 0{,}06186;$$

Так как нарушено условие $dx \rightarrow 0$,то оптимальное значение внешней верхней точки ячейки будет равно $ax = 3{,}58$.

Однако, значения dx при $m = 6{,}00$ и $m = 7{,}00$ равны. В этом случае выбор значения абсциссы внешней верхней точки ячейки будет зависеть от дополнительных ограничений аппроксимации или условий рационализации. Например, таким ограничением может выступить ограничение по максимальной сбавке (прибавке) в петельном ряду. Так как отсутствуют какие-либо дополнительные ограничения аппроксимации выберем $ax = 3{,}58$ при $m = 7{,}00$.

Тогда согласно формуле (2.21):

$$\text{сбавка} = \frac{3.58 - 3.30}{0.04} \approx 7;$$

$$\text{ряд} = \frac{9.84 - 9.79}{0.05} + 1 \approx 2;$$

Полученные значения запишем в таблицу 1:

ax = 3,58;

ay = 9,79;

xl = 3,56186;

сбавка = 7;

ряд = 2.

Таким образом, для получения второго петельного ряда, удовлетворяющего контуру кривой Безье, необходимо сбавить семь петельных столбиков.

Рисунок 2.17 – Кривая Безье

Данные расчета технологии вязания кривой Безье, представленной на рисунке 2.17, запишем в таблицу 1, а затем эти данные используем при проектировании в системе STOLL M1 3.15.

Таблица 1 – Результаты аппроксимации кривой Безье

ax	ay	xl	сбавка	Ряд
3,30	9,84	3,30000	0	1
3,58	9,79	3,56186	7	2
3,78	9,74	3,79012	5	3
3,98	9,69	3,98998	5	4
4,18	9,64	4,16833	5	5
4,34	9,59	4,32939	4	6
4,46	9,54	4,47642	3	7
4,62	9,49	4,61143	4	8
4,74	9,44	4,73638	3	9
4,86	9,39	4,85246	3	10
4,98	9,34	4,96086	3	11
5,06	9,29	5,06231	2	12
5,14	9,24	5,15757	2	13
5,26	9,19	5,24727	3	14
5,34	9,14	5,33190	2	15
5,42	9,09	5,41199	2	16
5,50	9,04	5,48774	2	17
5,54	8,99	5,55962	1	18
5,62	8,94	5,62786	2	19
5,70	8,89	5,69275	2	20
5,74	8,84	5,75447	1	21
5,82	8,79	5,81328	2	22
5,86	8,74	5,86928	1	23
5,94	8,69	5,92270	2	24
5,98	8,64	5,97364	1	25
6,02	8,59	6,02229	1	26
6,06	8,54	6,06874	1	27
6,10	8,49	6,11304	1	28
6,14	8,44	6,15545	1	29
6,18	8,39	6,19591	1	30
6,22	8,34	6,23459	1	31
6,26	8,29	6,27154	1	32
6,30	8,24	6,30686	1	33
6,34	8,19	6,34060	1	34
6,38	8,14	6,37285	1	35
6,42	8,09	6,40362	1	36
6,42	8,04	6,43302	0	37
6,46	7,99	6,46106	1	38
6,50	7,94	6,48782	1	39
6,50	7,89	6,51332	0	40

Продолжение таблицы 1

ax	ay	xl	сбавка	Ряд
6,54	7,84	6,53765	1	41
6,58	7,79	6,56079	1	42
6,58	7,74	6,58282	0	43
6,62	7,69	6,60379	1	44
6,62	7,64	6,62370	0	45
6,66	7,59	6,64261	1	46
6,66	7,54	6,66051	0	47
6,66	7,49	6,67745	0	48
6,70	7,44	6,69350	1	49
6,70	7,39	6,70864	0	50
6,74	7,34	6,72289	1	51
6,74	7,29	6,73631	0	52
6,74	7,24	6,74891	0	53
6,78	7,19	6,76070	1	54
6,78	7,14	6,77172	0	55
6,78	7,09	6,78197	0	56
6,78	7,04	6,79149	0	57
6,82	6,99	6,80028	1	58
6,82	6,94	6,80837	0	59
6,82	6,89	6,81577	0	60
6,82	6,84	6,82250	0	61
6,82	6,79	6,82858	0	62
6,82	6,74	6,83402	0	63
6,82	6,69	6,83883	0	64
6,86	6,64	6,84303	1	65
6,86	6,59	6,84664	0	66
6,86	6,54	6,84966	0	67
6,86	6,49	6,85211	0	68
6,86	6,44	6,85401	0	69
6,86	6,39	6,85535	0	70
6,86	6,34	6,85617	0	71
6,86	6,29	6,85645	0	72
6,86	6,24	6,85622	0	73
6,86	6,19	6,85549	0	74
6,86	6,14	6,85427	0	75
6,86	6,09	6,85256	0	76
6,86	6,04	6,85037	0	77
6,86	5,99	6,84772	0	78
6,86	5,94	6,84461	0	79
6,86	5,89	6,84105	0	80
6,82	5,84	6,83705	-1	81

Продолжение таблицы 1

ax	ay	xl	сбавка	Ряд
6,82	5,79	6,83262	0	82
6,82	5,74	6,82776	0	83
6,82	5,69	6,82248	0	84
6,82	5,64	6,81679	0	85
6,82	5,59	6,81071	0	86
6,82	5,54	6,80422	0	87
6,78	5,49	6,79735	-1	88
6,78	5,44	6,79009	0	89
6,78	5,39	6,78244	0	90
6,78	5,34	6,77444	0	91
6,78	5,29	6,76607	0	92
6,74	5,24	6,75733	-1	93
6,74	5,19	6,74825	0	94
6,74	5,14	6,73880	0	95
6,74	5,09	6,72903	0	96
6,70	5,04	6,71892	-1	97
6,70	4,99	6,70847	0	98
6,70	4,94	6,69769	0	99
6,70	4,89	6,68658	0	100
6,66	4,84	6,67516	-1	101
6,66	4,79	6,66346	0	102
6,66	4,74	6,65142	0	103
6,62	4,69	6,63908	-1	104
6,62	4,64	6,62641	0	105
6,62	4,59	6,61347	0	106
6,62	4,54	6,60026	0	107
6,58	4,49	6,58673	-1	108
6,58	4,44	6,57292	0	109
6,54	4,39	6,55884	-1	110
6,54	4,34	6,54449	0	111
6,54	4,29	6,52984	0	112
6,50	4,24	6,51494	-1	113
6,50	4,19	6,49978	0	114
6,50	4,14	6,48437	0	115
6,46	4,09	6,46868	-1	116
6,46	4,04	6,45271	0	117
6,42	3,99	6,43650	-1	118
6,42	3,94	6,42007	0	119
6,42	3,89	6,40340	0	120
6,38	3,84	6,38643	-1	121

Продолжение таблицы 1

ax	ay	xl	сбавка	Ряд
6,38	3,79	6,36928	0	122
6,34	3,74	6,35187	-1	123
6,34	3,69	6,33421	0	124
6,30	3,64	6,31634	-1	125
6,30	3,59	6,29822	0	126
6,26	3,54	6,27990	-1	127
6,26	3,49	6,26134	0	128
6,26	3,44	6,24253	0	129
6,22	3,39	6,22354	-1	130
6,22	3,34	6,20431	0	131
6,18	3,29	6,18491	-1	132
6,18	3,24	6,16528	0	133
6,14	3,19	6,14542	-1	134
6,14	3,14	6,12533	0	135
6,10	3,09	6,10509	-1	136
6,10	3,04	6,08462	0	137
6,06	2,99	6,06400	-1	138
6,06	2,94	6,04311	0	139
6,02	2,89	6,02206	-1	140
6,02	2,84	6,00081	0	141
5,98	2,79	5,97936	-1	142
5,94	2,74	5,95776	-1	143
5,94	2,69	5,93591	0	144
5,90	2,64	5,91392	-1	145
5,90	2,59	5,89174	0	146
5,86	2,54	5,86936	-1	147
5,86	2,49	5,84680	0	148
5,82	2,44	5,82411	-1	149
5,82	2,39	5,80118	0	150
5,78	2,34	5,77806	-1	151
5,74	2,29	5,75483	-1	152
5,74	2,24	5,73143	0	153
5,70	2,19	5,70785	-1	154
5,70	2,14	5,68402	0	155
5,66	2,09	5,66010	-1	156
5,62	2,04	5,63601	-1	157
5,62	1,99	5,61176	0	158
5,58	1,94	5,58734	-1	159
5,58	1,89	5,56276	0	160
5,54	1,84	5,53802	-1	161

Продолжение таблицы 1

ax	ay	xl	сбавка	Ряд
5,50	1,79	5,51312	-1	162
5,50	1,74	5,48807	0	163
5,46	1,69	5,46287	-1	164
5,42	1,64	5,43751	-1	165
5,42	1,59	5,41201	0	166
5,38	1,54	5,38636	-1	167
5,38	1,49	5,36049	0	168
5,34	1,44	5,33456	-1	169
5,34	1,39	5,33456	0	170
5,30	1,34	5,28219	-1	171
5,30	1,29	5,28219	0	172
5,22	1,24	5,22927	-2	173
5,22	1,19	5,20257	0	174
5,18	1,14	5,17574	-1	175
5,14	1,09	5,14878	-1	176
5,14	1,04	5,12170	0	177
5,14	0,99	5,12170	0	178
5,06	0,94	5,06707	-2	179
5,06	0,89	5,06707	0	180
5,02	0,84	5,01195	-1	181
4,98	0,79	4,98417	-1	182
4,94	0,74	4,95628	-1	183
4,94	0,69	4,95628	0	184
4,90	0,64	4,90005	-1	185
4,90	0,59	4,90005	0	186
4,90	0,54	4,90005	0	187
4,90	0,49	4,90005	0	188
4,90	0,44	4,90005	0	189
4,74	0,39	4,74000	-4	190
4,74	0,34	4,74000	0	191

Данные таблицы используем для построения кривой в подсистеме Shape Editor системы STOLL M1.

На рисунке 2.18 данная кривая представлена в аппроксимированном виде в системе проектирования STOLL M1.

Как видим кривая Безье, представленная на рисунке 2.17 и кривая, полученная в подсистеме Shape Editor системы проектирования STOLL M1, аппроксимированная ячейками, соответствующими петлям трикотажа, совпадают.

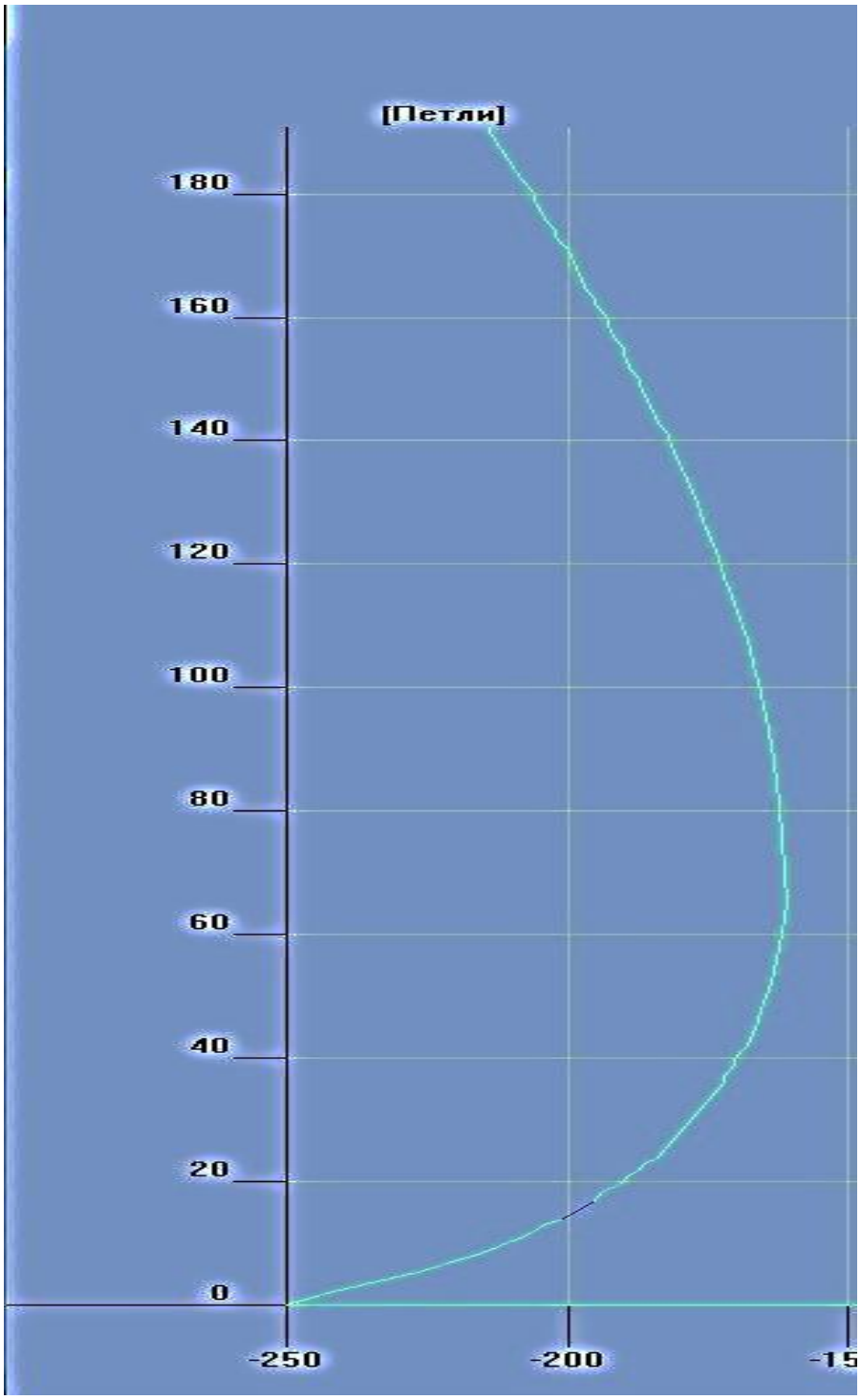


Рисунок 2.18 – Кривая Безье в подсистеме Shape Editor системы проектирования STOLL M1, аппроксимированная ячейками

ВЫВОДЫ ПО ГЛАВЕ 2

1. Разработана система перехода от конструирования к проектированию технологии вязания сложного цельновязаного изделия.
2. В качестве графических объектов выбраны кривые Безье, позволяющие выполнить переход от конструирования к проектированию технологии вязания.
3. Разработан способ аппроксимации графических объектов, позволяющий получить максимально приближенный профиль аппроксимированной кривой в сравнении с профилем исходной кривой.
4. Разработана технология проектирования сбавок (прибавок) по профилю графического объекта, позволяющая проектировать сбавки (прибавки) с учетом различных дополнительных ограничений, например, с ограничением максимальной сбавки в петельном ряду на заданном участке.
5. За счет введения дополнительных ограничений аппроксимации, разработанный способ позволяет выполнять в автоматическом режиме аппроксимацию кривой прямыми отрезками различной длины.

3 РАЗРАБОТКА ОСНОВНЫХ ТЕХНОЛОГИЧЕСКИХ МОДУЛЕЙ ДЛЯ ПРОЕКТИРОВАНИЯ СЛОЖНОГО ЦЕЛЬНОВЯЗАНОГО ИЗДЕЛИЯ

Сложное цельновязаное изделие представляет собой трикотажное изделие, у которого соединение основных и дополнительных деталей происходит в процессе вязания. Цельновязаные изделия могут быть простой (шапки, шарфы и т.п.) и сложной (плечевые, поясные, специального назначения и т.п.) конструкций [1.1].

Проектирование цельновязаного изделия, имеющего сложные узлы соединения, является сложным процессом, требующим большого количества времени и знаний технологических процессов [3.2].

Автоматизация проектирования технологии вязания цельновязаного изделия позволит сократить время, затрачиваемое на проектирование, а также даст возможность для интегрирования в производство новых, более сложных технологий вязания.

3.1 Разработка технологического модуля для проектирования узла соединения в цельновязаном изделии

Суть автоматизации проектирования технологии вязания на участке узла соединения цельновязаного изделия заключается в автоматическом расчете необходимых сбавок (прибавок) с учетом технологических ограничений (например, выполнять аппроксимацию с учетом максимальной сбавки в петельном ряду) и условий оптимизации.

Условия оптимизации соединения деталей в цельновязаном изделии представляют собой массив математических условий, необходимых для получения узла соединения, полностью удовлетворяющего своими параметрами предъявляемым требованиям. Например, для получения в готовом изделии линии соединения деталей необходимого профиля, следует задать требования, с учетом которых будет выполняться её построение. В дальнейшем эти требования будут

преобразованы в массив математических условий, используемых при расчете сбавок (прибавок) в узле соединения.

Внедрение в систему проектирования технологии её обучения, позволит вывести проектирование узла соединения, а также весь процесс проектирования, на путь автономного совершенствования. Например, в момент формирования узла соединения, система проектирования должна выполнять разработку технологии вязания, анализируя участки соединения и комбинируя различные способы соединения деталей (заложенные в нее разработчиком, а также полученными в результате обучения системы) с учетом предъявляемых требований для узла соединения в готовом изделии.

Проектирование узла соединения в цельновязаном изделии не должно выполняться по шаблону, так как это может привести:

- к значительному ограничению возможных соединений деталей;
- к невозможности получения готового цельновязаного изделия со сложным узлом соединения, если конструкция разработана по швейной технологии;
- к невозможности проектирования изделий со сложными линиями соединения;
- к невозможности проектирования изделий по заданной линии соединения;
- к большим затратам интеллектуальных ресурсов при разработке или внедрении в производство сложных технологий, не учтенных в шаблонах системы автоматизированного проектирования, используемых при разработке программ вязания.

Получение качественного узла соединения деталей в готовом цельновязаном изделии является приоритетной задачей для производителя. Качественно выполненный узел соединения в цельновязаном изделии это, прежде всего, хорошая посадка изделия и эстетичный внешний вид.

Для получения качественного узла соединения в сложном цельновязаном изделии необходимо:

- выполнить расчет сбавок (прибавок) по кромкам соединения деталей с учетом физико-механических свойств перерабатываемого сырья, типа переплетения и технологических ограничений вязального оборудования;
- выполнить балансирование элементов соединения на кромках соединяемых деталей;
- выполнить оптимизацию линии соединения согласно заданным требованиям;
- сформировать массив данных, необходимых для вязания цельновязаного изделия.

Расчет сбавок (прибавок) по кромкам соединения деталей с учетом различных дополнительных ограничений выполним при помощи аппроксимации ячейками (см. глава 2).

3.1.1 Разработка способа расчета высоты участка основы узла соединения

Участком основы узла соединения деталей цельновязаного изделия назовем нижний участок, имеющий одинаковое количество петельных столбиков на соединяемых кромках деталей. Верхней границей участка основы узла соединения будет являться линия, после которой кромки соединяемых деталей будут иметь разное количество петельных столбиков (рисунок 3.1).

Определение положения данной линии при проектировании узла соединения необходимо для дальнейшего балансирования петельных элементов на соединяемых кромках, что, в свою очередь, необходимо для создания технологически качественного узла соединения.

Балансированием элементов соединения на соединяемых кромках назовем введение или уменьшение сбавок (прибавок), а также дополнительных петельных рядов или столбиков, обеспечивающих получение качественного соединения деталей цельновязаного изделия, при этом обеспечивая технологически возможный, при вязании изделия, процесс соединения.

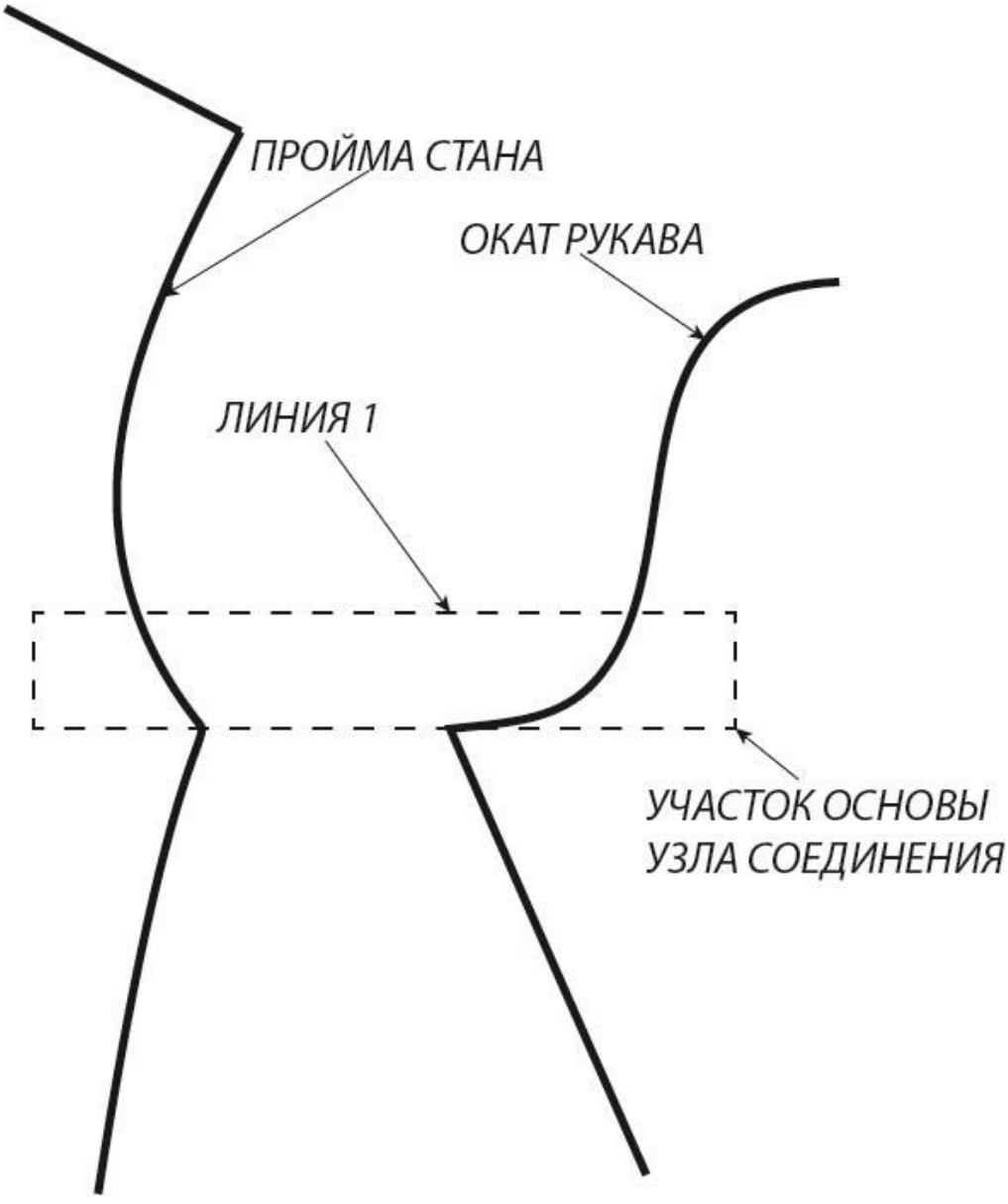


Рисунок 3.1 – Обозначение участка основы узла соединения в плечевом изделии

Согласно рисунку 3.1 окат рукава и пройма стана будут иметь одинаковое количество петельных рядов только до Линии 1, где Линия 1 является верхней границей участка основы узла соединения.

Рассмотрим пример узла соединения у швейного плечевого изделия с втачным типом рукава.

Как известно, при проектировании узла соединения в швейном изделии длина оката втачного рукава должна быть больше длины кривой проймы стана [1.4]. Это необходимо для устранения в готовом изделии различных сборок и дефектов соединения, возникающих в процессе стачивания деталей. Причиной возникновения этих складок может быть непрофессионализм швеи, сбой настройки швейной машины, погрешность в построении кривых оката рукава и проймы стана и т.д.

Для упрощения процесса стачивания трикотажных деталей по кромкам сложных кривых устанавливаются надсечки. Причем для трикотажных изделий расстояние от точки начала стачивания деталей до надсечки, между надсечками и от надсечки до точки окончания стачивания всегда одинаковое (рисунок 3.2).

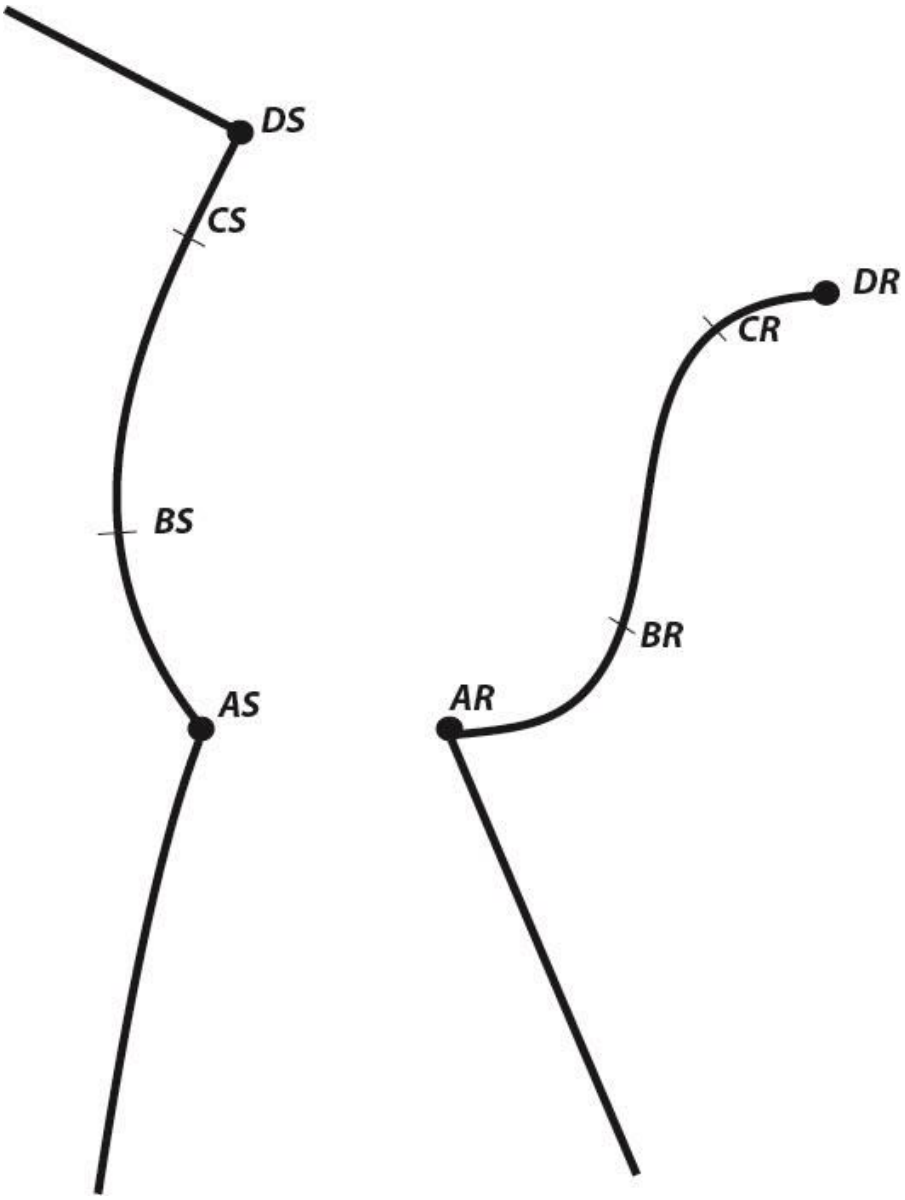


Рисунок 3.2 – Установленные надсечки на кромках соединения деталей

Согласно рисунку 3.2 длина участка кривой проймы ASBS будет равна длине участка кривой оката рукава ARBR, соответственно BSCS = BRCR, CSDS = CRDR [1.3, 1.4].

Выполним графическое наложение участка основы узла соединения, представленного на рисунке 3.1, на рисунок 3.2. Точки пересечения Линии 1 с кривыми проймы стана и оката рукава обозначим как LS и RS (рисунок 3.3).

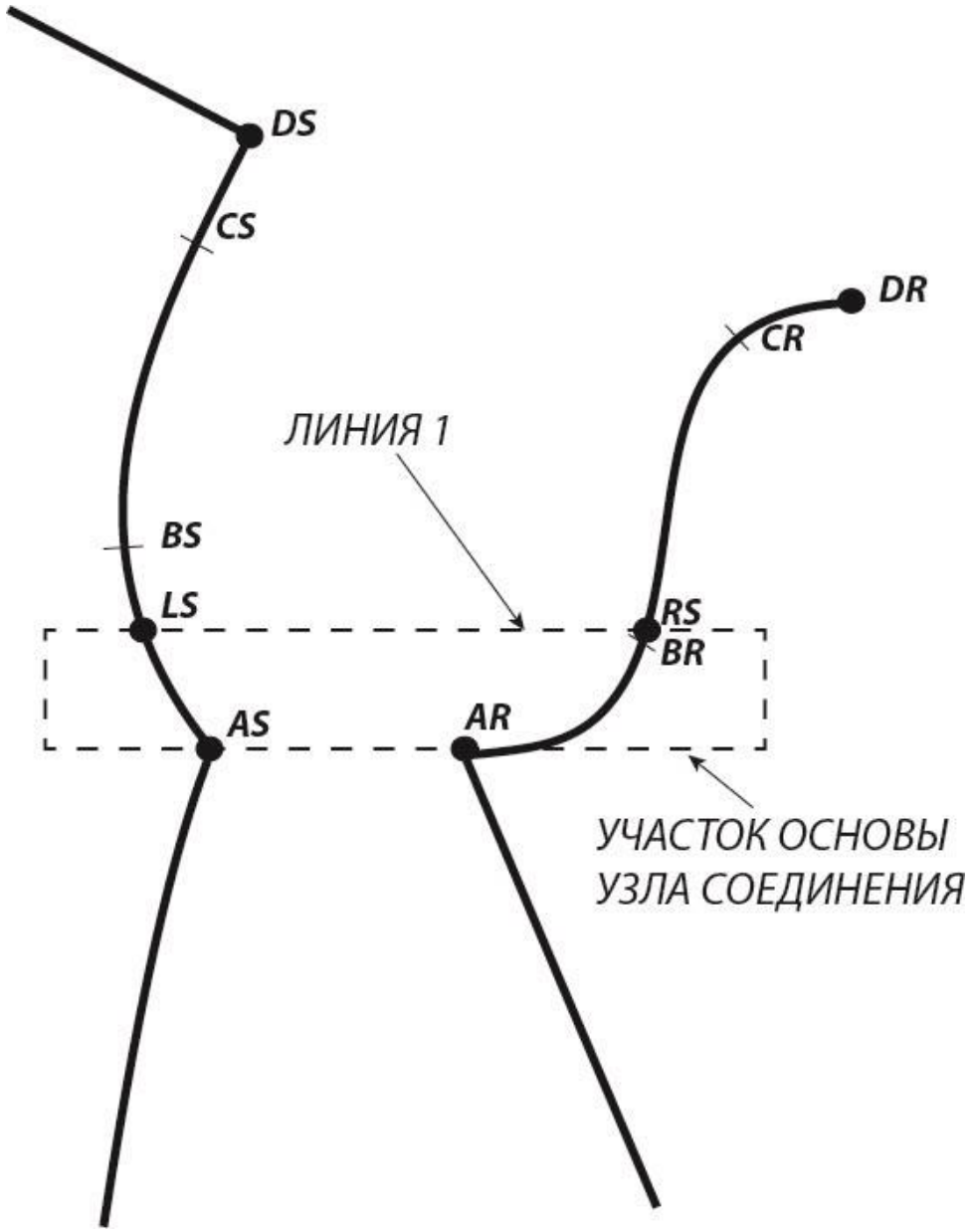


Рисунок 3.3 – Узел соединения в плечевом цельновязаном изделии

Согласно рисунку 3.3 длина участка ASLS будет меньше, чем длина участка ARRS. Но так как данные участки (ASLS и ARRS) имеют одинаковое расстояние по вертикали от точек начала соединения (AS, AR), то данные участки будут иметь одинаковое количество петельных рядов.

Если перенести надсечки BS, BR в точки LS, RS, то при стачивании деталей на участке ASLS будет сборка полотна со стороны рукава, что приведет к нарушению конструктивных характеристик узла соединения.

Поэтому при определении положения Линии 1 в узле соединения цельновязаного изделия необходимо на этапе аппроксимации кромок соединения деталей выполнять расчет длин участков кривых от точки начала кривой до точки, лежащей на кривой и имеющей ординату равную ординате внешней верхней точки ячейки аппроксимации (см. глава 2). На рисунке 3.3 точками, лежащими на кривой, являются точки LS и RS. Так как для различных узлов соединения положение Линии 1 будет всегда разным, следовательно, координаты точек LS и RS для каждого узла соединения будут различны. Причем в процессе аппроксимации с каждым шагом координаты данных точек будут меняться.

Участок кривой, начало которого лежит в нижней точке кромки соединения, а конец – в точке LS, если кромка соединения относительно детали, которой принадлежит LS, является правой, или в точке RS, если кромка соединения – является левой, назовем участком сравнения.

Тогда с каждым шагом аппроксимации должен выполняться расчет длины участка сравнения для каждой кромки соединения. Условием останова поиска будет являться следующее выражение:

$$|L_1 - L_2| \geq ks * B, \qquad (3.1)$$

где L_1 – длина участка сравнения левой кромки соединения;

L_2 – длина участка сравнения правой кромки соединения;

ks – коэффициент сравнения;

B – высота петельного ряда.

Рекомендуемое значение ks = 0,6. Данное значение коэффициента сравнения получено экспериментально за счет анализа различных профилей кромок соединения (30 профилей) и подбора к ним значения коэффициента сравнения (ks), при котором вероятность возникновения ошибки определения положения Линии 1 минимальна.

Суть данного условия заключается в определении абсолютной величины разности длин участков сравнений и последующем сравнении этого значения с высотой петельного ряда. Сравнение с высотой петельного ряда позволит

установить границу, после которой кромки соединения будут иметь разное количество петельных рядов.

Так как для получения качественного узла соединения начальные и конечные точки у кромок соединения должны совпадать (согласно рисунку 3.3 точка AS должна совпадать с точкой AR, точка DS должна совпадать с точкой DR), то соответственно петли в начале и в конце линии соединения могут быть соединены.

В целях устранения возможности образования некачественного соединения деталей в узле соединения, а также снижения напряжения на иглах в процессе выполнения узла соединения, количество соединяемых элементов по кромкам соединения должно быть одинаковым. Причем балансирование элементов соединения в узле соединения должно выполняться после Линии 1, иначе возможно появление отклонения профиля полученной линии соединения от профиля спроектированной линии в узле соединения.

Поэтому значение отклонения длин участков сравнения относительно друг друга $|L_1 - L_2|$ было принято меньше, чем высота петельного ряда, иначе рассчитанное положение Линии 1 будет ошибочно на один петельный ряд.

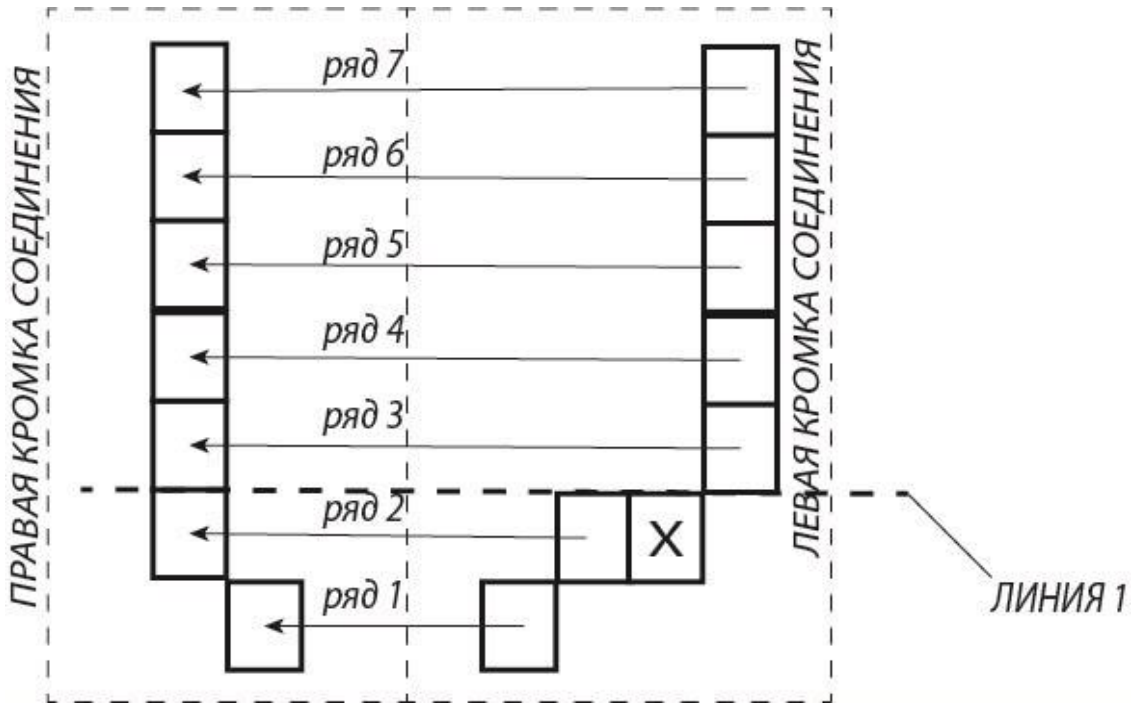


Рисунок 3.4 – Соединение деталей с несбалансированными кромками из-за ошибки определения положения Линии 1

Предположим определение положения Линии 1 выполнено с ошибкой на один петельный ряд, тогда, согласно рисунку 3.4, правая и левая кромки соединения имеют одинаковое количество петельных рядов, однако, левая кромка соединения длиннее, чем правая.

Так как необходимо получить качественный узел соединения, то количество соединяемых элементов должно быть одинаково. Однако левая кромка соединения во втором ряду имеет на один элемент соединения больше (ячейка “X”), чем правая кромка, поэтому ячейка, представляющая собой, например, петлю и обозначенная “X”, должна соединиться с петлей третьего ряда правой кромки соединения, в этом случае, положение Линии 1 определено ошибочно и балансирование элементов соединения не выполнено. Для выполнения соединения петлю в ячейке “X” необходимо сбавить-перенести на соседнюю петлю по левой кромке соединения, однако, в результате между вторым и третьим петельными рядами образуется отверстие, в данном случае на ширину двух петель.

Поиск положения Линии 1 имеет следующий алгоритм:

ПОЛУЧЕНИЕ КООРДИНАТ ТОЧЕК НАЧАЛА ПОИСКА >>> ШАГ АППРОКСИМАЦИИ >>> РАСЧЕТ ДЛИНЫ УЧАСТКА СРАВНЕНИЯ >>> РАСЧЕТ ОТКЛОНЕНИЯ ДЛИН УЧАСТКОВ СРАВНЕНИЯ >>> АНАЛИЗ

Графически алгоритм поиска положения Линии 1 представлен на рисунке 3.5.

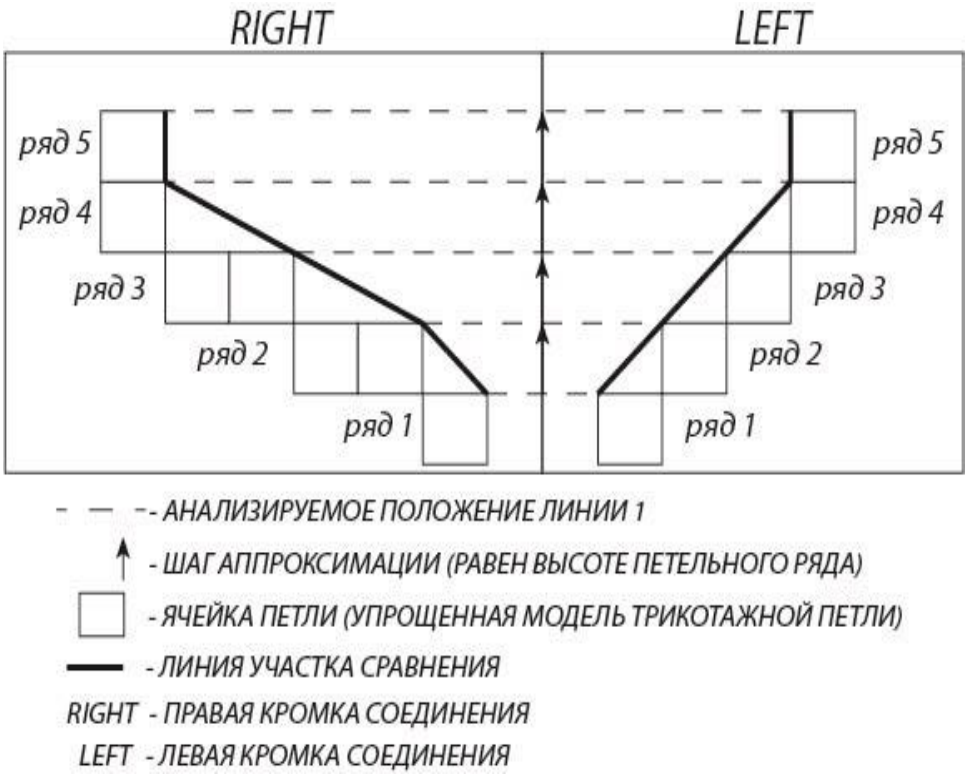


Рисунок 3.5 – Алгоритм графического поиска положения Линии 1

Так как после аппроксимации кривой Безье ячейками, профиль кромки меняется, а наиболее близкой к точке кривой является верхняя внешняя точка ячейки, то пусть для каждой кромки соединения на каждом шаге аппроксимации линия участка сравнения проходит через верхнюю внешнюю точку ячейки.

Точкой начала поиска положения Линии 1 будет являться верхняя внешняя точка ячейки первого петельного ряда в узле соединения.

С каждым шагом аппроксимации по рядам для каждой кромки соединения выполняется расчет длины линии участка сравнения.

Математически алгоритм поиска положения Линии 1 в узле соединения цельновязаного изделия примет следующий вид:

Пусть (x_{Li}, y_{Li}) – точка линии участка сравнения левой кромки соединения (верхняя внешняя точка ячейки левой кромки);

(x_{Ri}, y_{Ri}) – точка линии участка сравнения правой кромки соединения (верхняя внешняя точка ячейки правой кромки);

i – номер шага аппроксимации (петельного ряда);

B – высота петельного ряда;

LL – длина линии участка сравнения левой кромки соединения на текущем шаге аппроксимации;

LR – длина линии участка сравнения правой кромки соединения на текущем шаге аппроксимации;

dLR – значение отклонения длин линий участков сравнения кромок соединения относительно друг друга;

$L1$ – положение Линии 1 в узле соединения.

$$LL = \sqrt{(x_{Li} - x_{L(i+1)})^2 + (y_{Li} - y_{L(i+1)})^2}, \quad (3.2)$$

$$LR = \sqrt{(x_{Ri} - x_{R(i+1)})^2 + (y_{Ri} - y_{R(i+1)})^2}, \quad (3.3)$$

$$dLR = abs(LL - LR) \quad (3.4)$$

Анализ, полученного значения отклонения (dLR):

если $dLR \geq 0.6 * B,$ (3.5)

то $L1 = i + 1,$ (3.6)

где для первого петельного ряда в узле соединения $i = 0$.

Использование данного алгоритма поиска Линии 1 позволит:

- установить высоту участка в узле соединения цельновязаного изделия, на котором кромки соединения имеют одинаковое число петельных рядов;

– выполнить балансирование элементов соединения с малым отклонением (1-2 петли), полученной линии соединения от спроектированной линии;
– выполнить рационализацию узла соединения, то есть усовершенствовать технологию вязания узла соединения с целью повышения эффективности вязания изделия с учетом конструктивных требований, предъявляемых к узлу соединения с целью исключения возможности ухудшения посадки изделия.

3.1.2 Балансирование соединяемых элементов в узле соединения и рационализация линии соединения

Элементом соединения в узле соединения является трикотажная петля, участвующая в формировании линии соединения.

Количество элементов соединения в одном петельном ряду будет зависеть от количества сбавленных петель в текущем петельном ряду на кромке соединения [3.2]. В одном петельном ряду будет больше одного соединительного элемента, если в этом ряду больше одной сбавленной петли. Например, во втором петельном ряду (рисунок 3.6) количество элементов соединения равняется двум, так как петля 2 на левой кромке соединения будет соединена с петлей 2’ на правой кромке соединения, петля 2’’ – с петлей 3’. Общее количество элементов соединения на левой кромке соединения – 4, на правой кромке соединения – 4.

Таким образом, количество элементов соединения представим следующей формулой:

$$N_{(R||L)} = \sum_{i=1}^{n} |N_i - 1|, \qquad (3.7)$$

где $N_{(R||L)}$ – количество элементов соединения на правой (R) или левой (L) кромке соединения;

i – текущий петельный ряд;

n – общее количество петельных рядов на кромке соединения;

N_i – количество сбавок в i-ом петельном ряду.

Для того чтобы выполнить соединение всех сбавленных петель одной кромки с петлями другой необходимо выполнить балансирование элементов соединения.

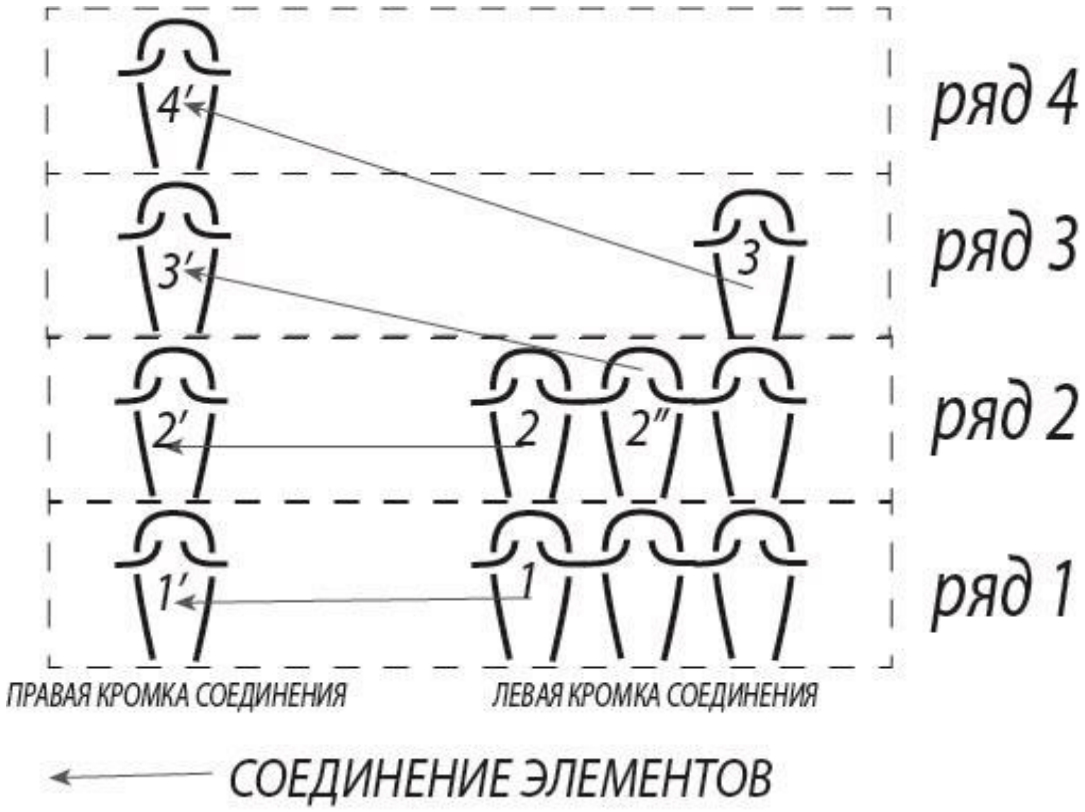


Рисунок 3.6 – Элементы соединения

Балансирование элементов соединения необходимо для получения качественного узла соединения в цельновязаном изделии. Балансирование должно выполняться после Линии 1 (рисунок 3.3), иначе это может привести к:

- Отклонению профиля линии соединения от профиля спроектированной линии;
- Нарушению посадки;
- Ухудшению внешнего вида изделия;
- Появлению технологических дефектов в узле соединения, например, появлению единичных увеличенных петель;

Балансирование, в зависимости от количества элементов соединения, может быть следующих типов (рисунок 3.7):

- Балансирование только по левой кромке соединения;
- Балансирование только по правой кромке соединения;
- Комбинированное балансирование элементов соединения по обеим кромкам соединения.

Рисунок 3.7 – Типы балансирования элементов соединения

В свою очередь каждый тип балансирования может иметь определенный вид, то есть балансирование может выполняться только по петельным столбикам, рядам или по петельным столбикам и рядам.

В случае балансирования по петельным рядам в кромку соединения необходимо добавить дополнительные петельные ряды с нулевой сбавкой (рисунок 3.8). При балансировании элементов соединения по петельным столбикам в определенных петельных рядах необходимо добавить дополнительное количество сбавок (рисунок 3.9). Комбинированный способ балансирования элементов соединения выполняется за счет добавления дополнительного количества петельных рядов и сбавок (рисунок 3.10).

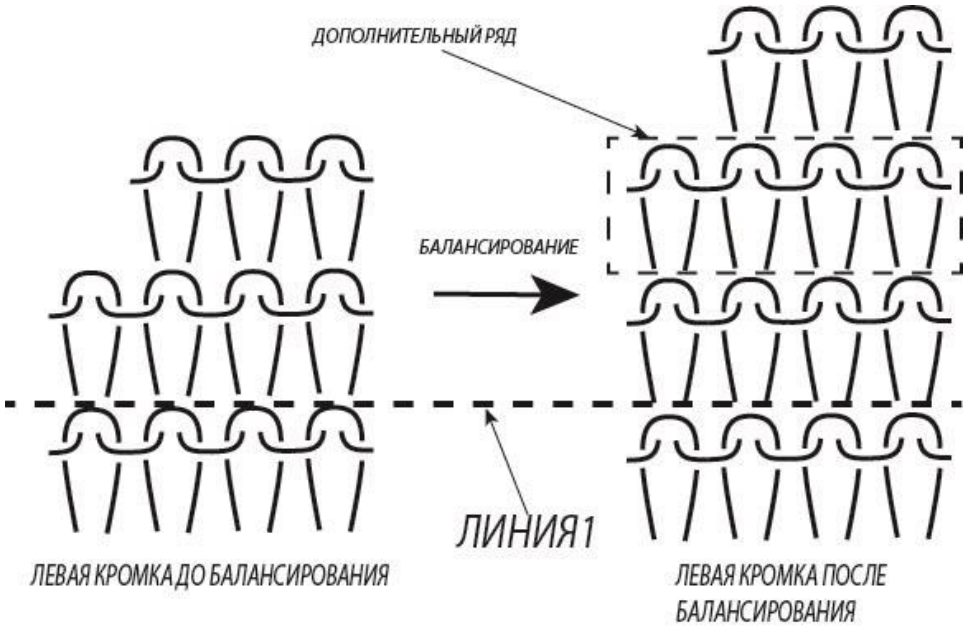


Рисунок 3.8 – Балансирование элементов соединения по петельному ряду

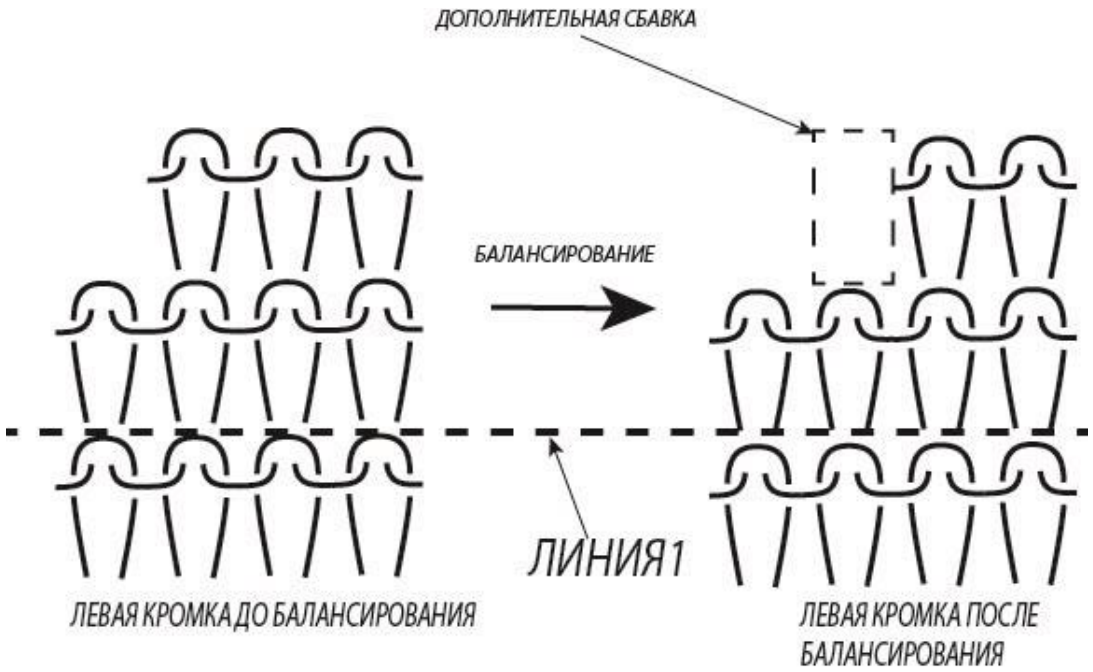


Рисунок 3.9 – Балансирование элементов соединения по петельному столбику

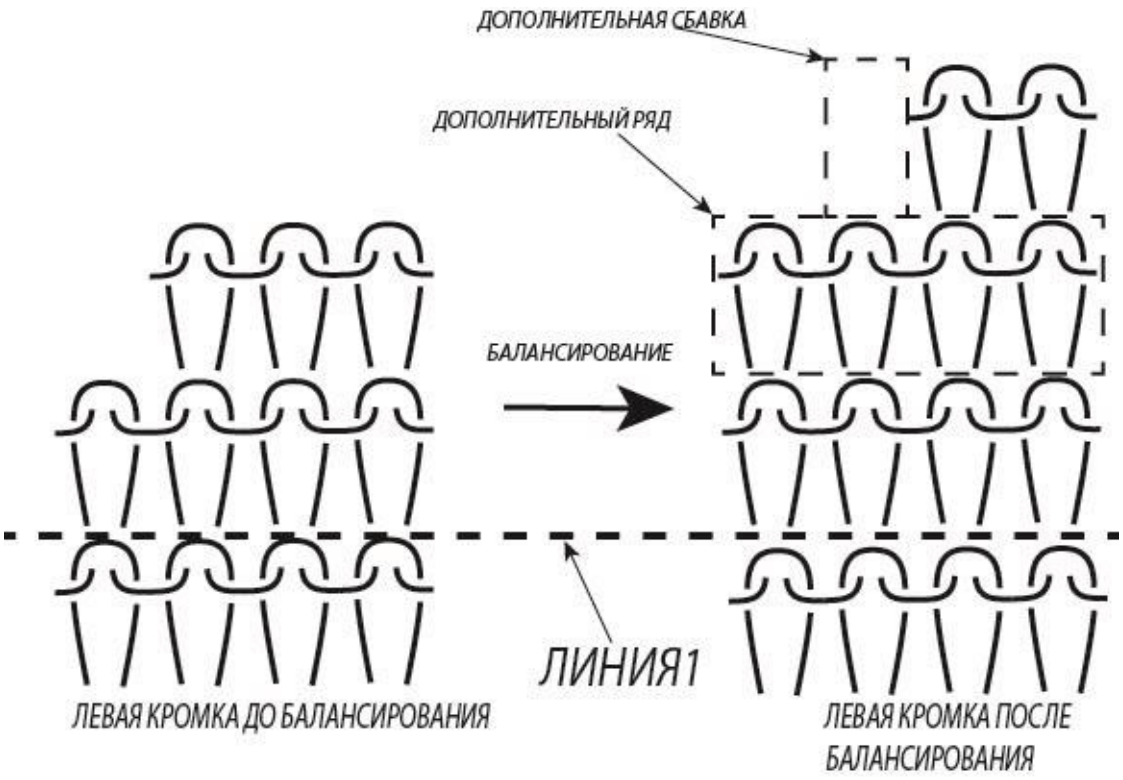


Рисунок 3.10 – Балансирование элементов соединения по петельному ряду и столбику

Графически сравним данные виды балансирования левой кромки на аппроксимированной кривой Безье третьей степени (рисунок 3.11). Предположим, что необходимо выполнить балансирование трех элементов соединения. Полученные кромки соединения сравним при помощи графического наложения (рисунок 3.12).

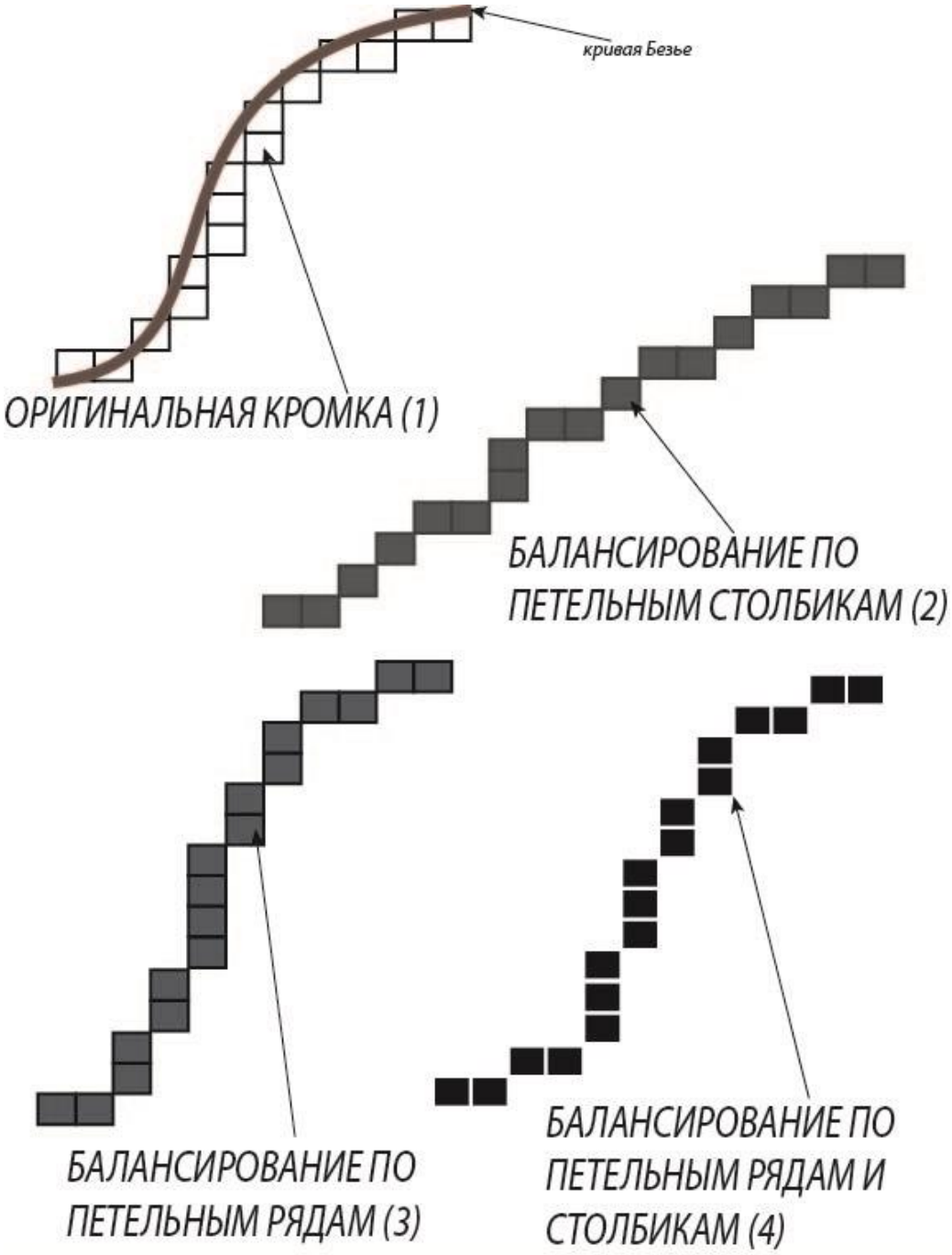


Рисунок 3.11 – Аппроксимированная кривая Безье третьей степени с различными видами балансирования элементов соединения

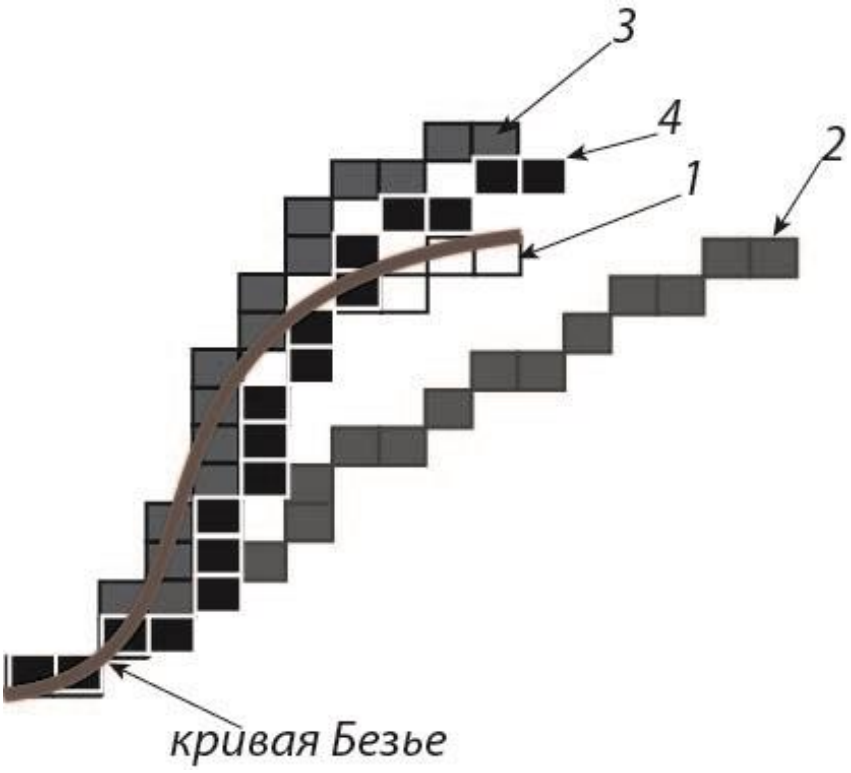


Рисунок 3.12 – Графическое сравнение различных видов балансирования

Согласно рисунку 3.12 вид балансирования по петельному столбику и петельному ряду (кривая 4) является наиболее оптимальным, так как имеет минимальное отклонение профиля кромки от оригинальной после процесса балансирования. Однако эффективность использования каждого вида балансирования будет зависеть от характеристик профиля кромки соединения: величины выпуклости, наличия технологических ограничений на конкретном участке и т.д. Кривая 1 представляет кромку до выполнения балансирования.

Согласно рисунку 3.7 балансирование элементов соединения может иметь три типа.

Балансирование по левой кромке соединения.

При данном типе балансирования, если общее количество элементов соединения меньше, чем у другой кромки, то элементы соединения будут добавляться и наоборот убавляться, если общее количество элементов соединения больше.

Эффективное использование данного типа балансирования возможно, когда кромки соединения имеют профили с малой (в одну или две петли) выпуклостью, а также небольшим расхождением (не более трех элементов) по количеству петельных элементов (рисунок 3.13). В этом случае кромки будут иметь минимальные отклонения профилей от профилей исходных кромок.

Балансирование по правой кромке соединения.

При данном типе балансирования, если общее количество элементов соединения меньше, чем у другой кромки, то элементы соединения будут добавляться и наоборот убавляться, если общее количество элементов соединения больше.

Также как и в случае с типом балансирования по левой кромке, эффективное использование данного типа балансирования возможно, когда кромки соединения имеют профили с малой (в одну или две петли) выпуклостью, а также небольшим расхождением (не более трех элементов) по количеству петельных элементов.

Комбинированный метод балансирования элементов соединения.

Данный тип балансирования основан на совместном использовании двух предыдущих типов.

Суть комбинированного типа балансирования элементов соединения состоит во взаимном изменении профилей кромок соединения.

Так как кромки соединения, в большинстве случаев, имеют разное количество элементов соединения, то в результате балансирования комбинированным типом, количество элементов соединения, соединяемых кромок, будет равно среднеарифметическому значению.

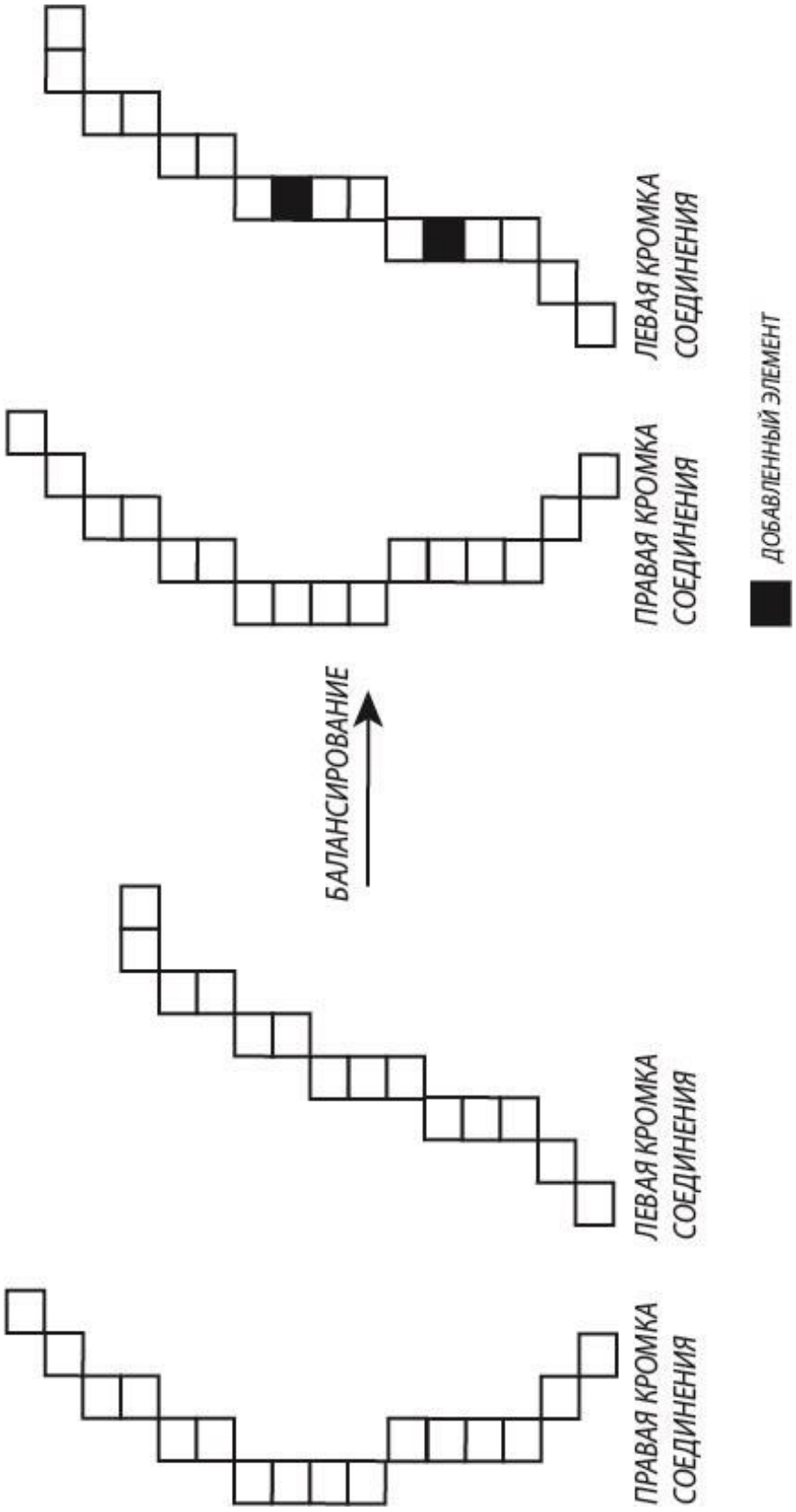


Рисунок 3.13 – Балансирование по левой кромке соединения по петельному ряду

Рассмотрим пример.

Пусть правая кромка соединения имеет 16 элементов, а левая кромка соединения – 14. Используя каждый из методов балансирования, получим:

Балансирование для левой кромки соединения.

Используя этот тип балансирования, количество элементов соединения левой кромки увеличится на две единицы.

Балансирование для правой кромки соединения.

Используя этот тип балансирования, количество элементов соединения правой кромки уменьшится на две единицы.

Балансирование комбинированным типом.

Используя этот тип балансирования, количество элементов соединения обеих кромок будет равно (16+14)/2 = 15, то есть по правой кромке соединения необходимо убрать один петельный ряд (например, третий), а по левой кромке соединения добавить один петельный ряд (например, восьмой).

На основе рисунка 3.13 выполним балансирование элементов соединения комбинированным типом (рисунок 3.14).

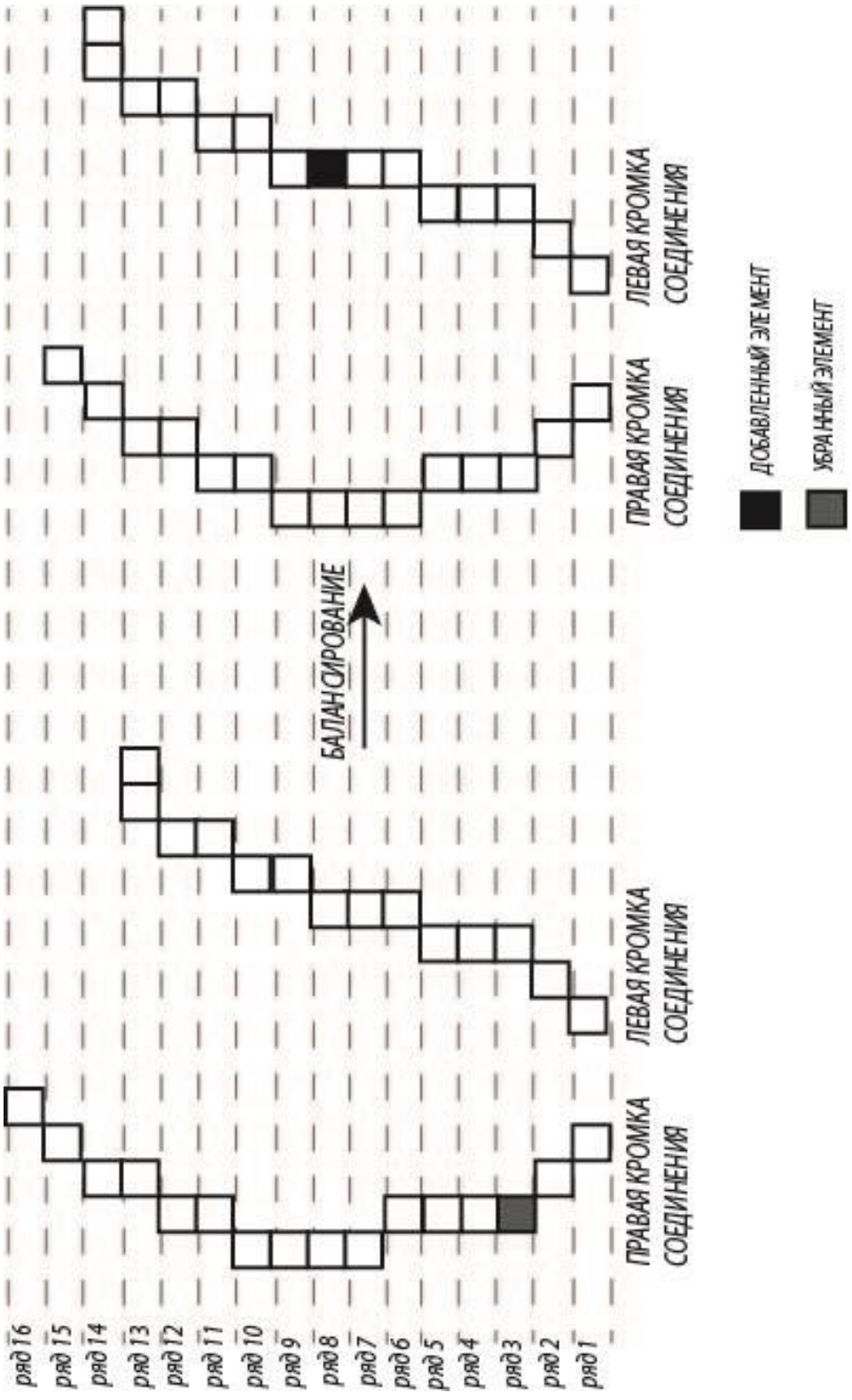


Рисунок 3.14 – Балансирование комбинированным типом

Таким образом, применение комбинированного метода для балансирования элементов соединения позволит:

- получить качественный узел соединения максимально приближенный к узлу, полученному на этапе конструкторской подготовки;
- улучшить посадку изделия;
- улучшить внешний вид узла соединения за счет сбалансированного распределения сбавок (прибавок) по кромкам соединения;
- получить сложный узел соединения в трикотажных изделиях специального и технического назначения, а также изделиях сувенирной группы;
- расширить ассортиментный ряд, выпускаемой продукции, за счет внедрения в производство сложных технологий вязания;
- сократить время и расход сырья при разработке;
- снизить затраты на проектирование цельновязаных изделий;
- сократить производственные площади.

3.2 Разработка технологического модуля для проектирования выпуклых участков

Основой качественной посадки изделия, способной удовлетворить потребителя, является использование в конструкции изделия измененных значений линейных размеров и форм тела человека, а также прибавок на свободу покроя значениями эмпирических коэффициентов, получаемых в результате исследования физико-механических показателей трикотажа с учетом выбранной пряжи и переплетения, а также применения в технологии вязания способов формирования рельефа (выпуклых участков) за счет группового переноса петель или провязывания дополнительных рядов.

При формировании участка выпуклости за счет группового переноса петель, на участке трикотажа будет иметь место столбик со сдвоенными петлями. Операция переноса петель – достаточно сложный процесс, который может

привести к дефектам на изделии, кроме того операция переноса петель может явиться ограничением при создании сложного переплетения на данном участке. Поэтому наиболее оптимальным способом формирования выпуклых участков трикотажа будет являться способ, при котором выпуклость участка достигается за счет провязывания дополнительных петельных рядов. Это даст возможность получения необходимого рельефа, создаваемого на изделии в целях получения качественной посадки, без локальных повреждений структуры трикотажа и ухудшения его эстетических характеристик.

3.2.1 Разработка метода проектирования выпуклого участка трикотажа при помощи частичного вязания

Суть получения выпуклого участка трикотажа за счет частичного вязания, состоит в провязывании дополнительных петельных рядов между петельными рядами основного переплетения. Причем ширина дополнительных рядов должна быть меньше, чем ширины соседних петельных рядов (рисунок 3.15).

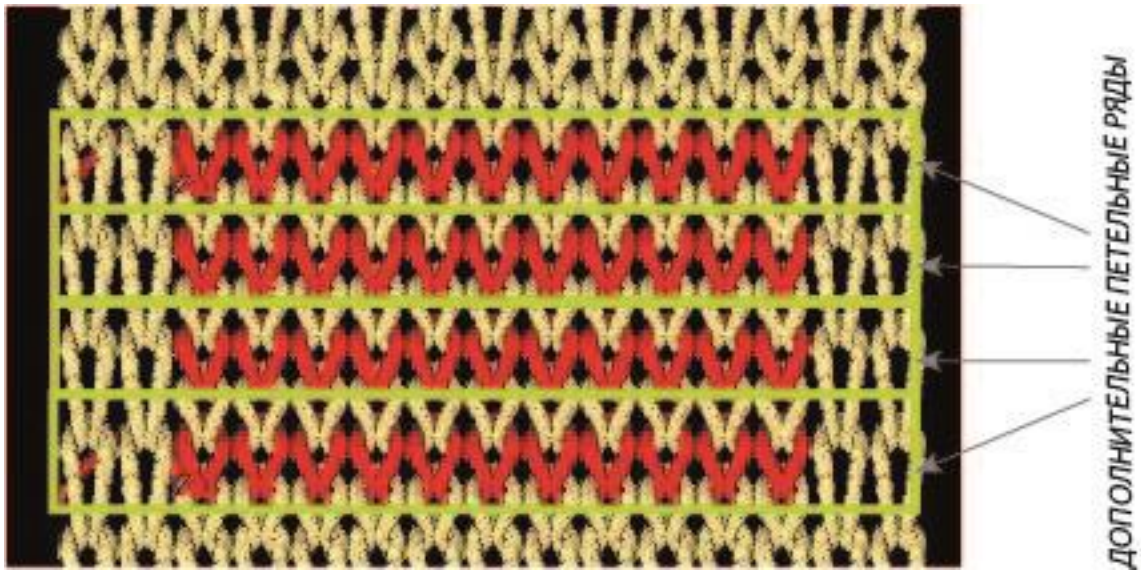


Рисунок 3.15 – Дополнительные петельные ряды в системе проектирования STOLL M1 3.15

При провязывании дополнительных петельных рядов не подряд, а чередуясь с основными петельными рядами, вырабатываемый участок трикотажа, будет иметь увеличенную высоту и очень маленькие глазки отверстий, равные высоте петельного ряда, в местах перехода от вязания основным нитеводителем к вязанию дополнительным нитеводителем. Причем высота, измененного участка трикотажа, будет увеличена на сумму высот дополнительных петельных рядов, провязываемых на данном участке.

При провязывании нескольких дополнительных рядов подряд точки начала и конца провязывания каждого дополнительного ряда должны меняться, чтобы в этих точках не было отверстий. Линии, ограничивающие участок дополнительных рядов, должны находиться внутри зоны основного полотна. В этом случае в зоне провязывания дополнительных рядов высота этого участка будет больше длины основного полотна, что позволит создать выпуклость (рисунок 3.16).

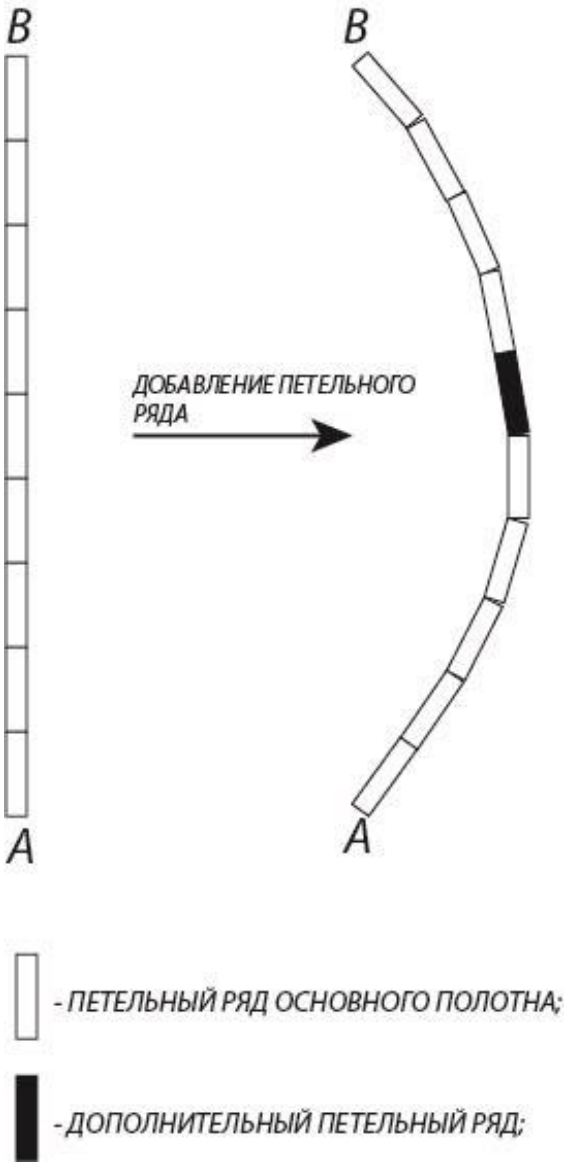


Рисунок 3.16 – Профиль выпуклого участка трикотажа

Согласно рисунку 3.16 участок трикотажа меняет свою выпуклость за счет провязывания одного дополнительного петельного ряда, причем верхние и нижние границы участка не меняются. Длину участка трикотажа с учетом выпуклости представим следующей формулой:

$$LV = \sum_{i=1}^{n} BO + \sum_{j=1}^{m} BD, \qquad (3.8)$$

где LV – длина профиля выпуклого участка трикотажа;

BO – высота петельного ряда основного полотна в равновесном состоянии;

BD – высота дополнительного петельного ряда в равновесном состоянии;

n – количество петельных рядов основного полотна на участке выпуклости;

m – количество дополнительных петельных рядов на участке выпуклости.

Так как трикотажная петля под воздействием различных внешних факторов может иметь нестабильные линейные размеры, то в формуле 3.8 необходимо ввести дополнительный корректирующий коэффициент.

$$LV = k * (\sum_{i=1}^{n} BO + \sum_{j=1}^{m} BD), \qquad (3.9)$$

где k – коэффициент, корректирующий выпуклость участка трикотажа.

Использование данного коэффициента необходимо при моделировании профиля выпуклости.

Так как выпуклость моделируется для трикотажа, петли которого находятся в равновесном состоянии, то основным фактором, влияющим на величину выпуклости при моделировании выпуклого участка трикотажа, будет являться способность петель менять свои параметры под воздействием сил, действующих внутри трикотажного полотна. Также данный коэффициент будет менять свое значение в зависимости от физико-механических свойств перерабатываемого сырья и типа переплетения.

На рисунке 3.17 представлена схема петельной структуры трикотажа при формировании выпуклого участка.

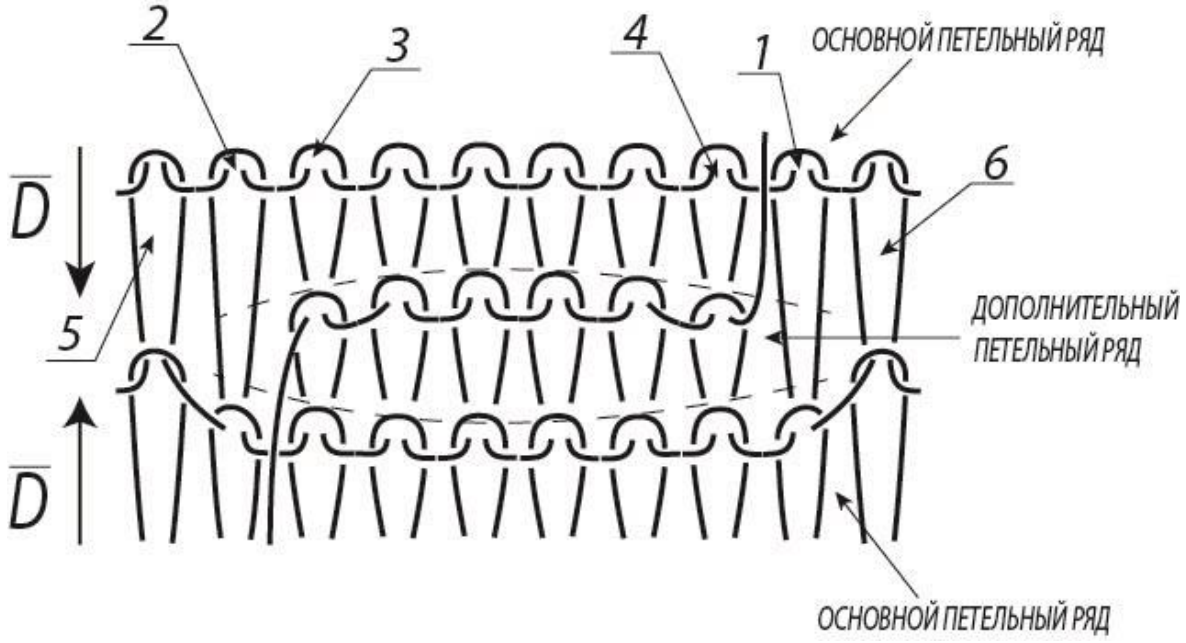


Рисунок 3.17 – Петельная структура выпуклого участка трикотажа

Так как при формировании выпуклого участка трикотажа ширина дополнительных петельных рядов должна быть меньше ширины основных петельных рядов, то под воздействием давления, оказываемыми основными петельными рядами, при приближении к краю, значения линейных параметров трикотажной петли дополнительного петельного ряда будут уменьшаться [2.5].

Кроме того значения линейных параметров трикотажных петель дополнительного петельного ряда, при движении от центра ряда к краю, будут меняться в зависимости от показателей жесткости пряжи, её линейной плотности, усадки (присадки), типа переплетения.

На рисунке 3.17 показано, что две крайние трикотажные петли дополнительного петельного ряда под воздействием сил D, формируемые основными петельными рядами, а именно петлями 1 и 2, имеют меньшие значения высота *B*, чем две петли, находящиеся по центру дополнительного петельного ряда.

Петли 1 и 2 из рисунка 3.17, имея одинаковую длину со всеми петлями основного трикотажа, сдавливают крайние петли дополнительного петельного ряда [2.5, 2.6]. Кроме того петли 1 и 2 вытягивают нить из соседних петель 3, 4, 5,

6 своего ряда, но их высота *B* никогда не будет равной *2*B*, поэтому появляется сила *D*, сдавливающая крайние петли дополнительного петельного ряда.

При расчете коэффициента, корректирующего выпуклость участка трикотажа, необходимо учитывать много различных факторов, влияющих на величину выпуклости в готовом изделии, что требует большого количества времени. Поэтому значение коэффициента *k* в промышленных условиях рационально, то есть, не затрачивая большого количества времени на его расчет, определять экспериментально.

$$k = \frac{LV_1}{LV_2}, \qquad (3.10)$$

где LV_1 – длина фактической выпуклости;

LV_2 – длина смоделированной выпуклости по формуле 3.8;

В таблице 2 представлены значения коэффициента *k* для некоторых трикотажных переплетений, полученных экспериментально.

Таблица 2 – Значения коэффициента *k* для различных типов переплетений

Тип переплетения	Тип сырья	Линейная плотность пряжи, Текс	Значение коэффициента *k*
Кулирная гладь	Хлопчатобумажная пряжа	26*2	0,945
	Полушерстяная пряжа (70% акрил, 30% меринос)	32*2	0,912
Ластик 1+1	Хлопчатобумажная пряжа	26*2	0,895
	Полушерстяная пряжа (70% акрил, 30% меринос)	32*2	0,864
Интерлок	Хлопчатобумажная пряжа	26*2	0,840
	Полушерстяная пряжа (70% акрил, 30% меринос)	32*2	0,810

Так как при движении от центра к краю дополнительного петельного ряда, значения линейных параметров трикотажных петель уменьшаются, то при провязывании дополнительных петельных рядов разной ширины можно получить выпуклый участок трикотажа сложной формы (рисунок 3.18). Выпуклые участки трикотажа сложной формы можно использовать в местах сгибания (например, на локтевых участках рукавов), в формировании плечевого участка изделия, в качестве элементов дизайна и т.д.

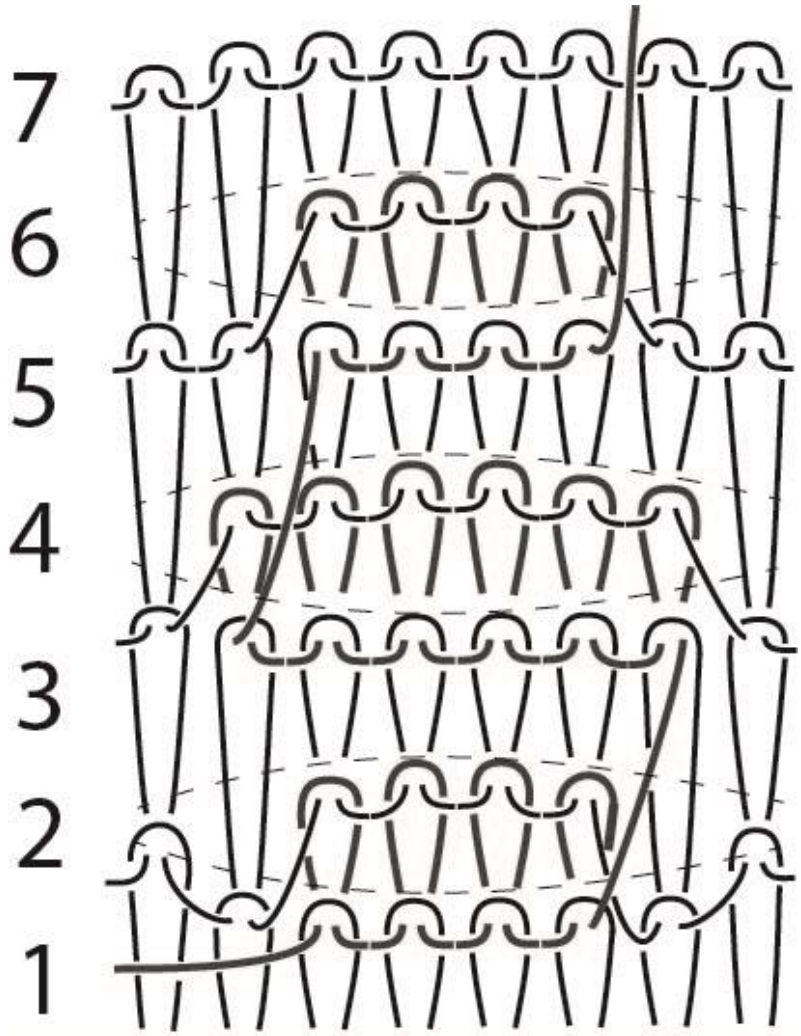


Рисунок 3.18 – Выпуклый участок трикотажа с дополнительными петельными рядами разной ширины

Согласно рисунку 3.18 петельные ряды (1), (3), (5), (7) являются основными петельными рядами, ряды (2), (4), (6) – дополнительными петельными рядами. В дополнительном петельном ряду меньшей ширины (2) соотношение количества деформированных петель к количеству недеформированных петель меньше, чем у дополнительного петельного ряда (4), поэтому величина выпуклости сегмента выпуклого участка трикотажа, образуемая дополнительным петельным рядом (2), будет меньше, чем у сегмента выпуклого участка трикотажа, образуемого дополнительным петельным рядом (4).

Ширина дополнительного петельного ряда рассчитывается исходя из формы края выпуклого участка трикотажа. Форма края выпуклого участка трикотажа необходимо формировать на этапе конструкторской подготовки изделия.

Так как разность ширин дополнительных петельных рядов дает от 3 до 10% от общего значения выпуклости, то форму края проектируемого выпуклого участка трикотажа можно сформировать исходя из способов получения рельефа, используемых методикой конструирования [1.4]. Причем, как было указано раннее, при построении конструкции при помощи кривых Безье, можно использовать любую методику конструирования.

На рисунке 3.19 представлены выпуклые участки трикотажа, полученные при одинаковой ширине дополнительных рядов – справа и с разной шириной дополнительных петельных рядов – слева. Как видим угол изгиба образцов практически одинаков.

Рисунок 3.19 – Сравнение выпуклых участков трикотажа

Однако на образце, представленном слева, по краю провязывания дополнительных рядов образуется небольшая складка, на правом образце такая складка отсутствует. Поэтому выпуклость с провязанными дополнительными рядами разной ширины более рационально использовать для формирования небольших участков, например, для формирования локтевых участков, выпуклой области плечевой точки и т.д. выпуклость с дополнительными петельными рядами одинаковой ширины целесообразно использовать для участков с большим радиусом изгиба, например, для изделий технического назначения.

Рассмотрим рисунок 3.16. Интерполируем профили участков трикотажа, представленных на нем, кривыми Безье второй степени (рисунок 3.20).

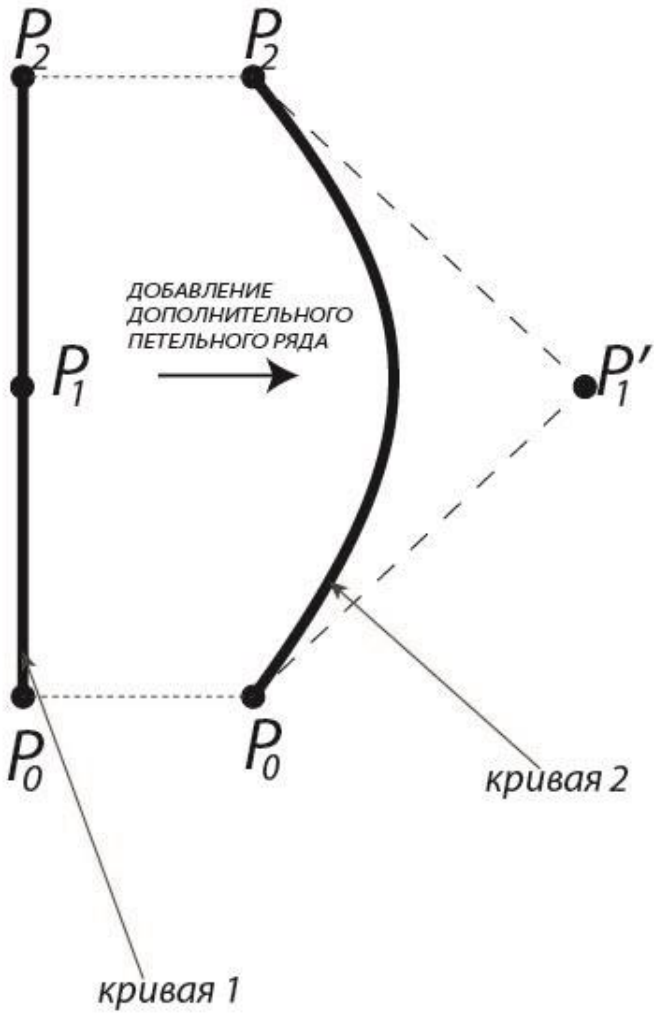


Рисунок 3.20 – Профили участков трикотажа в виде кривых Безье

Согласно рисунку 3.20 кривая 1 представляет собой первоначальное состояние участка трикотажа, кривая 2 – выпуклый участок трикотажа, с провязанным дополнительным петельным рядом. Причем кривая 1 является кривой Безье второй степени. Так как у кривой 1 промежуточная опорная вершина лежит на прямой, проходящей через точки P_0 и P_2 , то кривая Безье представлена в виде прямого отрезка.

Кривые Безье, представленные на рисунке 3.20, имеют одинаковые координаты начальной (P_0) и конечной (P_2) опорной вершины. Так как после провязывания дополнительного петельного ряда длина кривой Безье увеличится (кривая справа на рисунке 3.20), а координаты начальной и конечной опорных вершин не изменятся, то изменение длины кривой будет выполняться за счет перемещения промежуточной опорной вершины (P_1), тем самым формируя выпуклость.

Учитывая то, что начальная и конечная опорные вершины кривой не могут менять своего положения, то кривая 1 будет являться проекцией выпуклой кривой 2 на плоскость, перпендикулярную плоскости кривой и проходящей через начальную и конечную опорные вершины.

Таким образом, задача моделирования выпуклости участка трикотажа, формирование которой выполняется за счет провязывания дополнительных петельных рядов, сводится к поиску координат промежуточной опорной вершины кривой Безье второй степени.

Представим алгоритм поиска координат промежуточной опорной вершины кривой Безье второй степени на рисунке 3.21.

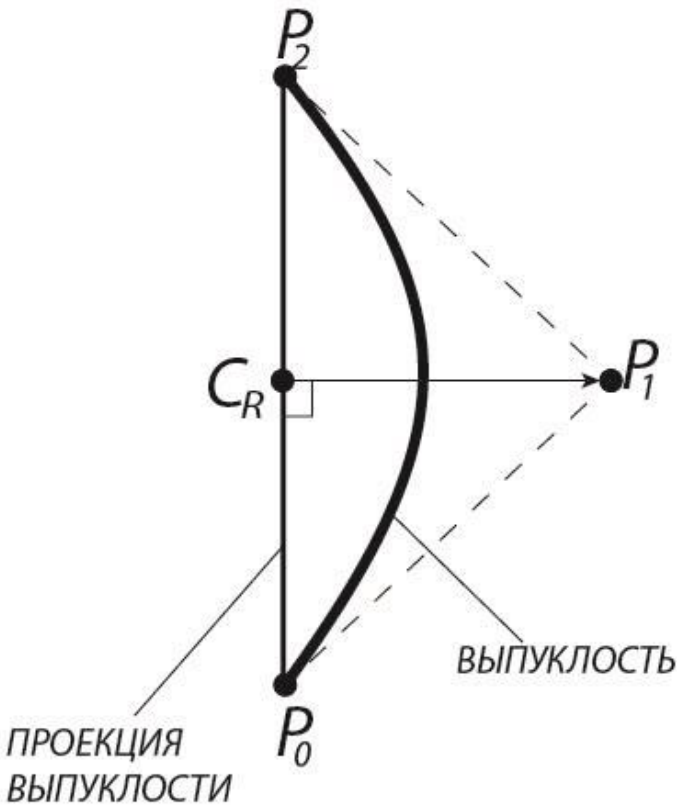


Рисунок 3.21 – Графический поиск промежуточной опорной вершины

Из рисунка 3.21 C_R – первоначальное положение промежуточной опорной вершины кривой Безье.

Согласно рисунку 3.21 поиск промежуточной опорной вершины (P_1) осуществляется из точки C_R вдоль перпендикуляра. Поиск вдоль перпендикуляра необходим для расчета распределения дополнительных петельных рядов по участку трикотажа.

Распределение дополнительных петельных рядов необходимо для получения выпуклости установленной формы. Центром распределения будет являться точка C_R.

Условием останова поиска будет являться соответствие значения длины, полученной кривой Безье, значению длины профиля выпуклости, рассчитанному по формуле 3.9.

Алгоритм формирования выпуклого участка трикотажа:

РАСЧЕТ ДЛИНЫ ПРОФИЛЯ ВЫПУКЛОГО УЧАСТКА ТРИКОТАЖА (ШАГ 1) >>> ОПРЕДЕЛЕНИЕ ФОРМЫ ВЫПУКЛОСТИ (ШАГ 2) >>> ПОИСК ПРОМЕЖУТОЧНОЙ ОПОРНОЙ ВЕРШИНЫ (ШАГ 3) >>> ФОРМИРОВАНИЕ ОТЧЕТА (ШАГ 4)

Математически алгоритм формирования выпуклого участка трикотажа примет следующий вид:

Пусть (x_{P0}, y_{P0}) – начальная опорная вершина кривой Безье (нижняя граница выпуклого участка трикотажа);

(x_{P1}, y_{P1}) – промежуточная опорная вершина кривой Безье;

(x_{P2}, y_{P2}) – конечная опорная вершина кривой Безье (верхняя граница выпуклого участка трикотажа);

(x_R, y_R) – точка центра распределения;

LV' – длина построенной кривой Безье в результате смещения промежуточной опорной вершины.

ШАГ 1

Рассчитаем длину профиля выпуклого участка трикотажа, воспользовавшись формулой 3.9:

$$LV = k * (\sum_{i=1}^{n} BO + \sum_{j=1}^{m} BD) \qquad (3.11)$$

Расчет данной длины необходим для моделирования выпуклого участка трикотажа при условии провязывания заданного количества дополнительных петельных рядов.

При формировании выпуклого участка трикотажа по заданному профилю выпуклости необходимо определить длину кривой Безье LV' по формулам (3.18)-(3.23). Далее необходимо определить количество дополнительных петельных рядов, воспользовавшись следующей формулой:

$$m = \frac{LV' - LV}{BD} \qquad (3.12)$$

ШАГ 2

Зададим координаты центра распределения.

В зависимости от расположения точки центра распределения C_R дополнительных петельных рядов на проекции выпуклости профиль выпуклого участка будет иметь различные виды (рисунки 3.22, 3.23).

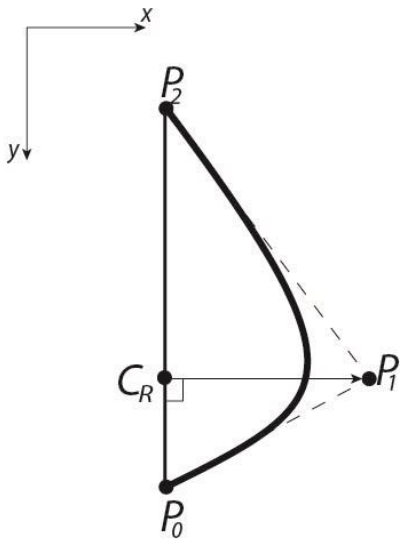


Рисунок 3.22 – Точка центра распределения смещена к начальной опорной вершине

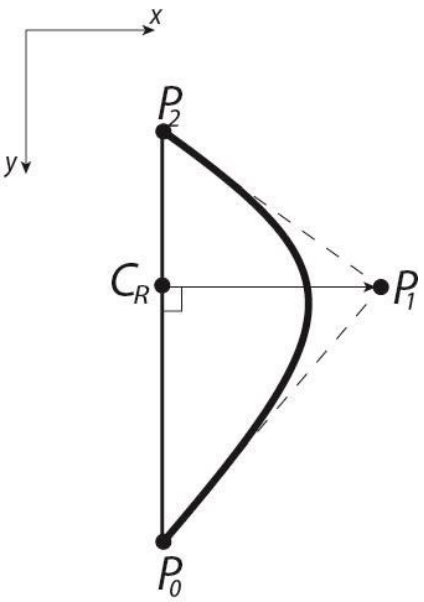


Рисунок 3.23 –Точка центра распределения смещена к конечной опорной вершине.

ШАГ 3

Поиск промежуточной опорной вершины осуществляется вдоль линии перпендикуляра, точкой старта поиска является точка центра распределения.

Уравнение линии проекции:

$$y = \frac{y_{P2} - y_{P0}}{x_{P2} - x_{P0}} * x - \frac{y_{P2} - y_{P0}}{x_{P2} - x_{P0}} * x_{P0} + y_{P0} \quad (3.13)$$

Уравнение линии, по которой выполняется поиск:

$$y = \frac{x_{P0} - x_{P2}}{y_{P2} - y_{P0}} * (x - x_R) + y_R \quad (3.14)$$

Тогда с каждым шагом поиска *dx*, получим текущие координаты промежуточной опорной вершины *(x_{P1}, y_{P1})*, где

$$x_{P1} = x_R + i * dx \quad (3.15)$$

где *i* – номер шага поиска;

i = 0, 1, 2 ... n;

dx – шаг поиска.

Рекомендуемая формула для расчета шага поиска:

$$dx = 0.01 * B, \quad (3.16)$$

где *B* – высота петельного ряда;

$$y_{P1} = \frac{x_{P0} - x_{P2}}{y_{P2} - y_{P0}} * dx * i + y_R \quad (3.17)$$

Определим длину, полученной кривой Безье, воспользовавшись следующим выражением:

$$LV' = \sum_{t=dt}^{1} \sqrt{(x_{B1} - x_{B2})^2 + (y_{B1} - y_{B2})^2}, \quad (3.18)$$

$$x_{B1} = (1 - t_1)^2 * x_{P0} + 2 * t_1 * (1 - t_1) * x_{P1} + {t_1}^2 * x_{P2}, \quad (3.19)$$

$$y_{B1} = (1 - t_1)^2 * y_{P0} + 2 * t_1 * (1 - t_1) * y_{P1} + {t_1}^2 * y_{P2}, \quad (3.20)$$

$$t_1 = t - dt, \quad (3.21)$$

где *dt* – шаг параметра t.

$$x_{B2} = (1 - t)^2 * x_{P0} + 2 * t * (1 - t) * x_{P1} + t^2 * x_{P2}, \quad (3.22)$$

$$y_{B2} = (1 - t)^2 * y_{P0} + 2 * t * (1 - t) * y_{P1} + t^2 * y_{P2} \quad (3.23)$$

Условие останова поиска:

$$LV' \geq LV \qquad (3.24)$$

В случае, формирования выпуклого участка трикотажа по заданному профилю необходимо определить положение центра распределения. Точка центра распределения будет лежать на пересечении линии проекции, формула (3.13), и линии по которой выполняется поиск, формула (3.14).

$$x_R = \frac{xx_1 - xx_2}{yy_1 - yy_2}, \qquad (3.25)$$

$$y_R = yy_1 * x_R - xx_1, \qquad (3.26)$$

где

$$xx_1 = \frac{y_{P2} - y_{P0}}{x_{P2} - x_{P0}} * x_{P0} + y_{P0}, \qquad (3.27)$$

$$xx_2 = \frac{x_{P0} - x_{P2}}{y_{P2} - y_{P0}} * x_{P1} + y_{P1}, \qquad (3.28)$$

$$yy_1 = \frac{y_{P2} - y_{P0}}{x_{P2} - x_{P0}}, \qquad (3.29)$$

$$yy_2 = \frac{x_{P0} - x_{P2}}{y_{P2} - y_{P0}} \qquad (3.30)$$

ШАГ 4

При формировании отчета моделирования выпуклого участка трикотажа необходимо выполнить расчет распределения дополнительных петельных рядов.

Проектирование технологии вязания выпуклого участка трикотажа в системе проектирования трикотажного производства выполняется согласно данным отчета.

Распределение, в зависимости от выбранной технологии провязывания дополнительных петельных рядов, может быть двух типов:

1. Провязывание дополнительных петельных рядов осуществляется при помощи одного нитеводителя;
2. Провязывание дополнительных петельных рядов осуществляется при помощи ввода дополнительного нитеводителя.

Способ вязания выпуклости с использованием двух нитеводителей представлен на рисунке 3.24. Данный способ подразумевает использование одного дополнительного нитеводителя на выпуклость, но это может привести к увеличению холостых ходов каретки и, следовательно, к увеличению времени вязания. Также к отрицательным характеристикам данного способа относится увеличенный расход пряжи, получаемый за счет длинных протяжек, формируемых при переходе от вязания одного дополнительного петельного рядя к вязанию другого дополнительного петельного ряда, а также за счет длинных протяжек, формируемых в момент включения и выключения нитеводителя, провязываемого дополнительные петельные ряды. На рисунке 3.24 изображена графическая схема технологии провязывания дополнительных петельных рядов при использовании двух нитеводителей в каретке, имеющей две петлеобразующие системы [1.1].

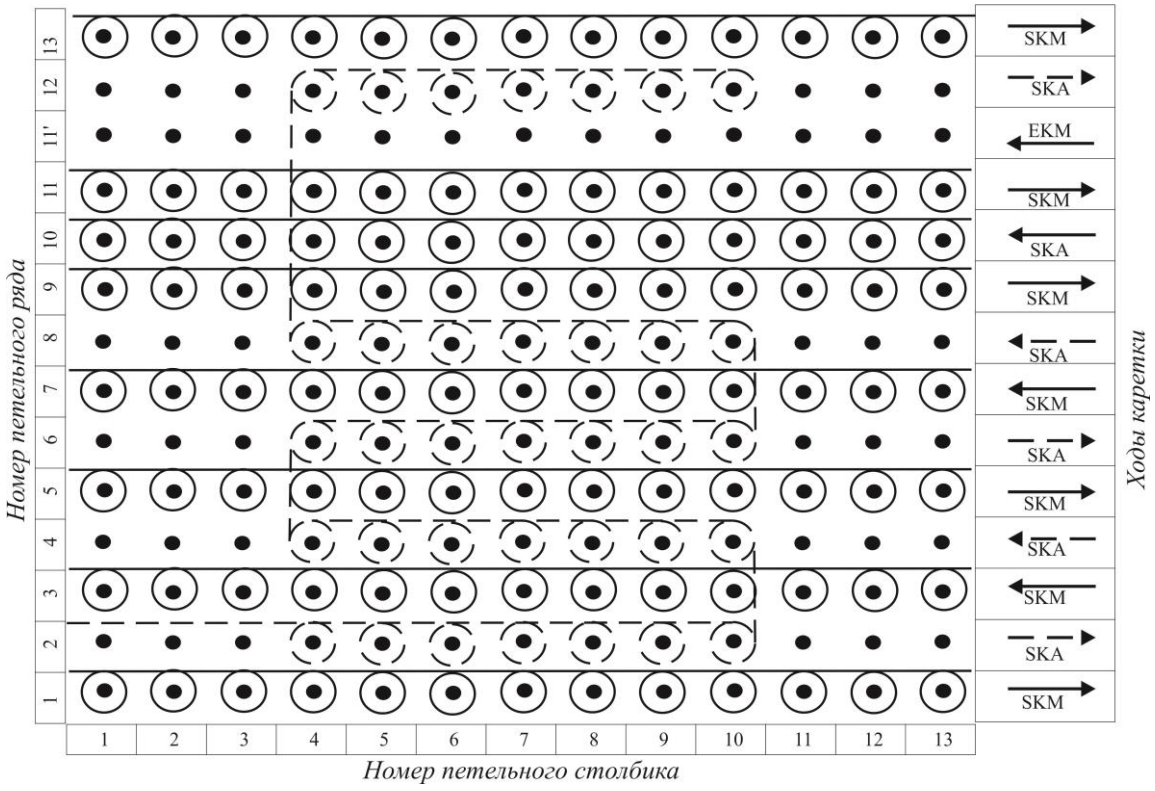


Рисунок 3.24 – Вязание выпуклого участка трикотажа с использованием дополнительного нитеводителя

Согласно рисунку 3.24:

- SKM и SKA – обозначают позицию ввода в начале вязания выпуклого участка трикотажа нитеводителя, провязывающего петельные ряды основного полотна и нитеводителя, провязывающего дополнительные петельные ряды соответственно, где КМ обозначает петельные ряды основного полотна, КА – дополнительные петельные ряды;
- EKM – холостой ход каретки.

Каждый раз при провязывании последующего дополнительного петельного ряда расход нити нитеводителя SKA увеличивается на величину равную, как минимум, величине высоты петельного ряда основного полотна, где данный расход будет зависеть от значений распределения дополнительных петельных рядов на участке выпуклости трикотажа.

После провязывания одиннадцатого петельного ряда, рассчитанное значение распределения дополнительных петельных рядов меняется, что говорит о нахождении на этом участке (9-11 петельные ряды) центра распределения дополнительных рядов, что приводит к появлению одного холостого хода каретки EKM перед началом вязания двенадцатого петельного ряда, являющимся дополнительным.

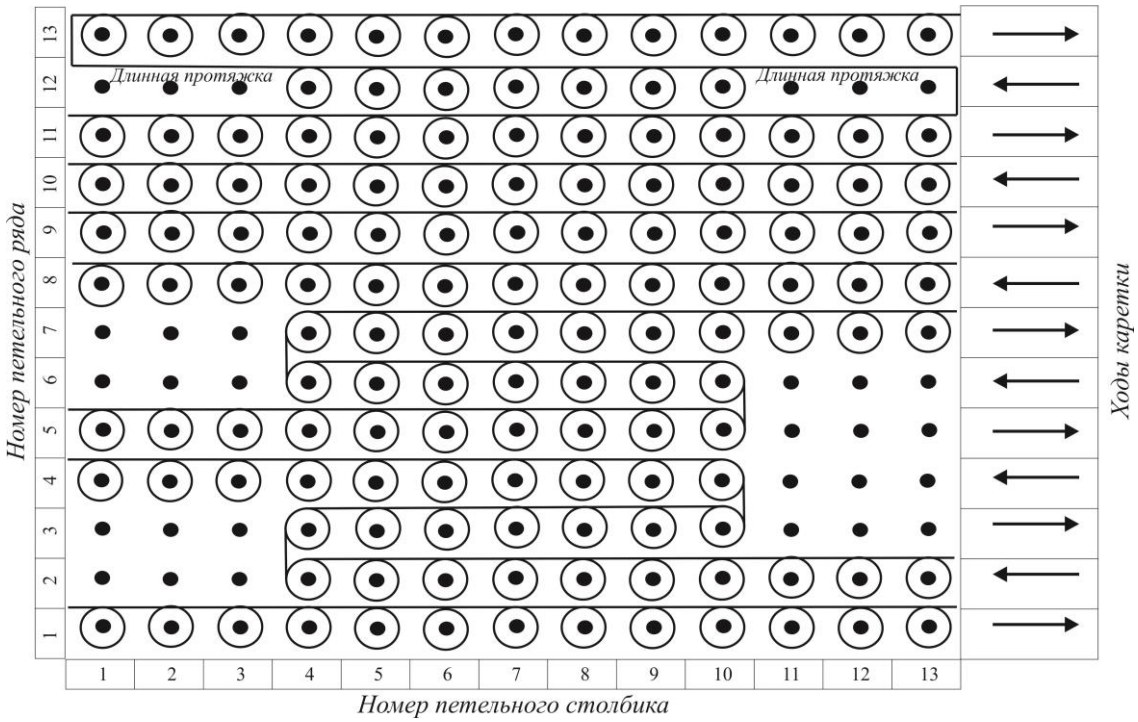


Рисунок 3.25 – Вязание выпуклого участка трикотажа без использования дополнительного нитеводителя

На рисунке 3.25 изображена графическая схема технологии провязывания дополнительных петельных рядов с использованием одного нитеводителя, также участвующего в вязании петельных рядов основного полотна. При таком способе вязания профиль, полученной выпуклости будет менее приближенным в сравнении со способом, предусматривающим использование двух нитеводителей. Это связано, прежде всего, с провязыванием участка дополнительных петельных рядов, содержащего в себе количество рядов больше 1, что необходимо для обеспечения условий, связанных с корректной установкой рабочего нитеводителя на начало вязания следующего участка петельных рядов основного полотна. Например, при провязывании одного дополнительного петельного ряда и дальнейшего перемещения нитеводителя к позиции начала вязания следующего участка петельных рядов основного полотна возникнет удлиненная протяжка, которая может вработаться в полотно, что также является недостатком. Кроме того при повороте нитеводителя при провязывании двух и более дополнительных рядов в месте поворота появляется отверстие (рисунок 3.26).

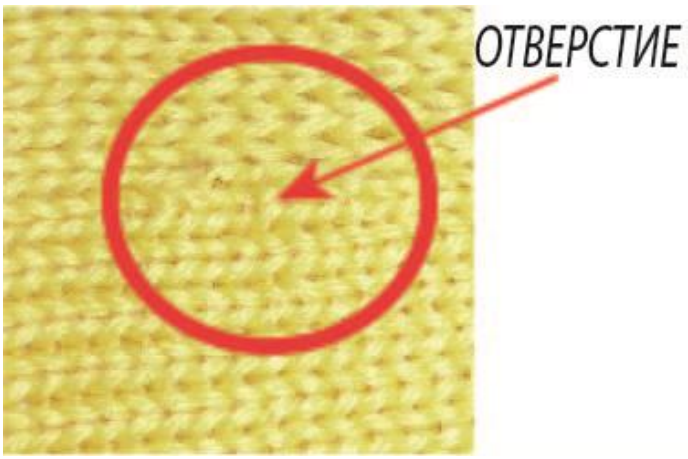


Рисунок 3.26 – Отверстие, получаемое при вязании выпуклости одним нитеводителем

Поэтому более предпочтительным при провязывании дополнительных петельных рядов является использование двух нитеводителей. Однако, нитеводитель дополнительных рядов должен быть интарзионным, чтобы основной нитеводитель мог пройти мимо, провязывая свой полный ряд.

Алгоритм распределения дополнительных петельных рядов по выпуклому участку трикотажа.

Распределение дополнительных петельных рядов выполняется от центра распределения в сторону границы участка, то есть в сторону начальной и конечной опорных вершин (рисунок 3.27).

Назовем верхним участком распределения участок от центра распределения до конечной опорной вершины, нижним участком распределения участок от центра распределения до начальной опорной вершины.

Плотностью петельных рядов назовем общее количество петельных рядов на участке распределения (P_0C_R’ и C_R’P_2 на рисунке 3.27). Плотность петельных рядов будет зависеть от величины высоты выпуклости на участке распределения.

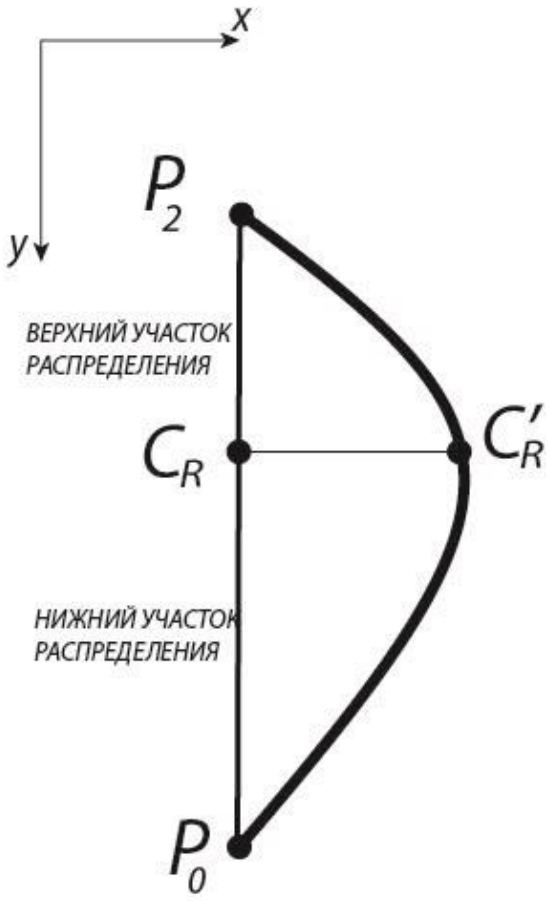


Рисунок 3.27 – Участки распределения

Определим количество дополнительных петельных рядов (*nl*) для нижнего участка распределения:

$$nl = Np * R, \qquad (3.30)$$

где *Np* – общее количество дополнительных петельных рядов на выпуклом участке трикотажа;

R – коэффициент распределения дополнительных петельных рядов, определяется согласно следующей формуле:

$$R = \frac{L(P_0C_R)}{L(P_0P_2)}, \qquad (3.31)$$

где $L(P_0C_R)$ – длина проекции P_0C_R нижнего участка распределения;

$L(P_0P_2)$ – длина проекции P_0P_2.

Определим количество дополнительных петельных рядов (nr) для верхнего участка распределения:

$$nr = Np - nl \qquad (3.32)$$

Для того чтобы выполнить распределение дополнительных петельных рядов по выпуклому участку трикотажа, необходимо определить положение центра распределения на трикотаже – *centr*.

$$centr = R * Nm + 1, \qquad (3.33)$$

где Nm – общее количество петельных рядов основного полотна на выпуклом участке трикотажа.

Чтобы не нарушать выпуклость участка, положение центра распределения примем между $(R * Nm + 1)$ петельным рядом и $(R * Nm + 2)$ петельными рядами.

Например, при расчете по формуле (3.33) получили, что *centr = 20*, тогда центр распределения будет находиться между 20 и 21 петельным рядом.

Далее определим количество пропускаемых петельных рядов *Rasl* и *Rasr*, через которое должен добавиться дополнительный петельный ряд. Движение распределения выполняется от центра распределения.

Для нижнего участка распределения:

$$Rasl = \frac{centr}{nl} \qquad (3.34)$$

Для верхнего участка распределения:

$$Rasr = \frac{Nm - centr}{nr} \qquad (3.35)$$

Распределение дополнительных петельных рядов по участкам распределения выполняется в бесконечном цикле. Поэтому в момент достижения конца участка распределения, процесс распределения начнется заново от центра распределения, но уже с учетом, распределенных дополнительных петельных рядов. Условием выхода из цикла распределения будет являться следующее выражение:

для нижнего участка распределения

$$nl = nl' \qquad (3.36)$$

для верхнего участка распределения

$$nr = nr', \qquad (3.37)$$

где *nl', nr'* – текущее значение количества распределённых дополнительных петельных рядов.

Однако при вязании выпуклого участка трикотажа с использованием одного нитеводителя распределение за один шаг должно выполняться по два петельных ряда, что необходимо для корректной доставки нитеводителя к началу вязания следующего петельного ряда и устранения длинной протяжки.

Тогда формула (3.34) и формула (3.35) примут следующий вид:

Для нижнего участка распределения:

$$Rasl = 2 * \frac{centr}{nl} \qquad (3.38)$$

Для верхнего участка распределения:

$$Rasr = 2 * \frac{Nm - centr}{nr} \qquad (3.39)$$

Условием выхода из цикла распределения будет являться следующее выражение:

для нижнего участка распределения

$$nl - 1 \leq nl' \qquad (3.40)$$

для верхнего участка распределения

$$nr - 1 \leq nr' \qquad (3.41)$$

Определим ширины дополнительных петельных рядов на выпуклом участке трикотажа.

Так как в формировании края выпуклого участка трикотажа используется кривая Безье, а отчет создается с учетом аппроксимированной развертки детали, то известна ордината верхней внешней точки ячейки (*ay*) основного петельного ряда, предшествующего дополнительному петельному ряду. Тогда необходимо

определить значения абсцисс точек, лежащих на краях выпуклого участка трикотажа.

Определим ординату для верхней внешней точки ячейки дополнительного петельного ряда:

$$ay' = ay - BD \qquad (3.42)$$

Так как при формировании края выпуклого участка трикотажа, координаты опорных вершин кривой Безье, формирующей край, известны, то определим параметр t, входящий в аналитическую запись кривой Безье, решая следующие уравнения относительно t:

для кривой Безье первой степени левого края:

$$ay' = (1 - tl) * yl0 + yl1 \qquad (3.43)$$

для кривой Безье второй степени левого края:

$$ay' = (1 - tl)^2 * yl0 + 2 * tl * (1 - tl) * yl1 + tl^2 * yl2 \qquad (3.44)$$

для кривой Безье третей степени левого края:

$$ay' = (1 - tl)^3 * yl0 + 3 * tl * (1 - tl)^2 * yl1 + 3 * tl^2 * (1 - tl) * yl2 + tl^3 * * yl3 \qquad (3.45)$$

Где *yl0, yl1, yl2, yl3* – значения ординат опорных вершин кривой Безье левого края;

$tl \in [0; 1]$.

для кривой Безье первой степени правого края:

$$ay' = (1 - tr) * yr0 + yr1 \qquad (3.46)$$

для кривой Безье второй степени правого края:

$$ay' = (1 - tr)^2 * yr0 + 2 * tr * (1 - tr) * yr1 + tr^2 * yr2 \qquad (3.47)$$

для кривой Безье третей степени правого края:

$$ay' = (1 - tr)^3 * yr0 + 3 * tr * (1 - tr)^2 * yr1 + 3 * tr^2 * (1 - tr) * yr2 + tr^3 * * yr3, \qquad (3.48)$$

где *yr0, yr1, yr2, yr3* – значения ординат опорных вершин кривой Безье правого края;

$tr \in [0; 1]$.

Рассчитав значения *tl* и *tr* для краев выпуклого участка трикотажа, определим значения абсцисс точек, лежащих на этих краях.

Для кривой Безье первой степени левого края:

$$ax' = (1 - tl) * xl0 + xl1 \quad (3.49)$$

Для кривой Безье второй степени левого края:

$$ax' = (1 - tl)^2 * xl0 + 2 * tl * (1 - tl) * xl1 + tl^2 * xl2 \quad (3.50)$$

Для кривой Безье третей степени левого края:

$$ax' = (1 - tl)^3 * xl0 + 3 * tl * (1 - tl)^2 * xl1 + 3 * tl^2 * (1 - tl) * xl2 + tl^3 * * xl3, \quad (3.51)$$

где *xl0, xl1, xl2, xl3* – значения абсцисс точек, лежащих на кривой Безье левого края.

Для кривой Безье первой степени правого края:

$$ax'' = (1 - tr) * xr0 + xr1 \quad (3.52)$$

Для кривой Безье второй степени правого края:

$$ax'' = (1 - tr)^2 * xr0 + 2 * tr * (1 - tr) * xr1 + tr^2 * xr2 \quad (3.53)$$

Для кривой Безье третей степени правого края:

$$ax'' = (1 - tr)^3 * xr0 + 3 * tr * (1 - tr)^2 * xr1 + 3 * tr^2 * (1 - tr) * xr2 + tr^3 * * xr3 \quad (3.54)$$

Тогда ширина дополнительного петельного ряда будет равна:

$$Na(ay') = \frac{ax'' - ax'}{A}, \quad (3.54)$$

где *A* – ширина петельного столбика;

Na(ay') – количество петельных столбиков в дополнительном петельном ряду при *ay'*.

Использование данного алгоритма в проектировании рельефа позволит получить выпуклый участок трикотажа любой сложной формы за счет провязывания дополнительных петельных рядов при использовании одного или двух нитеводителей, участвующих в вязании.

Решим обратную задачу.

Рассмотрим пример моделирования выпуклого участка трикотажа при заранее известном количестве дополнительных петельных рядов.

Пусть (x_R,y_R) – координаты центра распределения;

(x_0,y_0) – координаты начальной опорной вершины кривой Безье;

(x_1,y_1) – координаты промежуточной опорной вершины кривой Безье;

(x_2,y_2) – координаты конечной опорной вершины кривой Безье;

B – высота петельного ряда;

N_P – общее количество дополнительных петельных рядов.

Зададим значения координат начальной и конечной опорной вершины кривой Безье.

Пусть $x_0 = 2;\ y_0 = 14;\ x_2 = 2;\ y_2 = 2;$

$B = 0{,}14$ *см.*

Предположим, что необходимо смоделировать выпуклый участок трикотажа, высота (LO) которого равна 12 см за счет провязывания 20 дополнительных петельных рядов.

ШАГ 1

Рассчитаем длину профиля выпуклого участка трикотажа, воспользовавшись формулой (3.11):

$$LV = k * (\sum_{i=1}^{n} BO + \sum_{j=1}^{m} BD)$$

Примем $BO = BD = 0{,}14$ *см.*

n – общее количество основных петельных рядов на выпуклом участке трикотажа;

$$n = \frac{LO}{B} = \frac{12{,}00}{0{,}14} \approx 86 \text{ (петельных рядов)};$$

$$m = N_P = 20 \text{ (петельных рядов)};$$

m – количество дополнительных петельных рядов на выпуклом участке трикотажа.

Примем усадку основным и единственным фактором, влияющим на величину выпуклости после вязания и прохождения влажно-тепловой обработки.

Влажно-тепловую обработку выполним без дополнительного формирования профиля выпуклости.

Для вязания выпуклого участка трикотажа используем вязальную машину фирмы STOLL CMS 340 TC-KW класса 7.2 и хлопчатобумажную пряжу 26*2 Текс, тогда согласно таблице 2:

$$k = 0{,}945;$$

Тогда

$$LV = 0.945 * (\sum_{i=1}^{86} 0.14 + \sum_{j=1}^{20} 0.14) = 14.02 \text{ см};$$

ШАГ 2

Зададим координаты центра распределения:

$x_R = 2;$

$y_R = 8;$

ШАГ 3

По формуле (3.14) получим уравнение линии, по которой будет выполняться поиск координат промежуточной опорной вершины кривой Безье.

$$y = \frac{2-2}{14-2} * (x-2) + 8;$$

$$y = 8;$$

Так как с каждым шагом поиска значение ординаты промежуточной опорной вершины будет постоянным, то $y_1 = 8$.

Тогда по формуле (3.15) определим временное значение абсциссы промежуточной опорной вершины кривой Безье.

$$x_1 = x_R + i * dx,$$

$$dx = 0.01 * 0.14 = 0{,}0014 \text{ см};$$

При $i = 1$

$$x_1 = 2 + 1 * 0{,}0014 = 2{,}0014;$$

Определим длину (*LV'*), полученной кривой Безье, воспользовавшись формулами (3.17)-(3.23), получим:

$$LV' = 12{,}0001 \text{ см};$$

Так как *LV'* < *LV*, то продолжим поиск.

В результате дальнейших вычислений получим значение x_1 = *8,44*, при котором *LV' = LV = 14,02*.

Таким образом, кривая Безье, представляющая собой выпуклый участок трикотажа, выпуклость которого получена за счет провязывания двадцати дополнительных петельных рядов, будет иметь следующую аналитическую запись:

$$\begin{cases} x(t) = 2{,}00 * (1-t)^2 + 16{,}88 * t * (1-t) + 2{,}00 * t^2, \\ y(t) = 14{,}00 * (1-t)^2 + 16{,}00 * t * (1-t) + 2{,}00 * t^2; \end{cases}$$

Графически данная кривая Безье представлена на рисунке 3.28.

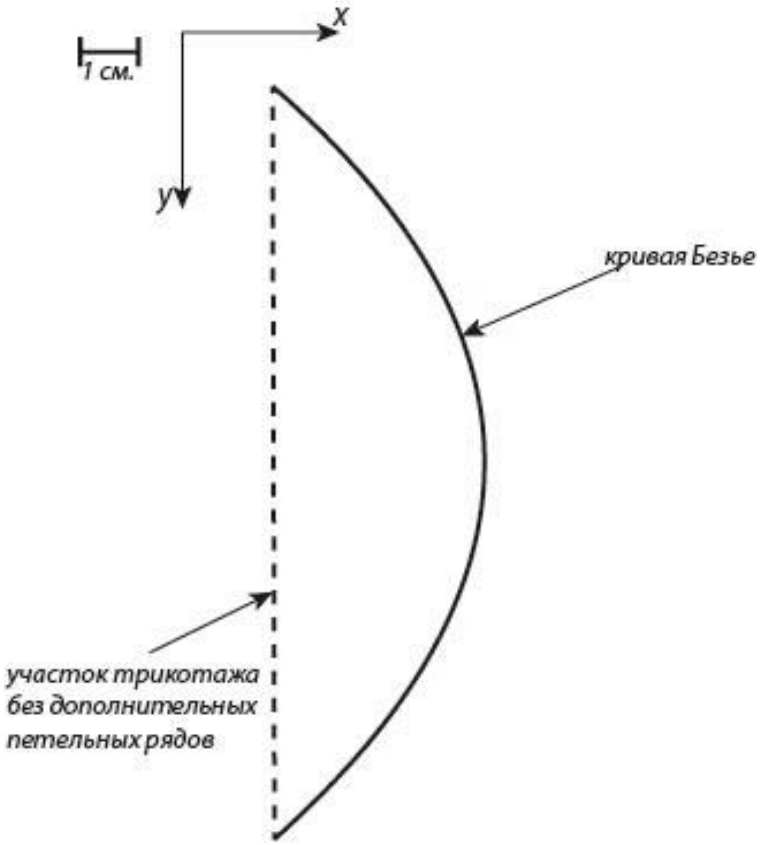


Рисунок 3.28 – кривая Безье второй степени

ШАГ 4

Сформируем два типа отчета:

1. Отчет с учетом вязания выпуклого участка трикотажа с использованием двух нитеводителей;
2. Отчет с учетом вязания выпуклого участка трикотажа с использованием одного нитеводителя.

Сформируем отчет для двух нитеводителей.

По формуле (3.31):

$$R = \frac{6}{12} = 0{,}5;$$

Тогда

$$nl = 20{,}0 * 0{,}5 = 10{,}0 \text{ (петельных рядов)};$$

$$nr = 20 - 10 = 10 \text{ (петельных рядов)};$$

Определим расположение центра распределения непосредственно на участке трикотажа:

$$centr = 0{,}5 * 86{,}0 + 1{,}0 = 44{,}0;$$

Тогда центр распределения будет находиться между 44 и 45 петельными рядами.

Определим число основных петельных рядов для нижнего участка распределения, через которое будет провязываться дополнительный петельный ряд:

$$Rasl = \frac{43}{10} \approx 4 \text{ (петельных ряда)};$$

Таким образом, для нижнего участка распределения через каждые четыре петельных ряда основного полотна необходимо провязать один дополнительный петельный ряд.

Определим число основных петельных рядов для верхнего участка распределения, через которое будет провязываться дополнительный петельный ряд:

$$Rasr = \frac{43}{10} \approx 4 \text{ (петельных ряда)};$$

Таким образом, для верхнего участка распределения через каждые четыре петельных ряда основного полотна необходимо провязать один дополнительный петельный ряд.

Представим результаты распределения в виде таблицы (таблица 3).

Таблица 3 – Отчет распределения дополнительных петельных рядов при использовании в вязании двух нитеводителей

Ряд	Кол-во доп. рядов
4	1
8	1
12	1
16	1
20	1
24	1
28	1
32	1
36	1
40	1
84	1
80	1
76	1
72	1
68	1
64	1
60	1
56	1
52	1
48	1

Полученный в результате распределения выпуклый участок трикотажа представлен на рисунке 3.29.

Рисунок 3.29 – Выпуклый участок трикотажа, связанный с использованием двух нитеводителей

Сформируем отчет для одного нитеводителя.

$$Rasl = 2 * \frac{43}{10} \approx 9 \text{ (петельных рядов)};$$

Таким образом, для нижнего участка распределения через каждые девять петельных рядов основного полотна необходимо провязать два дополнительных петельных ряда.

$$Rasr = 2 * \frac{43}{10} \approx 9 \text{ (петельных рядов)};$$

Таким образом, для верхнего участка распределения через каждые девять петельных рядов основного полотна необходимо провязать два дополнительных петельных ряда.

Представим результаты распределения в виде таблицы (таблица 4).

Таблица 4 – Отчет распределения дополнительных петельных рядов при использовании в вязании одного нитеводителя

Ряд	Кол-во доп. рядов
9	2
18	2
27	2
36	2
44	2
45	4
79	2
70	2
61	2
52	2

Полученный в результате распределения выпуклый участок трикотажа представлен на рисунке 3.30.

Рисунок 3.30 – Выпуклый участок трикотажа, связанный с использованием одного нитеводителя

Как видим, смоделированный профиль выпуклого участка трикотажа, представленный на рисунке 3.28 и профили связанных выпуклых участков трикотажа (рисунки 3.29 и 3.30) совпадают.

Таким образом, использование способа получения выпуклого участка трикотажа за счет провязывания дополнительных петельных рядов, а также использование способа его проектирования при помощи кривой Безье, позволит:

- Выполнить моделирование выпуклого участка трикотажа с максимально приближенным профилем к профилю связанного участка;
- Выполнить вязание выпуклости при использовании одного или двух нитеводителей;
- Улучшить посадку готового цельновязаного изделия за счет сформированного рельефа;

Ширина дополнительных петельных рядов в данном случае была одинаковой, что не повлияло на получение выпуклости заданной формы.

Ограничим края выпуклости, представленной на рисунке 3.29, кривыми Безье третей степени. Зададим значения координат промежуточных опорных вершин для кривой Безье левого и правого края.

Для левого края:

xll0 = 10,0;

yll0 = 14,0;

xll1 = 1.6;

yll1 = 13.4;

xll2 = 1.6;

yll2 = 3.6;

xll3 = 7.4;

yll3 = 2,0;

Для правого края:

xrr0 = 10,0;

yrr0 = 14,0;

xrr1 = 25,0;

yrr1 = 13,8;

xrr2 = 27,0;

yrr2 = 6,0;

xrr3 = 12,7;

yrr3 = 1,9;

Сформируем аналитические записи кривых Безье.

Для левого края:

$$\begin{cases} x(t) = 10{,}0 * (1 - tl)^3 + 4{,}8 * tl * (1 - tl)^2 + 4{,}8 * tl^2 * (1 - tl) + 7{,}4 * tl^3, \\ y(t) = 14{,}0 * (1 - tl)^3 + 40{,}2 * tl * (1 - tl)^2 + 10{,}8 * tl^2 * (1 - tl) + 2{,}0 * tl^3; \end{cases}$$

$$tl \in [0; 1];$$

Для правого края:

$$\begin{cases} x(t) = 10{,}0 * (1 - tr)^3 + 75{,}0 * tr * (1 - tr)^2 + 81{,}0 * tr^2 * (1 - tr) + 12{,}7 * tr^3, \\ y(t) = 14{,}0 * (1 - tr)^3 + 41{,}4 * tr * (1 - tr)^2 + 18{,}0 * tr^2 * (1 - tr) + 1{,}9 * tr^3; \end{cases}$$

$$tr \in [0; 1];$$

Согласно таблице 3, дополнительный петельный ряд провязывается после четвертого основного петельного ряда, а высота петельного ряда B = 0.14 см, тогда

$$ay' = ay'' - (rd + 1) * B = 14{,}00 - (4{,}00 + 1{,}00) * 0{,}14 = 13{,}30;$$

где $ay'' = yll0 = yrr0 = 14{,}00$;

$rd -$ номер основного петельного ряда;

$(rd + 1) -$ необходимо для получения значения ординаты верхней внешней точки дополнительного петельного ряда.

Определим значения абсцисс точек с ординатами *ay'*, лежащих на краях выпуклости.

Для левого края.

Определим значение параметра *tl*:

$$14{,}0 * (1 - tl)^3 + 40{,}2 * tl * (1 - tl)^2 + 10{,}8 * tl^2 * (1 - tl) + 2{,}0 * tl^3 = 13{,}3;$$

Решая данное уравнение относительно *tl*, получим

$$tl = 0{,}0402;$$

Тогда

$$ax' = 10{,}0000 * (1{,}0000 - 0{,}0402)^3 + 4{,}8000 * 0{,}0402 * (1{,}0000 - 0{,}0402)^2 + 4{,}8000 * 0{,}0402^2 * (1{,}0000 - 0{,}0402) + 7{,}4000 * 0{,}0402^3 \approx 8{,}55;$$

Для правого края.

Определим значение параметра *tr*:

$$14{,}0 * (1{,}0 - tr)^3 + 41{,}4 * tr * (1{,}0 - tr)^2 + 18{,}0 * tr^2 * (1{,}0 - tr) + 1{,}9 * tr^3 = 13{,}3;$$

Решая данное уравнение относительно *tr*, получим

$$tr = 0{,}06195;$$

Тогда

$$ax'' = 10{,}00000 * (1{,}00000 - 0{,}06195)^3 + 75{,}00000 * 0{,}06195 *$$

$$* (1{,}00000 - 0{,}06195)^2 + 81{,}00000 * 0.06195^2 * (1{,}00000 -$$

$$-0{,}06195) + 12{,}70000 * 0{,}06195^3 \approx 12{,}64000;$$

По формуле (3.54) определим количество петельных столбиков дополнительного петельного ряда, провязываемого после четвертого основного петельного ряда выпуклого участка трикотажа.

Примем ширину петельного столбика A = 0,17 см, тогда

$$Na = \frac{12{,}64 - 8{,}55}{0{,}17} \approx 24{,}00 \text{ (петельных столбика)};$$

Выполним аналогичные расчеты для остальных дополнительных петельных рядов таблицы 3. Результаты представим в виде сводной таблицы (таблица 5).

Таблица 5 – Отчет распределения дополнительных петельных рядов при использовании в вязании одного нитеводителя

Ряд	Кол-во доп. рядов	Ширина (столбики)
4	1	24
8	1	35
12	1	44
16	1	50
20	1	54
24	1	58
28	1	60
32	1	62
36	1	64
40	1	64
84	1	39
80	1	45

Продолжение таблицы 5

Ряд	Кол-во доп. рядов	Ширина (столбики)
76	1	50
72	1	54
68	1	57
64	1	60
60	1	62
56	1	64
52	1	64
48	1	65

Полученный в результате распределения выпуклый участок трикотажа представлен на рисунке 3.31.

Рисунок 3.31 – Выпуклый участок трикотажа с непрямыми краями

Полученный выпуклый участок трикотажа будет иметь одинаковую выпуклость с выпуклостями, представленными на рисунках 3.29 и 3.30. Использование данного типа выпуклости оптимально для зон выпуклости, имеющих небольшую площадь и большую выпуклость (например, локтевой участок рукава, плечевой участок).

Использование вывязанной выпуклости на плечевом участке изделия позволит улучшить посадку изделия.

Обычно при размножении размерного ряда изделий меняется ширина и высота изделия. Естественно при изменении размера изделия, например с 46 до 56, измениться и размер плечевого участка. Поэтому использование методики расчета технологии вязания плечевого участка для каждого размера позволит улучшить качество изделий и их посадку.

3.2.2 Разработка способа проектирования выпуклого участка трикотажа за счет замены швейной вытачки на частичное вязание

В швейном изделии основными конструктивными элементами, позволяющими сформировать рельеф, являются вытачки.

Так как суть вытачки заключается в формировании рельефа, то есть получения выпуклого участка, то профиль данного участка можно представить в виде кривой Безье второй степени.

На рисунке 3.32 представлена вертикальная вытачка в открытом виде.

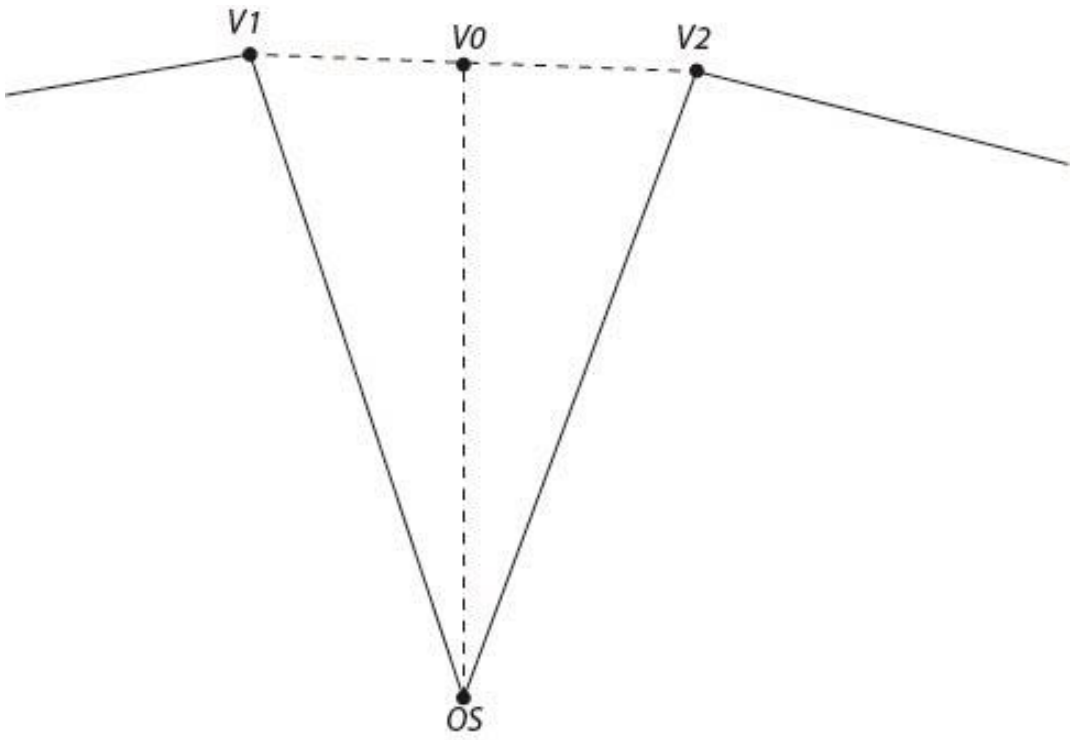


Рисунок 3.32 – Открытая вертикальная вытачка

Согласно рисунку 3.32:

V1, V2 – точки основания вытачки, где расстояние (V1V2) между этими точками является раствором вытачки;

OS – вершина вытачки;

V0 – точка закрытия вытачки;

V0OS – длина вытачки;

OSV1 = OSV2.

На рисунке 3.33 представлена вытачка в закрытом виде.

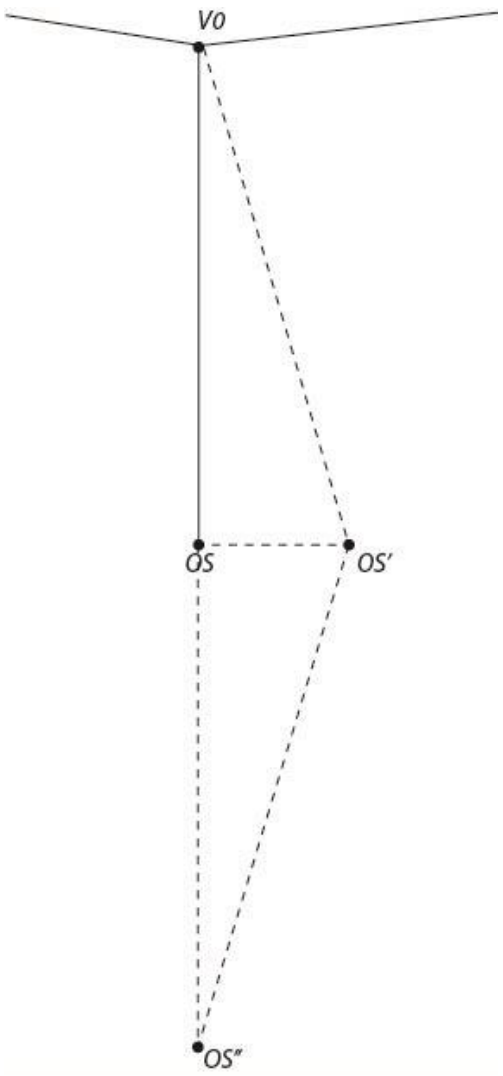


Рисунок 3.33– Закрытая вертикальная вытачка

При закрытии вытачки точка *OS'* (рисунок 3.33) будет являться вершиной выпуклого участка. Причем длина выпуклого участка *Lvyt* будет равна:

$$Lvyt = 2 * V0OS', \qquad (3.54)$$

где ломаная кривая V0OS'OS'' является профилем выпуклого участка, а V0OS'' – участком до закрытия вытачки.

При закрытии вытачки швейным способом с изнаночной стороны изделия прокладывается швейная строчка, в результате чего может быть образован угол V0OS'OS'' на самом изделии, что в некоторых случаях может являться фактором, ухудшающим внешний вид изделия, а также локально разрушающим целостность структуры полотна. Кроме того закрытие вытачки добавляет в технологическую

цепочку дополнительный этап обработки изделия и требует дополнительного расхода сырья, площадей и трудозатрат.

Поэтому замена швейной вытачки на геометрически аналогичную выпуклость, получаемую за счет провязывания дополнительных петельных рядов, позволит:

- сформировать рельеф на цельновязаном изделии, не нарушая целостность структуры трикотажа, и не ухудшая внешний вид изделия;
- исключить из последовательности технологической обработки изделия этап закрытия вытачек и сократить расход сырья;
- использовать методики конструирования, предназначенные для проектирования швейных изделий;
- улучшить посадку изделия;
- расширить ассортиментный ряд.

Суть операции замены швейной вытачки состоит в построении кривой Безье, представляющей собой профиль выпуклого участка и выполнения последующего расчета необходимого количества дополнительных петельных рядов.

Поэтому алгоритм замены вытачки примет следующий вид:

ФОРМИРОВАНИЕ ГРАНИЦ ВЫПУКЛОГО УЧАСТКА (ШАГ 1) >>> ПОСТРОЕНИЕ КРИВОЙ БЕЗЬЕ (ШАГ 2) >>> РАСЧЕТ НЕОБХОДИМОГО КОЛИЧЕСТВА ДОПОЛНИТЕЛЬНЫХ ПЕТЕЛЬНЫХ РЯДОВ (ШАГ 3) >>> ФОРМИРОВАНИЕ ОТЧЕТА (ШАГ 4)

Математически алгоритм замены вытачки примет следующий вид:

Примем положительное направление оси ординат сверху вниз, оси абсцисс – слева направо.

ШАГ 1

Определим координаты точки верхней границы профиля выпуклости – точка V0 на рисунке 3.33.

Так как точка V0 находится на одинаковом расстоянии от точек основания вытачки (точки V1 и V2 на рисунке 3.32), то

$$x_{V0} = x_{V1} + \frac{x_{V2} - x_{V1}}{2}, \qquad (3.55)$$

$$y_{V0} = y_{V1} + \frac{y_{V2} - y_{V1}}{2} \qquad (3.56)$$

Определим координаты нижней границы профиля выпуклости.

$$x_{OS''} = 2 * x_{OS} - x_{V0}, \qquad (3.57)$$

$$y_{OS''} = 2 * y_{OS} - y_{V0} \qquad (3.58)$$

Точками верхней и нижней границы, получаемого профиля выпуклости, за счет закрытия вытачки горизонтального типа будут являться точки её основания. На рисунке 3.34 данными точками являются точки *V1* и *V2*.

ШАГ 2

Пусть

$x_0 = x_{OS''}$;

$y_0 = y_{OS''}$;

$x_2 = x_{V0}$;

$y_2 = y_{V0''}$;

где $P_0(x_0, y_0)$, $P_2(x_2, y_2)$, $P_1(x_1, y_1)$ – координаты опорных вершин кривой Безье.

$x_R = x_{OS}$;

$y_R = y_{OS}$;

$C_R(x_R, y_R)$ – координаты центра распределения дополнительных петельных рядов.

Для вытачки горизонтального типа (рисунок 3.34):

$x_0 = x_{V1}$;

$y_0 = y_{V1''}$;

$x_2 = x_{V2}$;

$y_2 = y_{V2}$;

$x_R = x_{V0}$;

$y_R = y_{V0}$;

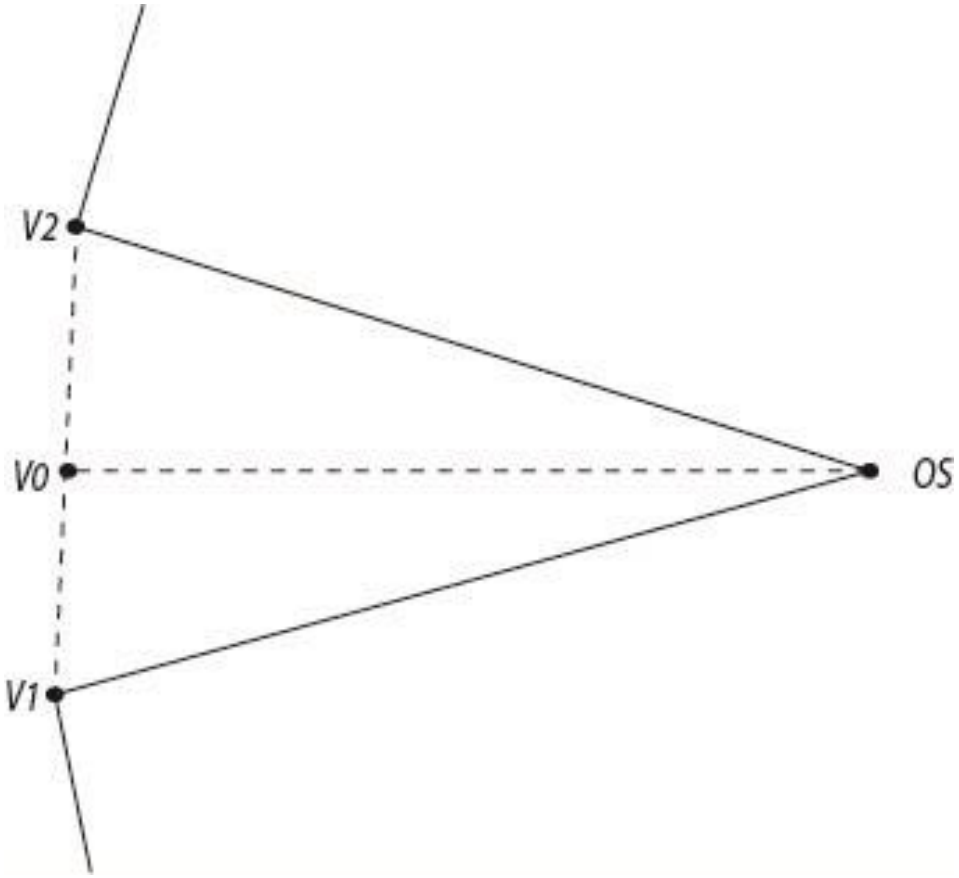


Рисунок 3.34 – Открытая вытачка горизонтального типа

Так как на рисунке 3.33 длина участка V0OS равна длине участка OSOS'', то коэффициент распределения дополнительных петельных рядов $R = 0{,}5$.

С учетом принятых изменений преобразуем рисунок 3.33 (рисунок 3.35).

При закрытой вытачке длина профиля выпуклости будет равна:

$$LV = 2 * L(V0OS'), \qquad (3.59)$$

где *L(V0OS')* – длина отрезка *V0OS'*.

Для вытачки горизонтального типа (рисунок 3.36):

$$LV = 2 * L(V1OS'), \qquad (3.59')$$

где *L(V1OS')* – длина отрезка *V1OS'*.

Так как длина отрезка *V1OS'* равна длине отрезка *V2OS'*.

Высота выпуклости будет равна половине раствора вытачки.

Примем за угол α угол C_RP_2OS' из рисунка 3.36.

Высота выпуклого участка:

$$VS = L(P_2P_0), \qquad (3.60)$$

где $L(P_2P_0)$ – длина проекции кривой Безье, представляющей профиль выпуклого участка трикотажа;

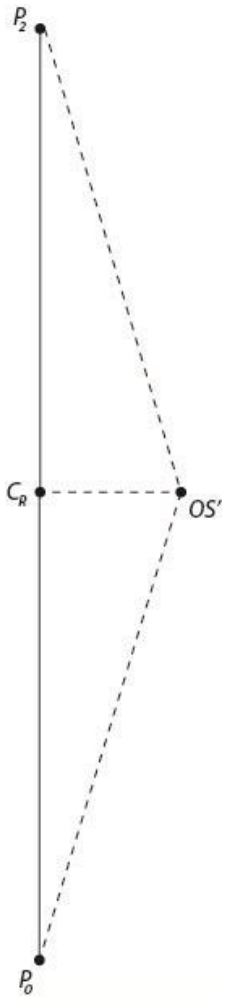


Рисунок 3.35 – Профиль выпуклого участка трикотажа при закрытой вертикальной вытачке

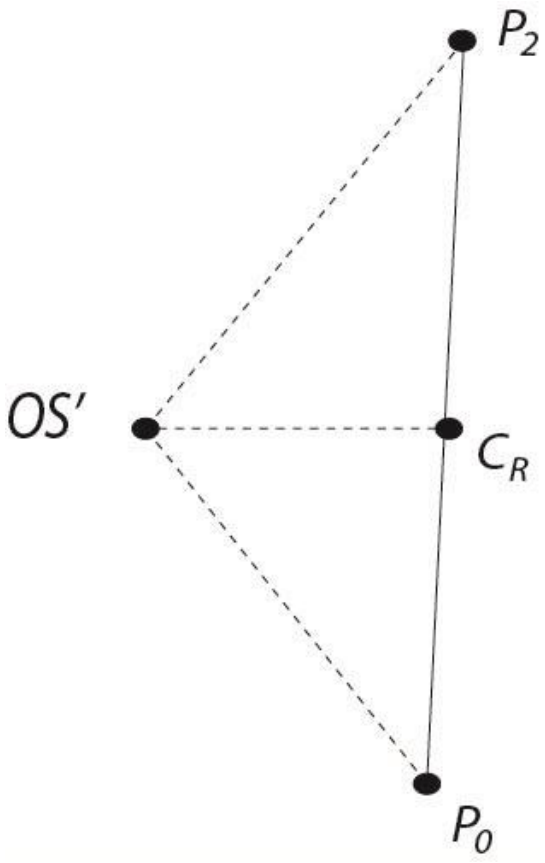


Рисунок 3.36 – Профиль выпуклого участка трикотажа при закрытой горизонтальной вытачке

$$\alpha = arctg\left(\frac{Rast}{VS}\right), \qquad (3.61)$$

$$L(V0OS') = \frac{VS}{\cos(\alpha)}, \qquad (3.62)$$

где *Rast* – величина раствора вытачки.

Подставляя формулу (3.62) в формулу (3.59), получим:

$$LV = 2 * \frac{VS}{\cos(\alpha)} \qquad (3.63)$$

Построение кривой Безье выполняется согласно формулам (3.11)-(3.21).

ШАГ 3

$$Np = \frac{LV}{B}, \qquad (3.64)$$

где Np – общее количество дополнительных петельных рядов на участке выпуклости;

B – высота петельного ряда.

Ширина дополнительного петельного ряда можно определить исходя из типа вытачки.

Для вертикальной вытачки ширина дополнительного петельного ряда (количество петельных столбиков) определяется следующей формулой:

$$Nv = Nu - Nt, \qquad (3.65)$$

где Nv – количество петельных столбиков дополнительного петельного ряда;

Nu – количество петельных столбиков основного петельного ряда;

Nt – количество петельных столбиков ограничения. Например, к данному ограничению можно отнести часть трикотажа, используемую для технологического припуска.

Рекомендуемое значение технологического припуска составляет от 0,5 до 1,0 см, поэтому, в зависимости от значения ширины петельного столбика, Nt для каждого полотна будет различно.

Как было установлено раннее: выпуклые участки трикотажа, имеющие одинаковое распределение, но разные формы границ выпуклых участков, будут иметь одинаковую выпуклость. Однако, построение границ выпуклого участка с использованием кривых Безье второй и третей степеней, позволит получить более качественную посадку изделия, чем с границами, являющимися отрезками прямых линий. Поэтому для построения границ выпуклого участка необходимо учитывать особенности локальной области первичного рельефа (если изделие предназначается для эксплуатации человеком, то первичным рельефом будет являться его тело). Причем преобразование форм границ, с целью улучшения посадки, необходимо осуществлять за счет интерполирования необходимого

участка первичного рельефа с использованием различных технических средств сканирования.

При замене вытачки горизонтального типа, края выпуклого участка трикотажа будут образовываться за счет продолжения стенок вытачки, до пересечения с нижней или верхней границей выпуклого участка трикотажа (рисунок 3.37).

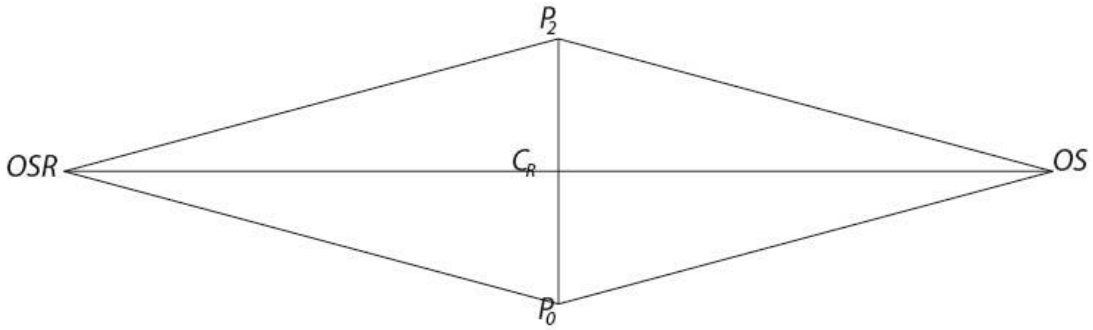


Рисунок 3.37 – Выпуклая область, образованная за счет закрытия горизонтальной вытачки

Согласно рисунку 3.37 границами выпуклого участка трикотажа, формируемого за счет закрытия вытачки, горизонтального типа будут являться кривые Безье первой степени (прямые отрезки): отрезок OSP_2, отрезок P_2OSR, отрезок $OSRP_0$, отрезок P_0OS, где длина отрезка OSC_R в два раза больше длины вытачки, а длина отрезка $OSRC_R$ равна длине отрезка OSC_R.

Так как координаты вершины (OS) вытачки задаются, то

$$x_R = x_{OS} \mp 2 * L(V0OS), \qquad (3.66)$$

$$y_R = y_{OS}, \qquad (3.67)$$

где $L(V0OS)$ – длина горизонтальной вытачки;

y_{OS} – ордината вершины вытачки;

x_{OS} – абсцисса вершины вытачки;

“-“ – вершина вытачки (OS) относительно точек основания (V1, V2 из рисунка 3.34) находится справа;

“+“ – вершина вытачки (OS) относительно точек основания (V1, V2 из рисунка 3.34) находится слева;

P_0, P_2 – начальная и конечная опорные вершины кривой Безье, представляющей профиль выпуклого участка трикотажа.

$$x_{OSR} = x_{OS} \mp 4 * L(V0OS), \qquad (3.68)$$

$$y_{OSR} = y_{OS} \qquad (3.69)$$

При построении выпуклого участка трикотажа с учетом особенностей первичного рельефа, кривые Безье первой степени, являющимися границами участка, можно заменить на кривые Безье второй и третей степени, что позволит получить более качественную посадку.

ШАГ 4

Процесс формирования отчета аналогичен ШАГУ 4 в алгоритме формирования выпуклого участка трикотажа.

В результате выполнения алгоритма получим выпуклость, сформированную за счет дополнительных петельных рядов, геометрически аналогичную с выпуклостью, получаемой в результате закрытия швейной вытачки.

Таким образом, данный способ замены швейной вытачки на аналогично выпуклый участок с провязанными дополнительными петельными рядами сводится к построению кривой Безье, представляющей собой профиль выпуклости, и дальнейшему расчету, по раннее приведенному алгоритму формирования выпуклого участка трикотажа.

Для улучшения качества цельновязаных изделий, для получения хорошей посадки на теле человека необходимо использовать технологии вывязывания выпуклых участков.

Для процесса проектирования технологии вязания цельновязаных изделий с выпуклыми участками целесообразно использовать разработанные технологии с использованием дополнительных рядов.

С целью ускорения процесса проектирования технологии вязания цельновязаных изделий с улучшенной посадкой требуется разработать специальную программу, результаты работы которой можно будет импортировать в САПР, используемую для разработки программы вязания, например, в САПР Stoll M1.

ВЫВОДЫ ПО ГЛАВЕ 3

1. Разработана технология балансирования-выравнивания числа элементов соединения по кромкам соединяемых деталей, обеспечивающих получение качественного узла соединения.
2. Предложен алгоритм нахождения линии в узлах соединения цельновязаных изделий, до которой число элементов соединения на кромках соединяемых деталей одинаково и после которой требуется выполнять балансирование элементов.
3. Предложены технология и метод расчета формирования выпуклых участков на трикотажных изделиях за счет провязывания дополнительных петельных рядов, позволяющих обеспечить необходимый рельеф для качественной посадки изделия.
4. Экспериментально определен коэффициент, корректирующий выпуклость участка трикотажа для кулирной глади, ластика и интерлока. Данный коэффициент учитывает свойства переплетений, физико-механические свойства пряжи и изменения, получаемые трикотажем после термической обработки, а также в процессе эксплуатации изделия.
5. Разработаны технологии получения выпуклостей при использовании одного и двух нитеводителей.
6. Установлено, что при вязании дополнительных петельных рядов с использованием двух нитеводителей, требуется применение специальных интарзионных нитеводителей.
7. Установлено, что более рациональным является способ получения выпуклостей при использовании одного нитеводителя. Этот способ позволяет исключить в технологическом процессе вязания холостые ходы и сократить расход пряжи на дополнительные петельные ряды, за счет уменьшения длин протяжек при переходе от вязания одного петельного ряда к другому, а также исключения протяжек, образуемых при включении и выключении нитеводителей.

8. Установлено, что получаемый при вывязывании радиус выпуклости определяется числом дополнительных петельных рядов и их чередованием с петельными рядами основного вязания и не зависит от изменения их ширины.
9. Установлено, что технология провязывания дополнительных петельных рядов с переменной шириной целесообразно применять для формирования выпуклостей на локтевых или плечевых участках.
10. Предложенный алгоритм расчета выпуклостей позволяет определить точное положение выпуклости на формируемом участке, что обеспечит правильную посадку проектируемого изделия на теле человека.
11. Разработан алгоритм для решения обратной задачи по определению формы выпуклого участка на трикотажном изделии при использовании технологии провязывания дополнительных петельных рядов, что позволит оценить возможную посадку на теле человека.
12. Экспериментальная проверка алгоритма для моделирования выпуклости заданной формы при заранее известном количестве дополнительных петельных рядов позволит определить длину профиля выпуклого участка трикотажа и сформировать технологии вязания выпуклости при использовании одного или двух нитеводителей.
13. Сравнение двух образцов выпуклостей, выработанных по разным технологиям при одинаковой и разной ширине дополнительных петельных рядов, показало сходимость полученных форм.
14. Использование расчетных данных для получения выпуклостей с разной шириной дополнительных петельных рядов для создания программы вязания в системе Stoll M1 3.15 проектирования цельновязаных изделий, позволило упростить процесс и сократить время на проектирование.
15. Разработана технология и алгоритм расчета горизонтальных и вертикальных вытачек на трикотажных изделиях за счет провязывания дополнительных петельных рядов, обеспечивающих получение необходимой трехмерной формы.

16. Для внедрения предложенных алгоритмов расчета для получения заданных выпуклостей за счет провязывания дополнительных петельных рядов требуется разработка специальной программы, которая ускорит процесс проектирования сложных трикотажных изделий и обеспечит их лучшую посадку на теле человека.

4 ПРОЕКТИРОВАНИЕ ЦЕЛЬНОВЯЗАНОГО ИЗДЕЛИЯ С ИСПОЛЬЗОВАНИЕМ РАЗРАБОТАННОГО ПРОГРАММНОГО ОБЕСПЕЧЕНИЯ “DESIGNER K-WEAR”

Согласно разработанной концепции комплексной системы проектирования было разработано программное обеспечение “DESIGNER K-WEAR” (свидетельство о государственной регистрации программы для ЭВМ № 2013618038).

Для разработки “DESIGNER K-WEAR” был выбран язык программирования C++, среда разработки MS Visual Studio 2010 [1.8, 1.10].

Текст исходного кода “DESIGNER K-WEAR” представлен в Приложении 1.

Учебное пособие по работе в “DESIGNER K-WEAR” представлено в Приложении 2.

Рассмотрим пример проектирования женского цельновязаного джемпера 46 размера (рисунок 4.1).

Разработку лекал, а также проектирование сбавок (прибавок) будем выполнять в “DESIGNER K-WEAR”, разработку программы вязания – в САПР Stoll M1 3.15.

Для вязания данного изделия используем вязальную машину Stoll CMS 340 TC-L класса 7.2, полушерстяную пряжу 32*2 Текс (50% шерсть, 50% полиамид).

Борт стана и рукавов, а также горловины выполним переплетением ластик 4+1, на рисунке 4.1 обозначены цифрой 1, рукава и стан (2) выполним переплетением кулирная гладь.

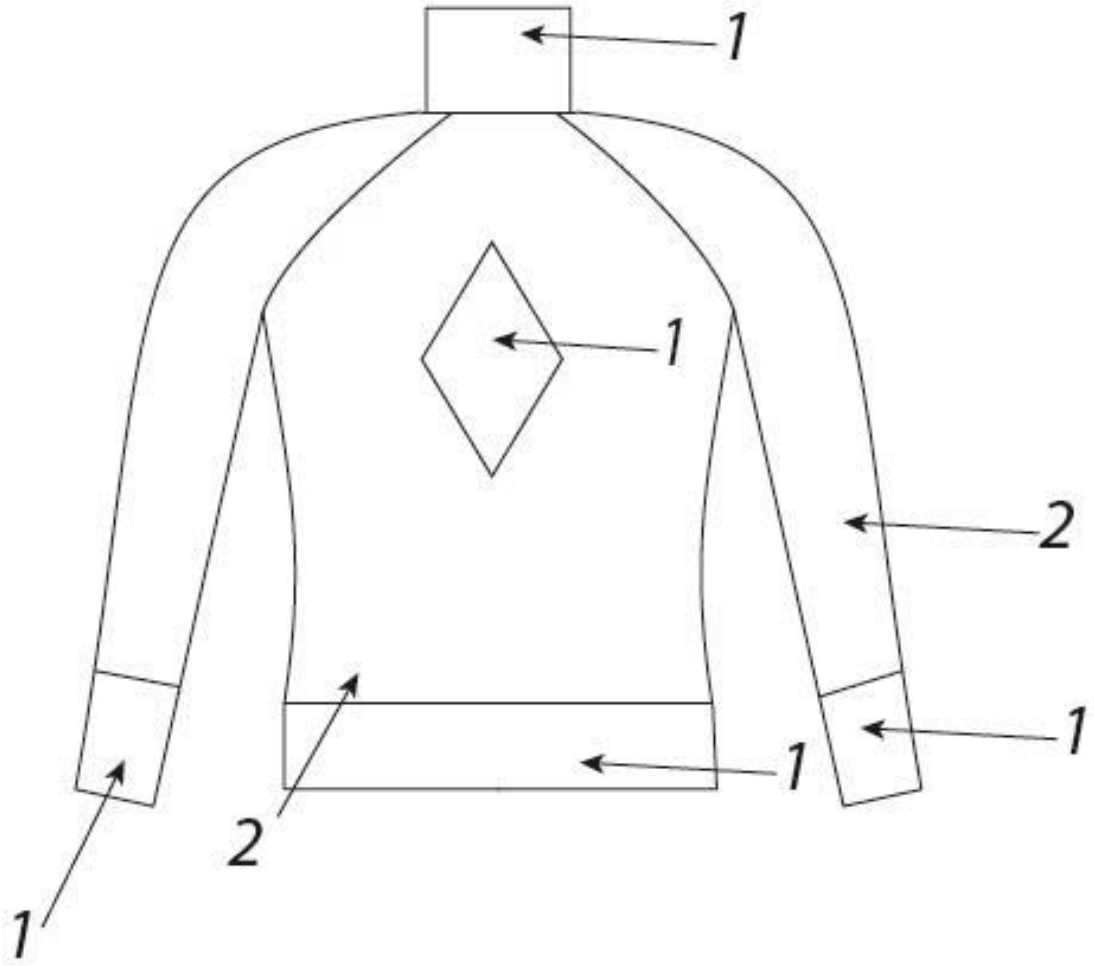


Рисунок 4.1 – Технический эскиз, проектируемого изделия

Высоту бортика стана и рукавов примем равной 120 петельным рядам.

Зададим плотность трикотажа основных петельных рядов:

- по горизонтали: $П_Г$ = 48 петельных столбиков на 100 мм;
- по вертикали: $П_В$ = 60 петельных рядов на 100 мм.

Сформируем пакет лекал, импортируемый в САПР Stoll M1 3.15.

Составим алгоритм построения. Осуществлять построение будем согласно учебному пособию, представленному в Приложении 2.

Построение графических объектов будем выполнять в неявной форме, что позволит в дальнейшем осуществлять автоматические градации разверток по размеро-ростам.

Так как проектируемое цельновязаное изделие симметричное, то будем выполнять построение только для одной половины детали.

Построим стан:

niz:line>100,100,122,100;

bok:bzr2>niz.x0,niz.y0,niz.x0+2,niz.y0-18,niz.x0,niz.y0-36;

reg1:bzr3>bok.x2,bok.y2,bok.x2+5,bok.y2-2,bok.x2+9.5,bok.y2-20,bok.x2+12.5,bok.y2-19;

gor:line>reg1.x3,reg1.y3,niz.x1,reg1.y3;

Воротник. В целях создания компактного отчета, высоту воротника будем задавать через раппорт в САПР М1.

nizv:line>gor.x0,gor.y0-2,gor.x0+11.6,gor.y0-2;

bokv:line>nizv.x0,nizv.y0,nizv.x0,nizv.y0-3;

Рукав:

nizr:line>niz.x0-20,niz.y0-3,niz.x0-29.6,niz.y0-3;

bokr:line>nizr.x0,nizr.y0,nizr.x0,nizr.y0-5;

bokr1:bzr2>bokr.x1,bokr.y1,bokr.x1,bokr.y1,bokr.x1+2.8,reg1.y0;

reg2:bzr3>bokr1.x2,bokr1.y2,bokr1.x2-5,bokr1.y2-2,bokr1.x2-9.5,bokr1.y2-20,nizr.x1,bokr1.y2-19;

Зададим параметры аппроксимации:

all:aprx>10/48,10/60;

reg2:aprxy>reg2.y0,reg2.y0-2.2,10/48,10/60,!!2,2;

reg1:aprxy>reg1.y0,reg1.y0-2.2,10/48,10/60,!!2,2;

reg1:aprxy>reg1.y3+9,reg1.y3+3.2,10/48,10/60,!!2,4;

reg1:aprxy>reg1.y3+3.2,reg1.y3,10/48,10/60,!!2,2;

reg2:aprxy>reg1.y3+9,reg1.y3+3.2,10/48,10/60,!!2,4;

reg2:aprxy>reg1.y3+3.2,reg1.y3,10/48,10/60,!!2,2;

reg2:aprx>10/48,10/60,!2,2;

reg1:aprx>10/48,10/60,!2,2;

Сформируем развертки деталей в режиме «кромка»:

rukav:side>nizr,bokr,bokr1,reg2|right;

stan:side>niz,bok,reg1,gor|left;

gorlo:side>nizv,bokv|left;

Выполним соединение рукава со станом:

rukav-stan:comb>reg2|reg1;

В области плечевого ската сформируем выпуклый участок трикотажа, вывязываемый при помощи одного нитеводителя

plecho:line>reg2.x3, reg2.y3,reg2.x3, reg2.y3+12;

vyp:side>plecho|right;

plecho:dopr>6,0.5,1,0.5*10/60;

krpl:bzr2>plecho.x0, plecho.y0,plecho.x0+2,plecho.y0+6,plecho.x1, plecho.y1;

vyp:formd>plecho|krpl|plecho;

Полученная конструкция цельновязаного изделия представлена на рисунке 4.2, узел соединения рукава со станом в аппроксимированном виде представлена на рисунке 4.3.

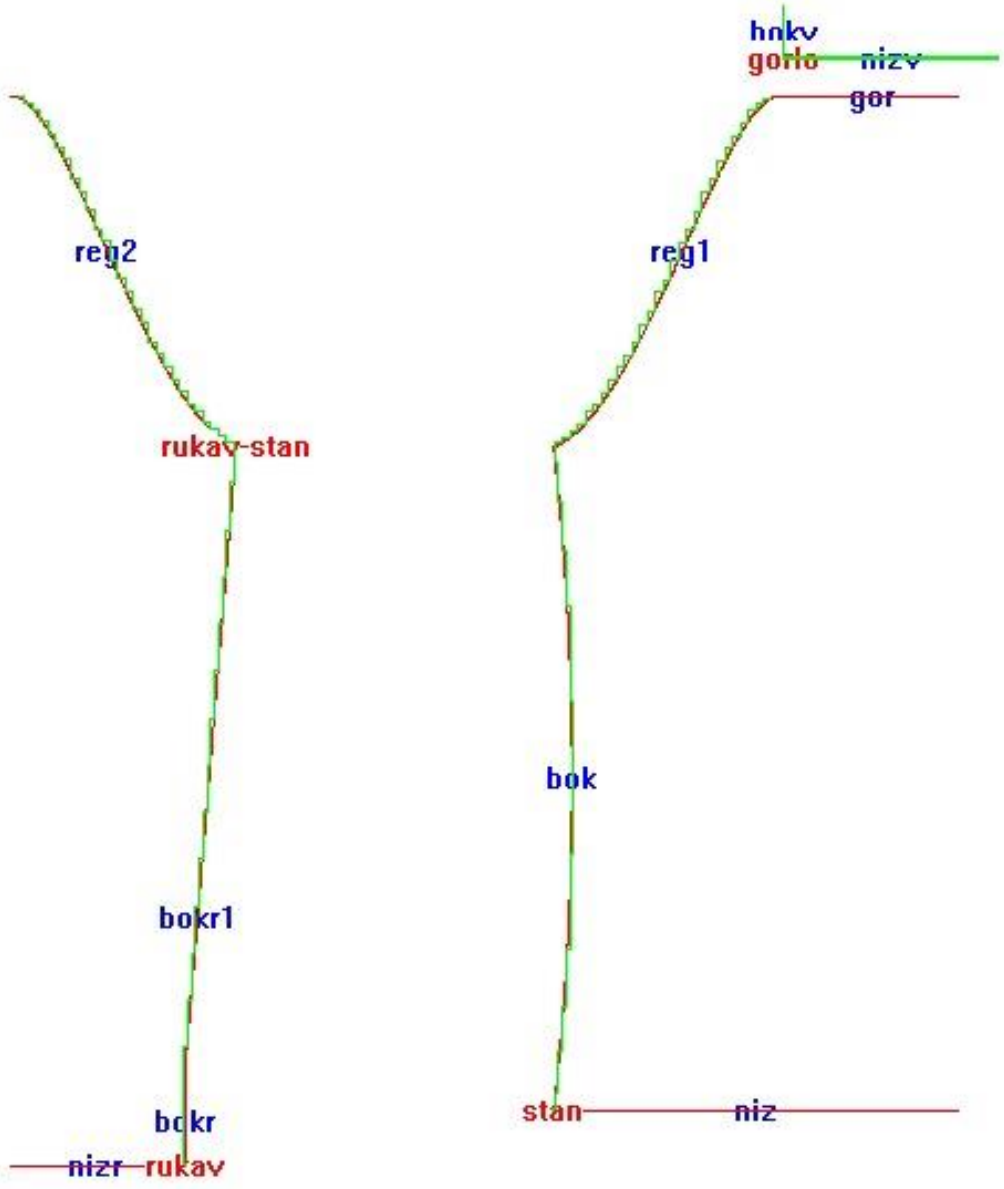


Рисунок 4.2 – Конструкция деталей цельновязаного джемпера, разработанная в “DESIGNER K-WEAR”

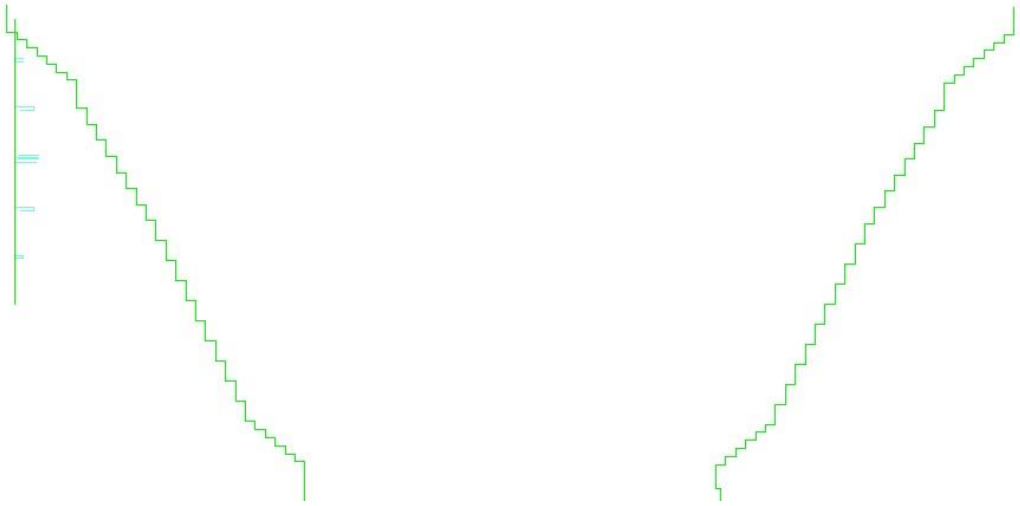


Рисунок 4.3 – Узел соединения в аппроксимированном виде

Данные, импортируемые в М1, представим в виде таблиц.

Таблица 5 – Рукав

Высота (ряды)	Ширина (петли)
1	-46
36	-1
24	-2
13	-1
24	-2
13	-1
12	-1
13	-1
24	-2
13	-1
12	-1
9	0
12	-12
50	-20
28	-14
5	0
14	-14
5	0

Таблица 6 – Стан

Высота (ряды)	Ширина (петли)
1	-105
5	1
13	1
14	1
19	1
30	1
49	-1
30	-1
18	-1
15	-1
13	-1
4	0
12	12
50	20
28	14
5	0
14	14
5	0

Таблица 7 – Воротник

Высота (ряды)	Ширина (петли)
0	55
1	-55
17	0

Таблица 8 – Выпуклость на плечевом участке

t	Ряд	Кол-во стол. слева	Кол-во доп. рядов	Кол-во стол. справа
0,11287	9	2	1	2
0,24039	18	3	1	3
0,36790	27	4	1	4
0,49541	36	5	1	5
0,50958	37	5	1	5
0,89212	64	2	1	2
0,76461	55	3	1	3
0,63709	46	4	1	4

Внешний вид цельновязаного изделия сформируем в САПР М1.

Импортируя данные из таблиц 5, 6, 7, 8 в САПР М1, получим форму цельновязаного изделия (рисунок 4.4).

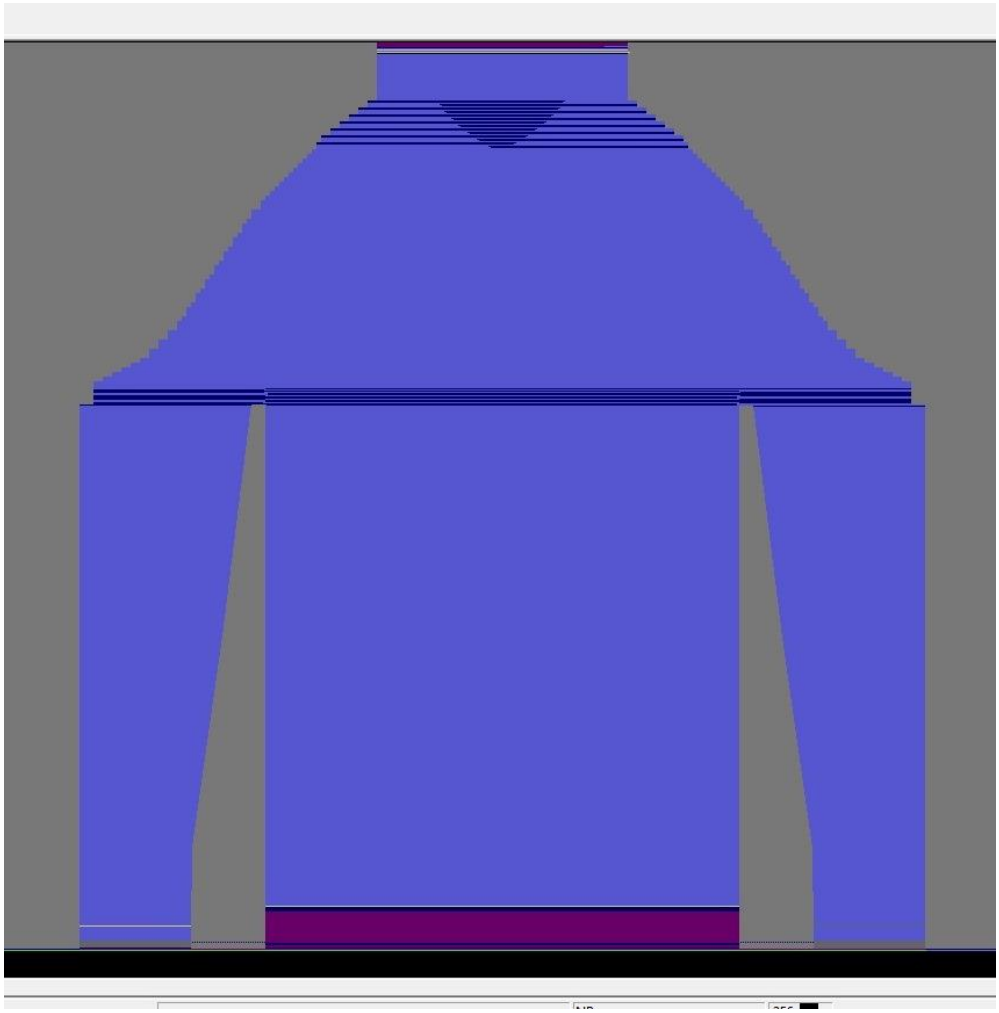


Рисунок 4.4 – Цельновязаное изделие в САПР M1

Так как машина CMS 340 TC-L оборудована гребенной системой оттяжки, то первый петельный ряд необходимо формировать из нити высокой растяжимости. Чаще всего в качестве такой нити используют спандекс 30 – 90 Текс.

Выполним распределение пряжи по работающим нитеводителям (таблица 9).

Таблица 9 – Распределение пряжи по нитеводителям

Номер нитеводителя		Назначение	Состав, Текс
Слева	Справа		
1	-	Формирование петельного ряда для захвата гребенкой	Спандекс, 40
2	-	Формирование разделительного петельного ряда	Полиамид, 20
-	3	Вязание стана	Полушерстяная пряжа, 32*2
4	-	Вязание левого рукава	Полушерстяная пряжа, 32*2
-	6	Вязание правого рукава	Полушерстяная пряжа, 32*2
-	8	Отработка	Хлопчатобумажная пряжа, 18,5*2

Установим скорость движения каретки.

Вязание петельных рядов основного полотна без переносов: 0,8 м/с.

Вязание петельных рядов основного полотна с переносами: 0,6 м/с.

Установим плотность трикотажа для петельных рядов отработки.

По горизонтали: 30 петельных столбиков на 100 мм;

По вертикали: 40 петельных рядов на 100 мм;

Выполним расчет времени вязания, а также расход сырья в САПР М1.

При условии, что вязальная машина оборудована автоматической системой смазки игольницы, примем проход каретки для смазки через каждые 200 проходов петлеобразующей системы. Так как CMS 340 TC-L имеет четыре петлеобразующие системы, то один проход каретки будет соответствовать

четырем проходам петлеобразующей системы. Тогда время вязания одного изделия согласно данным, определенным в САПР М1 3.15, составит 3514 секунд.

Расход, определенный в САПР М1 3.15, сырья представлен в таблице 10.

Таблица 10 – Расход сырья на одно изделие

Номер нитеводителя		Расход, м.	Расход, гр.
Слева	Справа		
1	-	30	1,2
2	-	15	0,3
-	3	1525	98,0
4	-	254	16,0
-	6	254	16,0
-	8	131	9,7

Разработанное цельновязаное изделие было выработано в ООО «Пафос» и представлено на рисунке 4.5.

Рисунок 4.5 – Женский цельновязаный джемпер 46 размера

Выполним расчет экономической эффективности использования "DESIGNER K-WEAR"

Как известно, производство сложных цельновязаных изделий экономически выгоднее производства трикотажных изделий, требуемых пошива. Однако,

трудоемкость при проектировании цельновязаных изделий выше, чем у вторых. Поэтому в целях установления экономической выгоды производства сложных цельновязаных изделий выполним расчет себестоимости данных изделий с учетом использования комплексной системы проектирования и без нее.

Как было установлено в главе 1: в настоящее время, чаще всего, конструирование трикотажных изделий выполняется при помощи ручных инструментов построения на масштабируемой бумаге. Поэтому эффективность работы отдела проектирования, в первую очередь, будет зависеть от степени загруженности сотрудников (дессинатора и конструктора), трудоемкости выполняемой ими работы и их квалификации.

Так как в широком понимании себестоимость – сумма денежных затрат на сырьё, обработку и реализацию на единицу продукции, то для определения эффективности производства цельновязаных изделий учтем только затраты, меняющие свое значение в зависимости от эффективности работы сотрудников.

С учетом использования комплексной системы проектирования в разработке цельновязаного изделия должна повышаться эффективность работы сотрудников отдела проектирования, что связано с автоматизацией ручных рутинных операций, созданием базы моделей и т.д. Исходя из этого, выполним расчет доли себестоимости единицы продукции и величины её изменения, что позволит определить экономическую выгоду при производстве одинаковых цельновязаных изделий, но с учетом разных методов проектирования.

Выполним планирование штата отдела проектирования (таблица 11).

Таблица 11 – Штат отдела проектирования

Наименование должности	Оклад (руб. в час)
Конструктор	190
Дессинатор	190

Приведем основные операции, выполняемые конструктором и дессинатором при проектировании единицы цельновязаного изделия (таблица 11).

Таблица 12 – Основные операции при проектировании

№	Конструктор	Дессинатор
1	Разработка конструкции	Аппроксимация лекал, полученных от конструктора
2	Формирование пакета лекал	Проектирование сбавок (прибавок)
3	Выдача дессинатору пакета лекал	Проектирование технологии вязания узлов соединения
4	Разработка формы для ВТО	Проектирование технологии вязания рельефных участков
5	-	Вязание образца
6	Анализ, связанного образца	
7	Выполнение технологических корректировок	
8	Градация по размеро-ростам	-

Анализ времени, требуемого конструктору на выполнение пунктов 1-4 таблицы 12, показал, что оно составляет 2 часа, дессинатору – 4 часа. Данный анализ выполнялся для проектирования цельновязаного изделия, представленного на рисунке 4.4. Вязание образца без учета заправки машины составляет 3514 секунд.

Выполнение пунктов 6-7 таблицы 12 составляет 1 час, причем выполнение технологических корректировок выполняет дессинатор, а конструктор осуществляет корректировки разверток согласно данным, полученным от дессинатора.

Время, необходимое для градации базового пакета лекал по размеро-ростам, составляет 10 минут на одну градацию. Количество градаций будет зависеть от результатов маркетинговых исследований рынка для разрабатываемого ассортимента. В среднем число градаций равно четырем. Исходя из этого, время, затрачиваемое конструктором на четыре градации, составляет 40 минут.

Данный анализ был выполнен при проектировании цельновязаного изделия без использования комплексной системы проектирования.

Таким образом, время, необходимое на разработку цельновязаного изделия, будет составлять: 2,00 + 4,00 + 3514,00/(60,00*60,00)+1,00 + 40,00/60,00 = 8,64 (часов), где 3,2 часа составляет работа конструктора, 5,44 часа – дессинатора.

Применение комплексной системы проектирования, позволило автоматизировать построение графических объектов, градацию, проектирование сбавок (прибавок), передачу дессинатору пакета лекал, частично автоматизировать проектирование узлов соединения, проектирование рельефа. Выполнение корректировок разверток выполняется автоматически в зависимости от поступления данных от дессинатора. Тогда таблица 12 примет следующий вид:

Таблица 13 – Основные операции при проектировании

№	Конструктор	Дессинатор
1	Разработка конструкции (сокращение времени на 15 %)	Аппроксимация лекал, полученных от конструктора (сокращение времени на 100 %)
2	Формирование пакета лекал (сокращение времени на 80 %)	Проектирование сбавок (прибавок) (сокращение времени на 95 %)

Продолжение таблицы 13

№	Конструктор	Дессинатор
3	Выдача дессинатору пакета лекал (сокращение времени 98 %)	Проектирование узлов соединения (сокращение времени на 30 %)
4	Разработка формы для ВТО	Проектирование рельефа (сокращение времени на 30 %)
5	-	Вязание образца
6	Анализ, связанного образца	
7	Выполнение технологических корректировок	
8	Градация по размеро-ростам (сокращение времени на 98 %)	-

Таким образом, время, затрачиваемое конструктором, составляет 59,9 минут или 1 час, дессинатором – 205,6 минут или 3,43 часа. Таким образом, общее время, затрачиваемое на разработку изделия, составит 4,43 часа.

Затраченное количество денежных средств на оплату работы:

- конструктора без использования комплексной системы проектирования составит 3,2*190,0 = 608,0 (рублей), с использованием комплексной системы проектирования – 1*190 = 190 (рублей);
- дессинатора без использования комплексной системы проектирования составит 5,44*190,00 = 1033,60 (рублей), с использованием комплексной системы проектирования – 4,43*190,00 = 841,70 (рублей);

Таким образом, общие затраты денежных средств на проектирование цельновязаного изделия конструктору и дессинатору:

- без использования комплексной системы проектирования, составит 1641,6 (рублей);
- с использованием комплексной системы проектирования, составит 1031,7.

Исходя из этого, экономическая эффективность использования комплексной системы проектирования будет составлять (1641,6 – 1031,7)*100,0 / 1641,6 = 36,8 (%), что позволит сократить издержки проектирования при расчете себестоимости.

При проектировании на предприятии 100 изделий в год экономическая эффективность составит 60990 рублей.

По разработанной технологии на предприятии ООО «Пафос» выпущено 200 штук изделий в следующем размерном ряду: 50 изделий размера 46-48, 100 изделий размера 50-52 и 50 изделий размера 54-56.

ВЫВОДЫ ПО ГЛАВЕ 4

1. Разработаны развертки деталей женского цельновязаного джемпера 46 размера с использованием специального программного обеспечения “DESIGNER K-WEAR”.
2. В “DESIGNER K-WEAR” выполнено проектирование сбавок (прибавок), а также сформирован узел соединения цельновязаного изделия.
3. С учетом, полученных данных из “DESIGNER K-WEAR” разработана программа вязания для плосковязальной машины CMS 340 TC-L при помощи САПР Stoll M1 3.15.
4. При помощи САПР M1 3.15 определено время вязания цельновязаного изделия, которое составило 3514 сек, а также определен расход сырья, затрачиваемого на вязание одного цельновязаного изделия.
5. Определена экономическая эффективность использования специального программного обеспечения “DESIGNER K-WEAR” при оплате труда конструктору и дессинатору составила 36,8 %, что соответствовало 609,9 рублей на одно проектируемое изделие, при выпуске 100 новых цельновязаных изделий экономический эффект составит 60909 рублей.
6. По разработанной технологии выработано 200 штук изделий на предприятии по выпуску верхнего трикотажа ООО «Пафос».

ОБЩИЕ ВЫВОДЫ

1. Проведенный анализ известных автоматизированных систем проектирования, применяемых в швейном и трикотажном производствах, показал, что проектирование сложных цельновязаных изделий является актуальным и перспективным направлением, что позволяет сократить время их разработки и повысить их качество.
2. Установлено, что использование расчетно-графических методов конструирования цельновязаных изделий позволит получить качественную посадку готового цельновязаного изделия на теле человека.
3. Разработан метод перевода лекал цельновязаных изделий, выполненных по швейной технологии, на технологию для трикотажного производства, реализованного на базе использования кривых Безье, широко применяемых в компьютерной графике.
4. Разработан метод проектирования сбавок (прибавок) за счет аппроксимации ячейками петли с учетом задаваемых ограничений аппроксимации, учитывающих технологические возможности вязального оборудования и параметры трикотажа.
5. Разработана технология балансирования-выравнивания элементов соединения по кромкам соединения, что необходимо для проектирования технологически и конструктивно качественного узла соединения деталей, имеющих разные по форме и не равные по размеру соединяемые кромки.
6. Разработан метод проектирования выпуклых участков трикотажа, получаемых за счет провязывания дополнительных петельных рядов, что позволяет улучшить качество посадки изделия.
7. Установлено, что наиболее рациональным способом получения выпуклого участка трикотажа является способ, при котором вязание основных и дополнительных петельных рядов выполняется одним

нитеводителем, однако, для получения более качественной выпуклости следует использовать специальный дополнительный нитеводитель, интарзионного типа.

8. Установлено, что для формирования сложных трехмерных выпуклостей целесообразно провязывать дополнительные петельные ряды переменной ширины.
9. Разработана технология и алгоритм расчета замены вертикальных и горизонтальных вытачек на геометрически аналогичную выпуклость, получаемую за счет провязывания дополнительных петельных рядов.
10. С использованием языка программирования С++, разработана программа “DESIGNER K-WEAR” (свидетельство о государственной регистрации программы для ЭВМ № 2013618038), а также учебное пособие по работе с ней (Приложение 2).
11. Выполнена апробация, разработанных методов проектирования, на примере проектирования женского цельновязаного джемпера 46 размера при использовании программы “DESIGNER K-WEAR” и САПР Stoll M1 3.15.
12. Определена экономическая эффективность внедрения комплексной системы проектирования, на примере программы “DESIGNER K-WEAR”, при оплате труда конструктору и дессинатору, которая составила 36,8 %.
13. Определены перспективы развития комплексной системы проектирования.

ТЕРМИНЫ И ОПРЕДЕЛЕНИЯ

1 **комплексная система проектирования**: Представляет собой совокупность взаимодополняющих программных и технических решений, используемых для частичной или полной автоматизации этапа проектирования.

2 **д-объект**: Представляет собой массив аналитических форм записей графических объектов, входящих в структуру развертки детали проектируемого изделия

3 **ячейка петли:** Условно представляет собой одну трикотажную петлю. Используется для проектирования сбавок (прибавок) по контуру развертки, в том числе с учетом дополнительных технологических ограничений, например, с учетом максимальной сбавки в петельном ряду.

4 **аппроксимация ячейками**: Процесс расчета сбавок (прибавок) по контуру развертки детали. Для аппроксимации каждой кромки развертки (левой или правой) применяется соответствующая ячейка петли.

5 **элемент соединения**: Элементом соединения является трикотажная петля, расположенная на кромке соединения и участвующая в соединении деталей в цельновязаном изделии.

6 **балансирование-выравнивание элементов соединения**: Процесс введения или уменьшения дополнительных петельных рядов или столбиков, обеспечивающих получение качественного соединения деталей цельновязаного изделия, при этом обеспечивая технологически возможный, при вязании изделия, процесс соединения.

7 **линия 1**: Верхняя граница участка основы узла соединения, до которой количество петельных рядов на кромках соединения деталей цельновязаного изделия одинаково.

8 **участок сравнения**: Представляет собой участки трикотажа на кромках соединения, длины которых сравниваются в процессе определения положения Линии 1.

9 **коэффициент сравнения (*ks*)**: Используется при определении положения Линии1. Суть данного коэффициента заключается в снижении вероятности возникновения ошибки в процессе определения положения Линии 1.

10 **центр распределения**: Является точка на проекции кривой Безье, которая представляет собой профиль выпуклого участка трикотажа, относительно которой выполняется распределение дополнительных петельных рядов. причем прямая, проходящая через промежуточную опорную вершину кривой Безье и центр распределения, будет перпендикулярна линии проекции кривой Безье.

СПИСОК ЛИТЕРАТУРЫ

1 Книги

1.1 Колесникова, Е.Н. Основы автоматизированного проектирования технологии вязания. – М: ТОО «Оргсервис ЛТД», 2000. – 240 с.

1.2 Шалов И.И., Кудрявин Л.А. Основы проектирования трикотажного производства с элементами САПР. – М. Легпромбытиздат, 1989. – 289 с.

1.3 Булатова Е.Б., Евсеева М.Н. Конструктиваное моделирование одежды.–М: «Академия», 2004. – 273 с.

1.4 Коблякова, Е.Б. Конструирование одежды с элементами САПР. – М.: Легпромбытиздат, 1988. – 462 с.

1.5 Карцева, А.А. Особенности конструирования изделий из трикотажа. –М: «Легкая индустрия», 1969. – 112 с.

1.6 Кудрявин Л.А., Шалов И.И. Основы технологии трикотажного производства: Учеб.пособие для вузов.- М.:Легпромбытиздат, 1991. – 496 с.

1.7 Дрожжин В.И., Орещенкова Н.В. Справочник по швейно-трикотажному производству. – М.: Легкая и пищевая промышленность, 1982. – 208 с.

1.8 Хортон, Айвор Visual C++ 2010: полный курс. – М.: ООО «И.Д. Вильямс», 2011. – 1216 с.

1.9 Кобляков В.А., Лукин А.С., Некоторые особенности моделирования структуры трикотажа. – М.: ЗАО «Экон-информ», 2013. – 115 с.

1.10 Ласло, М. Вычислительная геометрия и компьютерная графика на C++. – М: Бином, 2007. – 304 с.

2 Статьи

2.1 Киселева, С.Ф., Зорина Т.И. Обоснование наиболее эффективных вариантов организации производства верхнего трикотажа на малых предприятиях. // Текстильный клуб. – 2000, с.1-3.

2.2 Далидович, А.С. О направлениях в развитии техники трикотажного производства. // Текстильная промышленность. – 1960, № 8, 17-18 с.

2.3 Бюлер, Г. Тенденции, последние новинки и перспектива развития плосковязальных машин. // Трикотажная техника. – 1992, №2, 15-18 с.

2.4 Лазаренко В.М. Потребление нити в процессе кулирования. // Известия ВУЗов. Технология легкой промышленности. – 1969, №2, 119-121 с.

2.5 Болдырев, А.С. О перетяжке нити в процессе вязания. // Трикотажная промышленность. – 1940, №5, с.17-20.

2.6 Цитович И.Г., Большакова Н.И., Строганов Б.Б. Зависимость перетяжки нити в старых петлях полотна от усилия оттяжки и функциональных свойств нити. // Известия ВУЗов. Технология легкой промышленности. – 1975, №5.

2.7 Кальницкий, Л.Б. О соотношении между номерами пряжи и классом плоскофанговых машин. // Легкая промышленность. – 1955, №6, с. 40-33.

2.8 T.Stoll. Opportunities for using flat knitting machines for industrial textiles. // Knitting Technique. – 1991, V13, №2, p.120-125.

3 Диссертации

3.1 Сичкарь, Т.В. Разработка технологии и проектирования соединительных участков цельновязаных изделий. Диссертация на соискание ученой степени кандидата технических наук: 05.19.02 / Сичкарь Татьяна Валентиновна. – Спб., 2005. – 198 с.

3.2 Скопинцева, Е.А. Разработка технологии выработки трикотажных цельновязаных изделий сложной конструкции на плосковязальном оборудовании. – Диссертация на соискание ученой степени кандидата технических наук: 05.19.02 / Скопинцева Евгения Александровна. – М., 2009 г. – 258 с.

3.3 Лукин, А.С. Разработка методов проектирования и исследование процессов выработки регулярных трикотажных изделий сложных форм на вязальных машинах. – Диссертация на соискание ученой степени

кандидата технических наук: 05.19.02 / Лукин Александр Сергеевич. – М., 2003. – 149 с.

3.4 Щербакова, Н.В. Разработка и совершенствование технологии изготовления цельновязаных верхних трикотажных изделий на плосковязальных машинах типа ПА. – Диссертация на соискание ученой степени кандидата технических наук – М., 1985.

3.5 Ермохина, Т.Е. Разработка процесса выработки верхних трикотажных изделий сложных конструкций с минимальной швейной обработкой. – Диссертация на соискание ученой степени кандидата технических наук: 05.19.02 / Ермохина Татьяна Евгеньевна. – М., 2008. – 251 с.

3.6 Колесникова, Е.Н. Основы проектирования технологии петлеобразования. Диссертация на соискание ученой степени доктора технических наук: 05.19.02 / Колесникова Елена Николаевна. – М., 2001. – 477 с.

3.7 Муракаева, Т.В. Разработка автоматизированных методов проектирования верхнетрикотажных изделий с плосковязальных машин с целью ресурсосбережения. Диссертация на соискание ученой степени кандидата технических наук: 05.19.03 / Муракаева Татьяна вячеславовна. – М., 1997.

4 Патенты

4.1 Патент WO 3066947 A1 № 7D 04 B 7/00. Способ вязания трубчатого трикотажного изделия. Okamoto, Kazuyoshi.

4.2 Патент US 6581417 BB № 7D 04 B 7/10. Способ вязания трикотажного изделия. Yui, Manabu.

4.3 Патент JP 3541177 B2 № 7D 04 B 7/10. Способ вязания одежды. Yui, Manabu.

4.4 Патент WO 2004063447 A1 № 7D 04 B 1/24. Трубчатое трикотажное изделие и способ его вязания. Kosui, Tatsuya.

4.5 Патент EP 1394308 A1 № 7D 04 B 7/30. Трикотажное изделие с воротником, вырабатываемое на плосковязальной машине, и способ его вязания. Okamoto, Kazuyoshi.

4.6 Патент EP 1283290 A1 № 7D 04 B 7/32. Способ соединения деталей трикотажного изделия и готовое изделие. Okamoto, Kazuyoshi.

4.7 Патент US 6672113 BB № 7D 04 B 7/10. Способ вязания ворота трикотажного изделия на плосковязальной машине и применяемое для этого устройство. Okamoto, Kazuyoshi.

4.8 АС СССР № 320579 МКИ D 04 B 1/24. Способ вязания штучных трикотажных изделий типа свитера, джемпера на плоскофанговой машине. Ващинский Л.К., Ващинская Н.Н.

5 Стандарты

5.1 ГОСТ Р 7.0.11 – 2011 Система стандартов по информации, библиотечному и издательскому делу. Диссертация и автореферат диссертации. Структура и правила оформления. – М.: Стандартинформ, 2012. – 12 с.

Федеральное государственное бюджетное образовательное учреждение
высшего профессионального образования
«Московский государственный университет дизайна и технологии»

На правах рукописи

Ланшаков Денис Евгеньевич

Разработка технологий и конструкций сложных цельновязаных изделий на базе комплексной автоматизированной системы

Специальность 05.19.02 – Технология и первичная обработка текстильных материалов и сырья

ПРИЛОЖЕНИЕ 1
диссертации на соискание ученой степени
кандидата технических наук

Научный руководитель –
доктор технических наук
профессор Колесникова Елена Николаевна

Москва – 2014 г.

ОГЛАВЛЕНИЕ

1 ТЕКСТЫ ИСХОДНОГО КОДА ЗАГОЛОВОЧНЫХ ФАЙЛОВ

Файл “Approx.h”

```
#pragma once
#include <vector>
#include "AprObj.h"
#include "Objects.h"
#include "ReverseObj.h"
#include "Constants.h"
class CApprox
{
public:
	std::vector <CAprObj> sbavki;
	std::vector <CAprObj> sbavki_dop1; //
	CString name, name_dop, sopr_name;
	CApprox(std::vector <CReverseObj> obj, std::vector <CObjects> com, int side, CString dname, SGLAZH sgl);
	CApprox(void);
	~CApprox(void);
};
```

Файл “AprDobj.h”

```
#pragma once
#include <vector>
#include "DObjects.h"
#include "Objects.h"
#include "AprObj.h"
#include <cmath>
#include "Constants.h"
class CAprDobj
{
```

```
public:
        std::vector <CAprObj> sbavki_left;
        std::vector <CAprObj> sbavki_right;
        CString name;
        CString type;
        int line_r; // Линия 1
        int rzs; // Разность по рядам
        int rze; // Разность по элементами
        CString name_left, name_right;
        CAprDobj(std::vector <CReverseObj> objl, std::vector <CReverseObj> objr, std::vector <CObjects> com, CString dname, SGLAZH sgl, CString s_name);
        CAprDobj(void);
        ~CAprDobj(void);
};
```

Файл “AprObj.h”

```
#pragma once
class CAprObj
{
public:
        double ay; // Ордината при аппроксимации
        double kx; // Абсцисса кривой
        double ax; //Абсцисса при аппроксимации
        int sbav; // Сбавка в ряду r
        int r; // Ряд
        double t; // Параметр t
        int stol; // Количество столбиков
        int stol_l, stol_r;
```

```
        // Отчетность по аппроксимации
        double ap; // Ширина петельного столбика
        double bp; // Высота петельного ряда
        int maxsb; // Максимальная сбавка в ряду
        int rapapr; // Раппорт аппроксимации
        double axl, axr;
        // Для построения
        int pol;
        int side_pol;
        int sbav_all;
        double line_r;
        double dop_r; // дополнительный ряд
        CAprObj(void);
        ~CAprObj(void);
};
```

Файл “Console.h”

```
#pragma once
#include "afxwin.h"
#include <vector>
#include "Constants.h"
// диалоговое окно CConsole
class CConsole : public CDialog
{
        DECLARE_DYNAMIC(CConsole)
public:
        CConsole(CWnd* pParent = NULL);   // стандартный конструктор
        virtual ~CConsole();
// Данные диалогового окна
        enum { IDD = IDD_CONSOLE };
```

```
protected:
        virtual void DoDataExchange(CDataExchange* pDX);      // поддержка
DDX/DDV
        DECLARE_MESSAGE_MAP()
public:
        CEdit m_EditConsole;
        CString buffer_string;
        afx_msg void OnEnChangeEdit1();
        std::vector <CString> buff_inp;
        std::vector <CString> serial;
        CEdit m_Edit_Info;
        Check m_Check;
        afx_msg void OnBnClickedCheck1();
        afx_msg void OnBnClickedCheck2();
        CButton check1;
        CButton check2;
};
```

Файл “Constants.h”

```
#pragma once
enum Buttons {CONSOLE, EMPTY};
enum Grafika {CONSTRUCT, SBAVKI, ALL};
enum ED_IZM {SM, MM};
enum Info {PETLI, EMPTY_P};
enum Napr {EMPTY_N, NAPRAV};
enum Check{ZAMER, GRAFIKA, STAND};
enum SGLAZH {Ys, Ns};
enum PRINT {B, E};
enum M1TRANSFORM {Y, N};
enum VYTACHKI {CLOSE, OPEN};
enum Setup {ON, OFF};
```

Файл “Designer k-wear.h”

```
#pragma once
#ifndef __AFXWIN_H__
	#error "включить stdafx.h до включения этого файла в PCH"
#endif
#include "resource.h"		// основные символы
// CDesignerkwearApp:
// О реализации данного класса см. Designer k-wear.cpp
//
class CDesignerkwearApp : public CWinAppEx
{
public:
	CDesignerkwearApp();
// Переопределение
public:
	virtual BOOL InitInstance();
	virtual int ExitInstance();
// Реализация
	BOOL  m_bHiColorIcons;
	virtual void PreLoadState();
	virtual void LoadCustomState();
	virtual void SaveCustomState();
	afx_msg void OnAppAbout();
	DECLARE_MESSAGE_MAP()
};
extern CDesignerkwearApp theApp;
```

Файл “Designer k-wearDoc.h”

```
#pragma once
#include "Objects.h"
#include <vector>
#include "Console.h"
#include "Setup.h"
#include "DObjects.h"
#include "Approx.h"
#include "AprDobj.h"
#include "Constants.h"
#include "afxdlgs.h"
#include "Razmer.h"
#include "Info.h"
class CDesignerkwearDoc : public CDocument
{
protected: // создать только из сериализации
	CDesignerkwearDoc();
	DECLARE_DYNCREATE(CDesignerkwearDoc)
// Атрибуты
public:
	CSize m_DocSize;
// Операции
public:
// Переопределение
public:
	virtual BOOL OnNewDocument();
	virtual void Serialize(CArchive& ar);
#ifdef SHARED_HANDLERS
	virtual void InitializeSearchContent();
	virtual void OnDrawThumbnail(CDC& dc, LPRECT lprcBounds);
```

```
#endif // SHARED_HANDLERS
// Реализация
public:
	virtual ~CDesignerkwearDoc();
#ifdef _DEBUG
	virtual void AssertValid() const;
	virtual void Dump(CDumpContext& dc) const;
#endif
protected:
// Созданные функции схемы сообщений
protected:
	DECLARE_MESSAGE_MAP()
#ifdef SHARED_HANDLERS
	// Вспомогательная функция, задающая содержимое поиска для обработчика поиска
	void SetSearchContent(const CString& value);
#endif // SHARED_HANDLERS
public:
	afx_msg void OnConsole();
	afx_msg void OnBuild();
	std::vector <CObjects> obj; // Объекты
	std::vector <CObjects> com; // Команды
	std::vector <CDObjects> dobj; // Д-Объекты
	std::vector <CApprox> approximation_l; // Сбавки
	std::vector <CApprox> approximation_r; // Сбавки
	std::vector <CApprox> approximation_dl; // Сбавки
	std::vector <CApprox> approximation_dr; // Сбавки
	std::vector <CApprox> side_d; // Сбавки
	std::vector <CApprox> approx_side; // Сбавки
	std::vector <CAprDobj> approx_cs;
```

```
        std::vector <CObjects> aprox; // Параметры аппроксимации
        std::vector <CString> names;
        std::vector <CString> com_trans;
        afx_msg void OnReport();
        afx_msg void OnReportM1();
        std::vector <CObjects> buffer_vector; // Буферный вектор простого объекта
        std::vector <CRazmer> raz;
        std::vector<CString> vivod; // Только для View
    protected:
        Buttons m_Check;
    public:
        afx_msg void OnUpdateConsole(CCmdUI *pCmdUI);
        afx_msg void OnSm();
        // Единицы измерения (см или мм)
        ED_IZM m_Izm;
        int izmerenia;
        afx_msg void OnMm();
        afx_msg void OnUpdateSm(CCmdUI *pCmdUI);
        afx_msg void OnUpdateMm(CCmdUI *pCmdUI);
        afx_msg void OnRz();
        // Создание объекта диалогового окна клавиатурного ввода команд
        CConsole aConsole;
        std::vector <CString> savecom;
        std::vector<CInfo> info;
        CSetup aSetup;
        int tolshina;
        afx_msg void OnSglazh();
        SGLAZH m_Sgl;
        afx_msg void OnUpdateSglazh(CCmdUI *pCmdUI);
```

```
        int ris_list;
        M1TRANSFORM trans;
        afx_msg void OnM1Transform();
        afx_msg void OnUpdateM1Transform(CCmdUI *pCmdUI);
        afx_msg void OnVytClose();
        VYTACHKI m_Vyt;
        Setup m_prov;
        afx_msg void OnVytOpen();
        afx_msg void OnUpdateVytClose(CCmdUI *pCmdUI);
        afx_msg void OnUpdateVytOpen(CCmdUI *pCmdUI);
        afx_msg void OnSetup();
        afx_msg void OnUpdateSetup(CCmdUI *pCmdUI);
};
```

Файл “Designer k-wearView.h”

```
#pragma once
#include "ZoomWindow.h"
#include "Constants.h"
#include "atltypes.h"
class CDesignerkwearView : public CScrollView
{
protected: // создать только из сериализации
        CDesignerkwearView();
        DECLARE_DYNCREATE(CDesignerkwearView)
// Атрибуты
public:
        CDesignerkwearDoc* GetDocument() const;
        CZoomWindow zoomWin;
// Операции
public:
```

```
// Переопределение
public:
	virtual void OnDraw(CDC* pDC);  // переопределено для отрисовки этого представления
	virtual BOOL PreCreateWindow(CREATESTRUCT& cs);
protected:
	virtual void OnInitialUpdate(); // вызывается в первый раз после конструктора
	virtual BOOL OnPreparePrinting(CPrintInfo* pInfo);
	virtual void OnBeginPrinting(CDC* pDC, CPrintInfo* pInfo);
	virtual void OnEndPrinting(CDC* pDC, CPrintInfo* pInfo);
// Реализация
public:
	virtual ~CDesignerkwearView();
#ifdef _DEBUG
	virtual void AssertValid() const;
	virtual void Dump(CDumpContext& dc) const;
#endif
protected:
// Созданные функции схемы сообщений
protected:
	afx_msg void OnFilePrintPreview();
	afx_msg void OnRButtonUp(UINT nFlags, CPoint point);
	afx_msg void OnContextMenu(CWnd* pWnd, CPoint point);
	DECLARE_MESSAGE_MAP()
public:
	afx_msg void OnZoomIn();
	afx_msg void OnZoomOut();
	Grafika m_Graf;
```

```
        Info m_Info;
        Napr m_Naprav;
        afx_msg void OnGraf();
        afx_msg void OnUpdateGraf(CCmdUI *pCmdUI);
        afx_msg void OnSbav();
        afx_msg void OnUpdateSbav(CCmdUI *pCmdUI);
        afx_msg void OnPetliInfo();
        afx_msg void OnUpdatePetliInfo(CCmdUI *pCmdUI);
        afx_msg void OnNaprav();
        afx_msg void OnUpdateNaprav(CCmdUI *pCmdUI);
    //  afx_msg void OnLButtonDown(UINT nFlags, CPoint point);
        CPoint pvPoint;
        CPoint pvtPoint;
        CPoint pvePoint;
        PRINT m_Prnt;
        CRect PrintRect;
        afx_msg void OnLButtonUp(UINT nFlags, CPoint point);
        afx_msg void OnMouseMove(UINT nFlags, CPoint point);
    //  virtual void OnUpdate(CView* /*pSender*/, LPARAM /*lHint*/,
CObject* /*pHint*/);
    //  virtual BOOL OnScroll(UINT nScrollCode, UINT nPos, BOOL bDoScroll
= TRUE);
        int id_curve;
        int id_point;
        CString id_name;
        int zoom;
        afx_msg void OnLButtonDown(UINT nFlags, CPoint point);
        afx_msg void OnAll();
        afx_msg void OnUpdateAll(CCmdUI *pCmdUI);
        afx_msg void OnPrintObl();
```

```
        afx_msg void OnUpdatePrintObl(CCmdUI *pCmdUI);
};
#ifndef _DEBUG  // отладочная версия в Designer k-wearView.cpp
inline CDesignerkwearDoc* CDesignerkwearView::GetDocument() const
   { return reinterpret_cast<CDesignerkwearDoc*>(m_pDocument); }
#endif
```

Файл “DObjects.h”

```
#pragma once
#include "Objects.h"
#include "ReverseObj.h"
class CDObjects
{
public:
        // Форма
        std::vector <CReverseObj> d_object;
        std::vector <CReverseObj> d_object_left;
        std::vector <CReverseObj> d_object_right;
        // Кромка
        std::vector <CReverseObj> side;
        // Соединение кривых
        std::vector <CReverseObj> comb;
        std::vector <CReverseObj> comb_left;
        std::vector <CReverseObj> comb_right;
        int st; // Идентификатор
        CString dname; // Имя
        CString command;
        CString s_name;
        int side_const; // Const'та для указ положения зоны вязания
```

```
        double converter(CString argum, std::vector<CRazmer> rz, std::vector
<CObjects> vars);
        int prior(CString zn);
        int str;
        double kx, ky;
        CDObjects(CString buff_inp, std::vector <CObjects> obj,
std::vector<CRazmer> raz, std::vector<CObjects> vars, int strn, std::vector
<CDObjects> dobj, std::vector<CString> com_trans);
        CDObjects(void);
        ~CDObjects(void);
};
```

Файл “Info.h”

```
#pragma once
#include "Objects.h"
class CInfo
{
public:
        CString value;
        CString name;
        CInfo(CObjects obj);
        CInfo(void);
        ~CInfo(void);
};
```

Файл “MainFrm.h”

```
#pragma once
class CMainFrame : public CFrameWndEx
{
protected: // создать только из сериализации
```

```
        CMainFrame();
        DECLARE_DYNCREATE(CMainFrame)

    // Атрибуты
    public:
    // Операции
    public:
    // Переопределение
    public:
        virtual BOOL PreCreateWindow(CREATESTRUCT& cs);
        virtual BOOL LoadFrame(UINT nIDResource, DWORD dwDefaultStyle =
WS_OVERLAPPEDWINDOW | FWS_ADDTOTITLE, CWnd* pParentWnd = NULL,
CCreateContext* pContext = NULL);
    // Реализация
    public:
        virtual ~CMainFrame();
    #ifdef _DEBUG
        virtual void AssertValid() const;
        virtual void Dump(CDumpContext& dc) const;
    #endif
    protected:  // встроенные члены панели элементов управления
        CMFCMenuBar       m_wndMenuBar;
        CMFCToolBar       m_wndToolBar;
        CMFCStatusBar     m_wndStatusBar;
        CMFCToolBarImages m_UserImages;
    // Созданные функции схемы сообщений
    protected:
        afx_msg int OnCreate(LPCREATESTRUCT lpCreateStruct);
        afx_msg void OnViewCustomize();
        afx_msg LRESULT OnToolbarCreateNew(WPARAM wp, LPARAM lp);
```

```
        DECLARE_MESSAGE_MAP()
};
```

Файл “Objects.h”

```
// Класс распознавание команд, введенных с клавиатуры
#pragma once
#include <vector>
#include <cmath>
#include "Razmer.h"
class CObjects
{
public:
        // Координаты опорных вершин элементов
        double x0;
        double y0;
        double x1;
        double y1;
        double x2;
        double y2;
        double x3;
        double y3;
        double a;
        double b;
        int rapapr;
        int maxsb;
        // Рисование
        int tolshina;
        int red;
        int green;
        int black;
```

```
// Ключ элемента
int st;

// Наличие элемента
int p;
CString nit;
// Имя элемента
CString name;
// Команда
CString command;
CString fun;
CString sx0,sy0,sx1,sy1,sx2,sy2,sx3,sy3,sa1,sa2,sa3,sa4,st1,st2;
int str;
CString end_left, end_right, vyt_l, vyt_r, n_ryadov, vysota_ryada;
double pzx, pzy;
double left_line, right_line, ras;
// Частичное вязание
std::vector<CString> left;
std::vector<CString> right;
// Функция проебразование CString в double
double converter(CString argum, std::vector<CRazmer> rz, std::vector
<CObjects> obj);
int prior(CString zn);
CObjects(CString buff_inp, int mm, std::vector <CObjects> vars,
std::vector<CRazmer> rz,int strn,std::vector<CString> com_trans);
CObjects(void);
~CObjects(void);
double a1;
double a2;
double a3;
```

```
        double a4;
        double osnx, osny;

protected:
        double t;
        double t1;
};
```

Файл “Razmer.h”

```
#pragma once
class CRazmer
{
public:
        double rz;
        CString name;
        CString title;
        double converter(CString argum);
        CRazmer(CString buff, CString tit);
        CRazmer(void);
        ~CRazmer(void);
};
```

Файл “ReportM1.h”

```
#pragma once
#include "AprObj.h"
#include<vector>
class CReportM1
{
public:
        std::vector<CAprObj> sbavki;
```

```
        CReportM1(std::vector<CAprObj> input);
        CReportM1(void);
        ~CReportM1(void);
};
```

Файл “Resource.h”

```
#define IDP_OLE_INIT_FAILED          100
#define IDD_ABOUTBOX                 100
#define IDR_POPUP_EDIT               119
#define ID_STATUSBAR_PANE1           120
#define ID_STATUSBAR_PANE2           121
#define IDS_STATUS_PANE1             122
#define IDS_STATUS_PANE2             123
#define IDS_TOOLBAR_STANDARD         124
#define IDS_TOOLBAR_CUSTOMIZE        125
#define ID_VIEW_CUSTOMIZE            126
#define IDR_MAINFRAME                128
#define IDR_MAINFRAME_256            129
#define IDR_DesignerkwearTYPE        130
#define IDS_EDIT_MENU                306
#define IDD_CONSOLE                  310
#define IDD_DIALOG1                  312
#define IDD_DIALOG2                  313
#define IDD_DIALOG3                  315
#define IDC_EDIT1                    1000
#define IDCANCEL                     1007
#define IDC_EDIT2                    1015
#define IDC_CHECK1                   1025
#define IDC_CHECK2                   1026
#define ID_32771                     32771
```

```
#define ID_32772                32772
#define ID_BUILD                 32773
#define ID_32774                32774
#define ID_32775                32775
#define ID_ZOOM_IN                 32776
#define ID_ZOOM_OUT                  32777
#define ID_32778                32778
#define ID_REPORT                  32779
#define ID_32780                32780
#define ID_REPORT_M1                  32781
#define ID_32782                32782
#define ID_BUTTON32783                  32783
#define ID_BUTTON32784                  32784
#define ID_32788                32788
#define ID_32789                32789
#define ID_32790                32790
#define ID_GRAF                  32791
#define ID_SBAV                  32792
#define ID_32793                32793
#define ID_32797                32797
#define ID_32798                32798
#define ID_32799                32799
#define ID_SM                32800
#define ID_MM                 32801
#define ID_32802                 32802
#define ID_RZ                32803
#define ID_32804                 32804
#define ID_32805                 32805
#define ID_32806                 32806
#define ID_PETLI_INFO                  32807
```

```
#define ID_32808                32808
#define ID_NAPRAV               32809
#define ID_32810                32810
#define ID_LINE                 32811
#define ID_32812                32812
#define ID_                     32813
#define ID_32814                32814
#define ID_ALL                  32815
#define ID_32818                32818
#define ID_SGLAZH               32819
#define ID_32820                32820
#define ID_PRINT_OBL            32821
#define ID_32822                32822
#define ID_32823                32823
#define ID_32824                32824
#define ID_M1_TRANSFORM         32825
#define ID_32826                32826
#define ID_32827                32827
#define ID_32828                32828
#define ID_VYT_CLOSE            32829
#define ID_VYT_OPEN             32830
#define ID_32831                32831
#define ID_32832                32832
#define ID_32833                32833
#define ID_Menu                 32834
#define ID_32835                32835
#define ID_32836                32836
#define ID_32837                32837
#define ID_32838                32838
#define ID_32839                32839
```

```
#define ID_SETUP                    32840
// Next default values for new objects
//
#ifdef APSTUDIO_INVOKED
#ifndef APSTUDIO_READONLY_SYMBOLS
#define _APS_NEXT_RESOURCE_VALUE      317
#define _APS_NEXT_COMMAND_VALUE       32841
#define _APS_NEXT_CONTROL_VALUE       1028
#define _APS_NEXT_SYMED_VALUE         310
#endif
#endif
```

Файл “ReverseObj.h”

```
// Класс для создания реверсивных объетов
#pragma once
#include "Objects.h"
class CReverseObj
{
public:
        double x0;
        double x1;
        double x2;
        double x3;
        double y0;
        double y1;
        double y2;
        double y3;
        int p;
        int st;
        CString name;
```

```
        CReverseObj(void);
        ~CReverseObj(void);
        CReverseObj(CObjects Obj);
};
```

Файл “RzR.h”

```
#pragma once
#include "afxwin.h"
// диалоговое окно CRzR
class CRzR : public CDialog
{
        DECLARE_DYNAMIC(CRzR)
public:
        CRzR(CWnd* pParent = NULL);   // стандартный конструктор
        virtual ~CRzR();
// Данные диалогового окна
        enum { IDD = IDD_DIALOG1 };
protected:
        virtual  void  DoDataExchange(CDataExchange*  pDX);       //  поддержка
DDX/DDV
        DECLARE_MESSAGE_MAP()
};
```

Файл “Setup.h”

```
#pragma once
#include "Constants.h"
// диалоговое окно CSetup
class CSetup : public CDialog
{
        DECLARE_DYNAMIC(CSetup)
```

```
public:
        CSetup(CWnd* pParent = NULL);   // стандартный конструктор
        virtual ~CSetup();
// Данные диалогового окна
        enum { IDD = IDD_DIALOG2 };
protected:
        virtual  void  DoDataExchange(CDataExchange*  pDX);      //  поддержка
DDX/DDV
        DECLARE_MESSAGE_MAP()
public:
        afx_msg void OnCbnSelchangeCombo1();
};
```

Файл “ZoomWindow.h”

```
#pragma once
#include "afxwin.h"
class CZoomWindow
{
public:
        CZoomWindow();
        virtual ~CZoomWindow();
        CSize GetWin();          // получить размер окна
        CSize GetView();  // получить единицы разрешения
        void ZoomIn();           // увеличение
        void ZoomOut();          // уменьшение
private:
         CSize win;          // размер окна
         CSize view;              // единицы разрешения
};
```

2 ТЕКСТЫ ИСХОДНОГО КОДА ФАЙЛОВ РЕАЛИЗАЦИИ

Файл “Approx.cpp”

```
#include "StdAfx.h"
#include "Approx.h"
CApprox::CApprox(void)
{
}
CApprox::~CApprox(void)
{
}
CApprox::CApprox(std::vector <CReverseObj> obj, std::vector <CObjects> com, int side, CString dname, SGLAZH sgl)
{
	CAprObj report;
	std::vector<CAprObj> temp_sb;
	std::vector <CAprObj> sbavki_dop;
	name = dname;
	for (int i = 0; i < dname.GetLength(); i++)
	{
		if (dname[i] == '$')
		{
			name = dname.Mid(0, i);
			sopr_name = dname.Mid(i+1, dname.GetLength()-i);
			break;
		};
	};
	int ryad(1);
	double dx(0), dy(0);
	int sbavka_max(0);
	std::vector<double> min;
```

```
// Объявление переменных
double x0(0), y0(0), x1(0), y1(0), x2(0), y2(0), x3(0), y3(0);
double ap(0), bp(0);
int rapapr(0), maxsb(0);
double kony(0), konx(0);
double rapapr1(0);
double bp1(0);
        for (int iii = 0; iii < obj.size(); iii++)
        {
                // Перенос данных
                temp_sb.clear();
                if (obj[iii].st == 1)
                {
                        x0 = obj[iii].x0 + dx;      y0 = obj[iii].y0 + dy;
                        x1 = obj[iii].x1 + dx;      y1 = obj[iii].y1 + dy;
                        kony = y1;
                        konx = x1;
                }
                else if (obj[iii].st == 2)
                {
                        x0 = obj[iii].x0 + dx;      y0 = obj[iii].y0 + dy;
                        x1 = obj[iii].x1 + dx;      y1 = obj[iii].y1 + dy;
                        x2 = obj[iii].x2 + dx;              y2 = obj[iii].y2 + dy;
                        kony = y2;
                        konx = x2;
                }
                else if (obj[iii].st == 3)
                {
                        x0 = obj[iii].x0 + dx;      y0 = obj[iii].y0 + dy;
                        x1 = obj[iii].x1 + dx;      y1 = obj[iii].y1 + dy;
```

```
                x2 = obj[iii].x2 + dx;          y2 = obj[iii].y2 + dy;
                x3 = obj[iii].x3 + dx;          y3 = obj[iii].y3 + dy;
                kony = y3;
                konx = x3;
        };
        double aprx(x0);
        double apry(y0);
        double  xl(x0);
        double tt(0), ttt(0);
        double tt1(0), ttt1(0);
if (y0 == y1 && obj[iii].st == 1)
{
        // Определение с условиями аппроксимации
        CString t_n;
        for (int ii = 0; ii < com.size(); ii++)
        {
                t_n = com[ii].name;
                if (t_n.MakeLower() == "all")
                {
                        ap = com[ii].a;
                        bp = com[ii].b;
                        rapapr = com[ii].rapapr;
                        maxsb = com[ii].maxsb;
                        sbavka_max = com[ii].st;
                        break;
                };
        };
        for (int ii = 0; ii < com.size(); ii++)
        {
                if (com[ii].name == obj[iii].name)
```

```
                {
                if (com[ii].st == 101 || com[ii].st == 130 || com[ii].st ==
132)
                        {
                                ap = com[ii].a;
                                bp = com[ii].b;
                                rapapr = com[ii].rapapr;
                                maxsb = com[ii].maxsb;
                                sbavka_max = com[ii].st;
                                break;
                        };
                };
        };
        // Запись в вектор
        report.ax = x0;
        report.ay = y0;
        report.ap = ap;
        report.bp = bp;
        report.maxsb = 0;
        report.rapapr = 0;
        report.kx = x0;
        report.r = ryad;
        report.sbav = 0;
        report.t = 0;
        report.pol = 0;
        report.side_pol = side;
        sbavki.push_back(report);
        if (x0 < x1) // движение слева направо
        {
                report.sbav = floor((x1-x0)/ap + 0.5);
```

```
                report.pol = 1;
                report.ax += report.sbav*report.ap;
        }
        else
        {
                report.sbav = floor((x0-x1)/ap + 0.5);
                report.pol = -1;
                report.ax -= report.sbav*report.ap;
        };
        if (side == 1)
        {report.pol *= -1;};
        aprx = report.ax;
        apry = report.ay;
        report.ap = ap;
        report.bp = bp;
        report.maxsb = maxsb;
        report.rapapr = 0;
        report.kx = x1;
        report.r = ryad;
        report.t = abs((x1-x0)/(aprx-x0));
        report.side_pol = side;
        sbavki.push_back(report);
}
else
{
for (int zzz = 0; ; zzz++) //Аппроксимация по петельному ряду
{
        // Определение с условиями аппроксимации
        CString t_n;
        for (int ii = 0; ii < com.size(); ii++)
```

```
{
        t_n = com[ii].name;
        if (t_n.MakeLower() == "all")
        {
                ap = com[ii].a;
                bp = com[ii].b;
                rapapr = com[ii].rapapr;
                maxsb = com[ii].maxsb;
                sbavka_max = com[ii].st;
                break;
        };
};
for (int ii = 0; ii < com.size(); ii++)
{
        if (com[ii].name == obj[iii].name)
        {
        if (com[ii].st == 101 || com[ii].st == 130 || com[ii].st ==
132)
                {
                        ap = com[ii].a;
                        bp = com[ii].b;
                        rapapr = com[ii].rapapr;
                        maxsb = com[ii].maxsb;
                        sbavka_max = com[ii].st;
                        break;
                };
        };
};
for (int ii = 0; ii < com.size(); ii++)
{
```

```
                    if (com[ii].name == obj[iii].name)
                    {
                        if (com[ii].st == 102 && apry >= com[ii].y2+rapapr1*bp1 && apry <= com[ii].y1 ||
                            com[ii].st == 131 && apry >= com[ii].y2+rapapr1*bp1 && apry <= com[ii].y1 ||
                            com[ii].st == 133 && apry >= com[ii].y2+rapapr1*bp1 && apry <= com[ii].y1)
                        {
                            ap = com[ii].a;
                            bp = com[ii].b;
                            rapapr = com[ii].rapapr;
                            maxsb = com[ii].maxsb;
                            rapapr1 = rapapr;
                            bp1 = bp;
                            sbavka_max = com[ii].st;
                            break;
                        };
                    };
                };
            if (maxsb == 0)
            {
                if (apry <= kony)
                {break;};
                if(zzz >= 1)
                {
                    apry -= bp*rapapr;
                    ryad += rapapr;
                };
```

```
        report.ap = ap;
        report.bp = bp;
        report.maxsb = maxsb;
        report.rapapr = rapapr;
        report.ax = aprx;
        report.ay = apry;
        report.kx = xl;
        report.r = ryad;
        report.t = tt;
        report.side_pol = 1;
        sbavki.push_back(report);
        temp_sb.push_back(report);
}
else
{
        if (apry <= kony)
        {break;};
        if(zzz >= 1)
        {
                apry -= bp*rapapr;
                ryad += rapapr;
        };
        double prov(0);
        // Расчет абсциссы
        for (tt = tt1; tt <= 1; tt += 0.00001)
        {
                switch (obj[iii].st)
                {
```

```
                    case 1: {prov = (1. - tt) * y0 + tt * y1;break;};
            case 2: {prov = (1. - tt) * (1. - tt) * y0 + 2. * tt * (1. - tt) * y1 + tt * tt * y2;
    break;};
            case 3: {prov = (1. - tt) * (1. - tt) * (1. - tt) * y0 + 3. * tt * (1. - tt) * (1. - tt)
* y1 + 3. * tt * tt * (1. - tt) * y2 + tt * tt * tt * y3; break;};
                        default: break;
                };
                if (abs(apry / prov) > 0.9999 && abs(apry / prov) <= 1)
                {
                    tt1 = tt;
                    switch (obj[iii].st)
                    {
                        case 1: {xl = (1. - tt) * x0 + tt * x1;
    break;};
                        case 2: {xl = (1. - tt) * (1. - tt) * x0 + 2. * tt
* (1. - tt) * x1 + tt * tt * x2;     break;};
                        case 3: {xl = (1. - tt) * (1. - tt) * (1. - tt) *
x0 + 3. * tt * (1. - tt) * (1. - tt) * x1 + 3. * tt * tt * (1. - tt) * x2 + tt * tt * tt * x3; break;};
                        default: break;
                    };
                    break;
                };
            };
            if (apry - kony <= bp)
            {
                xl = konx;
                tt = 1;
            };
            // Расчет сбавок
```

```
if (sbavka_max == 132 && ryad > 1 || sbavka_max == 133 && ryad > 1)
{
	if (side == -1)
	{
		aprx += side*maxsb*ap;
	}
	else
	{
		aprx -= side*maxsb*ap;
	};
}
else
{
	if (aprx <= xl && ryad >1)
	{
		min.push_back(aprx);
		if (sbavka_max == 130 || sbavka_max == 131 || sbavka_max == 132 || sbavka_max == 133 )
		{
			aprx += maxsb*ap;
		}
		else
		{
			for (;;)
			{
				aprx += ap;
				maxsb--;
				if (aprx >= xl + ap)
				{break;}
```

```
                    else if (maxsb == 0)
                    {break;};
                    min.push_back(aprx);
                };
            };
            double minxl(0), minxl1(maxsb*ap);
            for (int i = 0; i < min.size(); i++)
            {
                minxl = abs(min[i] - xl);
                if (minxl < minxl1)
                {
                    minxl1 = minxl;
                    aprx = min[i];
                };
            };
        }
        else if (aprx > xl && ryad >1)
        {
            min.push_back(aprx);
            if (sbavka_max == 130 || sbavka_max == 131 ||
sbavka_max == 132 || sbavka_max == 133)
            {
                aprx -= maxsb*ap;
            }
            else
            {
                for (;;)
                {
                    aprx -= ap;
                    maxsb--;
```

```
                        if (aprx <= xl - ap)
                        {break;};
                        if (maxsb == 0)
                        {break;};
                        min.push_back(aprx);
                    };
                };
                double minxl(0), minxl1(maxsb*ap);
                for (int i = 0; i < min.size(); i++)
                {
                    minxl = abs(min[i] - xl);
                    if (minxl < minxl1)
                    {
                        minxl1 = minxl;
                        aprx = min[i];
                    };
                };
            }
            else
            {
                aprx = x0;
                apry = y0;
                ryad = 1;
            };
        };
        min.clear();
        report.ap = ap;
        report.bp = bp;
        report.maxsb = maxsb;
        report.rapapr = rapapr;
```

```
report.ax = floor(aprx*100+0.5)/100;
report.ay = apry;
report.kx = xl;
report.r = ryad;
report.t = tt;
// Сбавка / Прибавка
if (sbavki.size() > 0)
{
    report.sbav = floor(abs(report.ax-sbavki.back().ax)/sbavki.back().ap+0.5);
    if (side < 0)
    {
        if (report.ax - sbavki.back().ax < 0)
        {
            report.pol = -1;
        }
        else
        {
            report.pol = 1;
        };
    }
    else
    {
        if (report.ax - sbavki.back().ax < 0)
        {
            report.pol = 1;
        }
        else
        {
            report.pol = -1;
```

```
                                        };
                                };
                        }
                        else
                        {
                                report.sbav = 0;
                                report.pol = 1;
                        };
                        report.side_pol = side;
                        sbavki.push_back(report);
                        temp_sb.push_back(report);
                        min.clear();
                if (sbavki.size() > 1 && sbavki.back().r == sbavki[sbavki.size() - 2].r)
                        {
                                sbavki.pop_back();
                        };
                };

        };
        };
        // част. вяз.
        /* com[i].a2 - распределение, com[i].a1 - кол-во допол рядов, b - сторона, com[i].a3 - высота ряда */
        CString nit;
        double rasll(0), rasrr(0);
        for (int i = 0; i < com.size(); i++)
        {
                if (com[i].name == obj[iii].name && com[i].st == 222)
                {
```

```
// имя
name_dop = obj[iii].name;


int nl(0), nr(0);
nit = com[i].nit;
if (nit == "1")
{
        com[i].a1 = floor(com[i].a1 / 2 + 0.5);
};
// доп ряды
nl = floor(com[i].a1*(1 - com[i].a2) + 0.5); // левое распр
nr = com[i].a1 - nl; // правое распр
// распр в векторе
double centr(0);
double rasl(0), rasr(0);
centr = floor(com[i].a2*temp_sb.size() + 0.5);
if (abs(nl - nr) == 1 && com[i].a2 == 0.5)
{
        if (nl < nr)
        { nr -= 1; }
        else
        { nl -= 1;};
        sbavki_dop.push_back(temp_sb[centr]);
};
rasl = floor(centr / nl + 0.5);
rasr = floor((temp_sb.size() - centr) / nr +0.5);
rasll = rasl; rasrr = rasr;
 // left
for (int j = 0; j <= centr; j++)
{
```

```
if (nl > 0 && j < centr-1)
{
        if (rasl <= 1)
        {
                sbavki_dop.push_back(temp_sb[j]);
                rasl = rasll;
                nl--;
        }
        else
        {rasl--;};
}
else if (nl - 1 > 0 && j == centr-1)
{
        if (rasl <= 1)
        {
                sbavki_dop.push_back(temp_sb[j]);
                rasl = rasll;
                nl--;
        }
        else
        {rasl--;};
        j = 0;
}
else
{
        if (nl > 0)
        {sbavki_dop.push_back(temp_sb[j]);}
        else
        {break;};
};
```

```
};
 // right
for (int j = temp_sb.size() - 1; j >= centr; j--)
{
        if (nr > 0 && j > centr)
        {
                if (rasr <= 1)
                {
                        sbavki_dop.push_back(temp_sb[j]);
                        rasr = rasrr;
                        nr--;
                }
                else
                {rasr--;};
        }
        else if (nr - 1 > 0 && j == centr+1)
        {
                if (rasr <= 1)
                {
                        sbavki_dop.push_back(temp_sb[j]);
                        rasr = rasrr;
                        nr--;
                }
                else
                {rasr--;};
```

```
                        j = temp_sb.size() - 1;
                    }
                    else
                    {
                        if (nr > 0)
                        {sbavki_dop.push_back(temp_sb[j]);}
                        else
                        {break;};
                    };
                };
            };
        };
        // компоновка
        temp_sb.clear();
        std::vector<int> ryady;
        int count_r(0);
        for (int i = 0; i < sbavki_dop.size(); i++)
        {
            for (int j = 0; j < sbavki_dop.size(); j++)
            {
                if (sbavki_dop[j].r == sbavki_dop[i].r)
                {count_r++;};
            };
            if (ryady.size() > 0)
            {
                int match(0);
                for (int j = 0; j < ryady.size(); j++)
                {
                    if (ryady[j] == sbavki_dop[i].r)
                    {match++;};
```

```
                };
                if (match == 0)
                {
                        sbavki_dop[i].dop_r = count_r;
                        temp_sb.push_back(sbavki_dop[i]);
                };
                ryady.push_back(sbavki_dop[i].r);
        }
        else
        {
                sbavki_dop[i].dop_r = count_r;
                temp_sb.push_back(sbavki_dop[i]);
                ryady.push_back(sbavki_dop[i].r);
        };
        count_r = 0;
};
sbavki_dop.clear();
// перезапись
for (int j = 0; j < temp_sb.size(); j++)
{
        sbavki_dop1.push_back(temp_sb[j]);
};
if (nit == "1")
{
        for (int i = 0; i < sbavki_dop1.size(); i++)
        {
                sbavki_dop1[i].dop_r *= 2;
        };
};
// Расчет величины перемещения координат
```

```
        if (obj[iii].st == 1)
        {
                dx += -x1 + aprx;
                dy += -y1 + apry;
        }
        else if (obj[iii].st == 2)
        {
                dx += -x2 + aprx;
                dy += -y2 + apry;
        }
        else if (obj[iii].st == 3)
        {
                dx += -x3 + aprx;
                dy += -y3 + apry;
        };

};
// Сглаживание сбавок
if (sgl == Ys)
{
        for (int i = 1; i < sbavki.size()-1; i++)
        {
                if (sbavki[i+1].ax - sbavki[i-1].ax == 0)
                {
                        sbavki[i].ax = sbavki[i-1].ax;
                        sbavki[i].sbav = 0;
                        sbavki[i+1].sbav = 0;
                };
        };
};
```

```
}
```

Файл “AprDobj.cpp”

```
#include "StdAfx.h"
#include "AprDobj.h"
CAprDobj::CAprDobj(void)
{
}
CAprDobj::~CAprDobj(void)
{
}
CAprDobj::CAprDobj(std::vector <CReverseObj> objl, std::vector <CReverseObj> objr, std::vector <CObjects> com, CString dname, SGLAZH sgl, CString s_name)
{
    std::vector <CAprObj> temp_left;
    std::vector <CAprObj> temp_right;
    line_r = 0;
    CAprObj report;
    name = dname;
    type = CString(" (Тип: ") + s_name + CString(")");
    for (int j = 0; j < name.GetLength(); j++)
    {
        if (name[j] == '-')
        {
            name_left = name.Mid(0, j);
            name_right = name.Mid(j+1, name.GetLength()-j);
            break;
```

```
    };
};
int ryad(1);
double dx(0), dy(0);
int side(-1);
int sbavka_max(0);
std::vector<double> min;
// Объявление переменных
double x0(0), y0(0), x1(0), y1(0), x2(0), y2(0), x3(0), y3(0);
double ap(0), bp(0);
int rapapr(0), maxsb(0);
double kony(0), konx(0);
double rapapr1(0);
double bp1(0);
        for (int iii = 0; iii < objl.size(); iii++)
        {
                // Перенос данных
                if (objl[iii].st == 1)
                {
                        x0 = objl[iii].x0 + dx;     y0 = objl[iii].y0 + dy;
                        x1 = objl[iii].x1 + dx;     y1 = objl[iii].y1 + dy;
                        kony = y1;
                        konx = x1;
                }
                else if (objl[iii].st == 2)
                {
                        x0 = objl[iii].x0 + dx;     y0 = objl[iii].y0 + dy;
                        x1 = objl[iii].x1 + dx;     y1 = objl[iii].y1 + dy;
                        x2 = objl[iii].x2 + dx;             y2 = objl[iii].y2 + dy;
                        kony = y2;
```

```
            konx = x2;
        }
        else if (objl[iii].st == 3)
        {
            x0 = objl[iii].x0 + dx;     y0 = objl[iii].y0 + dy;
            x1 = objl[iii].x1 + dx;     y1 = objl[iii].y1 + dy;
            x2 = objl[iii].x2 + dx;          y2 = objl[iii].y2 + dy;
            x3 = objl[iii].x3 + dx;          y3 = objl[iii].y3 + dy;
            kony = y3;
            konx = x3;
        };
        double aprx(x0);
        double apry(y0);
        double  xl(x0);
        double tt(0), ttt(0);
        double tt1(0), ttt1(0);
if (y0 == y1)
{
        // Определение с условиями аппроксимации
        CString t_n;
        for (int ii = 0; ii < com.size(); ii++)
        {
            t_n = com[ii].name;
            if (t_n.MakeLower() == "all")
            {
                ap = com[ii].a;
                bp = com[ii].b;
                rapapr = com[ii].rapapr;
                maxsb = com[ii].maxsb;
                sbavka_max = com[ii].st;
```

```
                    break;
                };
            };
            for (int ii = 0; ii < com.size(); ii++)
            {
                if (com[ii].name == objl[iii].name)
                {
                if (com[ii].st == 101 || com[ii].st == 130 || com[ii].st ==
132)
                    {
                        ap = com[ii].a;
                        bp = com[ii].b;
                        rapapr = com[ii].rapapr;
                        maxsb = com[ii].maxsb;
                        sbavka_max = com[ii].st;
                        break;
                    };
                };
            };
            // Запись в вектор
            report.ax = x0;
            report.ay = y0;
            report.ap = ap;
            report.bp = bp;
            report.maxsb = 10000;
            report.rapapr = 0;
            report.kx = x0;
            report.r = ryad;
            report.sbav = 0;
            report.t = 0;
```

```
            report.pol = 1;
            sbavki_left.push_back(report);
            if (x0 < x1) // движение слева направо
            {
                  report.sbav = floor((x1-x0)/ap + 0.5);
                  report.pol = 1;
                  report.ax += report.sbav*report.ap;
            }
            else
            {
                  report.sbav = floor((x0-x1)/ap + 0.5);
                  report.pol = -1;
                  report.ax -= report.sbav*report.ap;
            };
            aprx = report.ax;
            apry = report.ay;
            report.ap = ap;
            report.bp = bp;
            report.maxsb = 10000;
            report.rapapr = 0;
            report.kx = x1;
            report.r = ryad;
            report.t = abs((x1-x0)/(aprx-x0));
            sbavki_left.push_back(report);
      }
      else
      {
      for (int zzz = 0; ; zzz++) //Аппроксимация по петельному ряду
      {
            // Определение с условиями аппроксимации
```

```
CString t_n;
for (int ii = 0; ii < com.size(); ii++)
{
	t_n = com[ii].name;
	if (t_n.MakeLower() == "all")
	{
		ap = com[ii].a;
		bp = com[ii].b;
		rapapr = com[ii].rapapr;
		maxsb = com[ii].maxsb;
		sbavka_max = com[ii].st;
		break;
	};
};
for (int ii = 0; ii < com.size(); ii++)
{
	if (com[ii].name == objl[iii].name)
	{
	if (com[ii].st == 101 || com[ii].st == 130 || com[ii].st ==
132)
		{
			ap = com[ii].a;
			bp = com[ii].b;
			rapapr = com[ii].rapapr;
			maxsb = com[ii].maxsb;
			sbavka_max = com[ii].st;
			break;
		};
	};
};
```

```
                for (int ii = 0; ii < com.size(); ii++)
                {
                    if (com[ii].name == objl[iii].name)
                    {
                        if (com[ii].st == 102 && apry >= com[ii].y2+rapapr1*bp1 && apry <= com[ii].y1 ||
                                com[ii].st == 131 && apry >= com[ii].y2+rapapr1*bp1 && apry <= com[ii].y1 ||
                                com[ii].st == 133 && apry >= com[ii].y2+rapapr1*bp1 && apry <= com[ii].y1)
                        {
                            ap = com[ii].a;
                            bp = com[ii].b;
                            rapapr = com[ii].rapapr;
                            maxsb = com[ii].maxsb;
                            rapapr1 = rapapr;
                            bp1 = bp;
                            sbavka_max = com[ii].st;
                            break;
                        };
                    };
                };
            if (maxsb == 0)
            {
                if (apry <= kony)
                {break;};
                if(zzz >= 1)
                {
                    apry -= bp*rapapr;
                    ryad += rapapr;
```

```
                };
                report.ap = ap;
                report.bp = bp;
                report.maxsb = maxsb;
                report.rapapr = rapapr;
                report.ax = aprx;
                report.ay = apry;
                report.kx = xl;
                report.r = ryad;
                report.t = tt;
                temp_left.push_back(report);
        }
        else
        {
                if (apry <= kony)
                {break;};
                if(zzz >= 1)
                {
                        apry -= bp*rapapr;
                        ryad += rapapr;
                };
                double prov(0);
                // Расчет абсциссы
                for (tt = tt1; tt <= 1; tt += 0.00001)
                {
                        switch (objl[iii].st)
                        {
                                case 1: {prov = (1. - tt) * y0 + tt * y1;break;};
                                case 2: {prov = (1. - tt) * (1. - tt) * y0 + 2. * tt *
(1. - tt) * y1 + tt * tt * y2;        break;};
```

```
                                    case 3: {prov = (1. - tt) * (1. - tt) * (1. - tt) * y0 +
3. * tt * (1. - tt) * (1. - tt) * y1 + 3. * tt * tt * (1. - tt) * y2 + tt * tt * tt * y3; break;};
                                    default: break;
                                };
                                if (abs(apry / prov) > 0.9999 && abs(apry / prov) <= 1)
                                {
                                    tt1 = tt;
                                    switch (objl[iii].st)
                                    {
                                        case 1: {xl = (1. - tt) * x0 + tt * x1;
        break;};
                                        case 2: {xl = (1. - tt) * (1. - tt) * x0 + 2. * tt
* (1. - tt) * x1 + tt * tt * x2;      break;};
                                        case 3: {xl = (1. - tt) * (1. - tt) * (1. - tt) *
x0 + 3. * tt * (1. - tt) * (1. - tt) * x1 + 3. * tt * tt * (1. - tt) * x2 + tt * tt * tt * x3; break;};
                                        default: break;
                                    };
                                    break;
                                };
                            };
                            if (abs(kony-apry) <= bp)
                            {
                                xl = konx;
                                tt = 1;
                            };

                            // Расчет сбавок
                        if (sbavka_max == 132 && ryad > 1 || sbavka_max == 133 && ryad
> 1)
                            {
```

```
                if (side == -1)
                {
                        aprx += side*maxsb*ap;
                }
                else
                {
                        aprx -= side*maxsb*ap;
                };
        }
        else
        {
                if (aprx <= xl && ryad >1)
                {
                        min.push_back(aprx);

                        if (sbavka_max == 130 || sbavka_max == 131 ||
sbavka_max == 132)
                        {
                                aprx += maxsb*ap;
                        }
                        else
                        {
                                for (;;)
                                {
                                        aprx += ap;
                                        maxsb--;

                                        if (aprx >= xl + ap)
                                        {break;}
                                        else if (maxsb == 0)
```

```
                    {break;};
                    min.push_back(aprx);
                };
            };
            double minxl(0), minxl1(maxsb*ap);
            for (int i = 0; i < min.size(); i++)
            {
                minxl = abs(min[i] - xl);
                if (minxl < minxl1)
                {
                    minxl1 = minxl;
                    aprx = min[i];
                };
            };
        }
        else if (aprx > xl && ryad >1)
        {
            min.push_back(aprx);
            if (sbavka_max == 130 || sbavka_max == 131 ||
sbavka_max == 132)
            {
                aprx -= maxsb*ap;
            }
            else
            {
                for (;;)
                {
                    aprx -= ap;
                    maxsb--;
```

```
                        if (aprx <= xl - ap)
                        {break;};
                        if (maxsb == 0)
                        {break;};
                        min.push_back(aprx);
                    };
                };

                double minxl(0), minxl1(maxsb*ap);
                for (int i = 0; i < min.size(); i++)
                {
                    minxl = abs(min[i] - xl);
                    if (minxl < minxl1)
                    {
                        minxl1 = minxl;
                        aprx = min[i];
                    };
                };
            }
            else
            {
                aprx = x0;
                apry = y0;
                ryad = 1;
            };
        };
        min.clear();
        report.ap = ap;
        report.bp = bp;
        report.maxsb = maxsb;
```

```
report.rapapr = rapapr;
report.ax = floor(aprx*100+0.5)/100;
report.ay = apry;
report.kx = xl;
report.r = ryad;
report.t = tt;


// Сбавка / Прибавка
if (temp_left.size() > 0)
{
        report.sbav = floor(abs(report.ax-
temp_left.back().ax)/temp_left.back().ap+0.5);
        if (side < 0)
        {
                if (report.ax - temp_left.back().ax < 0)
                {
                        report.pol = -1;
                }
                else
                {
                        report.pol = 1;
                };
        }
        else
        {
                if (report.ax - temp_left.back().ax < 0)
                {
                        report.pol = 1;
                }
```

```
                                else
                                {
                                        report.pol = -1;
                                };
                        };
                }
                else
                {
                        report.sbav = 0;
                        report.pol = 1;
                };
                temp_left.push_back(report);
                min.clear();
                if (sbavki_left.size() > 1 && sbavki_left.back().r == sbavki_left[sbavki_left.size() - 2].r)
                {
                        sbavki_left.pop_back();
                };
        };
};
};
        // Расчет величины перемещения координат
        if (objl[iii].st == 1)
        {
                dx += -x1 + aprx;
                dy += -y1 + apry;
        }
        else if (objl[iii].st == 2)
        {
                dx += -x2 + aprx;
```

```
                dy += -y2 + apry;
        }
        else if (objl[iii].st == 3)
        {
                dx += -x3 + aprx;
                dy += -y3 + apry;
        }

};
// Сглаживание сбавок
if (sgl == Ys)
{
        for (int i = 1; i < temp_left.size()-1; i++)
        {
                if (temp_left[i+1].ax - temp_left[i-1].ax == 0)
                {
                        temp_left[i].ax = temp_left[i-1].ax;
                        temp_left[i].sbav = 0;
                        temp_left[i+1].sbav = 0;
                };
        };
};
        ryad = 1;
        dx = 0; dy = 0;
        side = 1;
        for (int iii = 0; iii < objr.size(); iii++)
        {
                // Перенос данных

                if (objr[iii].st == 1)
```

```
        {
            x0 = objr[iii].x0 + dx;     y0 = objr[iii].y0 + dy;
            x1 = objr[iii].x1 + dx;     y1 = objr[iii].y1 + dy;
            kony = y1;
            konx = x1;
        }
        else if (objr[iii].st == 2)
        {
            x0 = objr[iii].x0 + dx;     y0 = objr[iii].y0 + dy;
            x1 = objr[iii].x1 + dx;     y1 = objr[iii].y1 + dy;
            x2 = objr[iii].x2 + dx;          y2 = objr[iii].y2 + dy;
            kony = y2;
            konx = x2;
        }
        else if (objr[iii].st == 3)
        {
            x0 = objr[iii].x0 + dx;     y0 = objr[iii].y0 + dy;
            x1 = objr[iii].x1 + dx;     y1 = objr[iii].y1 + dy;
            x2 = objr[iii].x2 + dx;          y2 = objr[iii].y2 + dy;
            x3 = objr[iii].x3 + dx;          y3 = objr[iii].y3 + dy;
            kony = y3;
            konx = x3;
        };
        double aprx(x0);
        double apry(y0);
        double  xl(x0);
        double tt(0), ttt(0);
        double tt1(0), ttt1(0);
if (y0 == y1)
{
```

```
// Определение с условиями аппроксимации
CString t_n;
for (int ii = 0; ii < com.size(); ii++)
{
	t_n = com[ii].name;
	if (t_n.MakeLower() == "all")
	{
		ap = com[ii].a;
		bp = com[ii].b;
		rapapr = com[ii].rapapr;
		maxsb = com[ii].maxsb;
		sbavka_max = com[ii].st;
		break;
	};
};
for (int ii = 0; ii < com.size(); ii++)
{
	if (com[ii].name == objr[iii].name)
	{
		if (com[ii].st == 101 || com[ii].st == 130 ||
com[ii].st == 132)
		{
			ap = com[ii].a;
			bp = com[ii].b;
			rapapr = com[ii].rapapr;
			maxsb = com[ii].maxsb;
			sbavka_max = com[ii].st;
			break;
		};
	};
```

```
};
// Запись в вектор
report.ax = x0;
report.ay = y0;
report.ap = ap;
report.bp = bp;
report.maxsb = 10000;
report.rapapr = 0;
report.kx = x0;
report.r = ryad;
report.sbav = 0;
report.t = 0;
report.pol = 1;
sbavki_right.push_back(report);
if (x0 < x1) // движение слева направо
{
        report.sbav = floor((x1-x0)/ap + 0.5);
        report.pol = 1;
        report.ax += report.sbav*report.ap;
}
else
{
        report.sbav = floor((x0-x1)/ap + 0.5);
        report.pol = -1;
        report.ax -= report.sbav*report.ap;
};
aprx = report.ax;
apry = report.ay;
report.ap = ap;
report.bp = bp;
```

```
            report.maxsb = 10000;
            report.rapapr = 0;
            report.kx = x1;
            report.r = ryad;
            report.t = abs((x1-x0)/(aprx-x0));
            sbavki_right.push_back(report);
}
else
{
for (int zzz = 0; ; zzz++) //Аппроксимация по петельному ряду
{
            // Определение с условиями аппроксимации
            CString t_n;
            for (int ii = 0; ii < com.size(); ii++)
            {
                        t_n = com[ii].name;
                        if (t_n.MakeLower() == "all")
                        {
                                    ap = com[ii].a;
                                    bp = com[ii].b;
                                    rapapr = com[ii].rapapr;
                                    maxsb = com[ii].maxsb;
                                    sbavka_max = com[ii].st;
                                    break;
                        };
            };
            for (int ii = 0; ii < com.size(); ii++)
            {
                        if (com[ii].name == objr[iii].name)
                        {
```

```
                    if (com[ii].st == 101 || com[ii].st == 130 || com[ii].st ==
132)
                        {
                            ap = com[ii].a;
                            bp = com[ii].b;
                            rapapr = com[ii].rapapr;
                            maxsb = com[ii].maxsb;
                            sbavka_max = com[ii].st;
                            break;
                        };
                    };
                };
                for (int ii = 0; ii < com.size(); ii++)
                {
                    if (com[ii].name == objr[iii].name)
                    {
                        if (com[ii].st == 102 && apry >=
com[ii].y2+rapapr1*bp1 && apry <= com[ii].y1 ||
                            com[ii].st == 131 && apry >=
com[ii].y2+rapapr1*bp1 && apry <= com[ii].y1 ||
                            com[ii].st == 133 && apry >=
com[ii].y2+rapapr1*bp1 && apry <= com[ii].y1)
                        {
                            ap = com[ii].a;
                            bp = com[ii].b;
                            rapapr = com[ii].rapapr;
                            maxsb = com[ii].maxsb;
                            rapapr1 = rapapr;
                            bp1 = bp;
                            sbavka_max = com[ii].st;
```

```
                    break;
                };
            };
        };
if (maxsb == 0)
{
        if (apry <= kony)
        {break;};

        if(zzz >= 1)
        {
            apry -= bp*rapapr;
            ryad += rapapr;
        };
        report.ap = ap;
        report.bp = bp;
        report.maxsb = maxsb;
        report.rapapr = rapapr;
        report.ax = aprx;
        report.ay = apry;
        report.kx = xl;
        report.r = ryad;
        report.t = tt;
        temp_right.push_back(report);
}
else
{
        if (apry <= kony)
        {break;};
```

```
if(zzz >= 1)
{
        apry -= bp*rapapr;
        ryad += rapapr;
};
double prov(0);
// Расчет абсциссы
for (tt = tt1; tt <= 1; tt += 0.00001)
{
        switch (objr[iii].st)
        {
                case 1: {prov = (1. - tt) * y0 + tt * y1;break;};
                case 2: {prov = (1. - tt) * (1. - tt) * y0 + 2. * tt *
(1. - tt) * y1 + tt * tt * y2;        break;};
                case 3: {prov = (1. - tt) * (1. - tt) * (1. - tt) * y0 +
3. * tt * (1. - tt) * (1. - tt) * y1 + 3. * tt * tt * (1. - tt) * y2 + tt * tt * tt * y3; break;};
                default: break;
        };
        if (abs(apry / prov) > 0.9999 && abs(apry / prov) <= 1)
        {
                tt1 = tt;
                switch (objr[iii].st)
                {
                        case 1: {xl = (1. - tt) * x0 + tt * x1;
    break;};
                        case 2: {xl = (1. - tt) * (1. - tt) * x0 + 2. * tt
* (1. - tt) * x1 + tt * tt * x2;       break;};
                        case 3: {xl = (1. - tt) * (1. - tt) * (1. - tt) *
x0 + 3. * tt * (1. - tt) * (1. - tt) * x1 + 3. * tt * tt * (1. - tt) * x2 + tt * tt * tt * x3; break;};
                        default: break;
```

```
                        };
                        break;
                    };
                };
                if (abs(kony-apry) <= bp)
                {
                    xl = konx;
                    tt = 1;
                };

                // Расчет сбавок
            if (sbavka_max == 132 && ryad > 1 || sbavka_max == 133 && ryad
> 1)
                {
                    if (side == 1)
                    {
                        aprx += side*maxsb*ap;
                    }
                    else
                    {
                        aprx -= side*maxsb*ap;
                    };
                }
                else
                {
                    if (aprx <= xl && ryad >1)
                    {
                        min.push_back(aprx);
```

```
if (sbavka_max == 130 || sbavka_max == 131 ||
sbavka_max == 132)
{
    aprx += maxsb*ap;
}
else
{
    for (;;)
    {
        aprx += ap;
        maxsb--;

        if (aprx >= xl + ap)
        {break;}
        else if (maxsb == 0)
        {break;};
        min.push_back(aprx);
    };
};
double minxl(0), minxl1(maxsb*ap);
for (int i = 0; i < min.size(); i++)
{
    minxl = abs(min[i] - xl);
    if (minxl < minxl1)
    {
        minxl1 = minxl;
        aprx = min[i];
    };
};
}
```

```
else if (aprx > xl && ryad >1)
{
        min.push_back(aprx);
        if (sbavka_max == 130 || sbavka_max == 131 ||
sbavka_max == 132)
        {
                aprx -= maxsb*ap;
        }
        else
        {
                for (;;)
                {
                        aprx -= ap;
                        maxsb--;

                        if (aprx <= xl - ap)
                        {break;};
                        if (maxsb == 0)
                        {break;};
                        min.push_back(aprx);
                };
        };
        double minxl(0), minxl1(maxsb*ap);
        for (int i = 0; i < min.size(); i++)
        {
                minxl = abs(min[i] - xl);
                if (minxl < minxl1)
                {
                        minxl1 = minxl;
                        aprx = min[i];
```

```
                    };
                };
            }
            else
            {
                aprx = x0;
                apry = y0;
                ryad = 1;
            };
        };
        min.clear();
        report.ap = ap;
        report.bp = bp;
        report.maxsb = maxsb;
        report.rapapr = rapapr;
        report.ax = floor(aprx*100+0.5)/100;
        report.ay = apry;
        report.kx = xl;
        report.r = ryad;
        report.t = tt;
        // Сбавка / Прибавка
        if (temp_right.size() > 0)
        {
            report.sbav = floor(abs(report.ax-
temp_right.back().ax)/temp_right.back().ap + 0.5);
            if (side < 0)
            {
                if (report.ax - temp_right.back().ax < 0)
                {
                    report.pol = -1;
```

```
                }
                else
                {
                    report.pol = 1;
                };
            }
            else
            {
                if (report.ax - temp_right.back().ax < 0)
                {
                    report.pol = -1;
                }
                else
                {
                    report.pol = 1;
                };
            };
        }
        else
        {
            report.sbav = 0;
            report.pol = 1;
        };
        temp_right.push_back(report);
        min.clear();
        if (sbavki_right.size() > 1 && sbavki_right.back().r ==
sbavki_right[sbavki_right.size() - 2].r)
        {
            sbavki_right.pop_back();
        };
```

```
        };
};
};
        // Расчет величины перемещения координат
        if (objr[iii].st == 1)
        {
                dx += -x1 + aprx;
                dy += -y1 + apry;
        }
        else if (objr[iii].st == 2)
        {
                dx += -x2 + aprx;
                dy += -y2 + apry;
        }
        else if (objr[iii].st == 3)
        {
                dx += -x3 + aprx;
                dy += -y3 + apry;
        }

};
// Сглаживание сбавок
if (sgl == Ys)
{
        for (int i = 1; i < temp_right.size()-1; i++)
        {
                if (temp_right[i+1].ax - temp_right[i-1].ax == 0)
                {
                        temp_right[i].ax = temp_right[i-1].ax;
                        temp_right[i].sbav = 0;
```

```
                        temp_right[i+1].sbav = 0;
                    };
                };
            };
            //Определение положения линии разностей
            double l1(0);
            double l2(0);
            int minimum(0);
            if (temp_left.size() > temp_right.size())
            {minimum = temp_right.size()-1;}
            else
            {minimum = temp_left.size()-1;};
            double tlr(0);
            for (int i = 0; i < minimum; i++)
            {
                l1 += sqrt((temp_left[i].ax - temp_left[i+1].ax)*(temp_left[i].ax - temp_left[i+1].ax) + (temp_left[i].ay - temp_left[i+1].ay)*(temp_left[i].ay - temp_left[i+1].ay));
                l2 += sqrt((temp_right[i].ax - temp_right[i+1].ax)*(temp_right[i].ax - temp_right[i+1].ax) + (temp_right[i].ay - temp_right[i+1].ay)*(temp_right[i].ay - temp_right[i+1].ay));
                tlr = abs(l1-l2);
                if (temp_left[i].bp == temp_right[i].bp)
                {
                    if (tlr >= temp_right[i].bp*0.6)
                    {
                        line_r = i+1;
                        break;
                    };
                }
```

```
                else
                {
                AfxMessageBox(CString("Различная высота рядов! Проверьте параметры аппроксимации!"),MB_OK);
                        break;
                };
        };
        //Разность по рядам
        rzs = temp_left.back().r-temp_right.back().r;
        int t_rzs(rzs);
        CAprObj dop_e;

        int count_ay(0);
        int count_r(0);
        int rasp(0);
        if (rzs != 0)
        {rze = abs((temp_right.back().r - line_r)/rzs);}
        else
        {rze = 0;};
        // Уравнивание количества соединяемых элементов
        // по столбикам и рядам
        if (s_name == "10")
        {
                int count_sbavok(0);
                int c_r(0);
                for (int i = 0; i < temp_left.size(); i++)
                {
                        if (temp_left[i].sbav == 0)
                        {
                                count_sbavok ++;
```

```
        }
        else
        {
                count_sbavok += temp_left[i].sbav;
        };
};
c_r = temp_right.back().r - temp_right[0].r + 1;
// Обнуление сбавок temp_right
for (int i = 1; i < temp_right.size(); i++)
{
        temp_right[i].ax = temp_right[0].ax;
        temp_right[i].sbav = 0;
};
if (c_r > count_sbavok)
{
        for (;;)
        {
                temp_right.pop_back();
                c_r--;
                if (c_r <= count_sbavok)
                {
                        break;
                };
        };
}
else if (c_r < count_sbavok)
{
        for (;;)
        {
                dop_e.ap = temp_right.back().ap;
```

```
                        dop_e.bp = temp_right.back().bp;
                        dop_e.maxsb = temp_right.back().maxsb;
                        dop_e.rapapr = temp_right.back().rapapr;
                        dop_e.ax = temp_right.back().ax;
                        dop_e.ay = temp_right.back().ay -
temp_right.back().bp;
                        dop_e.kx = temp_right.back().kx;
                        dop_e.r = temp_right.back().r + 1;
                        dop_e.sbav = 0;
                        dop_e.t = temp_right.back().t;
                        dop_e.pol = temp_right.back().pol;
                        temp_right.push_back(dop_e);
                        c_r += temp_right.back().rapapr;
                        if (c_r >= count_sbavok)
                        {break;};
                    };
                };
                // Запись векторов отчета
                for (int i = 0; i < temp_left.size(); i++)
                {
                    sbavki_left.push_back(temp_left[i]);
                };
                for (int i = 0; i < temp_right.size(); i++)
                {
                    sbavki_right.push_back(temp_right[i]);
                };
            }
            else if (s_name == "01")
            {
                int count_sbavok(0);
```

```
int c_r(0);
for (int i = 0; i < temp_right.size(); i++)
{
	if (temp_right[i].sbav == 0)
	{
		count_sbavok ++;
	}
	else
	{
		count_sbavok += temp_right[i].sbav;
	};
};

c_r = temp_left.back().r - temp_left[0].r + 1;
// Обнуление сбавок temp_left
for (int i = 1; i < temp_left.size(); i++)
{
	temp_left[i].ax = temp_left[0].ax;
	temp_left[i].sbav = 0;
};
if (c_r > count_sbavok)
{
	for (;;)
	{
		temp_left.pop_back();
		c_r--;
		if (c_r <= count_sbavok)
		{
			break;
		};
```

```
                };
            }
            else if (c_r < count_sbavok)
            {
                for (;;)
                {
                    dop_e.ap = temp_left.back().ap;
                    dop_e.bp = temp_left.back().bp;
                    dop_e.maxsb = temp_left.back().maxsb;
                    dop_e.rapapr = temp_left.back().rapapr;
                    dop_e.ax = temp_left.back().ax;
                    dop_e.ay = temp_left.back().ay -
temp_left.back().bp;
                    dop_e.kx = temp_left.back().kx;
                    dop_e.r = temp_left.back().r + 1;
                    dop_e.sbav = 0;
                    dop_e.t = temp_left.back().t;
                    dop_e.pol = temp_left.back().pol;
                    temp_left.push_back(dop_e);
                    c_r += temp_left.back().rapapr;
                    if (c_r >= count_sbavok)
                    {break;};
                };
            };
            // Запись векторов отчета
            for (int i = 0; i < temp_left.size(); i++)
            {
                sbavki_left.push_back(temp_left[i]);
            };
            for (int i = 0; i < temp_right.size(); i++)
```

```
                {
                        sbavki_right.push_back(temp_right[i]);
                };
        }
        else if (s_name == "11")
        {
        // по рядам (добав ряды согласно равномерному распределению)
        if (rzs > 0)
        {
                for (int i = 0; i < temp_right.size(); i++)
                {
                        if (temp_right[i].r > line_r)
                        {rasp++;};

                        // по условию распределения
                        if (temp_right[i].r > line_r && rasp == rze)
                        {
                                dop_e.ap = temp_right[i].ap;
                                dop_e.bp = temp_right[i].bp;
                                dop_e.maxsb = temp_right[i].maxsb;
                                dop_e.rapapr = temp_right[i].rapapr;
                                dop_e.ax = temp_right[i].ax;
                                dop_e.ay = temp_right[i].ay-
count_ay*temp_right[i].bp;
                                dop_e.kx = temp_right[i].kx;
                                dop_e.r = temp_right[i].r+count_r;
                                dop_e.sbav = temp_right[i].sbav;
                                dop_e.t = temp_right[i].t;
                                dop_e.pol = temp_right[i].pol;
                                sbavki_right.push_back(dop_e);
```

```
                count_ay++;
                if (rzs != 0)
                {
                        dop_e.ap = temp_right[i].ap;
                        dop_e.bp = temp_right[i].bp;
                        dop_e.maxsb = 0;
                        dop_e.rapapr = 1;
                        dop_e.ax = temp_right[i].ax;
                        dop_e.ay = temp_right[i].ay-
count_ay*temp_right[i].bp;
                        dop_e.kx = temp_right[i].kx;
                        dop_e.r++;
                        dop_e.sbav = 0;
                        dop_e.t = 777;
                        dop_e.pol = temp_right[i].pol;

                        sbavki_right.push_back(dop_e);
                };
                count_r++;
                rzs--;
                rasp = 0;
        }
        else
        {
                dop_e.ap = temp_right[i].ap;
                dop_e.bp = temp_right[i].bp;
                dop_e.maxsb = temp_right[i].maxsb;
                dop_e.rapapr = temp_right[i].rapapr;
                dop_e.ax = temp_right[i].ax;
```

```
                                dop_e.ay = temp_right[i].ay-
count_ay*temp_right[i].bp;
                                dop_e.kx = temp_right[i].kx;
                                dop_e.r = temp_right[i].r+count_r;
                                dop_e.sbav = temp_right[i].sbav;
                                dop_e.t = temp_right[i].t;
                                dop_e.pol = temp_right[i].pol;
                                sbavki_right.push_back(dop_e);
                        };
                };
                if (rzs > 0)
                {
                        for (;;)
                        {
                                if (rzs == 0)
                                {break;};

                                count_ay++;
                                dop_e.ap = temp_right.back().ap;
                                dop_e.bp = temp_right.back().bp;
                                dop_e.maxsb = 0;
                                dop_e.rapapr = 1;
                                dop_e.ax = temp_right.back().ax;
                                dop_e.ay = temp_right.back().ay-
count_ay*temp_right.back().bp;
                                dop_e.kx = temp_right.back().kx;
                                dop_e.r++;
                                dop_e.sbav = 0;
                                dop_e.t = 777;
                                dop_e.pol = temp_right.back().pol;
```

```
                    sbavki_right.push_back(dop_e);
                    rzs--;
                };
            };
            for (int i = 0; i <temp_left.size(); i++)
            {
                sbavki_left.push_back(temp_left[i]);
            };
        }
        else if (rzs < 0)
        {
            for (int i = 0; i < temp_left.size(); i++)
            {
                if (temp_left[i].r > line_r)
                {rasp++;};
                if (temp_left[i].r > line_r && rasp == rze && rzs <= 0)
                {

                    dop_e.ap = temp_left[i].ap;
                    dop_e.bp = temp_left[i].bp;
                    dop_e.maxsb = temp_left[i].maxsb;
                    dop_e.rapapr = temp_left[i].rapapr;
                    dop_e.ax = temp_left[i].ax;
                    dop_e.ay = temp_left[i].ay-
count_ay*temp_left[i].bp;
                    dop_e.kx = temp_left[i].kx;
                    dop_e.r = temp_left[i].r+count_r;
                    dop_e.sbav = temp_left[i].sbav;
                    dop_e.t = temp_left[i].t;
                    dop_e.pol = temp_left[i].pol;
```

```
                    sbavki_left.push_back(dop_e);
                    count_ay++;
                    if (rzs != 0)
                    {
                        dop_e.ap = temp_left[i].ap;
                        dop_e.bp = temp_left[i].bp;
                        dop_e.maxsb = 0;
                        dop_e.rapapr = 1;
                        dop_e.ax = temp_left[i].ax;
                        dop_e.ay = temp_left[i].ay-
count_ay*temp_left[i].bp;
                        dop_e.kx = temp_left[i].kx;
                        dop_e.r++;
                        dop_e.sbav = 0;
                        dop_e.t = 777;
                        dop_e.pol = temp_left[i].pol;
                        sbavki_left.push_back(dop_e);
                    };

                    count_r++;
                    rzs++;
                    rasp = 0;
                }
                else
                {
                    dop_e.ap = temp_left[i].ap;
                    dop_e.bp = temp_left[i].bp;
                    dop_e.maxsb = temp_left[i].maxsb;
                    dop_e.rapapr = temp_left[i].rapapr;
                    dop_e.ax = temp_left[i].ax;
```

```
                    dop_e.ay = temp_left[i].ay-count_ay*temp_left[i].bp;
                    dop_e.kx = temp_left[i].kx;
                    dop_e.r = temp_left[i].r+count_r;
                    dop_e.sbav = temp_left[i].sbav;
                    dop_e.t = temp_left[i].t;
                    dop_e.pol = temp_left[i].pol;
                    sbavki_left.push_back(dop_e);
                };
            };
            if (rzs < 0)
            {
                for (;;)
                {
                    if (rzs == 0)
                    {break;};
                    count_ay++;
                    dop_e.ap = temp_left.back().ap;
                    dop_e.bp = temp_left.back().bp;
                    dop_e.maxsb = 0;
                    dop_e.rapapr = 1;
                    dop_e.ax = temp_left.back().ax;
                    dop_e.ay = temp_left.back().ay-count_ay*temp_left.back().bp;
                    dop_e.kx = temp_left.back().kx;
                    dop_e.r++;
                    dop_e.sbav = 0;
                    dop_e.t = 777;
                    dop_e.pol = temp_left.back().pol;
                    sbavki_left.push_back(dop_e);
```

```
                        rzs++;
                    };
                };
                for (int i = 0; i <temp_right.size(); i++)
                {
                    sbavki_right.push_back(temp_right[i]);
                };
            }
            else
            {
                // Запись векторов отчета
                for (int i = 0; i <temp_left.size(); i++)
                {
                    sbavki_left.push_back(temp_left[i]);
                };
                for (int i = 0; i < temp_right.size(); i++)
                {
                    sbavki_right.push_back(temp_right[i]);
                };
            };
            };
            rzs = t_rzs;
}
```

Файл “AprObj.cpp”

```
#include "StdAfx.h"
#include "AprObj.h"
CAprObj::CAprObj(void)
{
}
```

```
CAprObj::~CAprObj(void)
{
}
```

Файл “Console.cpp”

```
#include "stdafx.h"
#include "Designer k-wear.h"
#include "Console.h"
#include "afxdialogex.h"
#include "Designer k-wearDoc.h"
// диалоговое окно CConsole
IMPLEMENT_DYNAMIC(CConsole, CDialog)
CConsole::CConsole(CWnd* pParent /*=NULL*/)
	: CDialog(CConsole::IDD, pParent), m_Check(STAND)
{
}
CConsole::~CConsole()
{
}
void CConsole::DoDataExchange(CDataExchange* pDX)
{
	CDialog::DoDataExchange(pDX);
	DDX_Control(pDX, IDC_EDIT1, m_EditConsole);
	DDX_Control(pDX, IDC_EDIT2, m_Edit_Info);
	DDX_Control(pDX, IDC_CHECK1, check1);
	DDX_Control(pDX, IDC_CHECK2, check2);
}
BEGIN_MESSAGE_MAP(CConsole, CDialog)
	ON_EN_CHANGE(IDC_EDIT1, &CConsole::OnEnChangeEdit1)
	ON_BN_CLICKED(IDC_CHECK1, &CConsole::OnBnClickedCheck1)
```

```
        ON_BN_CLICKED(IDC_CHECK2, &CConsole::OnBnClickedCheck2)
END_MESSAGE_MAP()
// обработчики сообщений CConsole
void CConsole::OnEnChangeEdit1()
{
        buff_inp.clear(); // Очистка вектора поля ввода CEdit
        int nLine = m_EditConsole.GetLineCount();
        CString aString;
        std::vector <CString> buffer;
        buffer_string.Empty();
        serial.clear();
        int jj(1);
        CString per("@");
        // Считывание строк из CEdit'a
 for (int i = 0; i < nLine; ++i)
 {
        int nLength = m_EditConsole.LineLength(m_EditConsole.LineIndex(i));
        m_EditConsole.GetLine(i, aString.GetBuffer(nLength), nLength);
  aString.ReleaseBuffer(nLength);
  aString.GetBuffer(0); // передаю полученную строку

        buffer_string += aString;
        serial.push_back(aString + per);
 }
  // Удаление пробелов
 CString bString;
 for (int j = 0; j < buffer_string.GetLength(); j++)
 {
          if (buffer_string[j] != ' ')
          {
```

```
                bString += buffer_string[j];
            };
    };
    buffer_string.Empty();
    buffer_string = bString;
// Формирование списка команд
    for (int j = 0; j < buffer_string.GetLength(); j++)
    {
            jj++;
            if (buffer_string[j] == ';')
            {
                    jj-=2;
                    buff_inp.push_back(buffer_string.Mid(j-jj, jj+1));
                    jj = 1;
            }
    };
}
int ch(0);
int ch1(0);
void CConsole::OnBnClickedCheck1()
{
        ch++;
        if (ch == 1 && ch1 == 0)
        {
                m_Check = ZAMER;
                check2.EnableWindow(false);
        }
        else
        {
                ch = 0;
```

```
            m_Check = STAND;
            check2.EnableWindow(true);
        };
}
void CConsole::OnBnClickedCheck2()
{
        ch1++;
        if (ch1 == 1 && ch == 0)
        {
            m_Check = GRAFIKA;
            check1.EnableWindow(false);
        }
        else
        {
            ch1 = 0;
            m_Check = STAND;
            check1.EnableWindow(true);
        };
}
```

Файл “Designer k-wear.cpp”

```
#include "stdafx.h"
#include "afxwinappex.h"
#include "afxdialogex.h"
#include "Designer k-wear.h"
#include "MainFrm.h"
#include "Designer k-wearDoc.h"
#include "Designer k-wearView.h"
#ifdef _DEBUG
#define new DEBUG_NEW
#endif
```

```
// CDesignerkwearApp
BEGIN_MESSAGE_MAP(CDesignerkwearApp, CWinAppEx)
	ON_COMMAND(ID_APP_ABOUT, &CDesignerkwearApp::OnAppAbout)
	// Стандартные команды по работе с файлами документов
	ON_COMMAND(ID_FILE_NEW, &CWinAppEx::OnFileNew)
	ON_COMMAND(ID_FILE_OPEN, &CWinAppEx::OnFileOpen)
	// Стандартная команда настройки печати
	ON_COMMAND(ID_FILE_PRINT_SETUP, &CWinAppEx::OnFilePrintSetup)
END_MESSAGE_MAP()
// создание CDesignerkwearApp
CDesignerkwearApp::CDesignerkwearApp()
{
	m_bHiColorIcons = TRUE;
	// поддержка диспетчера перезагрузки
	m_dwRestartManagerSupportFlags = AFX_RESTART_MANAGER_SUPPORT_ALL_ASPECTS;
#ifdef _MANAGED
	// Если приложение построено с поддержкой среды Common Language Runtime (/clr):
	//	1) Этот дополнительный параметр требуется для правильной поддержки работы диспетчера перезагрузки.
	//	2) В своем проекте для построения необходимо добавить ссылку на System.Windows.Forms.
	System::Windows::Forms::Application::SetUnhandledExceptionMode(System::Windows::Forms::UnhandledExceptionMode::ThrowException);
#endif
	// TODO: замените ниже строку идентификатора приложения строкой уникального идентификатора; рекомендуемый
```

```
        // формат для строки: ИмяКомпании.ИмяПродукта.СубПродукт.СведенияОВерсии
        SetAppID(_T("Designer k-wear.AppID.NoVersion"));

        // TODO: добавьте код создания,
        // Размещает весь важный код инициализации в InitInstance
    }
    // Единственный объект CDesignerkwearApp
    CDesignerkwearApp theApp;
    // инициализация CDesignerkwearApp
    BOOL CDesignerkwearApp::InitInstance()
    {
        // InitCommonControlsEx() требуются для Windows XP, если манифест
        // приложения использует ComCtl32.dll версии 6 или более поздней версии для включения
        // стилей отображения. В противном случае будет возникать сбой при создании любого окна.
        INITCOMMONCONTROLSEX InitCtrls;
        InitCtrls.dwSize = sizeof(InitCtrls);
        // Выберите этот параметр для включения всех общих классов управления, которые необходимо использовать
        // в вашем приложении.
        InitCtrls.dwICC = ICC_WIN95_CLASSES;
        InitCommonControlsEx(&InitCtrls);
        CWinAppEx::InitInstance();
        // Инициализация библиотек OLE
        if (!AfxOleInit())
        {
            AfxMessageBox(IDP_OLE_INIT_FAILED);
            return FALSE;
```

```
        }
        AfxEnableControlContainer();
        EnableTaskbarInteraction(FALSE);
        SetRegistryKey(_T("Локальные приложения, созданные с помощью мастера приложений"));
        LoadStdProfileSettings(4);  // Загрузите стандартные параметры INI-файла (включая MRU)
        InitContextMenuManager();
        InitKeyboardManager();
        InitTooltipManager();
        CMFCToolTipInfo ttParams;
        ttParams.m_bVislManagerTheme = TRUE;
        theApp.GetTooltipManager()->SetTooltipParams(AFX_TOOLTIP_TYPE_ALL,
                RUNTIME_CLASS(CMFCToolTipCtrl), &ttParams);
        CSingleDocTemplate* pDocTemplate;
        pDocTemplate = new CSingleDocTemplate(
                IDR_MAINFRAME,
                RUNTIME_CLASS(CDesignerkwearDoc),
                RUNTIME_CLASS(CMainFrame),       // основное окно рамки SDI
                RUNTIME_CLASS(CDesignerkwearView));
        if (!pDocTemplate)
                return FALSE;
        AddDocTemplate(pDocTemplate);
        // Синтаксический разбор командной строки на стандартные команды оболочки, DDE, открытие файлов
        CCommandLineInfo cmdInfo;
        ParseCommandLine(cmdInfo);
        // Включить открытие выполнения DDE
```

```
        EnableShellOpen();
        RegisterShellFileTypes(TRUE);
        if (!ProcessShellCommand(cmdInfo))
            return FALSE;
        m_pMainWnd->ShowWindow(SW_SHOW);
        m_pMainWnd->UpdateWindow();
        m_pMainWnd->DragAcceptFiles();
        return TRUE;
}
int CDesignerkwearApp::ExitInstance()
{   AfxOleTerm(FALSE);
    return CWinAppEx::ExitInstance();
}
// обработчики сообщений CDesignerkwearApp
// Диалоговое окно CAboutDlg используется для описания сведений о приложении
class CAboutDlg : public CDialogEx
{
public:
    CAboutDlg();
// Данные диалогового окна
    enum { IDD = IDD_ABOUTBOX };
protected:
    virtual void DoDataExchange(CDataExchange* pDX);    // поддержка DDX/DDV
// Реализация
protected:
    DECLARE_MESSAGE_MAP()
};
CAboutDlg::CAboutDlg() : CDialogEx(CAboutDlg::IDD)
```

```
{
}
void CAboutDlg::DoDataExchange(CDataExchange* pDX)
{
	CDialogEx::DoDataExchange(pDX);
}
BEGIN_MESSAGE_MAP(CAboutDlg, CDialogEx)
END_MESSAGE_MAP()
// Команда приложения для запуска диалога
void CDesignerkwearApp::OnAppAbout()
{
	CAboutDlg aboutDlg;
	aboutDlg.DoModal();
}
// CDesignerkwearApp настройка методов загрузки и сохранения
void CDesignerkwearApp::PreLoadState()
{
	BOOL bNameValid;
	CString strName;
	bNameValid = strName.LoadString(IDS_EDIT_MENU);
	ASSERT(bNameValid);
	GetContextMenuManager()->AddMenu(strName, IDR_POPUP_EDIT);
}

void CDesignerkwearApp::LoadCustomState()
{
}
void CDesignerkwearApp::SaveCustomState()
{
}
```

Файл “Designer k-wearDoc.cpp”

```
#ifndef SHARED_HANDLERS
#include "Designer k-wear.h"
#endif
#include "Designer k-wearDoc.h"
#include <propkey.h>
#ifdef _DEBUG
#define new DEBUG_NEW
#endif
#include <cmath>
#include "ReportM1.h"
#include "RzR.h"
// CDesignerkwearDoc
IMPLEMENT_DYNCREATE(CDesignerkwearDoc, CDocument)
BEGIN_MESSAGE_MAP(CDesignerkwearDoc, CDocument)
        ON_COMMAND(IDD_CONSOLE, &CDesignerkwearDoc::OnConsole)
        ON_COMMAND(ID_BUILD, &CDesignerkwearDoc::OnBuild)
        ON_COMMAND(ID_REPORT, &CDesignerkwearDoc::OnReport)
        ON_COMMAND(ID_REPORT_M1,
&CDesignerkwearDoc::OnReportM1)
        ON_COMMAND(ID_SM, &CDesignerkwearDoc::OnSm)
        ON_COMMAND(ID_MM, &CDesignerkwearDoc::OnMm)
        ON_UPDATE_COMMAND_UI(ID_SM,
&CDesignerkwearDoc::OnUpdateSm)
        ON_UPDATE_COMMAND_UI(ID_MM,
&CDesignerkwearDoc::OnUpdateMm)
        ON_COMMAND(ID_RZ, &CDesignerkwearDoc::OnRz)
        ON_COMMAND(ID_SGLAZH, &CDesignerkwearDoc::OnSglazh)
```

```
        ON_UPDATE_COMMAND_UI(ID_SGLAZH,
&CDesignerkwearDoc::OnUpdateSglazh)
        ON_COMMAND(ID_M1_TRANSFORM,
&CDesignerkwearDoc::OnM1Transform)
        ON_UPDATE_COMMAND_UI(ID_M1_TRANSFORM,
&CDesignerkwearDoc::OnUpdateM1Transform)
        ON_COMMAND(ID_VYT_CLOSE,
&CDesignerkwearDoc::OnVytClose)
        ON_COMMAND(ID_VYT_OPEN, &CDesignerkwearDoc::OnVytOpen)
        ON_UPDATE_COMMAND_UI(ID_VYT_CLOSE,
&CDesignerkwearDoc::OnUpdateVytClose)
        ON_UPDATE_COMMAND_UI(ID_VYT_OPEN,
&CDesignerkwearDoc::OnUpdateVytOpen)
        ON_COMMAND(ID_SETUP, &CDesignerkwearDoc::OnSetup)
        ON_UPDATE_COMMAND_UI(ID_SETUP,
&CDesignerkwearDoc::OnUpdateSetup)
    END_MESSAGE_MAP()
    // Импорт необходимых объектов для отчета в MS Excel
    #import "C:\Program Files\Common Files\Microsoft Shared\OFFICE12\mso.dll"
\
    rename("DocumentProperties", "DocumentPropertiesXL") \
    rename("RGB", "RBGXL")
    //Microsoft VBA objects
    #import "C:\Program Files\Common Files\Microsoft
Shared\VBA\VBA6\vbe6ext.olb"
    //Excel Application objects
    #import "C:\Program Files\Microsoft Office\OFFICE12\EXCEL.EXE" \
    rename("DialogBox", "DialogBoxXL") rename("RGB", "RBGXL") \
    rename("DocumentProperties", "DocumentPropertiesXL") \
    rename("ReplaceText", "ReplaceTextXL") \
```

```
rename("CopyFile", "CopyFileXL") \
exclude("IFont", "IPicture") no_dual_interfaces
// создание/уничтожение CDesignerkwearDoc
CDesignerkwearDoc::CDesignerkwearDoc():
        m_DocSize(CSize(3000,3000)), m_Check(EMPTY), m_Izm(SM)
                , tolshina(2), m_Sgl(Ys), trans(N), m_Vyt(OPEN), m_prov(OFF)
{
        aConsole.Create(IDD_CONSOLE);
        aSetup.Create(IDD_DIALOG2);
        CFileStatus status;
        if (CFile::GetStatus(CString("C:\\Dkw_com.txt"), status))
        {
        // чтение из файла строки
        CStdioFile File(CString("C:\\Dkw_com.txt"),CFile::modeRead);
        CString ref;
                do
                {
                        File.ReadString(ref);
                        if (!ref.IsEmpty())
                        {com_trans.push_back(ref);};
                // Действия с aStr
                } while(!ref.IsEmpty());
                File.Close();
        }
}
CDesignerkwearDoc::~CDesignerkwearDoc()
{
}

BOOL CDesignerkwearDoc::OnNewDocument()
```

```
{
	if (!CDocument::OnNewDocument())
		return FALSE;
	obj.clear();
	com.clear();
	approximation_l.clear();
	approximation_r.clear();
	aprox.clear();
	approx_side.clear();
	approx_cs.clear();
	dobj.clear();
	buffer_vector.clear();
	aConsole.m_EditConsole.SetWindowText(_T(""));
	aConsole.buff_inp.clear();
	names.clear();
	aConsole.serial.clear();
	vivod.clear();
	info.clear();
	UpdateAllViews(NULL);
	return TRUE;
}
// сериализация CDesignerkwearDoc
void CDesignerkwearDoc::Serialize(CArchive& ar)
{
if (m_prov == OFF)
{
	int jj(1);
	if (ar.IsStoring())
	{
		std::vector<CString> temp;
```

```
        CString stroka;
        if (!aConsole.serial.empty())
        {
                temp.clear();
                for (int i = 0; i < aConsole.serial.size(); i++)
                {
                        stroka += aConsole.serial[i];
                        temp.push_back(aConsole.serial[i]);
                };
                ar << stroka;
        }
        else
        {
                for (int i = 0; i < temp.size(); i++)
                {
                        stroka += temp[i];
                };
                ar << stroka;
        }
}
else
{
        obj.clear();
        com.clear();
        approximation_l.clear();
        approximation_r.clear();
        aprox.clear();
        approx_side.clear();
        approx_cs.clear();
        dobj.clear();
```

```
buffer_vector.clear();
aConsole.m_EditConsole.SetWindowText(_T(""));
aConsole.buff_inp.clear();
names.clear();
aConsole.serial.clear();
vivod.clear();
info.clear();
CString s;
CString ss;
CString sss;
ar >> s;
// Вывод в консоль
for (int j = 0; j < s.GetLength(); j++)
{
	jj++;
	if (s[j] == '@')
	{
		jj -= 2;
		ss += s.Mid(j-jj, jj)+_T("\r\n");
		aConsole.m_EditConsole.SetWindowText(ss);
		aConsole.serial.push_back(s.Mid(j-jj, jj+1));
		jj = 1;
	}
};
// Удаление символов перехода и пробелов
for (int j = 0; j < s.GetLength(); j++)
{
	if (s[j] != '@')
	{
		sss += s[j];
```

```
                }
            };
            // Построение
            for (int j = 0; j < sss.GetLength(); j++)
            {
                jj++;
                if (sss[j] == ';')
                {
                    jj -= 2;
                    aConsole.buff_inp.push_back(sss.Mid(j-jj, jj+1));
                    jj = 1;
                }
            };
        };
        OnBuild();
    }
    }
    #ifdef SHARED_HANDLERS
    // Поддержка для эскизов
    void CDesignerkwearDoc::OnDrawThumbnail(CDC& dc, LPRECT lprcBounds)
    {
        // Измените этот код для отображения данных документа
        dc.FillSolidRect(lprcBounds, RGB(255, 255, 255));

        CString strText = _T("TODO: implement thumbnail drawing here");
        LOGFONT lf;
        CFont* pDefaultGUIFont = CFont::FromHandle((HFONT)
GetStockObject(DEFAULT_GUI_FONT));
        pDefaultGUIFont->GetLogFont(&lf);
        lf.lfHeight = 36;
```

```
            CFont fontDraw;
            fontDraw.CreateFontIndirect(&lf);
            CFont* pOldFont = dc.SelectObject(&fontDraw);
            dc.DrawText(strText, lprcBounds, DT_CENTER | DT_WORDBREAK);
            dc.SelectObject(pOldFont);
        }
        // Поддержка обработчиков поиска
        void CDesignerkwearDoc::InitializeSearchContent()
        {
            CString strSearchContent;
            SetSearchContent(strSearchContent);
        }
        void CDesignerkwearDoc::SetSearchContent(const CString& value)
        {
            if (value.IsEmpty())
            {
                RemoveChunk(PKEY_Search_Contents.fmtid,
PKEY_Search_Contents.pid);
            }
            else
            {
                CMFCFilterChunkValueImpl *pChunk = NULL;
                ATLTRY(pChunk = new CMFCFilterChunkValueImpl);
                if (pChunk != NULL)
                {
                    pChunk->SetTextValue(PKEY_Search_Contents,      value,
CHUNK_TEXT);
                    SetChunkValue(pChunk);
                }
            }
```

```
}
#endif // SHARED_HANDLERS
// диагностика CDesignerkwearDoc
#ifdef _DEBUG
void CDesignerkwearDoc::AssertValid() const
{
	CDocument::AssertValid();
}
void CDesignerkwearDoc::Dump(CDumpContext& dc) const
{
	CDocument::Dump(dc);
}
#endif //_DEBUG
// команды CDesignerkwearDoc
void CDesignerkwearDoc::OnConsole()
{
	aConsole.ShowWindow(SW_SHOW);
}
void CDesignerkwearDoc::OnBuild()
{
 if (m_prov == OFF)
 {
	obj.clear();
	com.clear();
	approximation_l.clear();
	approximation_r.clear();
	approximation_dl.clear();
	approximation_dr.clear();
	side_d.clear();
	aprox.clear();
```

```
approx_side.clear();
approx_cs.clear();
dobj.clear();
names.clear();
info.clear();
std::vector<CObjects> temp;
if (m_Izm == SM)
{izmerenia = 10;}
else if (m_Izm == MM)
{izmerenia = 1;};
std::vector <CDObjects> dbuffer_vector;
CDObjects buff_dobject;
CObjects buff_object;
// Временный с об-ми
for (int i = 0; i < aConsole.buff_inp.size(); i++)
{
    buffer_vector.push_back(CObjects(aConsole.buff_inp[i], izmerenia,
buffer_vector, raz, i, com_trans));
    if (buffer_vector.back().st != 101 && buffer_vector.back().st != 102
&& buffer_vector.back().st != 130
        && buffer_vector.back().st != 131 &&
buffer_vector.back().st != 222 && buffer_vector.back().st != 132
        && buffer_vector.back().st != 133 &&
buffer_vector.back().st != 200 && buffer_vector.back().st != 10000
        && buffer_vector.back().st != 100 &&
buffer_vector.back().st != 190)
    {
        CString temp_string;
        temp_string = aConsole.buff_inp[i];
        for (int j = 0; j < temp_string.GetLength(); j++)
```

```
                {
                    if (temp_string[j] == ':')
                    {
                        names.push_back(temp_string.Mid(0, j));
                        break;
                    };
                };
            };
        };
        int names_count(0);
        // Построение векторов
        for (int i = 0; i < buffer_vector.size(); i++)
        {
            if (buffer_vector[i].st == 0 || buffer_vector[i].st == 1 ||
buffer_vector[i].st == 2 || buffer_vector[i].st == 3
                || buffer_vector[i].st == 4 || buffer_vector[i].st == 5
                || buffer_vector[i].st == 1000 || buffer_vector[i].st == 1001)
            {
                for (int k = 0; k < names.size(); k++)
                {
                    if (names[k] == buffer_vector[i].name &&
!names[k].IsEmpty())
                    {
                        names_count++;
                        if (names_count > 1)
                        {
                            AfxMessageBox(CString("Имя <") +
buffer_vector[i].name + CString("> используется \nнесколькими объектами! \n") +
buffer_vector[i].command);
                            names_count = 0;
```

```
                                        break;
                                    };
                                };
                            };
                            if (names_count == 1)
                            {
                                temp.push_back(buffer_vector[i]);
                                names_count = 0;
                            };
                        }
                        else if (buffer_vector[i].st == 100 || buffer_vector[i].st == 10000 ||
buffer_vector[i].st == 103 || buffer_vector[i].st == 200
                            || buffer_vector[i].st == 223 || buffer_vector[i].st == 190 ||
buffer_vector[i].st == 555)
                        {
                            com.push_back(buffer_vector[i]);          //          Команды
преобразования объектов
                        }
                        else if (buffer_vector[i].st == 101 || buffer_vector[i].st == 102 ||
buffer_vector[i].st == 222
                                || buffer_vector[i].st == 130 || buffer_vector[i].st == 131
                                || buffer_vector[i].st == 132 || buffer_vector[i].st ==
133)
                        {
                            aprox.push_back(buffer_vector[i]);          //          Команды
аппроксимации
                        };
                };
                buffer_vector.clear();
                for (int i = 0; i < temp.size(); i++)
```

```
{
        if (!temp[i].command.IsEmpty())
        {
                obj.push_back(temp[i]);
        };
};
temp.clear();
// Изменение объектов
for (int i = 0; i < com.size(); i++)
{
        if (com[i].st == 223 && m_Vyt == CLOSE)
        {
                // левая сторона
                for (int j = 0; j < com[i].left.size(); j++)
                {
                        for (int h = 0; h < obj.size(); h++)
                        {
                                if (com[i].left[j] == obj[h].name && j < com[i].left.size() - 1)
                                {
                                        if (com[i].osnx == obj[h].x0 && com[i].osny == obj[h].y0)
                                        {
                                                obj[h].x1 += com[i].a1;
                                                obj[h].y1 += com[i].a2;
                                                obj[h].x2 += com[i].a1;
                                                obj[h].y2 += com[i].a2;
                                                obj[h].x3 += com[i].a1;
                                                obj[h].y3 += com[i].a2;
                                        }
```

```
                else if (com[i].osnx == obj[h].x1 &&
com[i].osny == obj[h].y1)
                {
                    obj[h].x0 += com[i].a1;
                    obj[h].y0 += com[i].a2;
                    obj[h].x2 += com[i].a1;
                    obj[h].y2 += com[i].a2;
                    obj[h].x3 += com[i].a1;
                    obj[h].y3 += com[i].a2;
                }
                else if (com[i].osnx == obj[h].x2 &&
com[i].osny == obj[h].y2)
                {
                    obj[h].x0 += com[i].a1;
                    obj[h].y0 += com[i].a2;
                    obj[h].x1 += com[i].a1;
                    obj[h].y1 += com[i].a2;
                    obj[h].x3 += com[i].a1;
                    obj[h].y3 += com[i].a2;
                }
                else if (com[i].osnx == obj[h].x3 &&
com[i].osny == obj[h].y3)
                {
                    obj[h].x0 += com[i].a1;
                    obj[h].y0 += com[i].a2;
                    obj[h].x2 += com[i].a1;
                    obj[h].y2 += com[i].a2;
                    obj[h].x1 += com[i].a1;
                    obj[h].y1 += com[i].a2;
                }
```

```
            else
            {
                obj[h].x0 += com[i].a1;
                obj[h].y0 += com[i].a2;
                obj[h].x1 += com[i].a1;
                obj[h].y1 += com[i].a2;
                obj[h].x2 += com[i].a1;
                obj[h].y2 += com[i].a2;
                obj[h].x3 += com[i].a1;
                obj[h].y3 += com[i].a2;
            };
        }
        else if (com[i].left[j] == obj[h].name && j == com[i].left.size() - 1)
        {
            int l0(1),l1(1),l2(1),l3(1);
            for (int jj = 0; jj < com[i].end_left.GetLength(); jj++)
            {
                if (com[i].end_left[jj] == '0')
                {
                    l0 = 0;
                }
                else if (com[i].end_left[jj] == '1')
                {
                    l1 = 0;
                }
                else if (com[i].end_left[jj] == '2')
                {
                    l2 = 0;
```

```
                }
                else if (com[i].end_left[jj] == '3')
                {
                    l3 = 0;
                }
                else if (com[i].end_left[jj] == '4')
                {
                    l0 = l1 = l2 = l3 = 1;
                };
            };
            obj[h].x0 += com[i].a1*l0;
            obj[h].y0 += com[i].a2*l0;
            obj[h].x1 += com[i].a1*l1;
            obj[h].y1 += com[i].a2*l1;
            obj[h].x2 += com[i].a1*l2;
            obj[h].y2 += com[i].a2*l2;
            obj[h].x3 += com[i].a1*l3;
            obj[h].y3 += com[i].a2*l3;
        };
    };
};
// правая сторона
for (int j = 0; j < com[i].right.size(); j++)
{
    for (int h = 0; h < obj.size(); h++)
    {
        if (com[i].right[j] == obj[h].name && j < com[i].right.size() - 1)
        {
```

```
if (com[i].osnx == obj[h].x0 && com[i].osny == obj[h].y0)
{
    obj[h].x1 += com[i].a3;
    obj[h].y1 += com[i].a4;
    obj[h].x2 += com[i].a3;
    obj[h].y2 += com[i].a4;
    obj[h].x3 += com[i].a3;
    obj[h].y3 += com[i].a4;
}
else if (com[i].osnx == obj[h].x1 && com[i].osny == obj[h].y1)
{
    obj[h].x0 += com[i].a3;
    obj[h].y0 += com[i].a4;
    obj[h].x2 += com[i].a3;
    obj[h].y2 += com[i].a4;
    obj[h].x3 += com[i].a3;
    obj[h].y3 += com[i].a4;
}
else if (com[i].osnx == obj[h].x2 && com[i].osny == obj[h].y2)
{
    obj[h].x0 += com[i].a3;
    obj[h].y0 += com[i].a4;
    obj[h].x1 += com[i].a3;
    obj[h].y1 += com[i].a4;
    obj[h].x3 += com[i].a3;
    obj[h].y3 += com[i].a4;
}
```

```
                else if (com[i].osnx == obj[h].x3 &&
com[i].osny == obj[h].y3)
                {
                    obj[h].x0 += com[i].a3;
                    obj[h].y0 += com[i].a4;
                    obj[h].x2 += com[i].a3;
                    obj[h].y2 += com[i].a4;
                    obj[h].x1 += com[i].a3;
                    obj[h].y1 += com[i].a4;
                }
                else
                {
                    obj[h].x0 += com[i].a3;
                    obj[h].y0 += com[i].a4;
                    obj[h].x1 += com[i].a3;
                    obj[h].y1 += com[i].a4;
                    obj[h].x2 += com[i].a3;
                    obj[h].y2 += com[i].a4;
                    obj[h].x3 += com[i].a3;
                    obj[h].y3 += com[i].a4;
                };
            }
            else if (com[i].right[j] == obj[h].name && j ==
com[i].right.size() - 1)
            {
                int l0(1),l1(1),l2(1),l3(1);
                for (int jj = 0; jj <
com[i].end_right.GetLength(); jj++)
                {
                    if (com[i].end_right[jj] == '0')
```

```
                {
                    l0 = 0;
                }
                else if (com[i].end_right[jj] == '1')
                {
                    l1 = 0;
                }
                else if (com[i].end_right[jj] == '2')
                {
                    l2 = 0;
                }
                else if (com[i].end_right[jj] == '3')
                {
                    l3 = 0;
                }
                else if (com[i].end_right[jj] == '4')
                {
                    l0 = l1 = l2 = l3 = 1;
                };
            };
            obj[h].x0 += com[i].a3*l0;
            obj[h].y0 += com[i].a4*l0;
            obj[h].x1 += com[i].a3*l1;
            obj[h].y1 += com[i].a4*l1;
            obj[h].x2 += com[i].a3*l2;
            obj[h].y2 += com[i].a4*l2;
            obj[h].x3 += com[i].a3*l3;
            obj[h].y3 += com[i].a4*l3;
        };
    };
```

```
};
// перезапись вектора
for (int j = 0; j < obj.size(); j++)
{
	if (obj[j].name != com[i].vyt_r)
	{
		temp.push_back(obj[j]);
	};
};
obj.clear();
for (int j = 0; j < temp.size(); j++)
{
	obj.push_back(temp[j]);
	if (obj.back().name == com[i].vyt_l)
	{
		obj.back().x0 = com[i].pzx;
		obj.back().y0 = com[i].pzy;
		if (com[i].p == 3)
		{
			obj.back().x1 = com[i].pzx;
			obj.back().y1 = com[i].pzy - 2*com[i].left_line;
		}
		else if (com[i].p == 1)
		{
			obj.back().x1 = com[i].pzx;
			obj.back().y1 = com[i].pzy + 2*com[i].left_line;
		}
		else if (com[i].p == 2)
```

```
                {
                    obj.back().x1 = com[i].pzx;
                    obj.back().y1 = com[i].pzy + com[i].ras;
                }
                else if (com[i].p == 4)
                {
                    obj.back().x1 = com[i].pzx;
                    obj.back().y1 = com[i].pzy + com[i].ras;
                };

                if (!com[i].vyt_l.IsEmpty() && !com[i].vyt_r.IsEmpty())
                {
                    obj.back().name = com[i].vyt_l +"+"+ com[i].vyt_r;
                    obj.back().st = 1;
                    obj.back().p = 1;
                    // созд об-та частич вязания и отрезка
                    CString command1;
                    command1 = com[i].vyt_l +"+"+ com[i].vyt_r + CString(":dopr1>")
                        + com[i].n_ryadov + CString(",0.5,1,") + com[i].vysota_ryada + CString(";");
                    CObjects tobj(command1, izmerenia, obj, raz, obj.size(), com_trans);

                    aprox.push_back(tobj);
                };
            };
        };
```

```
                temp.clear();


            }
            else if (com[i].st == 100)
            {
                for (int j = 0; j < obj.size(); j++)
                {
                    if (com[i].name == obj[j].name)
                    {
                        double dx(0), dy(0);
                        if (com[i].x0 < 0)
                        {
                            com[i].x1 = com[i].y0*cos(com[i].x0*3.1415/180) + obj[j].x0;
                            com[i].y1 = -com[i].y0*sin(com[i].x0*3.1415/180) + obj[j].y0;
                        }
                        else
                        {
                            com[i].x1 = com[i].y0*cos(com[i].x0*3.1415/180) + obj[j].x0;
                            com[i].y1 = com[i].y0*sin(com[i].x0*3.1415/180) + obj[j].y0;
                        };
                        dx = com[i].x1 - obj[j].x0;
                        dy = com[i].y1 - obj[j].y0;
                        obj[j].x0 += dx;
                        obj[j].y0 += dy;
                        obj[j].x1 += dx;
                        obj[j].y1 += dy;
```

```
                        obj[j].x2 += dx;
                        obj[j].y2 += dy;
                        obj[j].x3 += dx;
                        obj[j].y3 += dy;
                        obj[j].sx0.Format(_T("%.2f"),obj[j].x0);
                        obj[j].sy0.Format(_T("%.2f"),obj[j].y0);
                        obj[j].sx1.Format(_T("%.2f"),obj[j].x1);
                        obj[j].sy1.Format(_T("%.2f"),obj[j].y1);
                        obj[j].sx2.Format(_T("%.2f"),obj[j].x2);
                        obj[j].sy2.Format(_T("%.2f"),obj[j].y2);
                        obj[j].sx3.Format(_T("%.2f"),obj[j].x3);
                        obj[j].sy3.Format(_T("%.2f"),obj[j].y3);
                        break;
                    };
                };
            }
            else if (com[i].st == 190)
            {
                for (int j = 0; j < obj.size(); j++)
                {
                    if (com[i].name == obj[j].name)
                    {
                        obj[j].x0 = obj[j].x0*cos(com[i].a1*3.1415/180) -
obj[j].y0*sin(com[i].a1*3.1415/180);
                        obj[j].y0 = obj[j].x0*sin(com[i].a1*3.1415/180) +
obj[j].y0*cos(com[i].a1*3.1415/180);
                        obj[j].x1 = obj[j].x1*cos(com[i].a1*3.1415/180) -
obj[j].y1*sin(com[i].a1*3.1415/180);
                        obj[j].y1 = obj[j].x1*sin(com[i].a1*3.1415/180) +
obj[j].y1*cos(com[i].a1*3.1415/180);
```

```
                    obj[j].sx0.Format(_T("%.2f"),obj[j].x0);
                    obj[j].sy0.Format(_T("%.2f"),obj[j].y0);
                    obj[j].sx1.Format(_T("%.2f"),obj[j].x1);
                    obj[j].sy1.Format(_T("%.2f"),obj[j].y1);
                    obj[j].sx2.Format(_T("%.2f"),obj[j].x2);
                    obj[j].sy2.Format(_T("%.2f"),obj[j].y2);
                    obj[j].sx3.Format(_T("%.2f"),obj[j].x3);
                    obj[j].sy3.Format(_T("%.2f"),obj[j].y3);
                    break;
                }
            };
        }
        else if (com[i].st == 555)
        {
            CString rx1, ry1, rx2, ry2, rax, rbx, ray, rby, new_name,
command;
            double tx(0), ty(0), t(com[i].x0);
            for (int j = 0; j < obj.size(); j++)
            {
                if (com[i].name == obj[j].name)
                {
                    if (obj[j].st == 2)
                    {
                        tx = (1-t)*(1-t)*obj[j].x0+2*t*(1-
t)*obj[j].x1+t*t*obj[j].x2;
                        ty = (1-t)*(1-t)*obj[j].y0+2*t*(1-
t)*obj[j].y1+t*t*obj[j].y2;
                        // построение 1 кривой
                        rx1.Format(_T("%.2f"), obj[j].x0);
                        ry1.Format(_T("%.2f"), obj[j].y0);
```

```
rx2.Format(_T("%.2f"), tx);
ry2.Format(_T("%.2f"), ty);

t *= 0.5;
tx = (1-t)*(1-t)*obj[j].x0+2*t*(1-t)*obj[j].x1+t*t*obj[j].x2;
ty = (1-t)*(1-t)*obj[j].y0+2*t*(1-t)*obj[j].y1+t*t*obj[j].y2;
rax.Format(_T("%.2f"), tx);
ray.Format(_T("%.2f"), ty);
if (com[i].sy0.IsEmpty())
{
    new_name = obj[j].name + CString("_1");
}
else
{
    new_name = com[i].sy0;
};
command = new_name + CString(":bzr2i>") + rx1 + CString(",") + ry1 + CString(",") + rx2 + CString(",") + ry2 + CString(",") + rax
    + CString(",")+ ray + CString(",") + CString("0.5;");
CObjects push_obj(command, izmerenia, obj, raz, obj.size(), com_trans);
obj.push_back(push_obj);
// построение 2 кривой
rx1.Format(_T("%.2f"), obj[j].x2);
```

```
ry1.Format(_T("%.2f"), obj[j].y2);
t = (1 - t) / 2;
tx = (1-t)*(1-t)*obj[j].x0+2*t*(1-t)*obj[j].x1+t*t*obj[j].x2;
ty = (1-t)*(1-t)*obj[j].y0+2*t*(1-t)*obj[j].y1+t*t*obj[j].y2;
rax.Format(_T("%.2f"), tx);
ray.Format(_T("%.2f"), ty);
if (com[i].sy0.IsEmpty())
{
    new_name = obj[j].name + CString("_2");
}
else
{
    new_name = com[i].sy0;
};
CString t1;
t1.Format(_T("%.2f"), t);
command = new_name + CString(":bzr2i>") + rx2 + CString(",") + ry2 + CString(",") + rx1 + CString(",") + ry1 + CString(",") + rax
    + CString(",")+ ray + CString(",") + t1 + CString(";");
CObjects push_obj1(command, izmerenia, obj, raz, obj.size(), com_trans);
obj.push_back(push_obj1);
com[i].command.Empty();
com[i].name.Empty();
}
```

```
                        };
                    };
                };
            };
            // Формирование Д-ОБЪЕКТОВ
            for (int i = 0; i < aConsole.buff_inp.size(); i++)
            {
                dobj.push_back(CDObjects(aConsole.buff_inp[i], obj, raz, obj, i, dobj, com_trans));
            };
            // Расчет сбавок
            if (!aprox.empty())
            {
                for (int i = 0; i < dobj.size(); i++)
                {
                    if (!dobj[i].d_object_left.empty() && !dobj[i].d_object_right.empty() && dobj[i].st != 53) // Форма
                    {

        approximation_l.push_back(CApprox(dobj[i].d_object_left, aprox, -1, dobj[i].dname, m_Sgl));

        approximation_r.push_back(CApprox(dobj[i].d_object_right, aprox, -1, dobj[i].dname, m_Sgl));
                        int rl(0), rr(0);
                        rl = approximation_l.back().sbavki.back().r;
                        rr = approximation_r.back().sbavki.back().r;
                        if (rl - rr != 0)
                        {
                            if (rl < rr)
```

```
                                        {
                                        CAprObj report;
                                        for (int iii = 1; iii <= rr - rl; iii++)
                                        {
                                                report.ap                       =
approximation_l.back().sbavki.back().ap;
                                                report.bp                       =
approximation_l.back().sbavki.back().bp;
                                                report.maxsb                    =
approximation_l.back().sbavki.back().maxsb;
                                                report.rapapr = 1;
                                                report.ax                       =
approximation_l.back().sbavki.back().ax;
                                                report.ay                       =
approximation_l.back().sbavki.back().ay - approximation_l.back().sbavki.back().bp;
                                                report.kx                       =
approximation_l.back().sbavki.back().kx;
                                                report.r                        =
approximation_l.back().sbavki.back().r + 1;
                                                report.sbav = 0;
                                                report.t = 1;
                                                report.pol                      =
approximation_l.back().sbavki.back().pol;
                                                report.side_pol                 =
approximation_l.back().sbavki.back().side_pol;

        approximation_l.back().sbavki.push_back(report);
                                        }
                                        }
                                        else if (rl > rr)
```

```
                                        {
                                        CAprObj report;
                                        for (int iii = 1; iii <= rl - rr; iii++)
                                        {
                                                report.ap                       =
approximation_r.back().sbavki.back().ap;
                                                report.bp                       =
approximation_r.back().sbavki.back().bp;
                                                report.maxsb                    =
approximation_r.back().sbavki.back().maxsb;
                                                report.rapapr = 1;
                                                report.ax                       =
approximation_r.back().sbavki.back().ax;
                                                report.ay                       =
approximation_r.back().sbavki.back().ay - approximation_r.back().sbavki.back().bp;
                                                report.kx                       =
approximation_r.back().sbavki.back().kx;
                                                report.r                        =
approximation_r.back().sbavki.back().r + 1;
                                                report.sbav = 0;
                                                report.t = 1;
                                                report.pol                      =
approximation_r.back().sbavki.back().pol;
                                                report.side_pol                 =
approximation_r.back().sbavki.back().side_pol;

        approximation_r.back().sbavki.push_back(report);
                                        }
                                        };
                                };
```

```
double dxx(0);
int dop_stol(0);
dxx = approximation_l.back().sbavki.back().ax / approximation_r.back().sbavki.back().ax;
if (dxx < 1)
{
dop_stol = abs(approximation_l.back().sbavki.back().ax - approximation_r.back().sbavki.back().ax)/approximation_r.back().sbavki.back().ap;
CString mess;
mess.Format(_T("%i"), dop_stol);
}
else if (dxx > 1)
{
dop_stol = abs(approximation_l.back().sbavki.back().ax - approximation_r.back().sbavki.back().ax)/approximation_r.back().sbavki.back().ap;
CString mess;
mess.Format(_T("%i"), dop_stol);
if (dop_stol >= 5)
{
AfxMessageBox(CString("Внимание! Наложение сторон контура ФОРМЫ друг на друга <") + dobj[i].dname + CString(">, \n необходимо увеличить начальную ширину на ") + mess + CString(" петельных столбиков!"), MB_OK);
}
else if (dop_stol != 0)
{

AfxMessageBox(CString("Внимание! Наложение сторон контура ФОРМЫ
```

```
друг на друга <") + dobj[i].dname + CString(">, \n необходимо увеличить начальную ширину на ") + mess + CString(" петельных столбика!"), MB_OK);
                                                }
                                        };
                                }
                                else if (!dobj[i].d_object_left.empty() && !dobj[i].d_object_right.empty() && dobj[i].st == 53) // Форма
                                {

        approximation_dl.push_back(CApprox(dobj[i].d_object_left, aprox, -1, dobj[i].dname, m_Sgl));

        approximation_dr.push_back(CApprox(dobj[i].d_object_right, aprox, -1, dobj[i].dname, m_Sgl));
                                        int rl(0), rr(0);
                                        rl = approximation_dl.back().sbavki.back().r;
                                        rr = approximation_dr.back().sbavki.back().r;
                                                if (rl - rr != 0)
                                                {
                                                        if (rl < rr)
                                                        {
                                                        CAprObj report;
                                                        for (int iii = 1; iii <= rr - rl; iii++)
                                                        {
                                                                report.ap = approximation_dl.back().sbavki.back().ap;
                                                                report.bp = approximation_dl.back().sbavki.back().bp;
                                                                report.maxsb = approximation_dl.back().sbavki.back().maxsb;
```

```
                                                report.rapapr = 1;
                                                report.ax =
approximation_dl.back().sbavki.back().ax;
                                                report.ay =
approximation_dl.back().sbavki.back().ay - approximation_dl.back().sbavki.back().bp;
                                                report.kx =
approximation_dl.back().sbavki.back().kx;
                                                report.r =
approximation_dl.back().sbavki.back().r + 1;
                                                report.sbav = 0;
                                                report.t = 1;
                                                report.pol =
approximation_dl.back().sbavki.back().pol;
                                                report.side_pol =
approximation_dl.back().sbavki.back().side_pol;

        approximation_dl.back().sbavki.push_back(report);
                                                }
                                                }
                                                else if (rl > rr)
                                                {
                                                CAprObj report;
                                                for (int iii = 1; iii <= rl - rr; iii++)
                                                {
                                                        report.ap =
approximation_dr.back().sbavki.back().ap;
                                                        report.bp =
approximation_dr.back().sbavki.back().bp;
                                                        report.maxsb =
approximation_dr.back().sbavki.back().maxsb;
```

```
                                                report.rapapr = 1;
                                                report.ax =
approximation_dr.back().sbavki.back().ax;
                                                report.ay =
approximation_dr.back().sbavki.back().ay - approximation_dr.back().sbavki.back().bp;
                                                report.kx =
approximation_dr.back().sbavki.back().kx;
                                                report.r =
approximation_dr.back().sbavki.back().r + 1;
                                                report.sbav = 0;
                                                report.t = 1;
                                                report.pol =
approximation_dr.back().sbavki.back().pol;
                                                report.side_pol =
approximation_dr.back().sbavki.back().side_pol;

        approximation_dr.back().sbavki.push_back(report);
                                                }
                                                };
                                        };
                                        double dxx(0);
                                        int dop_stol(0);
                                        dxx = approximation_dl.back().sbavki.back().ax /
approximation_dr.back().sbavki.back().ax;
                                        if (dxx < 1)
                                        {
                                                dop_stol =
abs(approximation_dl.back().sbavki.back().ax -
approximation_dr.back().sbavki.back().ax)/approximation_dr.back().sbavki.back().ap;
                                                CString mess;
```

```
                mess.Format(_T("%i"), dop_stol);
            }
            else if (dxx > 1)
            {
                dop_stol = abs(approximation_dl.back().sbavki.back().ax - approximation_dr.back().sbavki.back().ax)/approximation_dr.back().sbavki.back().ap;
                CString mess;
                mess.Format(_T("%i"), dop_stol);
                if (dop_stol >= 5)
                {

    AfxMessageBox(CString("Внимание! Наложение сторон контура ФОРМЫ друг на друга <") + dobj[i].dname + CString(">, \n необходимо увеличить начальную ширину на ") + mess + CString(" петельных столбиков!"), MB_OK);
                }
                else  if (dop_stol != 0)
                {

    AfxMessageBox(CString("Внимание! Наложение сторон контура ФОРМЫ друг на друга <") + dobj[i].dname + CString(">, \n необходимо увеличить начальную ширину на ") + mess + CString(" петельных столбика!"), MB_OK);
                }
            };
            side_d.push_back(CApprox(dobj[i].side, aprox, dobj[i].side_const, dobj[i].dname, m_Sgl));
            for (int u = 0; u < side_d.back().sbavki_dop1.size(); u++)
            {
```

```
for (int uu = 0; uu < approximation_dl.back().sbavki.size(); uu++)
{
if (approximation_dl.back().sbavki[uu].ay / side_d.back().sbavki_dop1[u].ay > 0.999
&&
approximation_dl.back().sbavki[uu].ay / side_d.back().sbavki_dop1[u].ay <= 1)
{

side_d.back().sbavki_dop1[u].stol_l =
floor(abs(approximation_dl.back().sbavki[uu].ax -

side_d.back().sbavki_dop1[u].ax)/side_d.back().sbavki_dop1[u].ap + 0.5);

side_d.back().sbavki_dop1[u].axl = approximation_dl.back().sbavki[uu].ax;
};
};
for (int uu = 0; uu < approximation_dr.back().sbavki.size(); uu++)
{
if (approximation_dr.back().sbavki[uu].ay / side_d.back().sbavki_dop1[u].ay > 0.999
&&
approximation_dr.back().sbavki[uu].ay / side_d.back().sbavki_dop1[u].ay <= 1)
{

side_d.back().sbavki_dop1[u].stol_r =
floor(abs(approximation_dr.back().sbavki[uu].ax -

side_d.back().sbavki_dop1[u].ax)/side_d.back().sbavki_dop1[u].ap + 0.5);
```

```
side_d.back().sbavki_dop1[u].axr = approximation_dr.back().sbavki[uu].ax;
                                                    };
                                                };
                                            };
                                }
                                else if (!dobj[i].side.empty() && dobj[i].st != 53) // Кромка
                                {
                                        approx_side.push_back(CApprox(dobj[i].side,      aprox,
dobj[i].side_const, dobj[i].dname, m_Sgl));
                                }
                                else if (!dobj[i].comb.empty() && dobj[i].st != 53)
                                {
                        approx_cs.push_back(CAprDobj(dobj[i].comb_left,dobj[i].comb_right,
aprox, dobj[i].dname, m_Sgl, dobj[i].s_name));
                                }
                        };
                };
                // Корректировка при комбинированном соединении
                if (!approx_cs.empty())
                {
                        CString name_cs_kor;
                        for (int j = 0; j < approx_cs.size(); j++)
                        {
                                double dxl(0), dyl(0), dxr(0), dyr(0);
                                std::vector<int> match;
                                name_cs_kor = approx_cs[j].name;

                                for (int i = 0; i < approx_cs.size(); i++)
                                {
```

```
                    if (name_cs_kor == approx_cs[i].name)
                    {
                        match.push_back(i);
                    };
                };
                if (match.size() >= 2)
                {
                    dxl = approx_cs[match[0]].sbavki_left.back().ax - approx_cs[match[1]].sbavki_left[0].ax;
                    dyl = approx_cs[match[0]].sbavki_left.back().ay - approx_cs[match[1]].sbavki_left[0].ay;
                    dxr = approx_cs[match[0]].sbavki_right.back().ax - approx_cs[match[1]].sbavki_right[0].ax;
                    dyr = approx_cs[match[0]].sbavki_right.back().ay - approx_cs[match[1]].sbavki_right[0].ay;
                    if (approx_cs[match[0]].sbavki_right.back().ay <= approx_cs[match[1]].sbavki_right.back().ay)
                    {
                        approx_cs[match[1]].sbavki_right.clear();
                        approx_cs[match[1]].sbavki_right.push_back(approx_cs[match[0]].sbavki_right.back());
                        approx_cs[match[1]].sbavki_right.push_back(approx_cs[match[0]].sbavki_right.back());
                        dxr = 0;
                        dyr = 0;
                    };
                    if (approx_cs[match[0]].sbavki_left.back().ay <= approx_cs[match[1]].sbavki_left.back().ay)
                    {
                        approx_cs[match[1]].sbavki_left.clear();
```

```
approx_cs[match[1]].sbavki_left.push_back(approx_cs[match[0]].sbavki_l
eft.back());
approx_cs[match[1]].sbavki_left.push_back(approx_cs[match[0]].sbavki_l
eft.back());
                    dxl = 0;
                    dyl = 0;
                };
                for (int v = 1; v < match.size(); v++)
                {
                    for (int ii = 0; ii < approx_cs[match[v]].sbavki_left.size(); ii++)
                    {
                        approx_cs[match[v]].sbavki_left[ii].ax += dxl;
                        approx_cs[match[v]].sbavki_left[ii].ay += dyl;
                    };
                for (int ii = 0; ii < approx_cs[match[v]].sbavki_right.size(); ii++)
                    {
                        approx_cs[match[v]].sbavki_right[ii].ax += dxr;
                        approx_cs[match[v]].sbavki_right[ii].ay += dyr;
                    };
                    if (v != match.size() - 1)
                    {
                        dxl = approx_cs[match[v]].sbavki_left.back().ax - approx_cs[match[v + 1]].sbavki_left[0].ax;
```

```
dyl = approx_cs[match[v]].sbavki_left.back().ay - approx_cs[match[v + 1]].sbavki_left[0].ay;
dxr = approx_cs[match[v]].sbavki_right.back().ax - approx_cs[match[v + 1]].sbavki_right[0].ax;
dyr = approx_cs[match[v]].sbavki_right.back().ay - approx_cs[match[v + 1]].sbavki_right[0].ay;
if (approx_cs[match[v]].sbavki_right.back().ay <= approx_cs[match[v + 1]].sbavki_right[0].ay)
{
    approx_cs[match[v + 1]].sbavki_right.clear();
    approx_cs[match[v + 1]].sbavki_right.push_back(approx_cs[match[v]].sbavki_right.back());
    approx_cs[match[v + 1]].sbavki_right.push_back(approx_cs[match[v]].sbavki_right.back());
    dxr = 0;
    dyr = 0;
};
if (approx_cs[match[v]].sbavki_left.back().ay <= approx_cs[match[v + 1]].sbavki_left[0].ay)
{
    approx_cs[match[v + 1]].sbavki_left.clear();
    approx_cs[match[v + 1]].sbavki_left.push_back(approx_cs[match[v]].sbavki_left.back());
```

```
                                                approx_cs[match[v                +
1]].sbavki_left.push_back(approx_cs[match[v]].sbavki_left.back());
                                                dxl = 0;
                                                dyl = 0;

                                            };
                                        };
                                    };
                                };
                                match.clear();
                            };
                        };
                        // Корректировка при соединении
                        // по кромке
                        std::vector<CAprObj> t_v;
                        for (int i = 0 ; i < approx_side.size(); i++)
                        {
                            for (int j = 0; j < approx_cs.size(); j++)
                            {
                                if (approx_side[i].name == approx_cs[j].name_left)
                                {
                                    int ii(0), jj(0);
                                    for (;;)
                                    {
                                        if (ii >= approx_side[i].sbavki.size() && jj >=
approx_cs[j].sbavki_left.size())
                                        {break;};
                                        if (jj < approx_cs[j].sbavki_left.size())
                                        {
                                            if (ii < approx_side[i].sbavki.size())
```

```
                                        {
                                            if  (approx_side[i].sbavki[ii].ay  <=
approx_cs[j].sbavki_left[jj].ay)
                                            {
                                t_v.push_back(approx_cs[j].sbavki_left[jj]);
                                                jj++;
                                            }
                                            else
                                        {

        t_v.push_back(approx_side[i].sbavki[ii]);
                                            };
                                        }
                                        else
                                        {

        t_v.push_back(approx_cs[j].sbavki_left[jj]);

                                                jj++;
                                        };
                                    };
                                    if (ii < approx_side[i].sbavki.size())
                                    {
                                        ii++;
                                    };
                                    t_v.back().r = 0;
                                };
                                int r(0);
                                int ost1(0);
                                double dx(0), dy(0);
```

```
std::vector<int> ost_corr;
approx_side[i].sbavki.clear();
for (int jjj = 1; jjj < t_v.size(); jjj++)
{
	if (t_v[jjj].t == 0)
	{
		ost_corr.push_back(jjj);
	};
};
for (int jjj = 0; jjj < t_v.size(); jjj++)
{
	if (ost1 < ost_corr.size() && jjj == ost_corr[ost1])
	{
		dx = t_v[jjj-1].ax - t_v[jjj].ax;
		dy = t_v[jjj-1].ay - t_v[jjj].ay;
		ost1++;
	};
	t_v[jjj].ax += dx;
	t_v[jjj].ay += dy;
};
approx_side[i].sbavki.push_back(t_v[0]);
r += approx_side[i].sbavki.back().rapapr;
approx_side[i].sbavki.back().side_pol = -1;
approx_side[i].sbavki.back().r = r;
for (int jjj = 1; jjj < t_v.size(); jjj++)
{
	if (t_v[jjj].ay != approx_side[i].sbavki.back().ay)
	{
		approx_side[i].sbavki.push_back(t_v[jjj]);
		r += approx_side[i].sbavki.back().rapapr;
```

```
                                approx_side[i].sbavki.back().side_pol = -1;
                                approx_side[i].sbavki.back().r = r;
                            };
                        };
                        CString izmenenie;
                        izmenenie.Format(_T("%i"), j + 1);
                        approx_cs[j].name += CString(" (") + izmenenie + CString(")");
                        t_v.clear();
                    }
                    else if (approx_side[i].name == approx_cs[j].name_right)
                    {
                        int ii(0), jj(0);
                        for (;;)
                        {
                            if (ii >= approx_side[i].sbavki.size() && jj >= approx_cs[j].sbavki_right.size())
                            {break;};
                            if (jj < approx_cs[j].sbavki_right.size())
                            {
                                if (ii < approx_side[i].sbavki.size())
                                {
                                    if (approx_side[i].sbavki[ii].ay <= approx_cs[j].sbavki_right[jj].ay)
                                    {

        t_v.push_back(approx_cs[j].sbavki_right[jj]);
                                        jj++;
                                    }
                                    else
```

```
                                        {

t_v.push_back(approx_side[i].sbavki[ii]);
                                        };
                                    }
                                    else
                                    {

t_v.push_back(approx_cs[j].sbavki_right[jj]);
        jj++;
                                        };
                                };
                                    if (ii < approx_side[i].sbavki.size())
                                {
                                    ii++;
                                };
                                t_v.back().r = 0;
                            };
                            int r(0);
                            int ost1(0);
                            double dx(0), dy(0);
                            std::vector<int> ost_corr;
                            approx_side[i].sbavki.clear();
                            for (int jjj = 1; jjj < t_v.size(); jjj++)
                            {
                                if (t_v[jjj].t == 0)
                                {
                                    ost_corr.push_back(jjj);
                                };
                            };
```

```
for (int jjj = 0; jjj < t_v.size(); jjj++)
{
	if (ost1 < ost_corr.size() && jjj == ost_corr[ost1])
	{
		dx = t_v[jjj-1].ax - t_v[jjj].ax;
		dy = t_v[jjj-1].ay - t_v[jjj].ay;
		ost1++;
	};
	t_v[jjj].ax += dx;
	t_v[jjj].ay += dy;
};
approx_side[i].sbavki.push_back(t_v[0]);
r += approx_side[i].sbavki.back().rapapr;
approx_side[i].sbavki.back().side_pol = -1;
approx_side[i].sbavki.back().r = r;
for (int jjj = 1; jjj < t_v.size(); jjj++)
{
	if (t_v[jjj].ay != approx_side[i].sbavki.back().ay)
	{
		approx_side[i].sbavki.push_back(t_v[jjj]);
		r += approx_side[i].sbavki.back().rapapr;
		approx_side[i].sbavki.back().side_pol = -1;
		approx_side[i].sbavki.back().r = r;
	};
};
CString izmenenie;
izmenenie.Format(_T("%i"), j + 1);
approx_cs[j].name += CString(" (") + izmenenie +
CString(")");

t_v.clear();
```

```
                    };
                };
            };
            t_v.clear();
            // по форме
            for (int i = 0 ; i < approximation_r.size(); i++)
            {
                for (int j = 0; j < approx_cs.size(); j++)
                {
                    if (approximation_r[i].name == approx_cs[j].name_left)
                    {
                        int ii(0), jj(0);
                        for (;;)
                        {
                            if (ii >= approximation_r[i].sbavki.size() && jj
>= approx_cs[j].sbavki_left.size())
                            {break;};
                            if (jj < approx_cs[j].sbavki_left.size())
                            {
                                if (ii < approximation_r[i].sbavki.size())
                                {
                                    if (approximation_r[i].sbavki[ii].ay
<= approx_cs[j].sbavki_left[jj].ay)
                                    {

        t_v.push_back(approx_cs[j].sbavki_left[jj]);
                                        jj++;
                                    }
                                    else
                                    {
```

```
t_v.push_back(approximation_r[i].sbavki[ii]);
                                        };
                                }
                                else
                                {

t_v.push_back(approx_cs[j].sbavki_left[jj]);
jj++;
                                        };
                                };
                                if (ii < approximation_r[i].sbavki.size())
                                {
                                        ii++;
                                };
                                t_v.back().r = 0;
                        };
                        int r(0);
                        int ost1(0);
                        double dx(0), dy(0);
                        std::vector<int> ost_corr;
                        approximation_r[i].sbavki.clear();
                        for (int jjj = 1; jjj < t_v.size(); jjj++)
                        {
                                if (t_v[jjj].t == 0)
                                {
                                        ost_corr.push_back(jjj);
                                };
                        };
                        for (int jjj = 0; jjj < t_v.size(); jjj++)
                        {
```

```
                    if (ost1 < ost_corr.size() && jjj == ost_corr[ost1])
                    {
                        dx = t_v[jjj-1].ax - t_v[jjj].ax;
                        dy = t_v[jjj-1].ay - t_v[jjj].ay;
                        ost1++;
                    };
                    t_v[jjj].ax += dx;
                    t_v[jjj].ay += dy;
                };
                approximation_r[i].sbavki.push_back(t_v[0]);
                r += approximation_r[i].sbavki.back().rapapr;
                approximation_r[i].sbavki.back().side_pol = -1;
                approximation_r[i].sbavki.back().r = r;
                for (int jjj = 1; jjj < t_v.size(); jjj++)
                {
                    if (t_v[jjj].ay != approximation_r[i].sbavki.back().ay)
                    {

        approximation_r[i].sbavki.push_back(t_v[jjj]);
                        r += approximation_r[i].sbavki.back().rapapr;
                        approximation_r[i].sbavki.back().side_pol = -1;
                        approximation_r[i].sbavki.back().r = r;
                    };
                };
                CString izmenenie;
                izmenenie.Format(_T("%i"), j + 1);
```

```
                    approx_cs[j].name += CString(" (") + izmenenie +
CString(")");
                    t_v.clear();
                };
            };
            // Компенсация другой кромки
            if (approximation_r[i].sbavki.back().r >
approximation_l[i].sbavki.back().r)
            {
                for (;;)
                {

    approximation_l[i].sbavki.push_back(approximation_l[i].sbavki.back());
                    approximation_l[i].sbavki.back().r++;
                    approximation_l[i].sbavki.back().ay -=
approximation_l[i].sbavki.back().bp;
                    if (approximation_r[i].sbavki.back().r ==
approximation_l[i].sbavki.back().r)
                    {break;}
                };
            };
            if (approximation_r[i].sbavki.back().r <
approximation_l[i].sbavki.back().r)
            {
                for (;;)
                {
                    approximation_l[i].sbavki.pop_back();

                    if (approximation_r[i].sbavki.back().r ==
approximation_l[i].sbavki.back().r)
```

```
                        {break;}
                    };
                };
            };
            t_v.clear();
            for (int i = 0 ; i < approximation_l.size(); i++)
            {
                for (int j = 0; j < approx_cs.size(); j++)
                {
                    if (approximation_l[i].name == approx_cs[j].name_left)
                    {
                        int ii(0), jj(0);
                        for (;;)
                        {
                            if (ii >= approximation_l[i].sbavki.size() && jj >=
approx_cs[j].sbavki_left.size())
                            {break;};
                            if (jj < approx_cs[j].sbavki_left.size())
                            {
                                if (ii < approximation_l[i].sbavki.size())
                                {
                                    if     (approximation_l[i].sbavki[ii].ay
<= approx_cs[j].sbavki_left[jj].ay)
                                    {
                                t_v.push_back(approx_cs[j].sbavki_left[jj]);
                                        jj++;
                                    }
                                    else
                                    {
                            t_v.push_back(approximation_l[i].sbavki[ii]);
```

```
                    };
                }
                else
                {

t_v.push_back(approx_cs[j].sbavki_left[jj]);

                    jj++;
                };
            };
            if (ii < approximation_l[i].sbavki.size())
            {
                ii++;
            };
            t_v.back().r = 0;
        };
        int r(0);
        int ost1(0);
        double dx(0), dy(0);
        std::vector<int> ost_corr;
        approximation_l[i].sbavki.clear();
        for (int jjj = 1; jjj < t_v.size(); jjj++)
        {
            if (t_v[jjj].t == 0)
            {
                ost_corr.push_back(jjj);
            };
        };
        for (int jjj = 0; jjj < t_v.size(); jjj++)
        {
```

```
                    if (ost1 < ost_corr.size() && jjj == ost_corr[ost1])
                    {
                        dx = t_v[jjj-1].ax - t_v[jjj].ax;
                        dy = t_v[jjj-1].ay - t_v[jjj].ay;
                        ost1++;
                    };
                    t_v[jjj].ax += dx;
                    t_v[jjj].ay += dy;
                };
                approximation_l[i].sbavki.push_back(t_v[0]);
                r += approximation_l[i].sbavki.back().rapapr;
                approximation_l[i].sbavki.back().side_pol = -1;
                approximation_l[i].sbavki.back().r = r;
                for (int jjj = 1; jjj < t_v.size(); jjj++)
                {
                    if (t_v[jjj].ay != approximation_l[i].sbavki.back().ay)
                    {

        approximation_l[i].sbavki.push_back(t_v[jjj]);
                        r += approximation_l[i].sbavki.back().rapapr;
                        approximation_l[i].sbavki.back().side_pol = -1;
                        approximation_l[i].sbavki.back().r = r;
                    };
                };
                CString izmenenie;
                izmenenie.Format(_T("%i"), j + 1);
```

```
                    approx_cs[j].name += CString(" (") + izmenenie +
CString(")");
                    t_v.clear();
                };
            };
            // Компенсация другой кромки
            if (approximation_l[i].sbavki.back().r >
approximation_r[i].sbavki.back().r)
            {
                for (;;)
                {

    approximation_r[i].sbavki.push_back(approximation_r[i].sbavki.back());
                    approximation_r[i].sbavki.back().r++;
                    approximation_r[i].sbavki.back().ay -=
approximation_r[i].sbavki.back().bp;
                    if (approximation_l[i].sbavki.back().r ==
approximation_r[i].sbavki.back().r)
                    {break;}
                };
            };
            if (approximation_l[i].sbavki.back().r <
approximation_r[i].sbavki.back().r)
            {
                for (;;)
                {
                    approximation_r[i].sbavki.pop_back();

                    if (approximation_l[i].sbavki.back().r ==
approximation_r[i].sbavki.back().r)
```

```
                                {break;}
                            };
                        };
                    };
                    t_v.clear();
                    // Синхронизация внутренних элементов
                    // кромка - кромка
                if (!approx_side.empty())
                {
                    for (int i = 0; i < approx_side.size(); i++)
                    {
                        if (!approx_side[i].sopr_name.IsEmpty() && approx_side.size() > 1)
                        {
                            for (int j = 0; j < approx_side.size(); j++)
                            {
                                if (approx_side[i].sopr_name == approx_side[j].name)
                                {
                                    for (int h = 0; h < approx_side[j].sbavki.size(); h++)
                                    {
                                        if (approx_side[j].sbavki[h].ay <= approx_side[i].sbavki[0].ay)
                                        {
                                            approx_side[i].sbavki[0].ay = approx_side[j].sbavki[h].ay;
                                            approx_side[i].sbavki[0].r = approx_side[j].sbavki[h].r;

                                            CString poz, stol;
                                            int pozx(0);
```

```
                                        double
prev(approx_side[i].sbavki[0].ax);
                                        pozx                                    =
floor(abs(approx_side[i].sbavki[0].ax       -       approx_side[j].sbavki[h].ax)       /
approx_side[j].sbavki[h].ap + 0.5);
                                        stol.Format(_T("%i"), pozx);
                                        if     (approx_side[i].sbavki[0].ax     -
approx_side[j].sbavki[h].ax < 0)
                                        {
                                            poz = CString("(") + stol +
CString("<-- ") + approx_side[i].sopr_name + CString(")");
                                            approx_side[i].sbavki[0].ax    =
approx_side[j].sbavki[h].ax - pozx * approx_side[j].sbavki[h].ap;

                                        }
                                        else
                                        {
                                            poz      =      CString("(")      +
approx_side[i].sopr_name + CString("--> ") + stol + CString(")");
                                            approx_side[i].sbavki[0].ax    =
approx_side[j].sbavki[h].ax + pozx * approx_side[j].sbavki[h].ap;
                                        };

                                        approx_side[i].name += poz;

                                        int r(approx_side[j].sbavki[h].r);

                                        double dax(0);
```

```
                                        dax = approx_side[i].sbavki[0].ax -
prev;
                                        for (int y = 1; y <
approx_side[i].sbavki.size(); y++)
                                        {
                                            r++;
                                            approx_side[i].sbavki[y].ay =
approx_side[i].sbavki[0].ay - y*approx_side[j].sbavki[h].bp;
                                            approx_side[i].sbavki[y].r = r;
                                            approx_side[i].sbavki[y].ax +=
dax;
                                        };
                                        break;
                                    };
                                };
                                break;
                            };
                        };
                    };
                    if (!approximation_l.empty())
                    {
                        for (int j = 0; j < approximation_l.size(); j++)
                        {
                            if (approx_side[i].sopr_name ==
approximation_l[j].name)
                            {
                                for (int h = 0; h <
approximation_l[j].sbavki.size(); h++)
                                {
```

```
if (approximation_l[j].sbavki[h].ay <= approx_side[i].sbavki[0].ay)
{
approx_side[i].sbavki[0].ay = approximation_l[j].sbavki[h].ay;
approx_side[i].sbavki[0].r = approximation_l[j].sbavki[h].r;
CString poz, stol;
int pozx(0);
double prev(approx_side[i].sbavki[0].ax);
pozx = floor(abs(approx_side[i].sbavki[0].ax - approximation_l[j].sbavki[h].ax) / approximation_l[j].sbavki[h].ap + 0.5);
stol.Format(_T("%i"), pozx);
if (approx_side[i].sbavki[0].ax - approximation_l[j].sbavki[h].ax < 0)
{
poz = CString("(") + stol + CString("<-- ") + approx_side[i].sopr_name + CString(")");
approx_side[i].sbavki[0].ax = approximation_l[j].sbavki[h].ax - pozx * approximation_l[j].sbavki[h].ap;
}
else
{
poz = CString("(") + approx_side[i].sopr_name + CString("--> ") + stol + CString(")");
approx_side[i].sbavki[0].ax = approximation_l[j].sbavki[h].ax + pozx * approximation_l[j].sbavki[h].ap;
};
```

```
                                        approx_side[i].name += poz;
                                        int r(approximation_l[j].sbavki[h].r);
                                        double dax(0);
                                        dax = approx_side[i].sbavki[0].ax -
prev;
                                        for (int y = 1; y <
approx_side[i].sbavki.size(); y++)
                                        {
                                                r++;
                                                approx_side[i].sbavki[y].ay =
approx_side[i].sbavki[0].ay - y*approximation_l[j].sbavki[h].bp;
                                                approx_side[i].sbavki[y].r = r;
                                                approx_side[i].sbavki[y].ax +=
dax;
                                        };
                                        break;
                                    };
                                };
                                break;
                            };
                        };
                    };
                };
            };
        // форма - форма
        if (!approximation_l.empty())
        {
            for (int i = 0; i < approximation_l.size(); i++)
            {
```

```
            if (!approximation_l[i].sopr_name.IsEmpty() && approximation_l.size() > 1)
                {
                    for (int j = 0; j < approximation_l.size(); j++)
                    {
                        if (approximation_l[i].sopr_name == approximation_l[j].name)
                        {
                            for (int h = 0; h < approximation_l[j].sbavki.size(); h++)
                            {
                                if (approximation_l[j].sbavki[h].ay <= approximation_l[i].sbavki[0].ay)
                                {
                                    approximation_l[i].sbavki[0].ay = approximation_l[j].sbavki[h].ay;
                                    approximation_l[i].sbavki[0].r = approximation_l[j].sbavki[h].r;
                                    CString poz, stol;
                                    int pozx(0);
                                    double prev(approximation_l[i].sbavki[0].ax);
                                    pozx = floor(abs(approximation_l[i].sbavki[0].ax - approximation_l[j].sbavki[h].ax) / approximation_l[j].sbavki[h].ap + 0.5);
                                    stol.Format(_T("%i"), pozx);
                                    if (approximation_l[i].sbavki[0].ax - approximation_l[j].sbavki[h].ax < 0)
                                    {
```

```
                poz = CString("(") + stol + CString("<-- ") + approximation_l[i].sopr_name + CString(")");

        approximation_l[i].sbavki[0].ax = approximation_l[j].sbavki[h].ax - pozx * approximation_l[j].sbavki[h].ap;
                }
                else
                {
                poz = CString("(") + approximation_l[i].sopr_name + CString("--> ") + stol + CString(")");

        approximation_l[i].sbavki[0].ax = approximation_l[j].sbavki[h].ax + pozx * approximation_l[j].sbavki[h].ap;
                };
                approximation_l[i].name += poz;
                int r(approximation_l[j].sbavki[h].r);
                double dax(0);
            dax = approximation_l[i].sbavki[0].ax - prev;
            for (int y = 1; y < approximation_l[i].sbavki.size(); y++)
                {
                r++;

        approximation_l[i].sbavki[y].ay = approximation_l[i].sbavki[0].ay - y*approximation_l[j].sbavki[h].bp;
                approximation_l[i].sbavki[y].r = r;
                approximation_l[i].sbavki[y].ax += dax;
```

```
                                        };
                                        break;
                                    };
                                };
                                break;
                            };
                        };
                    };
                    if (!approx_side.empty())
                    {
                        for (int j = 0; j < approx_side.size(); j++)
                        {
                            if (approximation_l[i].sopr_name == approx_side[j].name)
                            {
                                for (int h = 0; h < approx_side[j].sbavki.size(); h++)
                                {
                                    if (approx_side[j].sbavki[h].ay <= approximation_l[i].sbavki[0].ay)
                                    {
                                        approximation_l[i].sbavki[0].ay = approx_side[j].sbavki[h].ay;
                                        approximation_l[i].sbavki[0].r = approx_side[j].sbavki[h].r;
                                        CString poz, stol;
                                        int pozx(0);
                                        double prev(approximation_l[i].sbavki[0].ax);
```

```
                                        pozx =
floor(abs(approximation_l[i].sbavki[0].ax - approx_side[j].sbavki[h].ax) /
approx_side[j].sbavki[h].ap + 0.5);
                                        stol.Format(_T("%i"), pozx);
                                        if (approximation_l[i].sbavki[0].ax -
approx_side[j].sbavki[h].ax < 0)
                                        {
                                            poz = CString("(") + stol +
CString("<-- ") + approximation_l[i].sopr_name + CString(")");

        approximation_l[i].sbavki[0].ax = approx_side[j].sbavki[h].ax - pozx *
approx_side[j].sbavki[h].ap;
                                        }
                                        else
                                        {
                                            poz = CString("(") +
approximation_l[i].sopr_name + CString("--> ") + stol + CString(")");

        approximation_l[i].sbavki[0].ax = approx_side[j].sbavki[h].ax + pozx *
approx_side[j].sbavki[h].ap;
                                        };
                                        approximation_l[i].name += poz;
                                        int r(approx_side[j].sbavki[h].r);
                                        double dax(0);
                                    dax = approximation_l[i].sbavki[0].ax -
prev;
                                for (int y = 1; y <
approximation_l[i].sbavki.size(); y++)
                                {
                                        r++;
```

```
approximation_l[i].sbavki[y].ay = approximation_l[i].sbavki[0].ay - y*approx_side[j].sbavki[h].bp;
                                                    approximation_l[i].sbavki[y].r = r;
                                                    approximation_l[i].sbavki[y].ax += dax;
                                                    };
                                                    break;
                                                };
                                            };
                                            break;
                                        };
                                    };
                                };
                            };
                            for (int i = 0; i < approximation_r.size(); i++)
                            {
                                if (!approximation_r[i].sopr_name.IsEmpty() && approximation_r.size() > 1)
                                {
                                    for (int j = 0; j < approximation_r.size(); j++)
                                    {
                                        if (approximation_r[i].sopr_name == approximation_r[j].name)
                                        {
                                            for (int h = 0; h < approximation_r[j].sbavki.size(); h++)
                                            {
```

```
if (approximation_r[j].sbavki[h].ay <= approximation_r[i].sbavki[0].ay)
{
approximation_r[i].sbavki[0].ay = approximation_r[j].sbavki[h].ay;
approximation_r[i].sbavki[0].r = approximation_r[j].sbavki[h].r;
CString poz, stol;
int pozx(0);
double prev(approximation_r[i].sbavki[0].ax);
pozx = floor(abs(approximation_r[i].sbavki[0].ax - approximation_r[j].sbavki[h].ax) / approximation_r[j].sbavki[h].ap + 0.5);
stol.Format(_T("%i"), pozx);
if (approximation_r[i].sbavki[0].ax - approximation_r[j].sbavki[h].ax < 0)
{
poz = CString(" & (") + stol + CString("<-- ") + approximation_l[i].sopr_name + CString(")");
approximation_r[i].sbavki[0].ax = approximation_r[j].sbavki[h].ax - pozx * approximation_r[j].sbavki[h].ap;
}
else
{
poz = CString(" & (") + approximation_l[i].sopr_name + CString("--> ") + stol + CString(")");
```

```
approximation_r[i].sbavki[0].ax = approximation_r[j].sbavki[h].ax + pozx * approximation_r[j].sbavki[h].ap;
                                                                    };
                                                                    approximation_l[i].name += poz;
                                                                    int r(approximation_r[j].sbavki[h].r);
                                                                    double dax(0);
                                                                dax = approximation_r[i].sbavki[0].ax - prev;
                                                                    for (int y = 1; y < approximation_r[i].sbavki.size(); y++)
                                                                    {
                                                                        r++;

approximation_r[i].sbavki[y].ay = approximation_r[i].sbavki[0].ay - y*approximation_r[j].sbavki[h].bp;
                                                                        approximation_r[i].sbavki[y].r = r;
                                                                    approximation_r[i].sbavki[y].ax += dax;
                                                                    };
                                                                    break;
                                                                };
                                                            };
                                                        break;
                                                    };
                                                };
                                            };
                                            if (!approx_side.empty())
                                            {
```

```
for (int j = 0; j < approx_side.size(); j++)
{
if (approximation_r[i].sopr_name == approx_side[j].name)
{
for (int h = 0; h < approx_side[j].sbavki.size(); h++)
{
if (approx_side[j].sbavki[h].ay <= approximation_r[i].sbavki[0].ay)
{
approximation_r[i].sbavki[0].ay = approx_side[j].sbavki[h].ay;
approximation_r[i].sbavki[0].r = approx_side[j].sbavki[h].r;
CString poz, stol;
int pozx(0);
double prev(approximation_r[i].sbavki[0].ax);
pozx = floor(abs(approximation_r[i].sbavki[0].ax - approx_side[j].sbavki[h].ax) / approx_side[j].sbavki[h].ap + 0.5);
stol.Format(_T("%i"), pozx);
if (approximation_r[i].sbavki[0].ax - approx_side[j].sbavki[h].ax < 0)
{
poz = CString(" & (") + stol + CString("<-- ") + approximation_l[i].sopr_name + CString(")");
approximation_r[i].sbavki[0].ax = approx_side[j].sbavki[h].ax - pozx * approx_side[j].sbavki[h].ap;
```

```
                                        }
                                        else
                                        {
                    poz = CString(" & (") + approximation_l[i].sopr_name + CString("--
> ") + stol + CString(")");
                    approximation_r[i].sbavki[0].ax = approx_side[j].sbavki[h].ax +
pozx * approx_side[j].sbavki[h].ap;
                                        };
                                        approximation_l[i].name += poz;
                                        int r(approx_side[j].sbavki[h].r);
                                        double dax(0);
                                    dax = approximation_r[i].sbavki[0].ax -
prev;
                                for (int y = 1; y <
approximation_r[i].sbavki.size(); y++)
                                        {
                                            r++;

        approximation_r[i].sbavki[y].ay = approximation_r[i].sbavki[0].ay -
y*approx_side[j].sbavki[h].bp;
                                                approximation_r[i].sbavki[y].r
= r;

        approximation_r[i].sbavki[y].ax += dax;
                                        };
                                        break;
                                    };
                                };
                                break;
                            };
```

```
                    };
                };
            };
        };
        for (int i = 0; i < obj.size(); i++)
        {
            if (aConsole.m_Check == GRAFIKA)
            {
                if (obj[i].st != 4 && obj[i].st != 5)
                {info.push_back(obj[i]);};
            }
            else if (aConsole.m_Check == ZAMER)
            {
                if (obj[i].st == 4 || obj[i].st == 5)
                {info.push_back(obj[i]);};
            }
            else
            {
                if (obj[i].st != 1000 && obj[i].st != 1001)
                {info.push_back(obj[i]);}
            };
        };
        CString string;
        CString razm;
        if (m_Izm == SM)
        {razm = " (см.);";}
        else
        {razm = " (мм.);";}
        for (int i = 0; i < info.size(); i++)
        {
```

```
                if (!info[i].name.IsEmpty())
                {
                        string                  +=info[i].name+CString("                  =
")+info[i].value+razm+CString("\r\n");
                        aConsole.m_Edit_Info.SetWindowText(string);
                };
        };
        UpdateAllViews(NULL); // Обновить все представления (NULL)
        buffer_vector.clear(); // Очистка буферного вектора
    }
    else
    {
        for (int i = 0; i < aConsole.buff_inp.size(); i++) // добавить ограничения
        {
            com_trans.push_back(aConsole.buff_inp[i]);
        };
        CStdioFile      File(CString("C:\\Dkw_com.txt"),CFile::modeCreate      |
CFile::modeWrite);
        CString bbb;
        for (int i = 0; i < com_trans.size(); i++)
        {
            bbb = com_trans[i];
            if (!bbb.IsEmpty())
            {
                File.WriteString(bbb+CString("\r\n"));
            };
        };
        File.Close();
    };
    };
```

```
// Стандартный отчет
void CDesignerkwearDoc::OnReport()
{
	try
{
//Инициализация COM интерфейса
CoInitialize(NULL);
//Опеределение указателя на приложение Excel
Excel::_ApplicationPtr xl;
//Начало работы
xl.CreateInstance(L"Excel.Application");
//Сделать приложение видимым
xl->Visible = true;
//ДОбавить новую рабочую книгу
xl->Workbooks->Add(Excel::xlWorksheet);
int m(0);
ris_list = 0;
if (!approximation_l.empty())
{
//Получить указатель на активный рабочий лист
Excel::_WorksheetPtr pSheet = xl->ActiveSheet;
//Назвать рабочий лист
pSheet->Name = "Отчет по аппроксимации ФОРМЫ";
//ПОлучить указатель на ячейку активного рабочего листа
Excel::RangePtr pRange = pSheet->Cells;
//Заполняем таблицу
for (int i = 0; i < approximation_l.size(); i++)
{
	if (!approximation_l[i].name.IsEmpty())
	{
```

```
                for (int j = 0; j < approximation_l[i].sbavki.size(); j++)
                {
                        if (j == 1)
                        {
                                pRange->Item[1][1+m*20] = "Имя:";
                                pRange->Item[1][2+m*20]          =          (_bstr_t)
approximation_l[i].name;
                                pRange->Item[2][1+m*20] = "Левая кромка";
                                pRange->Item[3][1+m*20] = "ax";
                                pRange->Item[3][2+m*20] = "ay";
                                pRange->Item[3][3+m*20] = "t";
                                pRange->Item[3][4+m*20] = "xl";
                                pRange->Item[3][5+m*20] = "сбавка";
                                pRange->Item[3][6+m*20] = "Ряд";
                                pRange->Item[3][7+m*20] = "Направление";
                        };

                        pRange->Item[j+4][1+m*20]                                =
approximation_l[i].sbavki[j].ax;
                        pRange->Item[j+4][2+m*20]                                =
approximation_l[i].sbavki[j].ay;
                        pRange->Item[j+4][3+m*20] = approximation_l[i].sbavki[j].t;
                        pRange->Item[j+4][4+m*20]                                =
approximation_l[i].sbavki[j].kx;
                        pRange->Item[j+4][5+m*20]                                =
approximation_l[i].sbavki[j].sbav;
                        pRange->Item[j+4][6+m*20] = approximation_l[i].sbavki[j].r;
                        pRange->Item[j+4][7+m*20]                                =
approximation_l[i].sbavki[j].pol*(-approximation_l[i].sbavki[j].side_pol);
                };
```

```
                m++;
            }
        }

        m = 0;
        for (int i = 0; i < approximation_r.size(); i++)
        {
            if (!approximation_r[i].name.IsEmpty())
            {
                for (int j = 0; j < approximation_r[i].sbavki.size(); j++)
                {
                    if (j == 1)
                    {
                        pRange->Item[2][10+m*20] = "Правая кромка";
                        pRange->Item[3][10+m*20] = "ax";
                        pRange->Item[3][11+m*20] = "ay";
                        pRange->Item[3][12+m*20] = "t";
                        pRange->Item[3][13+m*20] = "xl";
                        pRange->Item[3][14+m*20] = "сбавка";
                        pRange->Item[3][15+m*20] = "Ряд";
                        pRange->Item[3][16+m*20] = "Направление";
                    };
                    pRange->Item[j+4][10+m*20]                    =
approximation_r[i].sbavki[j].ax;
                    pRange->Item[j+4][11+m*20]                    =
approximation_r[i].sbavki[j].ay;
                    pRange->Item[j+4][12+m*20]                    =
approximation_r[i].sbavki[j].t;
                    pRange->Item[j+4][13+m*20]                    =
approximation_r[i].sbavki[j].kx;
```

```
                    pRange->Item[j+4][14+m*20]                    =
approximation_r[i].sbavki[j].sbav;
                    pRange->Item[j+4][15+m*20]                    =
approximation_r[i].sbavki[j].r;
                    pRange->Item[j+4][16+m*20]                    =
approximation_r[i].sbavki[j].pol*(-approximation_r[i].sbavki[j].side_pol);
                };
                m++;
            };
        }
        m = 0;
        };
        for (int i = 0; i < approximation_l.size(); i++)
        {
            if (!approximation_l[i].sbavki_dop1.empty())
            {ris_list++;};

        };
        if (ris_list > 0)
        {
        // Создание нового листа
        Excel::_WorksheetPtr pChart4=xl->ActiveWorkbook->Sheets->Add();
            pChart4->Name = "Дополнительные ряды ФОРМЫ";
        Excel::_WorksheetPtr pSheet4 = xl->ActiveSheet;
            //ПОлучить указатель на ячейку активного рабочего листа
        Excel::RangePtr pRange4 = pSheet4->Cells;
        //Заполняем таблицу
        for (int i = 0; i < approximation_l.size(); i++)
        {
            if (!approximation_l[i].sbavki_dop1.empty())
```

```
            {
                for (int j = 0; j < approximation_l[i].sbavki_dop1.size(); j++)
                {
                    if (j == 1)
                    {
                        pRange4->Item[1][1+5*m] = "Имя:";
                        pRange4->Item[1][2+5*m] = (_bstr_t)
approximation_l[i].name + " (" + approximation_l[i].name_dop + ")";
                        pRange4->Item[2][1+5*m] = "t";
                        pRange4->Item[2][2+5*m] = "Ряд";
                        pRange4->Item[2][3+5*m] = "Кол-во доп. рядов";
                    };
                    pRange4->Item[j+3][1+5*m] =
approximation_l[i].sbavki_dop1[j].t;
                    pRange4->Item[j+3][2+5*m] =
approximation_l[i].sbavki_dop1[j].r;
                    pRange4->Item[j+3][3+5*m] =
approximation_l[i].sbavki_dop1[j].dop_r;
                };
                m++;
            };
        }
        m = 0;
        };
        ris_list = 0;
        for (int i = 0; i < approximation_r.size(); i++)
        {
            if (!approximation_r[i].sbavki_dop1.empty())
            {ris_list++;};
```

```
        };
        if (ris_list > 0)
        {
        // Создание нового листа
        Excel::_WorksheetPtr pChart4=xl->ActiveWorkbook->Sheets->Add();
                pChart4->Name = "Дополнительные ряды ФОРМЫ";
        Excel::_WorksheetPtr pSheet4 = xl->ActiveSheet;
                //ПОлучить указатель на ячейку активного рабочего листа
        Excel::RangePtr pRange4 = pSheet4->Cells;
        //Заполняем таблицу
        for (int i = 0; i < approximation_r.size(); i++)
        {
                if (!approximation_r[i].sbavki_dop1.empty())
                {
                        for (int j = 0; j < approximation_r[i].sbavki_dop1.size(); j++)
                        {
                                if (j == 1)
                                {
                                        pRange4->Item[1][1+5*m] = "Имя:";
                                        pRange4->Item[1][2+5*m]          =          (_bstr_t)
approximation_r[i].name + " (" + approximation_r[i].name_dop + ")";
                                        pRange4->Item[2][1+5*m] = "t";
                                        pRange4->Item[2][2+5*m] = "Ряд";
                                        pRange4->Item[2][3+5*m] = "Кол-во доп. рядов";
                                };
                                pRange4->Item[j+3][1+5*m]                                =
approximation_r[i].sbavki_dop1[j].t;
                                pRange4->Item[j+3][2+5*m]                                =
approximation_r[i].sbavki_dop1[j].r;
```

```
                    pRange4->Item[j+3][3+5*m]                    =
approximation_r[i].sbavki_dop1[j].dop_r;
                    };
                    m++;
                };
        }
        m = 0;
        };
        ris_list = 0;
        if (!approx_cs.empty())
        {
        // Создание нового листа
        Excel::_WorksheetPtr pChart=xl->ActiveWorkbook->Sheets->Add();
                pChart->Name = "Отчет о СОЕДИНЕНИИ";
        Excel::_WorksheetPtr pSheet1 = xl->ActiveSheet;
                //ПОлучить указатель на ячейку активного рабочего листа
        Excel::RangePtr pRange1 = pSheet1->Cells;
        //Заполняем таблицу
        for (int i = 0; i < approx_cs.size(); i++)
        {
                if (!approx_cs[i].name.IsEmpty())
                {
                //Заголовок таблицы
                pRange1->Item[2][1+22*m] = "Имя:";
                pRange1->Item[2][2+22*m]  =  (_bstr_t)  approx_cs[i].name  +  (_bstr_t)
approx_cs[i].type;
                pRange1->Item[4][1+22*m] = "ax";
                pRange1->Item[4][2+22*m] = "ay";
                pRange1->Item[4][3+22*m] = "t";
                pRange1->Item[4][4+22*m] = "xl";
```

```
pRange1->Item[4][5+22*m] = "сбавка";
pRange1->Item[4][6+22*m] = "Ряд";
pRange1->Item[4][7+22*m] = "Направление";
pRange1->Item[4][10+22*m] = "ax";
pRange1->Item[4][11+22*m] = "ay";
pRange1->Item[4][12+22*m] = "t";
pRange1->Item[4][13+22*m] = "xl";
pRange1->Item[4][14+22*m] = "сбавка";
pRange1->Item[4][15+22*m] = "Ряд";
pRange1->Item[4][16+22*m] = "Направление";
pRange1->Item[4][18+22*m] = "Линия разностей (ряд)";
pRange1->Item[7][18+22*m] = "Разность по рядам";
pRange1->Item[10][18+22*m] = "Распределение доп. рядов";
if (approx_cs[i].line_r == 0)
{
	pRange1->Item[5][18+22*m] = "отсутствует";
}
else
{
	pRange1->Item[5][18+22*m] = approx_cs[i].line_r;
}
CString r;
if (approx_cs[i].rze == 1)
{
	r = "ряд";
}
else if (approx_cs[i].rze >= 2 && approx_cs[i].rze <= 4)
{
	r = "ряда";
}
```

```
else
{
        r = "рядов";
};
pRange1->Item[8][18+22*m] = approx_cs[i].rzs;
pRange1->Item[11][18+22*m] = "через каждые ";
pRange1->Item[12][18+22*m] = approx_cs[i].rze;
pRange1->Item[12][19+22*m] = (_bstr_t) r;
for (int j = 0; j < approx_cs[i].sbavki_left.size(); j++)
{
        pRange1->Item[j+5][1+22*m] = approx_cs[i].sbavki_left[j].ax;
        pRange1->Item[j+5][2+22*m] = approx_cs[i].sbavki_left[j].ay;
        pRange1->Item[j+5][3+22*m] = approx_cs[i].sbavki_left[j].t;
        pRange1->Item[j+5][4+22*m] = approx_cs[i].sbavki_left[j].kx;
        pRange1->Item[j+5][5+22*m] = approx_cs[i].sbavki_left[j].sbav;
        pRange1->Item[j+5][6+22*m] = approx_cs[i].sbavki_left[j].r;
        pRange1->Item[j+5][7+22*m] = approx_cs[i].sbavki_left[j].pol;
};
for (int j = 0; j < approx_cs[i].sbavki_right.size(); j++)
{
        pRange1->Item[j+5][10+22*m] = approx_cs[i].sbavki_right[j].ax;
        pRange1->Item[j+5][11+22*m] = approx_cs[i].sbavki_right[j].ay;
        pRange1->Item[j+5][12+22*m] = approx_cs[i].sbavki_right[j].t;
        pRange1->Item[j+5][13+22*m] = approx_cs[i].sbavki_right[j].kx;
        pRange1->Item[j+5][14+22*m] = approx_cs[i].sbavki_right[j].sbav;
        pRange1->Item[j+5][15+22*m] = approx_cs[i].sbavki_right[j].r;
        pRange1->Item[j+5][16+22*m] = approx_cs[i].sbavki_right[j].pol;
}
m++;
};
```

```
}
m = 0;
}
if (!approx_side.empty())
{
// Создание нового листа
Excel::_WorksheetPtr pChart2=xl->ActiveWorkbook->Sheets->Add();
        pChart2->Name = "Отчет об аппроксимации КРОМКИ";
Excel::_WorksheetPtr pSheet2 = xl->ActiveSheet;
        //ПОлучить указатель на ячейку активного рабочего листа
Excel::RangePtr pRange2 = pSheet2->Cells;
//Заполняем таблицу
for (int i = 0; i < approx_side.size(); i++)
{
        if (!approx_side[i].name.IsEmpty())
        {
                for (int j = 0; j < approx_side[i].sbavki.size(); j++)
                {
                        if (j == 1)
                        {
                                pRange2->Item[1][1+9*m] = "Имя:";
                                pRange2->Item[1][2+9*m]          =          (_bstr_t)
approx_side[i].name;
                                pRange2->Item[2][1+9*m] = "ax";
                                pRange2->Item[2][2+9*m] = "ay";
                                pRange2->Item[2][3+9*m] = "t";
                                pRange2->Item[2][4+9*m] = "xl";
                                pRange2->Item[2][5+9*m] = "сбавка";
                                pRange2->Item[2][6+9*m] = "Ряд";
                                pRange2->Item[2][7+9*m] = "Направление";
```

```
                        };
                        pRange2->Item[j+3][1+9*m] = approx_side[i].sbavki[j].ax;
                        pRange2->Item[j+3][2+9*m] = approx_side[i].sbavki[j].ay;
                        pRange2->Item[j+3][3+9*m] = approx_side[i].sbavki[j].t;
                        pRange2->Item[j+3][4+9*m] = approx_side[i].sbavki[j].kx;
                        pRange2->Item[j+3][5+9*m] = approx_side[i].sbavki[j].sbav;
                        pRange2->Item[j+3][6+9*m] = approx_side[i].sbavki[j].r;
                        pRange2->Item[j+3][7+9*m] = approx_side[i].sbavki[j].pol*(-
approx_side[i].sbavki[j].side_pol);
                    };
                    m++;
                };
        }
        m = 0;
        };

        ris_list = 0;
        for (int i = 0; i < approx_side.size(); i++)
        {
            if (!approx_side[i].sbavki_dop1.empty())
            {ris_list++;};
        };
        if (ris_list > 0)
        {
        // Создание нового листа
        Excel::_WorksheetPtr pChart3=xl->ActiveWorkbook->Sheets->Add();
            pChart3->Name = "Дополнительные ряды КРОМКИ";
        Excel::_WorksheetPtr pSheet3 = xl->ActiveSheet;
            //ПОлучить указатель на ячейку активного рабочего листа
        Excel::RangePtr pRange3 = pSheet3->Cells;
```

```
//Заполняем таблицу
for (int i = 0; i < approx_side.size(); i++)
{
	if (!approx_side[i].sbavki_dop1.empty())
	{
		for (int j = 0; j < approx_side[i].sbavki_dop1.size(); j++)
		{
			if (j == 1)
			{
				pRange3->Item[1][1+5*m] = "Имя:";
				pRange3->Item[1][2+5*m] = (_bstr_t)
approx_side[i].name + " (" + approx_side[i].name_dop + ")";
				pRange3->Item[2][1+5*m] = "t";
				pRange3->Item[2][2+5*m] = "Ряд";
				pRange3->Item[2][3+5*m] = "Кол-во доп. рядов";
			};
			pRange3->Item[j+3][1+5*m] =
approx_side[i].sbavki_dop1[j].t;
			pRange3->Item[j+3][2+5*m] =
approx_side[i].sbavki_dop1[j].r;
			pRange3->Item[j+3][3+5*m] =
approx_side[i].sbavki_dop1[j].dop_r;
		};
		m++;
	};
}
m = 0;
};
f (!side_d.empty())
{
```

```
            // Создание нового листа
        Excel::_WorksheetPtr pChart4=xl->ActiveWorkbook->Sheets->Add();
            pChart4->Name = "Дополнительные ряды + область";
        Excel::_WorksheetPtr pSheet4 = xl->ActiveSheet;
            //ПОлучить указатель на ячейку активного рабочего листа
        Excel::RangePtr pRange4 = pSheet4->Cells;
        //Заполняем таблицу
        for (int i = 0; i < side_d.size(); i++)
        {
            if (!side_d[i].sbavki_dop1.empty())
            {
                for (int j = 0; j < side_d[i].sbavki_dop1.size(); j++)
                {
                    if (j == 1)
                    {
                        pRange4->Item[1][1+7*m] = "Имя:";
                        pRange4->Item[1][2+7*m] = (_bstr_t) side_d[i].name +
" (" + side_d[i].name_dop + ")";
                        pRange4->Item[2][1+7*m] = "t";
                        pRange4->Item[2][2+7*m] = "Ряд";
                        pRange4->Item[2][3+7*m] = "Кол-во стол. слева";
                        pRange4->Item[2][4+7*m] = "Кол-во доп. рядов";
                        pRange4->Item[2][5+7*m] = "Кол-во стол. справа";
                    };
                    pRange4->Item[j+3][1+7*m] = side_d[i].sbavki_dop1[j].t;
                    pRange4->Item[j+3][2+7*m] = side_d[i].sbavki_dop1[j].r;
                    pRange4->Item[j+3][3+7*m]                                =
side_d[i].sbavki_dop1[j].stol_l;
                    pRange4->Item[j+3][4+7*m]                                =
side_d[i].sbavki_dop1[j].dop_r;
```

```
                    pRange4->Item[j+3][5+7*m]                    =
side_d[i].sbavki_dop1[j].stol_r;
                    };
                    m++;
                };
        }
        m = 0;
        };
        }
        //Если ошибка, то
        catch(_com_error & error)
        {
                AfxMessageBox(CString("COM ERROR"), MB_OK);
        }
        //Закрываем COM interface
        CoUninitialize();
        }
        // Отчет для Stoll M1
        void CDesignerkwearDoc::OnReportM1()
        {
                    try
        {
        //Инициализация COM интерфейса
        CoInitialize(NULL);
        //Опеределение указателя на приложение Excel
        Excel::_ApplicationPtr xl;
        //Начало работы
        xl.CreateInstance(L"Excel.Application");
        //Сделать приложение видимым
        xl->Visible = true;
```

```
        //ДОбавить новую рабочую книгу
        xl->Workbooks->Add(Excel::xlWorksheet);
        int m(0);
        if(!approximation_l.empty())
        {
        //Получить указатель на активный рабочий лист
        Excel::_WorksheetPtr pSheet = xl->ActiveSheet;
        //Назвать рабочий лист
        pSheet->Name = "Отчет для Stoll M1 ФОРМА";
        //ПОлучить указатель на ячейку активного рабочего листа
        Excel::RangePtr pRange = pSheet->Cells;
        //Заполняем таблицу
        // Левая кромка
        int m(0);
        for (int i = 0; i < approximation_l.size(); i++)
        {
                if (!approximation_l[i].name.IsEmpty())
                {
                        //Заголовок таблицы
                        pRange->Item[1][1+7*m] = "Имя:";
                        pRange->Item[1][2+7*m] = (_bstr_t) approximation_l[i].name;
                        pRange->Item[2][1+7*m] = "Левая кромка";
                        pRange->Item[3][1+7*m] = "Высота (ряды)"; pRange-
>Item[3][2+7*m] = "Ширина (петли)";
                        CReportM1 m1(approximation_l[i].sbavki);
                        for (int j = 0; j < m1.sbavki.size(); j++)
                        {
                                pRange->Item[j+4][1+7*m] = m1.sbavki[j].ay;
                                pRange->Item[j+4][2+7*m] = m1.sbavki[j].ax *
m1.sbavki[j].pol;
```

```
                };
                m++;
            };
        };
        // Правая кромка
        m = 0;
        for (int i = 0; i < approximation_r.size(); i++)
        {
            if (!approximation_r[i].name.IsEmpty())
            {
                pRange->Item[2][4+7*m] = "Правая кромка";
                pRange->Item[3][4+7*m] = "Высота (ряды)"; pRange->Item[3][5+7*m] = "Ширина (петли)";
                CReportM1 m1(approximation_r[i].sbavki);
                for (int j = 0; j < m1.sbavki.size(); j++)
                {
                    pRange->Item[j+4][4+7*m] = m1.sbavki[j].ay;
                    pRange->Item[j+4][5+7*m] = m1.sbavki[j].ax * m1.sbavki[j].pol;
                };
                m++;
            };
        };
        m = 0;
        }
        if (!approx_side.empty())
        {
        // Создание нового листа
        Excel::_WorksheetPtr pChart=xl->ActiveWorkbook->Sheets->Add();
            pChart->Name = "Отчет для Stoll M1 КРОМКА";
```

```
Excel::_WorksheetPtr pSheet1 = xl->ActiveSheet;
        //ПОлучить указатель на ячейку активного рабочего листа
Excel::RangePtr pRange1 = pSheet1->Cells;
//Заполняем таблицу
for (int i = 0; i < approx_side.size(); i++)
{
        if (!approx_side[i].name.IsEmpty())
        {
                //Заголовок таблицы
                pRange1->Item[1][1+7*m] = "Имя:";
                pRange1->Item[1][2+7*m] = (_bstr_t) approx_side[i].name;
                pRange1->Item[2][1+7*m] = "Кромка";
                pRange1->Item[3][1+7*m] = "Высота (ряды)"; pRange1->Item[3][2+7*m] = "Ширина (петли)";
                CReportM1 m1(approx_side[i].sbavki);
                for (int j = 0; j < m1.sbavki.size(); j++)
                {
                        pRange1->Item[j+4][1+7*m] = m1.sbavki[j].ay;
                        if (trans == Y && m1.sbavki[j].side_pol == 1)
                        {
                                pRange1->Item[j+4][2+7*m] = -1 * m1.sbavki[j].ax * m1.sbavki[j].pol;
                        }
                        else
                        {
                                pRange1->Item[j+4][2+7*m] = m1.sbavki[j].ax * m1.sbavki[j].pol;
                        };
                };
                m++;
```

```
            };
        };
        m = 0;
        }
        if (!approx_cs.empty())
        {
        // Создание нового листа
        Excel::_WorksheetPtr pChart=xl->ActiveWorkbook->Sheets->Add();
            pChart->Name = "Отчет для Stoll M1 СОЕДИНЕНИЕ";
        Excel::_WorksheetPtr pSheet1 = xl->ActiveSheet;
            //ПОлучить указатель на ячейку активного рабочего листа
        Excel::RangePtr pRange2 = pSheet1->Cells;
        //Заполняем таблицу
        for (int i = 0; i < approx_cs.size(); i++)
        {
            if (!approx_cs[i].name.IsEmpty())
            {
                //Заголовок таблицы
                pRange2->Item[1][1+7*m] = "Имя:";
                pRange2->Item[1][2+7*m] = (_bstr_t) approx_cs[i].name + (_bstr_t)
approx_cs[i].type;
                pRange2->Item[2][1+7*m] = "Кромка левая";
                pRange2->Item[3][1+7*m]    =    "Высота    (ряды)";    pRange2-
>Item[3][2+7*m] = "Ширина (петли)";
                pRange2->Item[2][4+7*m] = "Кромка правая";
                pRange2->Item[3][4+7*m]    =    "Высота    (ряды)";    pRange2-
>Item[3][5+7*m] = "Ширина (петли)";
                CReportM1 m1(approx_cs[i].sbavki_left);
                for (int j = 0; j < m1.sbavki.size(); j++)
                {
```

```
                pRange2->Item[j+4][1+7*m] = m1.sbavki[j].ay;
                pRange2->Item[j+4][2+7*m] = m1.sbavki[j].ax;
            };
            CReportM1 m1r(approx_cs[i].sbavki_right);
            for (int j = 0; j < m1r.sbavki.size(); j++)
            {
                pRange2->Item[j+4][4+7*m] = m1r.sbavki[j].ay;
                pRange2->Item[j+4][5+7*m] = m1r.sbavki[j].ax;
            };
            m++;
        };
};
m = 0;
}
}
//Если ошибка, то
catch(_com_error & error)
{
    AfxMessageBox(CString("COM ERROR"), MB_OK);
}
//Закрываем COM interface
CoUninitialize();
}
// Единицы измерения
void CDesignerkwearDoc::OnSm()
{
    m_Izm = SM;
    OnBuild();
}
void CDesignerkwearDoc::OnMm()
```

```
{
	m_Izm = MM;
	OnBuild();
}
void CDesignerkwearDoc::OnUpdateSm(CCmdUI *pCmdUI)
{
	pCmdUI->SetCheck(m_Izm == SM);
}
void CDesignerkwearDoc::OnUpdateMm(CCmdUI *pCmdUI)
{
	pCmdUI->SetCheck(m_Izm == MM);
}
// Размерные признаки
void CDesignerkwearDoc::OnRz()
{
	raz.clear();
	CFileDialog DlgOpen(true);
	// отображение стандартной панели выбора файла Open
	if(DlgOpen.DoModal()==IDOK)
	{	// создание объекта и открытие файла для чтения
		CStdioFile File(DlgOpen.GetPathName(),CFile::modeRead);
		// чтение из файла строки
		CString ref;
		do
		{
			File.ReadString(ref);
			raz.push_back(CRazmer(ref, File.GetFileTitle()));
		// Действия с aStr
		} while(!ref.IsEmpty());
		File.Close();
```

```
        }
        OnBuild();
}
int sgl(1);
void CDesignerkwearDoc::OnSglazh()
{
        sgl++;
        if (sgl == 1)
        {m_Sgl = Ys;}
        else
        {m_Sgl = Ns; sgl = 0;}
        OnBuild();
}
void CDesignerkwearDoc::OnUpdateSglazh(CCmdUI *pCmdUI)
{
        pCmdUI->SetCheck(m_Sgl == Ys);
}
int trs(0);
void CDesignerkwearDoc::OnM1Transform()
{
        trs++;
        if (trs == 1)
        {trans = Y;}
        else
        {trans = N; trs = 0;}
        OnBuild();
}
void CDesignerkwearDoc::OnUpdateM1Transform(CCmdUI *pCmdUI)
{
        pCmdUI->SetCheck(trans == Y);
```

```
}
void CDesignerkwearDoc::OnVytClose()
{
	m_Vyt = CLOSE;
	OnBuild();
}
void CDesignerkwearDoc::OnVytOpen()
{
	m_Vyt = OPEN;
	OnBuild();
}
void CDesignerkwearDoc::OnUpdateVytClose(CCmdUI *pCmdUI)
{
	pCmdUI->SetCheck(m_Vyt == CLOSE);
}
void CDesignerkwearDoc::OnUpdateVytOpen(CCmdUI *pCmdUI)
{
	pCmdUI->SetCheck(m_Vyt == OPEN);
}
int set(0);
void CDesignerkwearDoc::OnSetup()
{
	set++;
	if (set == 1)
	{
		m_prov = ON;
		for (int i = 0; i < aConsole.buff_inp.size(); i++)
		{
			savecom.push_back(aConsole.buff_inp[i]);
		};
```

```
        obj.clear();
        com.clear();
        approximation_l.clear();
        approximation_r.clear();
        aprox.clear();
        approx_side.clear();
        approx_cs.clear();
        dobj.clear();
        buffer_vector.clear();
        aConsole.m_EditConsole.SetWindowText(_T(""));
        aConsole.buff_inp.clear();
        names.clear();
        aConsole.serial.clear();
        vivod.clear();
        info.clear();
        UpdateAllViews(NULL);
}
else
{
        aConsole.m_EditConsole.SetWindowText(_T(""));
        m_prov = OFF; set = 0;
        CString ss;
        for (int i = 0; i < savecom.size(); i++)
        {
                ss =  savecom[i] + _T("\r\n");
                aConsole.m_EditConsole.SetWindowText(ss);
                aConsole.buff_inp.push_back(savecom[i]);
                aConsole.serial.push_back(savecom[i] + CString("@"));
        };
        OnBuild();
```

```
            savecom.clear();
        };
    }
    void CDesignerkwearDoc::OnUpdateSetup(CCmdUI *pCmdUI)
    {
        pCmdUI->SetCheck(m_prov == ON);
    }
```

Файл “Designer k-wearView.cpp”

```
#ifndef SHARED_HANDLERS
#include "Designer k-wear.h"
#endif
#include "Designer k-wearDoc.h"
#include "Designer k-wearView.h"
#ifdef _DEBUG
#define new DEBUG_NEW
#endif
#include <cmath>
// CDesignerkwearView
IMPLEMENT_DYNCREATE(CDesignerkwearView, CScrollView)
BEGIN_MESSAGE_MAP(CDesignerkwearView, CScrollView)
    // Стандартные команды печати
    ON_COMMAND(ID_FILE_PRINT, &CScrollView::OnFilePrint)
    ON_COMMAND(ID_FILE_PRINT_DIRECT,
&CScrollView::OnFilePrint)
    ON_COMMAND(ID_FILE_PRINT_PREVIEW,
&CDesignerkwearView::OnFilePrintPreview)
    ON_WM_CONTEXTMENU()
    ON_WM_RBUTTONUP()
    ON_COMMAND(ID_ZOOM_IN, &CDesignerkwearView::OnZoomIn)
```

```
        ON_COMMAND(ID_ZOOM_OUT,
&CDesignerkwearView::OnZoomOut)
        ON_COMMAND(ID_GRAF, &CDesignerkwearView::OnGraf)
        ON_COMMAND(ID_SBAV, &CDesignerkwearView::OnSbav)
        ON_UPDATE_COMMAND_UI(ID_SBAV,
&CDesignerkwearView::OnUpdateSbav)
        ON_UPDATE_COMMAND_UI(ID_GRAF,
&CDesignerkwearView::OnUpdateGraf)
        ON_COMMAND(ID_PETLI_INFO,
&CDesignerkwearView::OnPetliInfo)
        ON_UPDATE_COMMAND_UI(ID_PETLI_INFO,
&CDesignerkwearView::OnUpdatePetliInfo)
        ON_COMMAND(ID_NAPRAV, &CDesignerkwearView::OnNaprav)
        ON_UPDATE_COMMAND_UI(ID_NAPRAV,
&CDesignerkwearView::OnUpdateNaprav)
        ON_WM_LBUTTONUP()
        ON_WM_MOUSEMOVE()
        ON_WM_LBUTTONDOWN()
        ON_COMMAND(ID_ALL, &CDesignerkwearView::OnAll)
        ON_UPDATE_COMMAND_UI(ID_ALL,
&CDesignerkwearView::OnUpdateAll)
        ON_COMMAND(ID_PRINT_OBL, &CDesignerkwearView::OnPrintObl)
        ON_UPDATE_COMMAND_UI(ID_PRINT_OBL,
&CDesignerkwearView::OnUpdatePrintObl)
    END_MESSAGE_MAP()

    // создание/уничтожение CDesignerkwearView
    CDesignerkwearView::CDesignerkwearView():
        m_Graf(CONSTRUCT), m_Info(EMPTY_P), m_Naprav(EMPTY_N)
            , pvPoint(0,0)
```

```
        , pvtPoint(0,0)// Перемещение
        , pvePoint(0,0)
        , id_curve(0)
        , id_point(-1)
        , id_name(_T(""))
        , zoom(0)
        , m_Prnt(E)
        , PrintRect(0,0,0,0)
    {
}
CDesignerkwearView::~CDesignerkwearView()
{
}
BOOL CDesignerkwearView::PreCreateWindow(CREATESTRUCT& cs)
{
    return CScrollView::PreCreateWindow(cs);
}
// рисование CDesignerkwearView
void CDesignerkwearView::OnDraw(CDC* pDC)
{
    CDesignerkwearDoc* pDoc = GetDocument();
    ASSERT_VALID(pDoc);
    if (!pDoc);
    double mm = pDoc->izmerenia*10.; // Масштаб

    CPen objPen, nPen, sbPen, dPen,dpPen, dopPen, prtPen;
    dPen.CreatePen(PS_SOLID, 5, RGB(250,0,0));
    dpPen.CreatePen(PS_DASH, 5, RGB(120,250,220));
    dopPen.CreatePen(PS_DASH, 1, RGB(0,250,0));
    nPen.CreatePen(PS_DASH, 2, RGB(250,0,250));
```

```
prtPen.CreatePen(PS_DASH, 0, RGB(0,0,200));
pDC->SetMapMode(MM_ANISOTROPIC); // режим
pDC->SetViewportExt(zoomWin.GetView());   // разрешение
pDC->SetWindowExt(zoomWin.GetWin());        // размер
if (m_Graf == CONSTRUCT || m_Graf == ALL)
{
        for (int i = 0; i < pDoc->aprox.size(); i++)
        {
                if (pDoc->aprox[i].st == 222 && m_Prnt == E)
                {
                        CPen *pOldPen = (CPen *)pDC->SelectObject(&dopPen);
                        CPoint starttext, endtext;
                        double x0(pDoc->aprox[i].x0*mm), y0(pDoc->aprox[i].y0*mm), x1(pDoc->aprox[i].x1*mm), y1(pDoc->aprox[i].y1*mm),
                                x2(pDoc->aprox[i].x2*mm), y2(pDoc->aprox[i].y2*mm);
                        CString name (pDoc->aprox[i].name);
                        pDC->MoveTo(x0, y0);
                        for ( double t = 0; t <= 1.001; t+=0.001)
                        {
                        pDC->LineTo((1-t)*(1-t)*x0+2*t*(1-t)*x1+t*t*x2,  (1-t)*(1-t)*y0+2*t*(1-t)*y1+t*t*y2);
                        };
                        pDC->SelectObject(pOldPen);
                        pDC->SetTextColor(RGB(0,0,250));

                        starttext.x = 0.5*0.5*x0+2*0.5*0.5*x1+0.5*0.5*x2;
                        starttext.y = 0.5*0.5*y0+2*0.5*0.5*y1+0.5*0.5*y2;
                        endtext.x = starttext.x;
                        endtext.y = starttext.y;
```

```
                    pDC->DrawText(name, CRect(starttext, endtext),
DT_CENTER | DT_VCENTER | DT_SINGLELINE | DT_NOCLIP);
                }
            };
            for (int i = 0; i < pDoc->obj.size(); i++)
            {
                objPen.CreatePen(PS_SOLID, pDoc->obj[i].tolshina,
RGB(pDoc->obj[i].red,pDoc->obj[i].green,pDoc->obj[i].black));
                if (pDoc->obj[i].st == 1000 && m_Prnt == E)
                {
                    CBrush aBrush(RGB(0,0,0));
                    CBrush *pOldBrush = (CBrush*) pDC-
>SelectObject(&aBrush);
                    CPen *pOldPen = (CPen *)pDC-
>SelectObject(&objPen);
                    CPoint starttext, endtext;
                    double x0(pDoc->obj[i].x0*mm), y0(pDoc-
>obj[i].y0*mm);
                    CString name (pDoc->obj[i].name);
                    if (pDoc->m_Izm == MM)
                    {pDC->Ellipse(CRect(x0,y0,x0+mm-3,y0+mm-3));}
                    else
                    {pDC->Ellipse(CRect(x0,y0,x0+mm/13,y0+mm/13));}
                    starttext.x = x0-5;
                    starttext.y = y0;
                    endtext.x = starttext.x;
                    endtext.y = starttext.y;

                    pDC->SetTextColor(RGB(0,0,250));
```

```
                    pDC->DrawText(name, CRect(starttext, endtext),
DT_CENTER | DT_VCENTER | DT_SINGLELINE | DT_NOCLIP);
                    pDC->SelectObject(pOldPen);
                    pDC->SelectObject(pOldBrush);
                }
                else if (pDoc->obj[i].st == 1001 && m_Prnt == E)
                {
                    CBrush aBrush(RGB(0,0,250));
                    CBrush *pOldBrush = (CBrush*) pDC-
>SelectObject(&aBrush);
                    CPen *pOldPen = (CPen *)pDC-
>SelectObject(&objPen);
                    CPoint starttext, endtext;
                    double x0(pDoc->obj[i].x0*mm), y0(pDoc-
>obj[i].y0*mm);
                    CString name (pDoc->obj[i].name);
                    if (pDoc->m_Izm == MM)
                    {pDC->Rectangle(CRect(x0-mm+5,y0-mm+5,x0+mm-
5,y0+mm-5));}
                    else
                    {pDC->Rectangle(CRect(x0-mm/15,y0-
mm/15,x0+mm/15,y0+mm/15));}
                    starttext.x = x0;
                    starttext.y = y0+15;
                    endtext.x = starttext.x;
                    endtext.y = starttext.y;
                    pDC->SetTextColor(RGB(0,0,250));
                    pDC->DrawText(name, CRect(starttext, endtext),
DT_CENTER | DT_VCENTER | DT_SINGLELINE | DT_NOCLIP);
                    pDC->SelectObject(pOldPen);
```

```
                pDC->SelectObject(pOldBrush);
            }
            else if (pDoc->obj[i].st == 1 && m_Prnt == E)
            {
                CPen *pOldPen = (CPen *)pDC->SelectObject(&objPen);
                CPoint starttext, endtext;
                double x0(pDoc->obj[i].x0*mm), y0(pDoc->obj[i].y0*mm), x1(pDoc->obj[i].x1*mm), y1(pDoc->obj[i].y1*mm);
                CString name (pDoc->obj[i].name);
                pDC->MoveTo(x0,y0);
                pDC->LineTo(x1,y1);
                pDC->SelectObject(pOldPen);
                starttext.x = 0.5*(x0 + x1);
                starttext.y = 0.5*(y0 + y1);
                endtext.x = starttext.x;
                endtext.y = starttext.y;
                pDC->SetTextColor(RGB(0,0,250));
                pDC->DrawText(name, CRect(starttext, endtext), DT_CENTER | DT_VCENTER | DT_SINGLELINE | DT_NOCLIP);
                if (m_Naprav == NAPRAV)
                {
                    CPen *pOldPen = (CPen *)pDC->SelectObject(&nPen);
                    pDC->SetTextColor(RGB(250,0,250));
                    pDC->DrawText(CString("P0(")+name+CString(")"), CRect(CPoint(x0,y0), CPoint(x0,y0)), DT_CENTER | DT_VCENTER | DT_SINGLELINE | DT_NOCLIP),
```

```
                    pDC->DrawText(CString("P1
(")+name+CString(")"), CRect(CPoint(x1,y1), CPoint(x1,y1)), DT_CENTER |
DT_VCENTER | DT_SINGLELINE | DT_NOCLIP),
                    pDC->SelectObject(pOldPen);
                }
            }
            else if (pDoc->obj[i].st == 2 && m_Prnt == E)
            {
                CPen *pOldPen = (CPen *)pDC-
>SelectObject(&objPen);
                CPoint starttext, endtext;
                double x0(pDoc->obj[i].x0*mm), y0(pDoc-
>obj[i].y0*mm), x1(pDoc->obj[i].x1*mm), y1(pDoc->obj[i].y1*mm),
                    x2(pDoc->obj[i].x2*mm), y2(pDoc-
>obj[i].y2*mm);
                CString name (pDoc->obj[i].name);
                pDC->MoveTo(x0, y0);
                for ( double t = 0; t <= 1.001; t+=0.001)
                {
                    pDC->LineTo((1-t)*(1-t)*x0+2*t*(1-
t)*x1+t*t*x2, (1-t)*(1-t)*y0+2*t*(1-t)*y1+t*t*y2);
                };
                pDC->SelectObject(pOldPen);
                pDC->SetTextColor(RGB(0,0,250));
                starttext.x = 0.5*0.5*x0+2*0.5*0.5*x1+0.5*0.5*x2;
                starttext.y = 0.5*0.5*y0+2*0.5*0.5*y1+0.5*0.5*y2;
                endtext.x = starttext.x;
                endtext.y = starttext.y;
                pDC->DrawText(name, CRect(starttext, endtext),
DT_CENTER | DT_VCENTER | DT_SINGLELINE | DT_NOCLIP);
```

```
                    if (m_Naprav == NAPRAV)
                    {
                        CPen *pOldPen = (CPen *)pDC->SelectObject(&nPen);
                        pDC->SetTextColor(RGB(250,0,250));
                        pDC->MoveTo(x0,y0);
                        pDC->LineTo(x1,y1);
                        pDC->LineTo(x2,y2);
                        pDC->DrawText(CString("P0(")+name+CString(")"), CRect(CPoint(x0,y0), CPoint(x0,y0)), DT_CENTER | DT_VCENTER | DT_SINGLELINE | DT_NOCLIP);
                        pDC->DrawText(CString("P1(")+name+CString(")"), CRect(CPoint(x1,y1), CPoint(x1,y1)), DT_CENTER | DT_VCENTER | DT_SINGLELINE | DT_NOCLIP);
                        pDC->DrawText(CString("P2(")+name+CString(")"), CRect(CPoint(x2,y2), CPoint(x2,y2)), DT_CENTER | DT_VCENTER | DT_SINGLELINE | DT_NOCLIP);
                        pDC->SelectObject(pOldPen);
                    }
                }
                else if (pDoc->obj[i].st == 3 && m_Prnt == E)
                {
                    CPen *pOldPen = (CPen *)pDC->SelectObject(&objPen);
                    CPoint starttext, endtext;
                    double x0(pDoc->obj[i].x0*mm), y0(pDoc->obj[i].y0*mm), x1(pDoc->obj[i].x1*mm), y1(pDoc->obj[i].y1*mm),
                        x2(pDoc->obj[i].x2*mm), y2(pDoc->obj[i].y2*mm), x3(pDoc->obj[i].x3*mm), y3(pDoc->obj[i].y3*mm);
                    CString name (pDoc->obj[i].name);
```

```
pDC->MoveTo(x0, y0);
for ( double t = 0; t <= 1.001; t+=0.001)
{
    pDC->LineTo((1-t)*(1-t)*(1-t)*x0+3*t*(1-t)*(1-t)*x1+3*t*t*(1-t)*x2+t*t*t*x3, (1-t)*(1-t)*(1-t)*y0+3*t*(1-t)*(1-t)*y1+3*t*t*(1-t)*y2+t*t*t*y3);
};
pDC->SelectObject(pOldPen);
pDC->SetTextColor(RGB(0,0,250));

starttext.x = 0.5*0.5*0.5*x0+3*0.5*0.5*0.5*x1+3*0.5*0.5*0.5*x2+0.5*0.5*0.5*x3;
starttext.y = 0.5*0.5*0.5*y0+3*0.5*0.5*0.5*y1+3*0.5*0.5*0.5*y2+0.5*0.5*0.5*y3;
endtext.x = starttext.x;
endtext.y = starttext.y;
pDC->DrawText(name, CRect(starttext, endtext), DT_CENTER | DT_VCENTER | DT_SINGLELINE | DT_NOCLIP);
if (m_Naprav == NAPRAV)
{
    CPen *pOldPen = (CPen *)pDC->SelectObject(&nPen);
    pDC->SetTextColor(RGB(250,0,250));
    pDC->MoveTo(x0,y0);
    pDC->LineTo(x1,y1);
    pDC->LineTo(x2,y2);
    pDC->LineTo(x3,y3);
    pDC->DrawText(CString("P0(")+name+CString(")"), CRect(CPoint(x0,y0), CPoint(x0,y0)), DT_CENTER | DT_VCENTER | DT_SINGLELINE | DT_NOCLIP);
```

```
                                pDC->DrawText(CString("P1
(")+name+CString(")"), CRect(CPoint(x1,y1), CPoint(x1,y1)), DT_CENTER |
DT_VCENTER | DT_SINGLELINE | DT_NOCLIP);
                                pDC->DrawText(CString("P2
(")+name+CString(")"), CRect(CPoint(x2,y2), CPoint(x2,y2)), DT_CENTER |
DT_VCENTER | DT_SINGLELINE | DT_NOCLIP);
                                pDC->DrawText(CString("P3
(")+name+CString(")"), CRect(CPoint(x3,y3), CPoint(x3,y3)), DT_CENTER |
DT_VCENTER | DT_SINGLELINE | DT_NOCLIP);
                                pDC->SelectObject(pOldPen);
                        }
                }
                objPen.DeleteObject();
        };
        CPen *pOldPen = (CPen *)pDC->SelectObject(&dPen);
        for (int i = 0; i < pDoc->dobj.size(); i++)
        {
                pDC->SetTextColor(RGB(250,0,0));
                if (pDoc->dobj[i].st == 51 && m_Prnt == E || pDoc-
>dobj[i].st == 51 && m_Prnt == B)
                {
                        for (int j = 0; j < pDoc->dobj[i].side.size(); j++)
                        {
                                double          x0(pDoc->dobj[i].side[j].x0*mm),
y0(pDoc->dobj[i].side[j].y0*mm);
                                if (pDoc->dobj[i].side[j].st == 1)
                                {
                                        double    x1(pDoc->dobj[i].side[j].x1*mm),
y1(pDoc->dobj[i].side[j].y1*mm);
                                        pDC->MoveTo(x0,y0);
```

```
                pDC->LineTo(x1,y1);
            }
            else if (pDoc->dobj[i].side[j].st == 2)
            {
                double    x1(pDoc->dobj[i].side[j].x1*mm),
y1(pDoc->dobj[i].side[j].y1*mm),
                  x2(pDoc->dobj[i].side[j].x2*mm),
y2(pDoc->dobj[i].side[j].y2*mm);
                pDC->MoveTo(x0,y0);
                for ( double t = 0; t <= 1.001; t+=0.001)
                {
                    pDC->LineTo((1-t)*(1-t)*x0+2*t*(1-
t)*x1+t*t*x2, (1-t)*(1-t)*y0+2*t*(1-t)*y1+t*t*y2);
                };
            }
            else if (pDoc->dobj[i].side[j].st == 3)
            {
                double    x1(pDoc->dobj[i].side[j].x1*mm),
y1(pDoc->dobj[i].side[j].y1*mm),
                  x2(pDoc->dobj[i].side[j].x2*mm),
y2(pDoc->dobj[i].side[j].y2*mm),    x3(pDoc->dobj[i].side[j].x3*mm),    y3(pDoc-
>dobj[i].side[j].y3*mm);
                pDC->MoveTo(x0,y0);
                for ( double t = 0; t <= 1.001; t+=0.001)
                {
                    pDC->LineTo((1-t)*(1-t)*(1-
t)*x0+3*t*(1-t)*(1-t)*x1+3*t*t*(1-t)*x2+t*t*t*x3,    (1-t)*(1-t)*(1-t)*y0+3*t*(1-t)*(1-
t)*y1+3*t*t*(1-t)*y2+t*t*t*y3);
                };
            };
```

```
if (j == 0 && m_Prnt == E)
{
        pDC->SetTextColor(RGB(250,0,0));
        pDC->DrawText(pDoc->dobj[i].dname,
CRect(CPoint(x0,y0),CPoint(x0,y0)), DT_CENTER | DT_VCENTER |
DT_SINGLELINE | DT_NOCLIP);
}
else if (j == 0 && m_Prnt == B)
{
        pDC->SetTextColor(RGB(250,0,0));

        for (int y = 0; y < pDoc->com.size(); y++)
        {
                if (pDoc->com[y].st == 200 &&
pDoc->com[y].name == pDoc->dobj[i].dname)
                {

                        pDC-
>DrawText(CString("Имя: ")+pDoc->com[y].name, CRect(CPoint(pDoc-
>com[y].x0,pDoc->com[y].y0),CPoint(pDoc->com[y].x0,pDoc->com[y].y0)),
DT_CENTER | DT_VCENTER | DT_SINGLELINE | DT_NOCLIP);
                        if (!pDoc->raz.empty())
                        {
                                pDC-
>DrawText(CString("Размер: ")+pDoc->raz.back().title, CRect(CPoint(pDoc-
>com[y].x0,pDoc->com[y].y0+5),CPoint(pDoc->com[y].x0,pDoc->com[y].y0+5)),
DT_CENTER | DT_VCENTER | DT_SINGLELINE | DT_NOCLIP);
                        };
                };
```

```
                    };
                };
            };
        }
        else if (pDoc->dobj[i].st == 53 && m_Prnt == E || pDoc->dobj[i].st == 53 && m_Prnt == B)
        {
            CPen *pOldPen = (CPen *)pDC->SelectObject(&dpPen);
            for (int j = 0; j < pDoc->dobj[i].d_object.size(); j++)
            {
                double x0(pDoc->dobj[i].d_object[j].x0*mm), y0(pDoc->dobj[i].d_object[j].y0*mm);
                if (pDoc->dobj[i].d_object[j].st == 1)
                {
                    double x1(pDoc->dobj[i].d_object[j].x1*mm), y1(pDoc->dobj[i].d_object[j].y1*mm);
                    pDC->MoveTo(x0,y0);
                    pDC->LineTo(x1,y1);
                }
                else if (pDoc->dobj[i].d_object[j].st == 2)
                {
                    double x1(pDoc->dobj[i].d_object[j].x1*mm), y1(pDoc->dobj[i].d_object[j].y1*mm),
                        x2(pDoc->dobj[i].d_object[j].x2*mm), y2(pDoc->dobj[i].d_object[j].y2*mm);
                    pDC->MoveTo(x0,y0);
                    for ( double t = 0; t <= 1.001; t+=0.001)
                    {
```

```
                pDC->LineTo((1-t)*(1-t)*x0+2*t*(1-t)*x1+t*t*x2, (1-t)*(1-t)*y0+2*t*(1-t)*y1+t*t*y2);
            };
        }
        else if (pDoc->dobj[i].d_object[j].st == 3)
        {
            double x1(pDoc->dobj[i].d_object[j].x1*mm), y1(pDoc->dobj[i].d_object[j].y1*mm),
                x2(pDoc->dobj[i].d_object[j].x2*mm), y2(pDoc->dobj[i].d_object[j].y2*mm),
                x3(pDoc->dobj[i].d_object[j].x3*mm), y3(pDoc->dobj[i].d_object[j].y3*mm);
            pDC->MoveTo(x0,y0);
            for ( double t = 0; t <= 1.001; t+=0.001)
            {
                pDC->LineTo((1-t)*(1-t)*(1-t)*x0+3*t*(1-t)*(1-t)*x1+3*t*t*(1-t)*x2+t*t*t*x3,
                (1-t)*(1-t)*(1-t)*y0+3*t*(1-t)*(1-t)*y1+3*t*t*(1-t)*y2+t*t*t*y3);
            };
        };
        if (j == 0 && m_Prnt == E)
        {
            pDC->SetTextColor(RGB(0,0,0));
            pDC->DrawText(pDoc->dobj[i].dname, CRect(CPoint(x0,y0),CPoint(x0,y0)),
            DT_CENTER | DT_VCENTER | DT_SINGLELINE | DT_NOCLIP);
        };
    };
    pDC->SelectObject(pOldPen);
}
```

```
                else if (pDoc->dobj[i].st == 50 && m_Prnt == E || pDoc->dobj[i].st == 50 && m_Prnt == B)
                {
                    for (int j = 0; j < pDoc->dobj[i].d_object.size(); j++)
                    {
                        double x0(pDoc->dobj[i].d_object[j].x0*mm), y0(pDoc->dobj[i].d_object[j].y0*mm);
                        if (pDoc->dobj[i].d_object[j].st == 1)
                        {
                            double x1(pDoc->dobj[i].d_object[j].x1*mm), y1(pDoc->dobj[i].d_object[j].y1*mm);
                            pDC->MoveTo(x0,y0);
                            pDC->LineTo(x1,y1);
                        }
                        else if (pDoc->dobj[i].d_object[j].st == 2)
                        {
                            double x1(pDoc->dobj[i].d_object[j].x1*mm), y1(pDoc->dobj[i].d_object[j].y1*mm),
                                x2(pDoc->dobj[i].d_object[j].x2*mm), y2(pDoc->dobj[i].d_object[j].y2*mm);
                            pDC->MoveTo(x0,y0);
                            for ( double t = 0; t <= 1.001; t+=0.001)
                            {
                                pDC->LineTo((1-t)*(1-t)*x0+2*t*(1-t)*x1+t*t*x2, (1-t)*(1-t)*y0+2*t*(1-t)*y1+t*t*y2);
                            };
                        }
                        else if (pDoc->dobj[i].d_object[j].st == 3)
                        {
```

```
                                        double x1(pDoc->dobj[i].d_object[j].x1*mm), y1(pDoc->dobj[i].d_object[j].y1*mm),
                                            x2(pDoc->dobj[i].d_object[j].x2*mm), y2(pDoc->dobj[i].d_object[j].y2*mm),
                                            x3(pDoc->dobj[i].d_object[j].x3*mm), y3(pDoc->dobj[i].d_object[j].y3*mm);
                                        pDC->MoveTo(x0,y0);
                                        for ( double t = 0; t <= 1.001; t+=0.001)
                                        {
                                            pDC->LineTo((1-t)*(1-t)*(1-t)*x0+3*t*(1-t)*(1-t)*x1+3*t*t*(1-t)*x2+t*t*t*x3, (1-t)*(1-t)*(1-t)*y0+3*t*(1-t)*(1-t)*y1+3*t*t*(1-t)*y2+t*t*t*y3);
                                        };
                                    };

                                    if (j == 0 && m_Prnt == E)
                                    {
                                        pDC->SetTextColor(RGB(250,0,0));
                                        pDC->DrawText(pDoc->dobj[i].dname, CRect(CPoint(x0,y0),CPoint(x0,y0)), DT_CENTER | DT_VCENTER | DT_SINGLELINE | DT_NOCLIP);
                                    };
                                };
                            }
                            else if (pDoc->dobj[i].st == 52 && m_Prnt == E || pDoc->dobj[i].st == 52 && m_Prnt == B)
                            {
                                for (int j = 0; j < pDoc->dobj[i].comb.size(); j++)
                                {
                                    double x0(pDoc->dobj[i].comb[j].x0*mm), y0(pDoc->dobj[i].comb[j].y0*mm);
```

```
if (pDoc->dobj[i].comb[j].st == 1)
{
	double  x1(pDoc->dobj[i].comb[j].x1*mm), y1(pDoc->dobj[i].comb[j].y1*mm);
	pDC->MoveTo(x0,y0);
	pDC->LineTo(x1,y1);
}
else if (pDoc->dobj[i].comb[j].st == 2)
{
	double  x1(pDoc->dobj[i].comb[j].x1*mm), y1(pDoc->dobj[i].comb[j].y1*mm),
	  x2(pDoc->dobj[i].comb[j].x2*mm), y2(pDoc->dobj[i].comb[j].y2*mm);
	pDC->MoveTo(x0,y0);
	for ( double t = 0; t <= 1.001; t+=0.001)
	{
		pDC->LineTo((1-t)*(1-t)*x0+2*t*(1-t)*x1+t*t*x2, (1-t)*(1-t)*y0+2*t*(1-t)*y1+t*t*y2);
	};
}
else if (pDoc->dobj[i].comb[j].st == 3)
{
	double  x1(pDoc->dobj[i].comb[j].x1*mm), y1(pDoc->dobj[i].comb[j].y1*mm),
	  x2(pDoc->dobj[i].comb[j].x2*mm), y2(pDoc->dobj[i].comb[j].y2*mm),  x3(pDoc->dobj[i].comb[j].x3*mm),  y3(pDoc->dobj[i].comb[j].y3*mm);
	pDC->MoveTo(x0,y0);
	for ( double t = 0; t <= 1.001; t+=0.001)
	{
```

```
                                        pDC->LineTo((1-t)*(1-t)*(1-
t)*x0+3*t*(1-t)*(1-t)*x1+3*t*t*(1-t)*x2+t*t*t*x3,    (1-t)*(1-t)*(1-t)*y0+3*t*(1-t)*(1-
t)*y1+3*t*t*(1-t)*y2+t*t*t*y3);
                                    };
                                };

                                if (j == 0 && m_Prnt == E)
                                {
                                    pDC->SetTextColor(RGB(250,0,0));
                                    pDC->DrawText(pDoc->dobj[i].dname,
CRect(CPoint(x0,y0),CPoint(x0,y0)),    DT_CENTER    |    DT_VCENTER    |
DT_SINGLELINE | DT_NOCLIP);
                                };
                            };
                        }
                    };
                    pDC->SelectObject(pOldPen);
                };
                if (m_Graf == SBAVKI || m_Graf == ALL)
                {
                    sbPen.CreatePen(PS_SOLID, 5, RGB(0,250,0));
                    // Доп р
                    for (int i = 0; i < pDoc->side_d.size(); i++)
                    {
                        CPen *pOldPen = (CPen *)pDC->SelectObject(&dpPen);
                        for (int j = 0; j < pDoc->side_d[i].sbavki_dop1.size(); j++)
                        {
                            CRect    aRect(pDoc->side_d[i].sbavki_dop1[j].axl*mm,
pDoc->side_d[i].sbavki_dop1[j].ay*mm,
```

```
                                pDoc->side_d[i].sbavki_dop1[j].axr*mm,  pDoc-
>side_d[i].sbavki_dop1[j].ay*mm - pDoc->side_d[i].sbavki_dop1[j].bp*mm);
                                pDC->MoveTo(pDoc-
>side_d[i].sbavki_dop1[j].axl*mm, pDoc->side_d[i].sbavki_dop1[j].ay*mm);
                                pDC->Rectangle(aRect);
                                if (m_Info == PETLI)
                                {
                                        pDC->MoveTo(pDoc-
>side_d[i].sbavki_dop1[j].axl*mm +
                                                (pDoc->side_d[i].sbavki_dop1[j].axr*mm -
pDoc->side_d[i].sbavki_dop1[j].axl*mm) / 2,
                                                pDoc->side_d[i].sbavki_dop1[j].ay*mm);
                                                CString ryad, left, right, nadpis, dr;
                                        dr.Format(_T("%.0f"),                   pDoc-
>side_d[i].sbavki_dop1[j].dop_r);
                                        ryad.Format(_T("%i"),pDoc-
>side_d[i].sbavki_dop1[j].r);
                                        left.Format(_T("%i"),abs(pDoc-
>side_d[i].sbavki_dop1[j].stol_l));
                                        right.Format(_T("%i"),abs(pDoc-
>side_d[i].sbavki_dop1[j].stol_r));
                                        nadpis = CString("Ряд ") + ryad +
                                                CString(" + (") + dr + CString(")") +
                                                CString(":  <-- ") + left + CString(" | ") +
right + CString(" -->");
                                        pDC->SetTextColor(RGB(0,0,0));
                                                pDC->DrawText(nadpis,
CRect(CPoint(pDoc->side_d[i].sbavki_dop1[j].axl*mm +
```

```
                                        (pDoc->side_d[i].sbavki_dop1[j].axr*mm -
pDoc->side_d[i].sbavki_dop1[j].axl*mm) / 2,pDoc->side_d[i].sbavki_dop1[j].ay*mm -
2),
                                                CPoint(pDoc-
>side_d[i].sbavki_dop1[j].axl*mm + (pDoc->side_d[i].sbavki_dop1[j].axr*mm -
                                                pDoc-
>side_d[i].sbavki_dop1[j].axl*mm) / 2,pDoc->side_d[i].sbavki_dop1[j].ay*mm - 2)),
                                                DT_CENTER | DT_VCENTER |
DT_SINGLELINE | DT_NOCLIP);
                                        };
                                };
                        };
                        // КРОМКА
                        for (int i = 0; i < pDoc->approx_side.size(); i++)
                        {
                                CPen *pOldPen = (CPen *)pDC->SelectObject(&sbPen);
                                for (int j = 0; j < pDoc->approx_side[i].sbavki.size(); j++)
                                {
                                        if (j == 0 )
                                        {
                                                pDC->MoveTo(pDoc-
>approx_side[i].sbavki[j].ax*mm,pDoc->approx_side[i].sbavki[j].ay*mm);
                                                pDC->LineTo(pDoc-
>approx_side[i].sbavki[j].ax*mm, pDoc->approx_side[i].sbavki[j+1].ay*mm);
                                        if (m_Info == PETLI)
                                                {
                                                pDC->MoveTo(pDoc-
>approx_side[i].sbavki[j].ax*mm,pDoc->approx_side[i].sbavki[j].ay*mm);

                                                CString ryad, sbavka, napr, nadpis;
```

```
                                        ryad.Format(_T("%i"),pDoc-
>approx_side[i].sbavki[j].r);
                                        sbavka.Format(_T("%i"),abs(pDoc-
>approx_side[i].sbavki[j].sbav));
                                        if (pDoc->approx_side[i].sbavki[j].pol == -
1)
                                        {
                                                napr = ", Прибавка: ";
                                        }
                                        else if (pDoc->approx_side[i].sbavki[j].pol
== 1)
                                        {
                                                napr = ", Сбавка: ";
                                        };

                                        if (pDoc->approx_side[i].sbavki[j].sbav ==
0)
                                        {
                                                napr = ", Прибавка / Сбавка: ";
                                        }

                                        nadpis = CString("Ряд: ") + ryad + napr +
sbavka;
                                        pDC->SetTextColor(RGB(0,250,0));
                                        pDC->DrawText(nadpis,
CRect(CPoint(pDoc->approx_side[i].sbavki[j].ax*mm - 2 - nadpis.GetLength(),pDoc-
>approx_side[i].sbavki[j].ay*mm
                                                - pDoc-
>approx_side[i].sbavki[j].bp*pDoc->approx_side[i].sbavki[j].rapapr*mm/3),
```

```
                                        CPoint(pDoc-
>approx_side[i].sbavki[j].ax*mm - 4 -
                                        nadpis.GetLength(),pDoc-
>approx_side[i].sbavki[j].ay*mm -
                                        pDoc-
>approx_side[i].sbavki[j].bp*pDoc->approx_side[i].sbavki[j].rapapr*mm/3)),
                                        DT_CENTER | DT_VCENTER |
DT_SINGLELINE | DT_NOCLIP);

                                    pDC->MoveTo(pDoc-
>approx_side[i].sbavki[j].ax*mm,  (pDoc->approx_side[i].sbavki[j].ay-pDoc-
>approx_side[i].sbavki[j].bp*pDoc->approx_side[i].sbavki[j].rapapr)*mm);
                                }
                                }
                                else
                                {
                                    pDC->LineTo(pDoc-
>approx_side[i].sbavki[j].ax*mm, pDoc->approx_side[i].sbavki[j].ay*mm);
                                    if ( j < pDoc->approx_side[i].sbavki.size()-
1)
                                    {pDC->LineTo(pDoc-
>approx_side[i].sbavki[j].ax*mm, pDoc->approx_side[i].sbavki[j+1].ay*mm);};


                                if (m_Info == PETLI)
                                {
                                    CString ryad, sbavka, napr, nadpis;
                                    ryad.Format(_T("%i"),pDoc-
>approx_side[i].sbavki[j].r);
```

```
sbavka.Format(_T("%i"),abs(pDoc->approx_side[i].sbavki[j].sbav));
if (pDoc->approx_side[i].sbavki[j].pol == -1)
{
napr = ", Прибавка: ";
}
else if (pDoc->approx_side[i].sbavki[j].pol == 1)
{
napr = ", Сбавка: ";
};

if (pDoc->approx_side[i].sbavki[j].sbav == 0)
{
napr = ", Прибавка / Сбавка: ";
}
nadpis = CString("Ряд: ") + ryad + napr + sbavka;
pDC->SetTextColor(RGB(0,250,0));
pDC->DrawText(nadpis, CRect(CPoint(pDoc->approx_side[i].sbavki[j].ax*mm - 2 - nadpis.GetLength(),pDoc->approx_side[i].sbavki[j].ay*mm-pDoc->approx_side[i].sbavki[j].bp*pDoc->approx_side[i].sbavki[j].rapapr*mm/3), CPoint(pDoc->approx_side[i].sbavki[j].ax*mm - 4 - nadpis.GetLength(),pDoc->approx_side[i].sbavki[j].ay*mm-pDoc->approx_side[i].sbavki[j].bp*pDoc->approx_side[i].sbavki[j].rapapr*mm/3)),DT_CENTER | DT_VCENTER | DT_SINGLELINE | DT_NOCLIP);
```

```
                                        pDC->MoveTo(pDoc-
>approx_side[i].sbavki[j].ax*mm,                (pDoc->approx_side[i].sbavki[j].ay-pDoc-
>approx_side[i].sbavki[j].bp*pDoc->approx_side[i].sbavki[j].rapapr)*mm);
                                    }
                                    };
                                };
                        };
                        // ФОРМА
                        for (int i = 0; i < pDoc->approximation_l.size(); i++)
                        {
                                CPen *pOldPen = (CPen *)pDC->SelectObject(&sbPen);
                                for (int j = 0; j < pDoc->approximation_l[i].sbavki.size(); j++)
                                {
                                        if (j == 0)
                                        {
                                                pDC->MoveTo(pDoc-
>approximation_l[i].sbavki[j].ax*mm,pDoc->approximation_l[i].sbavki[j].ay*mm);
                                        pDC->LineTo(pDoc-
>approximation_l[i].sbavki[j].ax*mm, pDoc->approximation_l[i].sbavki[j+1].ay*mm);

                                        if (m_Info == PETLI)
                                        {
                                                pDC->MoveTo(pDoc-
>approximation_l[i].sbavki[j].ax*mm,pDoc->approximation_l[i].sbavki[j].ay*mm);

                                                CString ryad, sbavka, napr, nadpis;
                                                ryad.Format(_T("%i"),pDoc-
>approximation_l[i].sbavki[j].r);
                                                sbavka.Format(_T("%i"),abs(pDoc-
>approximation_l[i].sbavki[j].sbav));
```

```
                    if (pDoc->approximation_l[i].sbavki[j].pol == -1 && pDoc->approximation_l[i].sbavki[j].t != 777)
                    {
                        napr = ", Прибавка: ";
                    }
                    else if (pDoc->approximation_l[i].sbavki[j].pol == 1 && pDoc->approximation_l[i].sbavki[j].t != 777)
                    {
                        napr = ", Сбавка: ";
                    };
                    if (pDoc->approximation_l[i].sbavki[j].sbav == 0 && pDoc->approximation_l[i].sbavki[j].t != 777)
                    {
                        napr = ", Прибавка / Сбавка: ";
                    }
                    if (pDoc->approximation_l[i].sbavki[j].pol == 1 && pDoc->approximation_l[i].sbavki[j].t == 777)
                    {
                        napr = ", Прибавка: ";
                    }
                    else if (pDoc->approximation_l[i].sbavki[j].pol == -1 && pDoc->approximation_l[i].sbavki[j].t == 777)
                    {
                        napr = ", Сбавка: ";
                    };
                    nadpis = CString("Ряд: ") + ryad + napr + sbavka;
```

```
                    pDC->SetTextColor(RGB(0,250,0));
                    pDC->DrawText(nadpis,
CRect(CPoint(pDoc->approximation_l[i].sbavki[j].ax*mm - 2 -
nadpis.GetLength(),pDoc->approximation_l[i].sbavki[j].ay*mm - pDoc-
>approximation_l[i].sbavki[j].bp*pDoc->approximation_l[i].sbavki[j].rapapr*mm/3),
CPoint(pDoc->approximation_l[i].sbavki[j].ax*mm - 4 - nadpis.GetLength(),pDoc-
>approximation_l[i].sbavki[j].ay*mm - pDoc->approximation_l[i].sbavki[j].bp*pDoc-
>approximation_l[i].sbavki[j].rapapr*mm/3)),DT_CENTER | DT_VCENTER |
DT_SINGLELINE | DT_NOCLIP);
                    pDC->MoveTo(pDoc-
>approximation_l[i].sbavki[j].ax*mm, (pDoc->approximation_l[i].sbavki[j].ay-pDoc-
>approximation_l[i].sbavki[j].bp)*mm);
                }

                }
                else
                {
                    pDC->LineTo(pDoc-
>approximation_l[i].sbavki[j].ax*mm, pDoc->approximation_l[i].sbavki[j].ay*mm);
                    if ( j < pDoc-
>approximation_l[i].sbavki.size()-1)
                    {pDC->LineTo(pDoc-
>approximation_l[i].sbavki[j].ax*mm, pDoc-
>approximation_l[i].sbavki[j+1].ay*mm);};

                if (m_Info == PETLI)
                {
                    CString ryad, sbavka, napr, nadpis;
                    ryad.Format(_T("%i"),pDoc-
>approximation_l[i].sbavki[j].r);
```

```
sbavka.Format(_T("%i"),abs(pDoc->approximation_l[i].sbavki[j].sbav));
if (pDoc->approximation_l[i].sbavki[j].pol == -1 && pDoc->approximation_l[i].sbavki[j].t != 777)
{
napr = ", Прибавка: ";
}
else if (pDoc->approximation_l[i].sbavki[j].pol == 1 && pDoc->approximation_l[i].sbavki[j].t != 777)
{
napr = ", Сбавка: ";
};

if (pDoc->approximation_l[i].sbavki[j].sbav == 0 && pDoc->approximation_l[i].sbavki[j].t != 777)
{
napr = ", Прибавка / Сбавка: ";
}

if (pDoc->approximation_l[i].sbavki[j].pol == 1 && pDoc->approximation_l[i].sbavki[j].t == 777)
{
napr = ", Прибавка: ";
}
else if (pDoc->approximation_l[i].sbavki[j].pol == -1 && pDoc->approximation_l[i].sbavki[j].t == 777)
{
napr = ", Сбавка: ";
```

```
                                        };

                                        nadpis = CString("Ряд: ") + ryad + napr + sbavka;
                                        pDC->SetTextColor(RGB(0,250,0));
                                        pDC->DrawText(nadpis, CRect(CPoint(pDoc->approximation_l[i].sbavki[j].ax*mm - 2 - nadpis.GetLength(),pDoc->approximation_l[i].sbavki[j].ay*mm-pDoc->approximation_l[i].sbavki[j].bp*pDoc->approximation_l[i].sbavki[j].rapapr*mm/3), CPoint(pDoc->approximation_l[i].sbavki[j].ax*mm - 4 - nadpis.GetLength(),pDoc->approximation_l[i].sbavki[j].ay*mm-pDoc->approximation_l[i].sbavki[j].bp*pDoc->approximation_l[i].sbavki[j].rapapr*mm/3)),DT_CENTER | DT_VCENTER | DT_SINGLELINE | DT_NOCLIP);
                                        pDC->MoveTo(pDoc->approximation_l[i].sbavki[j].ax*mm, (pDoc->approximation_l[i].sbavki[j].ay-pDoc->approximation_l[i].sbavki[j].bp*pDoc->approximation_l[i].sbavki[j].rapapr)*mm);
                                    }
                                    };
                                };
                            };
                            for (int i = 0; i < pDoc->approximation_r.size(); i++)
                            {
                                CPen *pOldPen = (CPen *)pDC->SelectObject(&sbPen);
                                for (int j = 0; j < pDoc->approximation_r[i].sbavki.size(); j++)
                                {
                                    if (j == 0)
                                    {
                                        pDC->MoveTo(pDoc->approximation_r[i].sbavki[j].ax*mm,pDoc->approximation_r[i].sbavki[j].ay*mm);
```

```
                                pDC->LineTo(pDoc->approximation_r[i].sbavki[j].ax*mm, pDoc->approximation_r[i].sbavki[j+1].ay*mm);
                            if (m_Info == PETLI)
                            {
                                pDC->MoveTo(pDoc->approximation_r[i].sbavki[j].ax*mm,pDoc->approximation_r[i].sbavki[j].ay*mm);
                                CString ryad, sbavka, napr, nadpis;
                                ryad.Format(_T("%i"),pDoc->approximation_r[i].sbavki[j].r);
                                sbavka.Format(_T("%i"),abs(pDoc->approximation_r[i].sbavki[j].sbav));
                                if (pDoc->approximation_r[i].sbavki[j].pol == -1 && pDoc->approximation_r[i].sbavki[j].t != 777)
                                {
                                    napr = ", Сбавка: ";
                                }
                                else if (pDoc->approximation_r[i].sbavki[j].pol == 1 && pDoc->approximation_r[i].sbavki[j].t != 777)
                                {
                                    napr = ", Прибавка: ";
                                };
                                if (pDoc->approximation_r[i].sbavki[j].sbav == 0 && pDoc->approximation_r[i].sbavki[j].t != 777)
                                {
                                    napr = ", Прибавка / Сбавка: ";
                                }
                                if (pDoc->approximation_r[i].sbavki[j].pol == 1 && pDoc->approximation_r[i].sbavki[j].t == 777)
```

```
{
napr = ", Сбавка: ";
}
else if (pDoc->approximation_r[i].sbavki[j].pol == -1 && pDoc->approximation_r[i].sbavki[j].t == 777)
{
napr = ", Прибавка: ";
};

nadpis = CString("Ряд: ") + ryad + napr + sbavka;
pDC->SetTextColor(RGB(0,250,0));
pDC->DrawText(nadpis, CRect(CPoint(pDoc->approximation_r[i].sbavki[j].ax*mm - 2 - nadpis.GetLength(),pDoc->approximation_r[i].sbavki[j].ay*mm - pDoc->approximation_r[i].sbavki[j].bp*pDoc->approximation_r[i].sbavki[j].rapapr*mm/3), CPoint(pDoc->approximation_r[i].sbavki[j].ax*mm - 4 - nadpis.GetLength(),pDoc->approximation_r[i].sbavki[j].ay*mm - pDoc->approximation_r[i].sbavki[j].bp*pDoc->approximation_r[i].sbavki[j].rapapr*mm/3)),DT_CENTER | DT_VCENTER | DT_SINGLELINE | DT_NOCLIP);
pDC->MoveTo(pDoc->approximation_r[i].sbavki[j].ax*mm, (pDoc->approximation_r[i].sbavki[j].ay-pDoc->approximation_r[i].sbavki[j].bp)*mm);
}
}
else
{
pDC->LineTo(pDoc->approximation_r[i].sbavki[j].ax*mm, pDoc->approximation_r[i].sbavki[j].ay*mm);
```

```
                    if ( j < pDoc->approximation_r[i].sbavki.size()-1)
                    {pDC->LineTo(pDoc->approximation_r[i].sbavki[j].ax*mm, pDoc->approximation_r[i].sbavki[j+1].ay*mm);};
                if (m_Info == PETLI)
                {
                    CString ryad, sbavka, napr, nadpis;
                    ryad.Format(_T("%i"),pDoc->approximation_r[i].sbavki[j].r);
                    sbavka.Format(_T("%i"),abs(pDoc->approximation_r[i].sbavki[j].sbav));
                    if (pDoc->approximation_r[i].sbavki[j].pol == -1 && pDoc->approximation_r[i].sbavki[j].t != 777)
                    {
                        napr = ", Сбавка: ";
                    }
                    else if (pDoc->approximation_r[i].sbavki[j].pol == 1 && pDoc->approximation_r[i].sbavki[j].t != 777)
                    {
                        napr = ", Прибавка: ";
                    };
                    if (pDoc->approximation_r[i].sbavki[j].sbav == 0 && pDoc->approximation_r[i].sbavki[j].t != 777)
                    {
                        napr = ", Прибавка / Сбавка: ";
                    }
```

```
                                    if (pDoc->approximation_r[i].sbavki[j].pol
== 1 && pDoc->approximation_r[i].sbavki[j].t == 777)
                                    {
                                        napr = ", Сбавка: ";
                                    }
                                    else if (pDoc-
>approximation_r[i].sbavki[j].pol == -1 && pDoc->approximation_r[i].sbavki[j].t ==
777)
                                    {
                                        napr = ", Прибавка: ";
                                    };
                                    nadpis = CString("Ряд: ") + ryad + napr +
sbavka;
                                    pDC->SetTextColor(RGB(0,250,0));
                                    pDC->DrawText(nadpis,
CRect(CPoint(pDoc->approximation_r[i].sbavki[j].ax*mm - 2 -
nadpis.GetLength(),pDoc->approximation_r[i].sbavki[j].ay*mm-pDoc-
>approximation_r[i].sbavki[j].bp*pDoc->approximation_r[i].sbavki[j].rapapr*mm/3),
CPoint(pDoc->approximation_r[i].sbavki[j].ax*mm - 4 - nadpis.GetLength(),pDoc-
>approximation_r[i].sbavki[j].ay*mm-pDoc->approximation_r[i].sbavki[j].bp*pDoc-
>approximation_r[i].sbavki[j].rapapr*mm/3)),DT_CENTER | DT_VCENTER |
DT_SINGLELINE | DT_NOCLIP);
                                    pDC->MoveTo(pDoc-
>approximation_r[i].sbavki[j].ax*mm, (pDoc->approximation_r[i].sbavki[j].ay-pDoc-
>approximation_r[i].sbavki[j].bp*pDoc->approximation_r[i].sbavki[j].rapapr)*mm);
                                }
                                };
                        };
                };
                for (int i = 0; i < pDoc->approx_cs.size(); i++)
```

```
{
        CPen *pOldPen = (CPen *)pDC->SelectObject(&sbPen);
            if (pDoc->approx_cs[i].name_left.IsEmpty() && pDoc->approx_cs[i].name_right.IsEmpty())
{
        for (int j = 0; j < pDoc->approx_cs[i].sbavki_left.size(); j++)
        {
                if (j == 0)
                {
                        pDC->MoveTo(pDoc->approx_cs[i].sbavki_left[j].ax*mm,pDoc->approx_cs[i].sbavki_left[j].ay*mm);
                        pDC->LineTo(pDoc->approx_cs[i].sbavki_left[j].ax*mm, pDoc->approx_cs[i].sbavki_left[j+1].ay*mm);
                if (m_Info == PETLI)
                {
                        pDC->MoveTo(pDoc->approx_cs[i].sbavki_left[j].ax*mm,pDoc->approx_cs[i].sbavki_left[j].ay*mm);

                        CString ryad, sbavka, napr, nadpis;
                        ryad.Format(_T("%i"),pDoc->approx_cs[i].sbavki_left[j].r);
                        sbavka.Format(_T("%i"),pDoc->approx_cs[i].sbavki_left[j].sbav);
                        if      (pDoc->approx_cs[i].sbavki_left[j].pol == -1)
                        {
                                napr = ", Прибавка: ";
                        }
                else  if  (pDoc->approx_cs[i].sbavki_left[j].pol  == 1)
```

```
{
napr = ", Сбавка: ";
};
if (pDoc->approx_cs[i].sbavki_left[j].sbav == 0)
{
napr = ", Прибавка / Сбавка: ";
}
nadpis = CString("Ряд: ") + ryad + napr + sbavka;
pDC->SetTextColor(RGB(0,250,0));
pDC->DrawText(nadpis, CRect(CPoint(pDoc->approx_cs[i].sbavki_left[j].ax*mm - 2 - nadpis.GetLength(),pDoc->approx_cs[i].sbavki_left[j].ay*mm - pDoc->approx_cs[i].sbavki_left[j].bp*pDoc->approx_cs[i].sbavki_left[j].rapapr*mm/3), CPoint(pDoc->approx_cs[i].sbavki_left[j].ax*mm - 4 - nadpis.GetLength(),pDoc->approx_cs[i].sbavki_left[j].ay*mm - pDoc->approx_cs[i].sbavki_left[j].bp*pDoc->approx_cs[i].sbavki_left[j].rapapr*mm/3)),DT_CENTER | DT_VCENTER | DT_SINGLELINE | DT_NOCLIP);

pDC->MoveTo(pDoc->approx_cs[i].sbavki_left[j].ax*mm, (pDoc->approx_cs[i].sbavki_left[j].ay-pDoc->approx_cs[i].sbavki_left[j].bp)*mm);
}
}
else
{
pDC->LineTo(pDoc->approx_cs[i].sbavki_left[j].ax*mm, pDoc->approx_cs[i].sbavki_left[j].ay*mm);
```

```
if ( j < pDoc->approx_cs[i].sbavki_left.size()-1)
{pDC->LineTo(pDoc->approx_cs[i].sbavki_left[j].ax*mm, pDoc->approx_cs[i].sbavki_left[j+1].ay*mm);};
if (m_Info == PETLI)
{
CString ryad, sbavka, napr, nadpis;
ryad.Format(_T("%i"),pDoc->approx_cs[i].sbavki_left[j].r);
sbavka.Format(_T("%i"),pDoc->approx_cs[i].sbavki_left[j].sbav);
if (pDoc->approx_cs[i].sbavki_left[j].pol == -1)
{
napr = ", Прибавка: ";
}
else if (pDoc->approx_cs[i].sbavki_left[j].pol == 1)
{
napr = ", Сбавка: ";
};
if (pDoc->approx_cs[i].sbavki_left[j].sbav == 0)
{
napr = ", Прибавка / Сбавка: ";
}
nadpis = CString("Ряд: ") + ryad + napr + sbavka;
pDC->SetTextColor(RGB(0,250,0));
```

```
                                    pDC->DrawText(nadpis,
CRect(CPoint(pDoc->approx_cs[i].sbavki_left[j].ax*mm - 2 -
nadpis.GetLength(),pDoc->approx_cs[i].sbavki_left[j].ay*mm-pDoc-
>approx_cs[i].sbavki_left[j].bp*pDoc->approx_cs[i].sbavki_left[j].rapapr*mm/3),
CPoint(pDoc->approx_cs[i].sbavki_left[j].ax*mm - 4 - nadpis.GetLength(),pDoc-
>approx_cs[i].sbavki_left[j].ay*mm-pDoc->approx_cs[i].sbavki_left[j].bp*pDoc-
>approx_cs[i].sbavki_left[j].rapapr*mm/3)),DT_CENTER | DT_VCENTER |
DT_SINGLELINE | DT_NOCLIP);
                                    pDC->MoveTo(pDoc-
>approx_cs[i].sbavki_left[j].ax*mm, (pDoc->approx_cs[i].sbavki_left[j].ay-pDoc-
>approx_cs[i].sbavki_left[j].bp*pDoc->approx_cs[i].sbavki_left[j].rapapr)*mm);
                                }
                                };
                        };
                }
                if (pDoc->approx_cs[i].name_left.IsEmpty() && pDoc-
>approx_cs[i].name_right.IsEmpty())
                                {
                                for (int j = 0; j < pDoc-
>approx_cs[i].sbavki_right.size(); j++)
                                {
                                    if (j == 0)
                                    {
                                        pDC->MoveTo(pDoc-
>approx_cs[i].sbavki_right[j].ax*mm,pDoc->approx_cs[i].sbavki_right[j].ay*mm);
                                    pDC->LineTo(pDoc-
>approx_cs[i].sbavki_right[j].ax*mm, pDoc->approx_cs[i].sbavki_right[j+1].ay*mm);


                                    if (m_Info == PETLI)
```

```
{
pDC->MoveTo(pDoc->approx_cs[i].sbavki_right[j].ax*mm,pDoc->approx_cs[i].sbavki_right[j].ay*mm);

CString ryad, sbavka, napr, nadpis;
ryad.Format(_T("%i"),pDoc->approx_cs[i].sbavki_right[j].r);
sbavka.Format(_T("%i"),pDoc->approx_cs[i].sbavki_right[j].sbav);
if (pDoc->approx_cs[i].sbavki_right[j].pol == -1)
{
napr = ", Прибавка: ";
}
else if (pDoc->approx_cs[i].sbavki_right[j].pol == 1)
{
napr = ", Сбавка: ";
};

if (pDoc->approx_cs[i].sbavki_right[j].sbav == 0)
{
napr = ", Прибавка / Сбавка: ";
}

nadpis = CString("Ряд: ") + ryad + napr + sbavka;
pDC->SetTextColor(RGB(0,250,0));
```

```
pDC->DrawText(nadpis,
CRect(CPoint(pDoc->approx_cs[i].sbavki_right[j].ax*mm - 2 -
nadpis.GetLength(),pDoc->approx_cs[i].sbavki_right[j].ay*mm - pDoc-
>approx_cs[i].sbavki_right[j].bp*pDoc->approx_cs[i].sbavki_right[j].rapapr*mm/3),
CPoint(pDoc->approx_cs[i].sbavki_right[j].ax*mm - 4 - nadpis.GetLength(),pDoc-
>approx_cs[i].sbavki_right[j].ay*mm - pDoc->approx_cs[i].sbavki_right[j].bp*pDoc-
>approx_cs[i].sbavki_right[j].rapapr*mm/3)),DT_CENTER | DT_VCENTER |
DT_SINGLELINE | DT_NOCLIP);
pDC->MoveTo(pDoc-
>approx_cs[i].sbavki_right[j].ax*mm, (pDoc->approx_cs[i].sbavki_right[j].ay-pDoc-
>approx_cs[i].sbavki_right[j].bp)*mm);
}
}
else
{
pDC->LineTo(pDoc-
>approx_cs[i].sbavki_right[j].ax*mm, pDoc->approx_cs[i].sbavki_right[j].ay*mm);
if ( j < pDoc-
>approx_cs[i].sbavki_right.size()-1)
{pDC->LineTo(pDoc-
>approx_cs[i].sbavki_right[j].ax*mm, pDoc-
>approx_cs[i].sbavki_right[j+1].ay*mm);};
if (m_Info == PETLI)
{
CString ryad, sbavka, napr, nadpis;
ryad.Format(_T("%i"),pDoc-
>approx_cs[i].sbavki_right[j].r);
sbavka.Format(_T("%i"),pDoc-
>approx_cs[i].sbavki_right[j].sbav);
```

```
if (pDoc->approx_cs[i].sbavki_right[j].pol == -1)
{
napr = ", Прибавка: ";
}
else if (pDoc->approx_cs[i].sbavki_right[j].pol == 1)
{
napr = ", Сбавка: ";
};

if (pDoc->approx_cs[i].sbavki_right[j].sbav == 0)
{
napr = ", Прибавка / Сбавка: ";
}
nadpis = CString("Ряд: ") + ryad + napr + sbavka;
pDC->SetTextColor(RGB(0,250,0));
pDC->DrawText(nadpis, CRect(CPoint(pDoc->approx_cs[i].sbavki_right[j].ax*mm - 2 - nadpis.GetLength(),pDoc->approx_cs[i].sbavki_right[j].ay*mm-pDoc->approx_cs[i].sbavki_right[j].bp*pDoc->approx_cs[i].sbavki_right[j].rapapr*mm/3), CPoint(pDoc->approx_cs[i].sbavki_right[j].ax*mm - 4 - nadpis.GetLength(),pDoc->approx_cs[i].sbavki_right[j].ay*mm-pDoc->approx_cs[i].sbavki_right[j].bp*pDoc->approx_cs[i].sbavki_right[j].rapapr*mm/3)),DT_CENTER | DT_VCENTER | DT_SINGLELINE | DT_NOCLIP);
```

```
                                                pDC->MoveTo(pDoc-
>approx_cs[i].sbavki_right[j].ax*mm,     (pDoc->approx_cs[i].sbavki_right[j].ay-pDoc-
>approx_cs[i].sbavki_right[j].bp*pDoc->approx_cs[i].sbavki_right[j].rapapr)*mm);
                                                }
                                        };
                                };
                                };
                        };
                        }
        }
        void CDesignerkwearView::OnInitialUpdate()
        {
                CScrollView::OnInitialUpdate();
                CSize sizeTotal;
                sizeTotal.cx = sizeTotal.cy = 10000;
                SetScrollSizes(MM_TEXT, sizeTotal);
        }
        // печать CDesignerkwearView
        void CDesignerkwearView::OnFilePrintPreview()
        {
        #ifndef SHARED_HANDLERS
                AFXPrintPreview(this);
        #endif
        }
        BOOL CDesignerkwearView::OnPreparePrinting(CPrintInfo* pInfo)
        {
                CDesignerkwearDoc* pDoc = GetDocument();
                ASSERT_VALID(pDoc);
                if (!pDoc);
                return DoPreparePrinting(pInfo);
```

```
}
void CDesignerkwearView::OnBeginPrinting(CDC* pDC, CPrintInfo*
/*pInfo*/)
{
	// TODO: добавьте дополнительную инициализацию перед печатью
}

void CDesignerkwearView::OnEndPrinting(CDC* /*pDC*/, CPrintInfo*
/*pInfo*/)
{
	// TODO: добавьте очистку после печати
}
void CDesignerkwearView::OnRButtonUp(UINT /* nFlags */, CPoint point)
{
	ClientToScreen(&point);
	OnContextMenu(this, point);
}
void CDesignerkwearView::OnContextMenu(CWnd* /* pWnd */, CPoint point)
{
#ifndef SHARED_HANDLERS
	theApp.GetContextMenuManager()-
>ShowPopupMenu(IDR_POPUP_EDIT, point.x, point.y, this, TRUE);
#endif
}
// диагностика CDesignerkwearView
#ifdef _DEBUG
void CDesignerkwearView::AssertValid() const
{
	CScrollView::AssertValid();
}
```

```
void CDesignerkwearView::Dump(CDumpContext& dc) const
{
	CScrollView::Dump(dc);
}
CDesignerkwearDoc*  CDesignerkwearView::GetDocument()  const  //  встроена неотлаженная версия
{
	ASSERT(m_pDocument->IsKindOf(RUNTIME_CLASS(CDesignerkwearDoc)));
	return (CDesignerkwearDoc*)m_pDocument;
}
#endif //_DEBUG
// обработчики сообщений CDesignerkwearView
void CDesignerkwearView::OnZoomIn()
{
	zoomWin.ZoomIn();
	zoom++;
	Invalidate(TRUE);
}
void CDesignerkwearView::OnZoomOut()
{
	zoomWin.ZoomOut();
	zoom--;
	Invalidate(TRUE);
}
void CDesignerkwearView::OnGraf()
{
	m_Graf = CONSTRUCT;
	CDesignerkwearDoc* pDoc = GetDocument();
	ASSERT_VALID(pDoc);
```

```
        if (!pDoc);
        pDoc->UpdateAllViews(NULL);
}
void CDesignerkwearView::OnSbav()
{
        m_Graf = SBAVKI;
        CDesignerkwearDoc* pDoc = GetDocument();
        ASSERT_VALID(pDoc);
        if (!pDoc);
        pDoc->UpdateAllViews(NULL);
}
void CDesignerkwearView::OnAll()
{
        m_Graf = ALL;
        CDesignerkwearDoc* pDoc = GetDocument();
        ASSERT_VALID(pDoc);
        if (!pDoc);
        pDoc->UpdateAllViews(NULL);
}
void CDesignerkwearView::OnUpdateAll(CCmdUI *pCmdUI)
{
        pCmdUI->SetCheck(m_Graf == ALL);
}
void CDesignerkwearView::OnUpdateSbav(CCmdUI *pCmdUI)
{
        pCmdUI->SetCheck(m_Graf == SBAVKI);
}
void CDesignerkwearView::OnUpdateGraf(CCmdUI *pCmdUI)
{
        pCmdUI->SetCheck(m_Graf == CONSTRUCT);
```

```
}
int petli(0);
void CDesignerkwearView::OnPetliInfo()
{
	petli++;
	if (petli == 1)
	{
		m_Info = PETLI;
	}
	else
	{
		m_Info = EMPTY_P;
		petli = 0;
	};
	CDesignerkwearDoc* pDoc = GetDocument();
	ASSERT_VALID(pDoc);
	if (!pDoc);
	pDoc->UpdateAllViews(NULL);
}
void CDesignerkwearView::OnUpdatePetliInfo(CCmdUI *pCmdUI)
{
	pCmdUI->SetCheck(m_Info == PETLI);
}
int napravv(0);
void CDesignerkwearView::OnNaprav()
{
	CDesignerkwearDoc* pDoc = GetDocument();
	ASSERT_VALID(pDoc);
	if (!pDoc);
	napravv++;
```

```
        if (napravv == 1)
        {
                m_Naprav = NAPRAV;
        }
        else
        {
                m_Naprav = EMPTY_N;
                napravv = 0;
        };
        pDoc->UpdateAllViews(NULL);
}
void CDesignerkwearView::OnUpdateNaprav(CCmdUI *pCmdUI)
{
        CDesignerkwearDoc* pDoc = GetDocument();
        ASSERT_VALID(pDoc);
        if (!pDoc);
        pCmdUI->Enable(pDoc->m_Vyt == OPEN);
        pCmdUI->SetCheck(m_Naprav == NAPRAV);
}
void CDesignerkwearView::OnLButtonDown(UINT nFlags, CPoint point)
{
        CDC *pDC=GetDC();
        OnPrepareDC(pDC);
        pDC->DPtoLP(&point);
        CDesignerkwearDoc* pDoc = GetDocument();
        ASSERT_VALID(pDoc);
        if (!pDoc);
        double mm (0);
        if (zoom > 0)
        {
```

```
            mm = pDoc->izmerenia*10.*(zoom+1);
        }
        else if (zoom < 0)
        {
            mm = pDoc->izmerenia*10./abs(zoom-1);
        }
        else
        {
            mm = pDoc->izmerenia*10.;
        };
        pvPoint.x = point.x;
        pvPoint.y = point.y;
        int nn = 10;
        int nv = nn;
        if (zoom < 0 && zoom > -2)
        {
            nn = 10*abs(zoom-1);
            nv = nn;
        };
        for (int i = 0; i < pDoc->obj.size(); i++)
            {
                if (pDoc->obj[i].st != 4 && pDoc->obj[i].st != 1001 &&
pDoc->obj[i].st != 22 && pDoc->obj[i].st != 5)
                {
                if (pvPoint.y >= pDoc->obj[i].y0*mm-nn && pvPoint.y <=
pDoc->obj[i].y0*mm+nv)
                {
                    if (pvPoint.x >= pDoc->obj[i].x0*mm-nn && pvPoint.x
<= pDoc->obj[i].x0*mm+nv)
                    {
```

```
                        id_name = pDoc->obj[i].name;
                        id_point = 0;
                        if (pDoc->obj[i].st == 1)
                        {id_curve = 1;}
                        else if (pDoc->obj[i].st == 2)
                        {id_curve = 2;}
                        if (pDoc->obj[i].st == 3)
                        {id_curve = 3;}
                        if (m_Naprav == NAPRAV)
                        {
                                pDoc->obj[i].tolshina = 5;
                                pDoc->obj[i].red = 50;
                                pDoc->obj[i].green = 250;
                                pDoc->obj[i].black = 250;
                        }
                }
        }
        if (pvPoint.y >= pDoc->obj[i].y1*mm-nn && pvPoint.y <=
pDoc->obj[i].y1*mm+nv)
        {
                if (pvPoint.x >= pDoc->obj[i].x1*mm-nn && pvPoint.x
<= pDoc->obj[i].x1*mm+nv)
                {
                        id_name = pDoc->obj[i].name;
                        id_point = 1;
                        if (pDoc->obj[i].st == 1)
                        {id_curve = 1;}
                        else if (pDoc->obj[i].st == 2)
                        {id_curve = 2;}
                        if (pDoc->obj[i].st == 3)
```

```
                    {id_curve = 3;}
                    if (m_Naprav == NAPRAV)
                    {
                        pDoc->obj[i].tolshina = 5;
                        pDoc->obj[i].red = 50;
                        pDoc->obj[i].green = 250;
                        pDoc->obj[i].black = 250;
                    }
                }
            }
            if (pvPoint.y >= pDoc->obj[i].y2*mm-nn && pvPoint.y <= pDoc->obj[i].y2*mm+nv)
            {
                if (pvPoint.x >= pDoc->obj[i].x2*mm-nn && pvPoint.x <= pDoc->obj[i].x2*mm+nv)
                {
                    id_name = pDoc->obj[i].name;
                    id_point = 2;
                    if (pDoc->obj[i].st == 2)
                    {id_curve = 2;}
                    if (pDoc->obj[i].st == 3)
                    {id_curve = 3;}
                    if (m_Naprav == NAPRAV)
                    {
                        pDoc->obj[i].tolshina = 5;
                        pDoc->obj[i].red = 50;
                        pDoc->obj[i].green = 250;
                        pDoc->obj[i].black = 250;
                    }
                }
```

```
                    }
                    if (pvPoint.y >= pDoc->obj[i].y3*mm-nn && pvPoint.y <= pDoc->obj[i].y3*mm+nv)
                    {
                        if (pvPoint.x >= pDoc->obj[i].x3*mm-nn && pvPoint.x <= pDoc->obj[i].x3*mm+nv)
                        {
                            id_name = pDoc->obj[i].name;
                            id_point = 3;
                            if (pDoc->obj[i].st == 3)
                            {id_curve = 3;}
                            if (m_Naprav == NAPRAV)
                            {
                                pDoc->obj[i].tolshina = 5;
                                pDoc->obj[i].red = 50;
                                pDoc->obj[i].green = 250;
                                pDoc->obj[i].black = 250;
                            }
                        }
                    };
                };
            };
            pDoc->UpdateAllViews(NULL);
            CScrollView::OnLButtonDown(nFlags, point);
        }
        void CDesignerkwearView::OnLButtonUp(UINT nFlags, CPoint point)
        {
            CDesignerkwearDoc* pDoc = GetDocument();
            ASSERT_VALID(pDoc);
            if (!pDoc);
```

```
        for (int i = 0; i < pDoc->obj.size(); i++)
        {
                pDoc->obj[i].tolshina = 2;
                pDoc->obj[i].red = 0;
                pDoc->obj[i].green = 0;
                pDoc->obj[i].black = 0;
        };
        pDoc->UpdateAllViews(NULL);
        id_curve = 0;
        id_point = -1;
        id_name = _T(" ");
        CScrollView::OnLButtonUp(nFlags, point);
        pDoc->OnBuild();
}
void CDesignerkwearView::OnMouseMove(UINT nFlags, CPoint point)
{
        CDC *pDC=GetDC();
        OnPrepareDC(pDC);
        pDC->DPtoLP(&point);
        CDesignerkwearDoc* pDoc = GetDocument();
        ASSERT_VALID(pDoc);
        if (!pDoc);
        double mm (0);
        CString temp_str;
        if (zoom > 0)
        {
                mm = pDoc->izmerenia*10.*(zoom+1);
        }
        else if (zoom < 0)
        {
```

```
                mm = pDoc->izmerenia*10./abs(zoom-1);
        }
        else
        {
                mm = pDoc->izmerenia*10.;
        };

        double temp(0);
        if (nFlags && MK_LBUTTON && m_Naprav == NAPRAV && m_Prnt
== E && pDoc->m_Vyt == OPEN)
        {
                for (int i = 0; i < pDoc->obj.size(); i++)
                {
                        if (id_name == pDoc->obj[i].name)
                        {
                                if (id_point == 0 && pDoc->obj[i].st != 4
                                        && pDoc->obj[i].st != 5)
                                {
                                        temp = point.x;
                                        pDoc->obj[i].x0 = temp/mm;
                                        temp = point.y;
                                        pDoc->obj[i].y0 = temp/mm;
                                        CString bbb;
                                        bbb.Format(_T("%.2f"), pDoc->obj[i].x0);
                                        pDoc->obj[i].sx0 = bbb;
                                        bbb.Format(_T("%.2f"), pDoc->obj[i].y0);
                                        pDoc->obj[i].sy0 = bbb;
                                }
                                else if (id_point == 1 && pDoc->obj[i].st != 4
                                        && pDoc->obj[i].st != 5)
```

```
{
        temp = point.x;
        pDoc->obj[i].x1 = temp/mm;
        temp = point.y;
        pDoc->obj[i].y1 = temp/mm;
        CString bbb;
        bbb.Format(_T("%.2f"), pDoc->obj[i].x1);
        pDoc->obj[i].sx1 = bbb;
        bbb.Format(_T("%.2f"), pDoc->obj[i].y1);
        pDoc->obj[i].sy1 = bbb;
}
else if (id_point == 2 && pDoc->obj[i].st != 4
        && pDoc->obj[i].st != 5)
{
        temp = point.x;
        pDoc->obj[i].x2 = temp/mm;
        temp = point.y;
        pDoc->obj[i].y2 = temp/mm;
        CString bbb;
        bbb.Format(_T("%.2f"), pDoc->obj[i].x2);
        pDoc->obj[i].sx2 = bbb;
        bbb.Format(_T("%.2f"), pDoc->obj[i].y2);
        pDoc->obj[i].sy2 = bbb;
}
else if (id_point == 3 && pDoc->obj[i].st != 4
        && pDoc->obj[i].st != 5)
{
        temp = point.x;
        pDoc->obj[i].x3 = temp/mm;
        temp = point.y;
```

```
                                pDoc->obj[i].y3 = temp/mm;
                                CString bbb;
                                bbb.Format(_T("%.2f"), pDoc->obj[i].x3);
                                pDoc->obj[i].sx3 = bbb;
                                bbb.Format(_T("%.2f"), pDoc->obj[i].y3);
                                pDoc->obj[i].sy3 = bbb;
                        };
                };
        };
        pDoc->aConsole.serial.clear();
        CString stroka;
        for (int i = 0; i < pDoc->obj.size(); i++)
        {
                if (pDoc->obj[i].st == 1)
                {
                        CString buff_com;
                        for (int j = 0; pDoc->obj[i].command; j++)
                        {
                                if (pDoc->obj[i].command[j] == ':')
                                {
                                        buff_com = pDoc->obj[i].command.Mid(j+1,
pDoc->obj[i].command.GetLength()-j);
                                        break;
                                };
                        };
                        if (pDoc->obj[i].x3 == 1)
                        {
                                stroka = pDoc->obj[i].name+_T(":") + pDoc->obj[i].fun
+ pDoc->obj[i].sx0+_T(",")+pDoc->obj[i].sy0+_T(",")
```

```
                                +pDoc->obj[i].sx1+_T(",")+pDoc-
>obj[i].sy1+_T(";");
                        pDoc->obj[i].command = stroka;
                }
                else if(pDoc->obj[i].x3 == 2)
                {
                        double dl(0), ug(0);

                        dl = sqrt((pDoc->obj[i].x1-pDoc->obj[i].x0)*(pDoc-
>obj[i].x1-pDoc->obj[i].x0)+(pDoc->obj[i].y1-pDoc->obj[i].y0)*(pDoc->obj[i].y1-
pDoc->obj[i].y0));
                        pDoc->obj[i].sy2.Format(_T("%.2f"), dl);
                        if (pDoc->obj[i].y0 > pDoc->obj[i].y1)
                        {
                                ug = 360 - acos((pDoc->obj[i].x1-pDoc-
>obj[i].x0)/dl)*180/3.1415;
                                pDoc->obj[i].sx2.Format(_T("%.2f"), ug);
                        }
                        else
                        {
                                ug = acos((pDoc->obj[i].x1-pDoc-
>obj[i].x0)/dl)*180/3.1415;
                                pDoc->obj[i].sx2.Format(_T("%.2f"), ug);
                        };
                        stroka = pDoc->obj[i].name+_T(":")+pDoc-
>obj[i].fun+pDoc->obj[i].sx0+_T(",")+pDoc->obj[i].sy0+_T(",")
                                +pDoc->obj[i].sx2+_T(",")+pDoc-
>obj[i].sy2+_T(";");
                        pDoc->obj[i].command = stroka;
                };
```

```
                }
                else if (pDoc->obj[i].st == 2)
                {
                        stroka = pDoc->obj[i].name+_T(":")+pDoc->obj[i].fun+pDoc->obj[i].sx0+_T(",")+pDoc->obj[i].sy0+_T(",")
                                        +pDoc->obj[i].sx1+_T(",")+pDoc->obj[i].sy1+_T(",")+pDoc->obj[i].sx2+_T(",")+pDoc->obj[i].sy2+_T(";");
                        pDoc->obj[i].command = stroka;
                }
                else if (pDoc->obj[i].st == 3)
                {
                        stroka = pDoc->obj[i].name+_T(":")+pDoc->obj[i].fun+pDoc->obj[i].sx0+_T(",")+pDoc->obj[i].sy0+_T(",")
                                        +pDoc->obj[i].sx1+_T(",")+pDoc->obj[i].sy1+_T(",")
                                        +pDoc->obj[i].sx2+_T(",")+pDoc->obj[i].sy2+_T(",")
                                        +pDoc->obj[i].sx3+_T(",")+pDoc->obj[i].sy3+_T(";");
                        pDoc->obj[i].command = stroka;
                }
                else if (pDoc->obj[i].st == 1000)
                {
                        CString buff_com;
                        for (int j = 0; pDoc->obj[i].command; j++)
                        {
                                if (pDoc->obj[i].command[j] == ':')
                                {
                                        buff_com = pDoc->obj[i].command.Mid(j+1, pDoc->obj[i].command.GetLength()-j);
```

```
                        break;
                    };
                };
                if (buff_com.Left(5).MakeLower() == "point" || !pDoc->obj[i].fun.IsEmpty() && pDoc->obj[i].fun.GetLength() < buff_com.GetLength() && buff_com.Left(pDoc->obj[i].fun.GetLength()).MakeLower() == pDoc->obj[i].fun)
                {
                    stroka = pDoc->obj[i].name+_T(":")+pDoc->obj[i].fun+pDoc->obj[i].sx0+_T(",")+pDoc->obj[i].sy0+_T(";");
                    pDoc->obj[i].command = stroka;
                };
            }
            else if (pDoc->obj[i].st == 1001)
            {
                CString buff_com;
                for (int j = 0; pDoc->obj[i].command; j++)
                {
                    if (pDoc->obj[i].command[j] == ':')
                    {
                        buff_com = pDoc->obj[i].command.Mid(j+1, pDoc->obj[i].command.GetLength()-j);
                        break;
                    };
                };
                if (buff_com.Left(8).MakeLower() == "point_to" || !pDoc->obj[i].fun.IsEmpty() && pDoc->obj[i].fun.GetLength() < buff_com.GetLength() && buff_com.Left(pDoc->obj[i].fun.GetLength()).MakeLower() == pDoc->obj[i].fun)
                {
```

```
                    stroka = pDoc->obj[i].name+_T(":")+pDoc->obj[i].fun+pDoc->obj[i].sx2+_T(",")+pDoc->obj[i].sy2+_T(",")+pDoc->obj[i].sa1+_T(";");
                    pDoc->obj[i].command = stroka;
                }
            }
            else if (pDoc->obj[i].st == 5)
            {
                CString buff_com;
                for (int j = 0; pDoc->obj[i].command; j++)
                {
                    if (pDoc->obj[i].command[j] == ':')
                    {
                        buff_com = pDoc->obj[i].command.Mid(j+1, pDoc->obj[i].command.GetLength()-j);
                        break;
                    };
                };
                if (buff_com.Left(1).MakeLower() == "=")
                {
                    stroka = pDoc->obj[i].name+_T(":")+_T("=")+pDoc->obj[i].sx0+_T(";");
                    pDoc->obj[i].command = stroka;
                };
            }
        };
        if (!pDoc->obj.empty())
        {
            for (int j = 0; j < pDoc->obj.size(); j++)
            {
```

```
                    if (!pDoc->obj[j].name.IsEmpty())
                    {pDoc->vivod.push_back(pDoc->obj[j].command);};
                };
            };
            if (!pDoc->com.empty())
            {
                for (int j = 0; j < pDoc->com.size(); j++)
                {
                    if (!pDoc->com[j].name.IsEmpty())
                    {pDoc->vivod.push_back(pDoc->com[j].command);};
                };
            };
            CString match;
            for (int i = 0; i < pDoc->dobj.size(); i++)
            {
                if (!pDoc->dobj[i].command.IsEmpty() && match != pDoc->dobj[i].command)
                {
                    pDoc->vivod.push_back(pDoc->dobj[i].command);
                    match = pDoc->dobj[i].command;
                };
            };
            if (!pDoc->aprox.empty())
            {
                for (int j = 0; j < pDoc->aprox.size(); j++)
                {
                    if (!pDoc->aprox[j].name.IsEmpty())
                    {pDoc->vivod.push_back(pDoc->aprox[j].command);};
                };
            };
```

```
		pDoc->aConsole.buff_inp.clear();
		// Вывод в консоль
			for (int i = 0; i < pDoc->vivod.size(); i++)
			{
				temp_str += pDoc->vivod[i]+_T("\r\n");
				pDoc->aConsole.m_EditConsole.SetWindowText(temp_str);
				pDoc->aConsole.serial.push_back(pDoc->vivod[i]+_T("@"));
				pDoc->aConsole.buff_inp.push_back(pDoc->vivod[i]);
			};
			pDoc->vivod.clear();
			pDoc->UpdateAllViews(NULL);
		};
		CScrollView::OnMouseMove(nFlags, point);
}

// Маркировка, подготовка
int prt(0);
void CDesignerkwearView::OnPrintObl()
{
		CDesignerkwearDoc* pDoc = GetDocument();
		ASSERT_VALID(pDoc);
		if (!pDoc);

		prt++;
		if (prt == 1)
		{
			m_Prnt = B;
		}
		else
		{
```

```
            m_Prnt = E;
            prt = 0;
        };

        pDoc->UpdateAllViews(NULL);
    }
    void CDesignerkwearView::OnUpdatePrintObl(CCmdUI *pCmdUI)
    {
        pCmdUI->SetCheck(m_Prnt == B);
    }
```

Файл “DObjects.cpp”

```
#include "DObjects.h"
CDObjects::CDObjects(void)
{
}
CDObjects::~CDObjects(void)
{
}
CDObjects::CDObjects(CString buff_inp, std::vector <CObjects> obj, std::vector<CRazmer> raz, std::vector<CObjects> vars, int strn, std::vector <CDObjects> dobj, std::vector<CString> com_trans)
{
    CString buff_com;
    CString aName;
    CString adname;
```

```
kx = 1; ky = 1;
CString skx, sky;
// Создание имен переопределенных команд
CString form_name, formd_name, side_name, comb_name;
// Формирование имени Д-Объекта
for (int j = 0; j < buff_inp.GetLength(); j++)
{
	if (buff_inp[0] == '/')
	{break;};
	if (buff_inp[j] == ':')
	{
		adname = buff_inp.Mid(0, j);
		buff_com = buff_inp.Mid(j+1, buff_inp.GetLength()-j);
		break;
	};
};
if (!com_trans.empty())
{
for (int i = 0; i < com_trans.size(); i++)
{
	if (com_trans[i].GetLength() > 6 &&
com_trans[i].Left(6).MakeLower() == "form->")
	{
		int z(0);
		int start(0);
		int arg(0);
		CString buff_str(com_trans[i]);
		for (int j = 0; j < buff_str.GetLength(); j++)
		{
			if (buff_str[j] == ';')
```

```
                {break;};
                if (buff_str[j] == '>')
                {z++; start = j+1; arg = 0;};
                if (z == 1 && buff_str[j] != '>')
                {
                        arg++;
                        form_name = buff_str.Mid(start,arg);
                };
        };
}
else if (com_trans[i].GetLength() > 7 &&
com_trans[i].Left(7).MakeLower() == "formd->")
{
        int z(0);
        int start(0);
        int arg(0);
        CString buff_str(com_trans[i]);
        for (int j = 0; j < buff_str.GetLength(); j++)
        {
                if (buff_str[j] == ';')
                {break;};
                if (buff_str[j] == '>')
                {z++; start = j+1; arg = 0;};
                if (z == 1 && buff_str[j] != '>')
                {
                        arg++;
                        form_name = buff_str.Mid(start,arg);
                };
        };
}
```

```
else if (com_trans[i].GetLength() > 6 &&
com_trans[i].Left(6).MakeLower() == "side->")
{
    int z(0);
    int start(0);
    int arg(0);
    CString buff_str(com_trans[i]);
    for (int j = 0; j < buff_str.GetLength(); j++)
    {
        if (buff_str[j] == ';')
        {break;};
        if (buff_str[j] == '>')
        {z++; start = j+1; arg = 0;};
        if (z == 1 && buff_str[j] != '>')
        {
            arg++;
            side_name = buff_str.Mid(start,arg);
        };
    };
}
else if (com_trans[i].GetLength() > 6 &&
com_trans[i].Left(6).MakeLower() == "comb->"
    || com_trans[i].GetLength() > 7 &&
com_trans[i].Left(7).MakeLower() == "comb11->"
    || com_trans[i].GetLength() > 7 &&
com_trans[i].Left(7).MakeLower() == "comb01->"
    || com_trans[i].GetLength() > 7 &&
com_trans[i].Left(7).MakeLower() == "comb10->")///////////////
{
    int z(0);
```

```
                int start(0);
                int arg(0);
                CString buff_str(com_trans[i]);
                for (int j = 0; j < buff_str.GetLength(); j++)
                {
                        if (buff_str[j] == ';')
                        {break;};
                        if (buff_str[j] == '>')
                        {z++; start = j+1; arg = 0;};
                        if (z == 1 && buff_str[j] != '>')
                        {
                                arg++;
                                comb_name = buff_str.Mid(start,arg);
                        };
                };
        };
};
};
// Форма
        if (buff_com.Left(4).MakeLower() == "form" && buff_com.Left(5).MakeLower() != "formd"
                || !form_name.IsEmpty() && form_name.GetLength() < buff_com.GetLength() && buff_com.Left(form_name.GetLength()).MakeLower() == form_name)
                {
                str = strn;
                command = buff_inp;
                std::vector<CString> names;
                int z(0);
                int start(0);
```

```
int arg(0);
int r(0);
side_const = 1;
st = 50;
dname = adname;
for (int i = 0; ; i++)
{
	if (buff_com[i] == ';')
	{
		names.push_back(aName);
		break;
	};
	if (buff_com[i] == '>' || buff_com[i] == ',')
	{
		z++; start = i+1; arg = 0;
		if (!aName.IsEmpty())
		{
			names.push_back(aName);
		};
	};
	if (buff_com[i] == '|')
	{
		z++; start = i+1; arg = -1;
		names.push_back(aName);
		CString razd;
		razd = buff_com[i];
		names.push_back(razd);
	};
	if (z == 1 && buff_com[i] != '>')
	{
```

```
                arg++;
                aName = buff_com.Mid(start,arg);
        };
        if (z == 2 && buff_com[i] != ',')
        {
                arg++;
                aName = buff_com.Mid(start,arg);
        };
        if (z == 3 && buff_com[i] != ',')
        {
                arg++;
                aName = buff_com.Mid(start,arg);
                z = 2;
        };
};
std::vector<CString> left;
std::vector<CString> right;
int g(0);
for (int i = 0; i < names.size(); i++)
{
        if (names[i]=="|")
        {
                r++;
        }
        if (r == 1 && names[i] != "|")
        {
                right.push_back(names[i]);
        }
        else if(r == 0)
        {
```

```
                left.push_back(names[i]);
            }
            else if(r == 2 && names[i] != "|")
            {
                if (g == 0)
                {skx = names[i];  g++;}
                else if (g == 1)
                {sky = names[i];}
            };

        };
        for (int i = 0; i < left.size(); i++)
        {
            for (int j = 0; j < obj.size(); j++)
            {
                if (left[i] == obj[j].name)
                {

d_object_left.push_back(CReverseObj(obj[j]));
                    break;
                }
            };
        };
        for (int i = 0; i < right.size(); i++)
        {
            for (int j = 0; j < obj.size(); j++)
            {
                if (right[i] == obj[j].name)
                {
```

```
d_object_right.push_back(CReverseObj(obj[j]));
                                    break;
                            }
                    };
            };
            // Усадка
            if (!skx.IsEmpty())
            {kx = converter(skx,raz,vars);}
            if (!sky.IsEmpty())
            {ky = converter(sky,raz,vars);}
            // Расчет на усадку
            for (int i = 0; i < d_object_left.size(); i++)
            {
                    if (d_object_left[i].st == 1)
                    {
                            d_object_left[i].x0 = d_object_left[i].x0*kx;
                            d_object_left[i].y0 = d_object_left[i].y0*ky;
                            d_object_left[i].x1 = d_object_left[i].x1*kx;
                            d_object_left[i].y1 = d_object_left[i].y1*ky;
                    }
                    else if (d_object_left[i].st == 2)
                    {
                            d_object_left[i].x0 = d_object_left[i].x0*kx;
                            d_object_left[i].y0 = d_object_left[i].y0*ky;
                            d_object_left[i].x1 = d_object_left[i].x1*kx;
                            d_object_left[i].y1 = d_object_left[i].y1*ky;
                            d_object_left[i].x2 = d_object_left[i].x2*kx;
                            d_object_left[i].y2 = d_object_left[i].y2*ky;
                    }
```

```
        else if (d_object_left[i].st == 3)
        {
            d_object_left[i].x0 = d_object_left[i].x0*kx;
            d_object_left[i].y0 = d_object_left[i].y0*ky;
            d_object_left[i].x1 = d_object_left[i].x1*kx;
            d_object_left[i].y1 = d_object_left[i].y1*ky;
            d_object_left[i].x2 = d_object_left[i].x2*kx;
            d_object_left[i].y2 = d_object_left[i].y2*ky;
            d_object_left[i].x3 = d_object_left[i].x3*kx;
            d_object_left[i].y3 = d_object_left[i].y3*ky;
        };
        d_object.push_back(d_object_left[i]);
};
for (int i = 0; i < d_object_right.size(); i++)
{
        if (d_object_right[i].st == 1)
        {
            d_object_right[i].x0 = d_object_right[i].x0*kx;
            d_object_right[i].y0 = d_object_right[i].y0*ky;
            d_object_right[i].x1 = d_object_right[i].x1*kx;
            d_object_right[i].y1 = d_object_right[i].y1*ky;
        }
        else if (d_object_right[i].st == 2)
        {
            d_object_right[i].x0 = d_object_right[i].x0*kx;
            d_object_right[i].y0 = d_object_right[i].y0*ky;
            d_object_right[i].x1 = d_object_right[i].x1*kx;
            d_object_right[i].y1 = d_object_right[i].y1*ky;
            d_object_right[i].x2 = d_object_right[i].x2*kx;
            d_object_right[i].y2 = d_object_right[i].y2*ky;
```

```
                    }
                    else if (d_object_right[i].st == 3)
                    {
                        d_object_right[i].x0 = d_object_right[i].x0*kx;
                        d_object_right[i].y0 = d_object_right[i].y0*ky;
                        d_object_right[i].x1 = d_object_right[i].x1*kx;
                        d_object_right[i].y1 = d_object_right[i].y1*ky;
                        d_object_right[i].x2 = d_object_right[i].x2*kx;
                        d_object_right[i].y2 = d_object_right[i].y2*ky;
                        d_object_right[i].x3 = d_object_right[i].x3*kx;
                        d_object_right[i].y3 = d_object_right[i].y3*ky;
                    };
                    d_object.push_back(d_object_right[i]);
                };
            }
            // Форма доп
            else if (buff_com.Left(5).MakeLower() == "formd"
                || !formd_name.IsEmpty() && formd_name.GetLength() < buff_com.GetLength() && buff_com.Left(formd_name.GetLength()).MakeLower() == formd_name)
                {
                    str = strn;
                    command = buff_inp;
                    std::vector<CString> names;
                    int z(0);
                    int start(0);
                    int arg(0);
                    int r(0);
                    side_const = 1;
                    st = 53;
```

```
dname = adname;
for (int i = 0; ; i++)
{
	if (buff_com[i] == ';')
	{
		names.push_back(aName);
		break;
	};
	if (buff_com[i] == '>' || buff_com[i] == ',')
	{
		z++; start = i+1; arg = 0;
		if (!aName.IsEmpty())
		{
			names.push_back(aName);
		};
	};

	if (buff_com[i] == '|')
	{
		z++; start = i+1; arg = -1;
		names.push_back(aName);
		CString razd;
		razd = buff_com[i];
		names.push_back(razd);
	};
	if (z == 1 && buff_com[i] != '>')
	{
		arg++;
		aName = buff_com.Mid(start,arg);
	};
```

```
        if (z == 2 && buff_com[i] != ',')
        {
                arg++;
                aName = buff_com.Mid(start,arg);
        };
        if (z == 3 && buff_com[i] != ',')
        {
                arg++;
                aName = buff_com.Mid(start,arg);
                z = 2;
        };
};
std::vector<CString> left;
std::vector<CString> right;
std::vector<CString> dopol;
int g(0);
for (int i = 0; i < names.size(); i++)
{
        if (names[i]=="|")
        {
                r++;
        }
        if (r == 1 && names[i] != "|")
        {
                right.push_back(names[i]);
        }
        else if(r == 0)
        {
                left.push_back(names[i]);
        }
```

```
                else if(r == 2)
                {
                        dopol.push_back(names[i]);
                }
                else if(r == 3 && names[i] != "|")
                {
                        if (g == 0)
                        {skx = names[i];  g++;}
                        else if (g == 1)
                        {sky = names[i];}
                };
        };
        for (int i = 0; i < left.size(); i++)
        {
                for (int j = 0; j < obj.size(); j++)
                {
                        if (left[i] == obj[j].name)
                        {

d_object_left.push_back(CReverseObj(obj[j]));
                                break;
                        }
                };
        };
        for (int i = 0; i < right.size(); i++)
        {
                for (int j = 0; j < obj.size(); j++)
                {
                        if (right[i] == obj[j].name)
                        {
```

```
d_object_right.push_back(CReverseObj(obj[j]));
                                    break;
                            }
                    };
            };
            for (int i = 0; i < dopol.size(); i++)
            {
                    for (int j = 0; j < obj.size(); j++)
                    {
                            if (dopol[i] == obj[j].name)
                            {
                                    side.push_back(CReverseObj(obj[j]));
                                    break;
                            }
                    };
            };


            // Усадка
            if (!skx.IsEmpty())
            {kx = converter(skx,raz,vars);}
            if (!sky.IsEmpty())
            {ky = converter(sky,raz,vars);}
            // Расчет на усадку
            for (int i = 0; i < d_object_left.size(); i++)
            {
                    if (d_object_left[i].st == 1)
                    {
                            d_object_left[i].x0 = d_object_left[i].x0*kx;
```

```
                d_object_left[i].y0 = d_object_left[i].y0*ky;
                d_object_left[i].x1 = d_object_left[i].x1*kx;
                d_object_left[i].y1 = d_object_left[i].y1*ky;
            }
            else if (d_object_left[i].st == 2)
            {
                d_object_left[i].x0 = d_object_left[i].x0*kx;
                d_object_left[i].y0 = d_object_left[i].y0*ky;
                d_object_left[i].x1 = d_object_left[i].x1*kx;
                d_object_left[i].y1 = d_object_left[i].y1*ky;
                d_object_left[i].x2 = d_object_left[i].x2*kx;
                d_object_left[i].y2 = d_object_left[i].y2*ky;
            }
            else if (d_object_left[i].st == 3)
            {
                d_object_left[i].x0 = d_object_left[i].x0*kx;
                d_object_left[i].y0 = d_object_left[i].y0*ky;
                d_object_left[i].x1 = d_object_left[i].x1*kx;
                d_object_left[i].y1 = d_object_left[i].y1*ky;
                d_object_left[i].x2 = d_object_left[i].x2*kx;
                d_object_left[i].y2 = d_object_left[i].y2*ky;
                d_object_left[i].x3 = d_object_left[i].x3*kx;
                d_object_left[i].y3 = d_object_left[i].y3*ky;
            };
            d_object.push_back(d_object_left[i]);
        };
        for (int i = 0; i < d_object_right.size(); i++)
        {
            if (d_object_right[i].st == 1)
            {
```

```
                    d_object_right[i].x0 = d_object_right[i].x0*kx;
                    d_object_right[i].y0 = d_object_right[i].y0*ky;
                    d_object_right[i].x1 = d_object_right[i].x1*kx;
                    d_object_right[i].y1 = d_object_right[i].y1*ky;
                }
                else if (d_object_right[i].st == 2)
                {
                    d_object_right[i].x0 = d_object_right[i].x0*kx;
                    d_object_right[i].y0 = d_object_right[i].y0*ky;
                    d_object_right[i].x1 = d_object_right[i].x1*kx;
                    d_object_right[i].y1 = d_object_right[i].y1*ky;
                    d_object_right[i].x2 = d_object_right[i].x2*kx;
                    d_object_right[i].y2 = d_object_right[i].y2*ky;
                }
                else if (d_object_right[i].st == 3)
                {
                    d_object_right[i].x0 = d_object_right[i].x0*kx;
                    d_object_right[i].y0 = d_object_right[i].y0*ky;
                    d_object_right[i].x1 = d_object_right[i].x1*kx;
                    d_object_right[i].y1 = d_object_right[i].y1*ky;
                    d_object_right[i].x2 = d_object_right[i].x2*kx;
                    d_object_right[i].y2 = d_object_right[i].y2*ky;
                    d_object_right[i].x3 = d_object_right[i].x3*kx;
                    d_object_right[i].y3 = d_object_right[i].y3*ky;
                };
                d_object.push_back(d_object_right[i]);
            };
    }
    // Кромка
    else if (buff_com.Left(4).MakeLower() == "side"
```

```
                || !side_name.IsEmpty() && side_name.GetLength() < buff_com.GetLength() && buff_com.Left(side_name.GetLength()).MakeLower() == side_name)
        {
            str = strn;
                command = buff_inp;
                int z(0);
                int start(0);
                int arg(0);
                int r(0);
                int dob(0);
                st = 51;
                dname = adname;
                for (int i = 0; ; i++)
                {
                    if (buff_com[i] == ';')
                    { break; };

                    if (buff_com[i] == '>' || buff_com[i] == ',')
                    {
                        z++; start = i+1; arg = 0;
                        for (int j = 0; j < obj.size(); j++)
                        {
                            if (aName == obj[j].name && !obj[j].name.IsEmpty())
                            {

    side.push_back(CReverseObj(obj[j]));
                                break;
                            };
```

```
                    };
                };
                if (buff_com[i] == '|')
                {
                    z = 0; start = i+1; arg = 0; r++;
                    for (int j = 0; j < obj.size(); j++)
                    {
                        if (aName == obj[j].name && !obj[j].name.IsEmpty())
                        {
                            side.push_back(CReverseObj(obj[j]));
                            break;
                        };
                    };
                };
            if (r == 0)
            {
                if (z == 1 && buff_com[i] != '>')
                {
                    arg++;
                    aName = buff_com.Mid(start,arg);
                };
                if (z == 2 && buff_com[i] != ',')
                {
                    arg++;
                    aName = buff_com.Mid(start,arg);
                };
                if (z == 3 && buff_com[i] != ',')
                {
```

```
                arg++;
                aName = buff_com.Mid(start,arg);
                z = 2;
        }
}
else if (r == 1)
{
        if (z == 0 && buff_com[i] != '|')
        {
                arg++;
                aName = buff_com.Mid(start,arg);
                if (aName.MakeLower() == "left")
                {
                        side_const = -1;
                }
                else if (aName.MakeLower() == "right")
                {
                        side_const = 1;
                };
        };
}
else if (r == 2)
{
        if (z == 0 && buff_com[i] != '|')
        {
                arg++;
                skx = buff_com.Mid(start,arg);
        };
        if (z == 1 && buff_com[i] != ',')
        {
```

```
                arg++;
                sky = buff_com.Mid(start,arg);
            };
        };
};
        if (!skx.IsEmpty())
        {kx = converter(skx,raz,vars);}
        if (!sky.IsEmpty())
        {ky = converter(sky,raz,vars);}
        // Расчет на усадку
        for (int i = 0; i < side.size(); i++)
        {
            if (side[i].st == 1)
            {
                side[i].x0 = side[i].x0*kx;
                side[i].y0 = side[i].y0*ky;
                side[i].x1 = side[i].x1*kx;
                side[i].y1 = side[i].y1*ky;
            }
            else if (side[i].st == 2)
            {
                side[i].x0 = side[i].x0*kx;
                side[i].y0 = side[i].y0*ky;
                side[i].x1 = side[i].x1*kx;
                side[i].y1 = side[i].y1*ky;
                side[i].x2 = side[i].x2*kx;
                side[i].y2 = side[i].y2*ky;
            }
            else if (side[i].st == 3)
            {
```

```
                                side[i].x0 = side[i].x0*kx;
                                side[i].y0 = side[i].y0*ky;
                                side[i].x1 = side[i].x1*kx;
                                side[i].y1 = side[i].y1*ky;
                                side[i].x2 = side[i].x2*kx;
                                side[i].y2 = side[i].y2*ky;
                                side[i].x3 = side[i].x3*kx;
                                side[i].y3 = side[i].y3*ky;
                            };
                        };
                }
                else if (buff_com.Left(4).MakeLower() == "comb"
                        || !comb_name.IsEmpty() && comb_name.GetLength() < buff_com.GetLength() && buff_com.Left(comb_name.GetLength()).MakeLower() == comb_name)
                {
                        str = strn;
                        command = buff_inp;
                        std::vector<CString> names;
                        CString sbs;

                            int z(0);
                            int start(0);
                            int arg(0);
                            int r(0);
                            side_const = 1;
                            st = 52;
                            dname = adname;
                            for (int i = 0; ; i++)
                            {
```

```
if (buff_com[i] == ';')
{
        names.push_back(aName);
        break;
};
if (buff_com[i] == '>' || buff_com[i] == ',')
{
        z++; start = i+1; arg = 0;
        if (!aName.IsEmpty())
        {
                names.push_back(aName);
        };
};
if (z == 0)
{ sbs += buff_com[i];};
if (buff_com[i] == '|')
{
        z++; start = i+1; arg = -1;
        names.push_back(aName);
        CString razd;
        razd = buff_com[i];
        names.push_back(razd);

};
if (z == 1 && buff_com[i] != '>')
{
        arg++;
        aName = buff_com.Mid(start,arg);
};
if (z == 2 && buff_com[i] != ',')
```

```
		{
			arg++;
			aName = buff_com.Mid(start,arg);
		};
		if (z == 3 && buff_com[i] != ',')
		{
			arg++;
			aName = buff_com.Mid(start,arg);
			z = 2;
		};
};
std::vector<CString> left;
std::vector<CString> right;
s_name = sbs.Right(2);
if (s_name != "01" && s_name != "10")
{s_name = "11";};
int g(0);
for (int i = 0; i < names.size(); i++)
{
		if (names[i] == "|")
		{
			r++;
		};
		if (r == 1)
		{
			right.push_back(names[i]);
		}
		else if(r == 0)
		{
			left.push_back(names[i]);
```

```
                }
                else if(r == 2 && names[i] != "|")
                {
                        if (g == 0)
                        {skx = names[i];  g++;}
                        else if (g == 1)
                        {sky = names[i];}
                };
        };
        for (int i = 0; i < left.size(); i++)
        {
                for (int j = 0; j < obj.size(); j++)
                {
                        if (left[i] == obj[j].name)
                        {

comb_left.push_back(CReverseObj(obj[j]));
                                break;
                        }
                };
        };


        for (int i = 0; i < right.size(); i++)
        {
                for (int j = 0; j < obj.size(); j++)
                {
                        if (right[i] == obj[j].name)
                        {
```

```
comb_right.push_back(CReverseObj(obj[j]));
                                        break;
                                }
                        };
                };
                // Усадка
                if (!skx.IsEmpty())
                {kx = converter(skx,raz,vars);}
                if (!sky.IsEmpty())
                {ky = converter(sky,raz,vars);}
                CString name_left, name_right;
                for (int j = 0; j < dname.GetLength(); j++)
                {
                        if (dname[j] == '-')
                        {
                                name_left = dname.Left(j);
                                name_right  =  dname.Right(dname.GetLength()-j-
1);
                                break;
                        };
                };
                for (int i = 0; i < dobj.size(); i++)
                {
                        if (name_left == dobj[i].dname)
                        {
                                kx = dobj[i].kx;
                                ky = dobj[i].ky;
                        };
                };
```

```
// Расчет на усадку
for (int i = 0; i < comb_left.size(); i++)
{
	if (comb_left[i].st == 1)
	{
		comb_left[i].x0 = comb_left[i].x0*kx;
		comb_left[i].y0 = comb_left[i].y0*ky;
		comb_left[i].x1 = comb_left[i].x1*kx;
		comb_left[i].y1 = comb_left[i].y1*ky;
	}
	else if (comb_left[i].st == 2)
	{
		comb_left[i].x0 = comb_left[i].x0*kx;
		comb_left[i].y0 = comb_left[i].y0*ky;
		comb_left[i].x1 = comb_left[i].x1*kx;
		comb_left[i].y1 = comb_left[i].y1*ky;
		comb_left[i].x2 = comb_left[i].x2*kx;
		comb_left[i].y2 = comb_left[i].y2*ky;
	}
	else if (comb_left[i].st == 3)
	{
		comb_left[i].x0 = comb_left[i].x0*kx;
		comb_left[i].y0 = comb_left[i].y0*ky;
		comb_left[i].x1 = comb_left[i].x1*kx;
		comb_left[i].y1 = comb_left[i].y1*ky;
		comb_left[i].x2 = comb_left[i].x2*kx;
		comb_left[i].y2 = comb_left[i].y2*ky;
		comb_left[i].x3 = comb_left[i].x3*kx;
		comb_left[i].y3 = comb_left[i].y3*ky;
	};
```

```
            comb.push_back(comb_left[i]);
    };
    for (int i = 0; i < dobj.size(); i++)
    {
            if (name_right == dobj[i].dname)
            {
                    kx = dobj[i].kx;
                    ky = dobj[i].ky;
            };
    };
    for (int i = 0; i < comb_right.size(); i++)
    {
            if (comb_right[i].st == 1)
            {
                    comb_right[i].x0 = comb_right[i].x0*kx;
                    comb_right[i].y0 = comb_right[i].y0*ky;
                    comb_right[i].x1 = comb_right[i].x1*kx;
                    comb_right[i].y1 = comb_right[i].y1*ky;
            }
            else if (comb_right[i].st == 2)
            {
                    comb_right[i].x0 = comb_right[i].x0*kx;
                    comb_right[i].y0 = comb_right[i].y0*ky;
                    comb_right[i].x1 = comb_right[i].x1*kx;
                    comb_right[i].y1 = comb_right[i].y1*ky;
                    comb_right[i].x2 = comb_right[i].x2*kx;
                    comb_right[i].y2 = comb_right[i].y2*ky;
            }
            else if (comb_right[i].st == 3)
            {
```

```
                                comb_right[i].x0 = comb_right[i].x0*kx;
                                comb_right[i].y0 = comb_right[i].y0*ky;
                                comb_right[i].x1 = comb_right[i].x1*kx;
                                comb_right[i].y1 = comb_right[i].y1*ky;
                                comb_right[i].x2 = comb_right[i].x2*kx;
                                comb_right[i].y2 = comb_right[i].y2*ky;
                                comb_right[i].x3 = comb_right[i].x3*kx;
                                comb_right[i].y3 = comb_right[i].y3*ky;
                        };
                        comb.push_back(comb_right[i]);
                };
        };
        }
        double CDObjects::converter(CString argum, std::vector<CRazmer> rz,
std::vector <CObjects> vars)
    {
        double argument (0);
        std::vector<CString> temp1;
        std::vector<CString> temp2;
        CString aName;
        int z(1);
        int start(0);
        int arg(0);
        int pr(0);
        int v2_count(0);

        std::vector <CString> v1; // выходная строка
        std::vector <CString> v2; // знаки
        std::vector <double> result;
        double n1(0), n2(0), res(0);
```

```
if (argum[0] == '-')
{
	argum = CString("0.00")+argum;
}
else
{
	argum = CString("0.00+")+argum;
};
for (int i = 0; i < argum.GetLength() ; i++)
{
		if (argum[i] == '+' || argum[i] == '-' || argum[i] == '*' || argum[i]
== '/'
			|| argum[i] == '(' || argum[i] == ')')
		{
			if (argum[i] == '-' && argum[i-1] == '(')
			{
				z++; start = i+1; arg = 0;

				temp1.push_back(CString("-1.00"));

				temp1.push_back(CString("*"));
			}
			else
			{
				z++; start = i+1; arg = 0;
				temp1.push_back(aName);
				temp1.push_back(CString(argum[i]));
			};
		};
		if (z == 1)
```

```
                {
                        arg++;
                        aName = argum.Mid(start,arg);
                }
                else if (z == 2)
                {
                        arg++;
                        aName = argum.Mid(start,arg-1);
                }
                else if (z == 3)
                {
                        arg++;
                        aName = argum.Mid(start,arg-1);
                        z = 2;
                };
};
temp1.push_back(aName);
for (int i = 0; i < temp1.size(); i++)
{
                for (int j = 0; j < temp1[i].GetLength(); j++)
                {
                        if (temp1[i][j] == '.')
                        {
                                for (int h = 0; h < vars.size(); h++)
                                {
                                        if (vars[h].name == temp1[i].Left(j))
                                        {
                                                if (temp1[i].Right(2) == "x0")
                                                {
                                                        CString bbb;
```

```
                bbb.Format(_T("%.2f"),
vars[h].x0);
                temp1[i] = bbb;
            }
            else if (temp1[i].Right(2) == "y0")
            {
                CString bbb;
                bbb.Format(_T("%.2f"),
vars[h].y0);
                temp1[i] = bbb;
            }
            else if (temp1[i].Right(2) == "x1")
            {
                CString bbb;
                bbb.Format(_T("%.2f"),
vars[h].x1);
                temp1[i] = bbb;
            }
            else if (temp1[i].Right(2) == "y1")
            {
                CString bbb;
                bbb.Format(_T("%.2f"),
vars[h].y1);
                temp1[i] = bbb;
            }
            else if (temp1[i].Right(2) == "x2")
            {
                CString bbb;
                bbb.Format(_T("%.2f"),
vars[h].x2);
```

```
                temp1[i] = bbb;
            }
            else if (temp1[i].Right(2) == "y2")
            {
                CString bbb;
                bbb.Format(_T("%.2f"),
vars[h].y2);
                temp1[i] = bbb;
            }
            else if (temp1[i].Right(2) == "x3")
            {
                CString bbb;
                bbb.Format(_T("%.2f"),
vars[h].x3);
                temp1[i] = bbb;
            }
            else if (temp1[i].Right(2) == "y3")
            {
                CString bbb;
                bbb.Format(_T("%.2f"),
vars[h].y3);
                temp1[i] = bbb;
            }
            else                                    if
(temp1[i].Right(1).MakeLower() == "l")
            {
                double line(0);
                if (vars[h].st == 1)
                {
```

```
                line = sqrt((vars[h].x1 -
vars[h].x0)*(vars[h].x1 - vars[h].x0)
                    +(vars[h].y1 -
vars[h].y0)*(vars[h].y1 - vars[h].y0));
            }
            else if (vars[h].st == 2)
            {
                double a1(vars[h].x0),
b1(vars[h].y0), a2(0), b2(0);
                for (double t = 0.001; t
<= 1; t += 0.001)
                {
                    a2 = (1-t)*(1-
t)*vars[h].x0+2*t*(1-t)*vars[h].x1+t*t*vars[h].x2;
                    b2 = (1-t)*(1-
t)*vars[h].y0+2*t*(1-t)*vars[h].y1+t*t*vars[h].y2;
                    line += sqrt((a1-
a2)*(a1-a2)+(b1-b2)*(b1-b2));
                    a1 = a2;
                    b1 = b2;
                };
            }
            else if (vars[h].st == 3)
            {
                double a1(vars[h].x0),
b1(vars[h].y0), a2(0), b2(0);
                for (double t = 0.001; t
<= 1; t += 0.001)
                {
```

```
                                                    a2 = (1-t)*(1-t)*(1-t)*vars[h].x0+3*t*(1-t)*(1-t)*vars[h].x1+3*t*t*(1-t)*vars[h].x2+t*t*t*vars[h].x3;
                                                    b2 = (1-t)*(1-t)*(1-t)*vars[h].y0+3*t*(1-t)*(1-t)*vars[h].y1+3*t*t*(1-t)*vars[h].y2+t*t*t*vars[h].y3;
                                                    line += sqrt((a1-a2)*(a1-a2)+(b1-b2)*(b1-b2));
                                                    a1 = a2;
                                                    b1 = b2;
                                                };
                                            }
                                            CString bbb;
                                            bbb.Format(_T("%.2f"), line);
                                            temp1[i] = bbb;
                                        };;
                                    };
                                };
                                break;
                            };
                        };
                        for (int j = 0; j < rz.size(); j++)
                        {
                            if (rz[j].name.MakeLower() == temp1[i].MakeLower())
                            {
                                CString bbb;
                                bbb.Format(_T("%.2f"), rz[j].rz);
                                temp1[i] = bbb;
                            }
                        };
                        for (int j = 0; j < vars.size(); j++)
                        {
```

```
                        if (vars[j].name == temp1[i])
                        {
                                CString bbb;
                                bbb.Format(_T("%.2f"), vars[j].x0);
                                temp1[i] = bbb;
                        };
                };
        };
        for (int i = 0; i < temp1.size(); i++)
        {
                if (!temp1[i].IsEmpty())
                {
                        temp2.push_back(temp1[i]);
                };
        };
        temp1.clear();
        for (int i = 0; i < temp2.size(); i++)
        {
                if (temp2[i] == "+" || temp2[i] == "-" || temp2[i] == "/" || temp2[i] ==
"*"
                         || temp2[i] == "(" || temp2[i] == ")")
                {
                        pr = prior(temp2[i]);
                        v2.push_back(temp2[i]);

                        if (temp2[i] == ")")
                        {
                                for (int j = v2.size()-1; ; j--)
                                {
                                        v2_count++;
```

```
                    if (v2[j] == "(")
                    {
                        v2.pop_back();
                        break;
                    };
                    v1.push_back(v2[j]);

                    if (v1.back() == "(")
                    {
                        v1.pop_back();
                    };
                    if (v1.back() == ")")
                    {
                        v1.pop_back();
                    };
                };
                for (int j = 0; j < v2_count-1; j++)
                {
                    v2.pop_back();
                };
                v2_count = 0;
            };
        }
        else
        {
            v1.push_back(temp2[i]);
            if (pr == 1)
            {
                v1.push_back(v2.back());
                v2.pop_back();
```

```
                };
        };
};
for (int j = v2.size()-1; j >= 0; j--)
{
        if(v2[j] != "(")
        {v1.push_back(v2[j]);};
};
std::stack<double> st1;
for (int i = 0; i < v1.size(); i++)
{
                if (v1[i] == "+")
                {
                        n1 = st1.top();
                        st1.pop();
                        n2 = st1.top();
                        st1.pop();
                        res = n2+n1;
                        st1.push(res);
                }
                else if (v1[i] == "-")
                {
                        n1 = st1.top();
                        st1.pop();
                        n2 = st1.top();
                        st1.pop();
                        res = n2-n1;
                        st1.push(res);
                }
                else if (v1[i] == "/")
```

```
{
        n1 = st1.top();
        st1.pop();
        n2 = st1.top();
        st1.pop();
        if (n1 == 0.000)
        {
                AfxMessageBox(CString("Внимание!    Деление на ноль!"), MB_OK);
                break;
        }
        else
        {
                res = n2/n1;
                st1.push(res);
        };
}
else if (v1[i] == "*")
{
        n1 = st1.top();
        st1.pop();
        n2 = st1.top();
        st1.pop();
        res = n2*n1;
        st1.push(res);
}
else
{
        st1.push(_tstof(v1[i]));
};
```

```
        };
        return st1.top();
};
int CDObjects::prior(CString zn)
{
        if (zn == '+' || zn == '-' || zn == '(' || zn == ')')
        {return 0;}
        else if (zn == '*' || zn == '/')
        {return 1;}
        else
        {return 2;};
};
```

Файл “Info.cpp”

```
#include "Info.h"
CInfo::CInfo(void)
{
}
CInfo::~CInfo(void)
{
}
CInfo::CInfo(CObjects obj)
{
        double line(0);
        if (obj.st == 1 || obj.st == 4)
        {
                CString st;
                if (obj.st == 1)
                {
                        st.Format(_T("%i"), obj.st);
```

```
                }
                else
                {
                        st = "L";
                };
                name = obj.name+CString(" (")+st+CString(")");
                line = sqrt((obj.x1 - obj.x0)*(obj.x1 - obj.x0)+(obj.y1 - obj.y0)*(obj.y1 - obj.y0));
                value.Format(_T("%.1f"), line);
        }
        else if (obj.st == 2)
        {
                CString st;
                st.Format(_T("%i"), obj.st);
                name = obj.name+CString(" (")+st+CString(")");
                double a1(obj.x0), b1(obj.y0), a2(0), b2(0);
                for (double t = 0.001; t <= 1; t += 0.001)
                {
                        a2 = (1-t)*(1-t)*obj.x0+2*t*(1-t)*obj.x1+t*t*obj.x2;
                        b2 = (1-t)*(1-t)*obj.y0+2*t*(1-t)*obj.y1+t*t*obj.y2;
                        line += sqrt((a1-a2)*(a1-a2)+(b1-b2)*(b1-b2));
                        a1 = a2;
                        b1 = b2;
                };
                value.Format(_T("%.1f"), line);
        }
        else if (obj.st == 3)
        {
                CString st;
                st.Format(_T("%i"), obj.st);
```

```
                name = obj.name+CString(" (")+st+CString(")");
                double a1(obj.x0), b1(obj.y0), a2(0), b2(0);
                for (double t = 0.001; t <= 1; t += 0.001)
                {
                        a2 = (1-t)*(1-t)*(1-t)*obj.x0+3*t*(1-t)*(1-t)*obj.x1+3*t*t*(1-
t)*obj.x2+t*t*t*obj.x3;
                        b2 = (1-t)*(1-t)*(1-t)*obj.y0+3*t*(1-t)*(1-t)*obj.y1+3*t*t*(1-
t)*obj.y2+t*t*t*obj.y3;
                        line += sqrt((a1-a2)*(a1-a2)+(b1-b2)*(b1-b2));
                        a1 = a2;
                        b1 = b2;
                };
                value.Format(_T("%.1f"), line);
        }
        else if (obj.st == 5)
        {
                CString st;
                st = "V";
                name = obj.name+CString(" (")+st+CString(")");
                value.Format(_T("%.2f"), obj.x0);
        };
}
```

Файл “MainFrm.cpp”

```
#include "Designer k-wear.h"
#include "MainFrm.h"
#ifdef _DEBUG
#define new DEBUG_NEW
#endif
// CMainFrame
```

```
IMPLEMENT_DYNCREATE(CMainFrame, CFrameWndEx)
const int  iMaxUserToolbars = 10;
const UINT uiFirstUserToolBarId = AFX_IDW_CONTROLBAR_FIRST + 40;
const UINT uiLastUserToolBarId = uiFirstUserToolBarId + iMaxUserToolbars -
1;
BEGIN_MESSAGE_MAP(CMainFrame, CFrameWndEx)
	ON_WM_CREATE()
	ON_COMMAND(ID_VIEW_CUSTOMIZE,
&CMainFrame::OnViewCustomize)
	ON_REGISTERED_MESSAGE(AFX_WM_CREATETOOLBAR,
&CMainFrame::OnToolbarCreateNew)
END_MESSAGE_MAP()
static UINT indicators[] =
{
	ID_SEPARATOR,           // индикатор строки состояния
	ID_INDICATOR_CAPS,
	ID_INDICATOR_NUM,
	ID_INDICATOR_SCRL,
};
// создание/уничтожение CMainFrame
CMainFrame::CMainFrame()
{
}
CMainFrame::~CMainFrame()
{
}
int CMainFrame::OnCreate(LPCREATESTRUCT lpCreateStruct)
{
	if (CFrameWndEx::OnCreate(lpCreateStruct) == -1)
		return -1;
```

```
        BOOL bNameValid;
        if (!m_wndMenuBar.Create(this))
        {
            TRACE0("Не удалось создать строку меню\n");
            return -1;      // не удалось создать
        }
        m_wndMenuBar.SetPaneStyle(m_wndMenuBar.GetPaneStyle()
CBRS_SIZE_DYNAMIC | CBRS_TOOLTIPS | CBRS_FLYBY);
        // предотвращение фокусировки строки меню на активации
        CMFCPopupMenu::SetForceMenuFocus(FALSE);
        if (!m_wndToolBar.CreateEx(this, TBSTYLE_FLAT, WS_CHILD |
WS_VISIBLE | CBRS_TOP | CBRS_GRIPPER | CBRS_TOOLTIPS | CBRS_FLYBY |
CBRS_SIZE_DYNAMIC) ||
            !m_wndToolBar.LoadToolBar(theApp.m_bHiColorIcons ?
IDR_MAINFRAME_256 : IDR_MAINFRAME))
        {
            TRACE0("Не удалось создать панель инструментов\n");
            return -1;      // не удалось создать
        }
        CString strToolBarName;
        bNameValid =
strToolBarName.LoadString(IDS_TOOLBAR_STANDARD);
        ASSERT(bNameValid);
        m_wndToolBar.SetWindowText(strToolBarName);
        CString strCustomize;
        bNameValid = strCustomize.LoadString(IDS_TOOLBAR_CUSTOMIZE);
        ASSERT(bNameValid);
        m_wndToolBar.EnableCustomizeButton(TRUE,
ID_VIEW_CUSTOMIZE, strCustomize);
        // Разрешить операции с пользовательскими панелями инструментов:
```

```
InitUserToolbars(NULL, uiFirstUserToolBarId, uiLastUserToolBarId);
if (!m_wndStatusBar.Create(this))
{
	TRACE0("Не удалось создать строку состояния\n");
	return -1;	// не удалось создать
}
m_wndStatusBar.SetIndicators(indicators,
sizeof(indicators)/sizeof(UINT));

m_wndMenuBar.EnableDocking(CBRS_ALIGN_ANY);
m_wndToolBar.EnableDocking(CBRS_ALIGN_ANY);
EnableDocking(CBRS_ALIGN_ANY);
DockPane(&m_wndMenuBar);
DockPane(&m_wndToolBar);
CDockingManager::SetDockingMode(DT_SMART);
EnableAutoHidePanes(CBRS_ALIGN_ANY);
// Включить функцию замены меню панелей инструментов и
закрепляемых окон
EnablePaneMenu(TRUE, ID_VIEW_CUSTOMIZE, strCustomize,
ID_VIEW_TOOLBAR);
// включить быструю (Alt+перетаскивание) настройку панелей
инструментов
CMFCToolBar::EnableQuickCustomization();
if (CMFCToolBar::GetUserImages() == NULL)
{
	// загрузить изображения пользовательских панелей
инструментов
	if (m_UserImages.Load(_T(".\\UserImages.bmp")))
	{
		CMFCToolBar::SetUserImages(&m_UserImages);
```

```
                }
        }
        return 0;
}
BOOL CMainFrame::PreCreateWindow(CREATESTRUCT& cs)
{
        if( !CFrameWndEx::PreCreateWindow(cs) )
                return FALSE;
                return TRUE;
}
// диагностика CMainFrame
#ifdef _DEBUG
void CMainFrame::AssertValid() const
{
        CFrameWndEx::AssertValid();
}
void CMainFrame::Dump(CDumpContext& dc) const
{
        CFrameWndEx::Dump(dc);
}
#endif //_DEBUG
// обработчики сообщений CMainFrame
void CMainFrame::OnViewCustomize()
{
        CMFCToolBarsCustomizeDialog* pDlgCust = new
CMFCToolBarsCustomizeDialog(this, TRUE /* сканировать меню*/);
        pDlgCust->EnableUserDefinedToolbars();
        pDlgCust->Create();
}
LRESULT CMainFrame::OnToolbarCreateNew(WPARAM wp,LPARAM lp)
```

```
{
        LRESULT lres = CFrameWndEx::OnToolbarCreateNew(wp,lp);
        if (lres == 0)
        {
                return 0;
        }
        CMFCToolBar* pUserToolbar = (CMFCToolBar*)lres;
        ASSERT_VALID(pUserToolbar);

        BOOL bNameValid;
        CString strCustomize;
        bNameValid = strCustomize.LoadString(IDS_TOOLBAR_CUSTOMIZE);
        ASSERT(bNameValid);
        pUserToolbar->EnableCustomizeButton(TRUE, ID_VIEW_CUSTOMIZE,
strCustomize);
        return lres;
}
BOOL CMainFrame::LoadFrame(UINT nIDResource, DWORD dwDefaultStyle,
CWnd* pParentWnd, CCreateContext* pContext)
{
        // базовый класс не работает
        if      (!CFrameWndEx::LoadFrame(nIDResource,      dwDefaultStyle,
pParentWnd, pContext))
        {
                return FALSE;
        }
        // включить кнопку настройки для всех пользовательских панелей
инструментов
        BOOL bNameValid;
        CString strCustomize;
```

```
        bNameValid = strCustomize.LoadString(IDS_TOOLBAR_CUSTOMIZE);
        ASSERT(bNameValid);
        for (int i = 0; i < iMaxUserToolbars; i ++)
        {
            CMFCToolBar* pUserToolbar = GetUserToolBarByIndex(i);
            if (pUserToolbar != NULL)
            {
                pUserToolbar->EnableCustomizeButton(TRUE,
ID_VIEW_CUSTOMIZE, strCustomize);
            }
        }
        return TRUE;
    }
```

Файл “Objects.cpp”

```
#include "Objects.h"
CObjects::CObjects(void)
{
}
CObjects::~CObjects(void)
{
}
CObjects::CObjects(CString buff_inp, int mm, std::vector <CObjects> vars,
std::vector<CRazmer> rz, int strn,std::vector<CString> com_trans)
{
    CString buff_com;
    CString buff_arg;
    CString aname;
    command = buff_inp;
    p = 0;
```

```
tolshina = 2;
red = 0;
green = 0;
black = 0;
// Создание имен переопределенных команд
CString aprx, aprxy, comment, point, point_cc, point_to,
        line, razd, longline, bzr2, bzr2i, bzr3, bzr3i, move,
        turn, ravno, rgb, line_to, dopr, dopr1, vyt;

if (!com_trans.empty())
{
for (int i = 0; i < com_trans.size(); i++)
{
        if (com_trans[i].GetLength() > 6 &&
com_trans[i].Left(6).MakeLower() == "aprx->")
        {
                int z(0);
                int start(0);
                int arg(0);
                CString buff_str(com_trans[i]);
                for (int j = 0; j < buff_str.GetLength(); j++)
                {
                        if (buff_str[j] == ';')
                        {break;};
                        if (buff_str[j] == '>')
                        {z++; start = j+1; arg = 0;};
                        if (z == 1 && buff_str[j] != '>')
                        {
                                arg++;
                                aprx = buff_str.Mid(start,arg);
```

```
                    };
                };
            }
            else if (com_trans[i].GetLength() > 7 &&
com_trans[i].Left(7).MakeLower() == "aprxy->")
            {
                int z(0);
                int start(0);
                int arg(0);
                CString buff_str(com_trans[i]);
                for (int j = 0; j < buff_str.GetLength(); j++)
                {
                    if (buff_str[j] == ';')
                    {break;};
                    if (buff_str[j] == '>')
                    {z++; start = j+1; arg = 0;};
                    if (z == 1 && buff_str[j] != '>')
                    {
                        arg++;
                        aprxy = buff_str.Mid(start,arg);
                    };
                };
            }
            else if (com_trans[i].GetLength() > 7 &&
com_trans[i].Left(7).MakeLower() == "point->")
            {
                int z(0);
                int start(0);
                int arg(0);
                CString buff_str(com_trans[i]);
```

```
                for (int j = 0; j < buff_str.GetLength(); j++)
                {
                        if (buff_str[j] == ';')
                        {break;};
                        if (buff_str[j] == '>')
                        {z++; start = j+1; arg = 0;};
                        if (z == 1 && buff_str[j] != '>')
                        {
                                arg++;
                                point = buff_str.Mid(start,arg);
                        };
                };
        }
        else if (com_trans[i].GetLength() > 6 &&
com_trans[i].Left(6).MakeLower() == "line->")
        {
                int z(0);
                int start(0);
                int arg(0);
                CString buff_str(com_trans[i]);
                for (int j = 0; j < buff_str.GetLength(); j++)
                {
                        if (buff_str[j] == ';')
                        {break;};
                        if (buff_str[j] == '>')
                        {z++; start = j+1; arg = 0;};
                        if (z == 1 && buff_str[j] != '>')
                        {
                                arg++;
                                line = buff_str.Mid(start,arg);
```

```
                    };
                };
            }
            else if (com_trans[i].GetLength() > 10 &&
com_trans[i].Left(10).MakeLower() == "point_cc->")
            {
                int z(0);
                int start(0);
                int arg(0);
                CString buff_str(com_trans[i]);
                for (int j = 0; j < buff_str.GetLength(); j++)
                {
                    if (buff_str[j] == ';')
                    {break;};
                    if (buff_str[j] == '>')
                    {z++; start = j+1; arg = 0;};
                    if (z == 1 && buff_str[j] != '>')
                    {
                        arg++;
                        point_cc = buff_str.Mid(start,arg);
                    };
                };
            }
            else if (com_trans[i].GetLength() > 10 &&
com_trans[i].Left(10).MakeLower() == "point_to->")
            {
                int z(0);
                int start(0);
                int arg(0);
                CString buff_str(com_trans[i]);
```

```
                for (int j = 0; j < buff_str.GetLength(); j++)
                {
                        if (buff_str[j] == ';')
                        {break;};
                        if (buff_str[j] == '>')
                        {z++; start = j+1; arg = 0;};
                        if (z == 1 && buff_str[j] != '>')
                        {
                                arg++;
                                point_to = buff_str.Mid(start,arg);
                        };
                };
        }
        else if (com_trans[i].GetLength() > 6 &&
com_trans[i].Left(6).MakeLower() == "long->")
        {
                int z(0);
                int start(0);
                int arg(0);
                CString buff_str(com_trans[i]);
                for (int j = 0; j < buff_str.GetLength(); j++)
                {
                        if (buff_str[j] == ';')
                        {break;};
                        if (buff_str[j] == '>')
                        {z++; start = j+1; arg = 0;};
                        if (z == 1 && buff_str[j] != '>')
                        {
                                arg++;
                                longline = buff_str.Mid(start,arg);
```

```
                    };
                };
            }
            else if (com_trans[i].GetLength() > 6 &&
com_trans[i].Left(6).MakeLower() == "bzr2->")
            {
                int z(0);
                int start(0);
                int arg(0);
                CString buff_str(com_trans[i]);
                for (int j = 0; j < buff_str.GetLength(); j++)
                {
                    if (buff_str[j] == ';')
                    {break;};
                    if (buff_str[j] == '>')
                    {z++; start = j+1; arg = 0;};
                    if (z == 1 && buff_str[j] != '>')
                    {
                        arg++;
                        bzr2 = buff_str.Mid(start,arg);
                    };
                };
            }
            else if (com_trans[i].GetLength() > 7 &&
com_trans[i].Left(7).MakeLower() == "bzr2i->")
            {
                int z(0);
                int start(0);
                int arg(0);
                CString buff_str(com_trans[i]);
```

```
                for (int j = 0; j < buff_str.GetLength(); j++)
                {
                        if (buff_str[j] == ';')
                        {break;};
                        if (buff_str[j] == '>')
                        {z++; start = j+1; arg = 0;};
                        if (z == 1 && buff_str[j] != '>')
                        {
                                arg++;
                                bzr2i = buff_str.Mid(start,arg);
                        };
                };
        }
        else if (com_trans[i].GetLength() > 6 &&
com_trans[i].Left(6).MakeLower() == "bzr3->")
        {
                int z(0);
                int start(0);
                int arg(0);
                CString buff_str(com_trans[i]);
                for (int j = 0; j < buff_str.GetLength(); j++)
                {
                        if (buff_str[j] == ';')
                        {break;};
                        if (buff_str[j] == '>')
                        {z++; start = j+1; arg = 0;};
                        if (z == 1 && buff_str[j] != '>')
                        {
                                arg++;
                                bzr3 = buff_str.Mid(start,arg);
```

```
                    };
                };
            }
            else if (com_trans[i].GetLength() > 7 &&
com_trans[i].Left(7).MakeLower() == "bzr3i->")
            {
                int z(0);
                int start(0);
                int arg(0);
                CString buff_str(com_trans[i]);
                for (int j = 0; j < buff_str.GetLength(); j++)
                {
                    if (buff_str[j] == ';')
                    {break;};
                    if (buff_str[j] == '>')
                    {z++; start = j+1; arg = 0;};
                    if (z == 1 && buff_str[j] != '>')
                    {
                        arg++;
                        bzr3i = buff_str.Mid(start,arg);
                    };
                };
            }
            else if (com_trans[i].GetLength() > 6 &&
com_trans[i].Left(6).MakeLower() == "move->")
            {
                int z(0);
                int start(0);
                int arg(0);
                CString buff_str(com_trans[i]);
```

```
                for (int j = 0; j < buff_str.GetLength(); j++)
                {
                        if (buff_str[j] == ';')
                        {break;};
                        if (buff_str[j] == '>')
                        {z++; start = j+1; arg = 0;};
                        if (z == 1 && buff_str[j] != '>')
                        {
                                arg++;
                                move = buff_str.Mid(start,arg);
                        };
                };
        }
        else if (com_trans[i].GetLength() > 6 &&
com_trans[i].Left(6).MakeLower() == "turn->")
        {
                int z(0);
                int start(0);
                int arg(0);
                CString buff_str(com_trans[i]);
                for (int j = 0; j < buff_str.GetLength(); j++)
                {
                        if (buff_str[j] == ';')
                        {break;};
                        if (buff_str[j] == '>')
                        {z++; start = j+1; arg = 0;};
                        if (z == 1 && buff_str[j] != '>')
                        {
                                arg++;
                                turn = buff_str.Mid(start,arg);
```

```
                    };
                };
            }
            else if (com_trans[i].GetLength() > 3 &&
com_trans[i].Left(3).MakeLower() == "=->")
            {
                int z(0);
                int start(0);
                int arg(0);
                CString buff_str(com_trans[i]);
                for (int j = 0; j < buff_str.GetLength(); j++)
                {
                    if (buff_str[j] == ';')
                    {break;};
                    if (buff_str[j] == '>')
                    {z++; start = j+1; arg = 0;};
                    if (z == 1 && buff_str[j] != '>')
                    {
                        arg++;
                        ravno = buff_str.Mid(start,arg);
                    };
                };
            }
            else if (com_trans[i].GetLength() > 9 &&
com_trans[i].Left(9).MakeLower() == "line_to->")
            {
                int z(0);
                int start(0);
                int arg(0);
                CString buff_str(com_trans[i]);
```

```
                for (int j = 0; j < buff_str.GetLength(); j++)
                {
                        if (buff_str[j] == ';')
                        {break;};
                        if (buff_str[j] == '>')
                        {z++; start = j+1; arg = 0;};
                        if (z == 1 && buff_str[j] != '>')
                        {
                                arg++;
                                line_to = buff_str.Mid(start,arg);
                        };
                };
        }
        else if (com_trans[i].GetLength() > 6 &&
com_trans[i].Left(6).MakeLower() == "dopr->")
        {
                int z(0);
                int start(0);
                int arg(0);
                CString buff_str(com_trans[i]);
                for (int j = 0; j < buff_str.GetLength(); j++)
                {
                        if (buff_str[j] == ';')
                        {break;};
                        if (buff_str[j] == '>')
                        {z++; start = j+1; arg = 0;};
                        if (z == 1 && buff_str[j] != '>')
                        {
                                arg++;
                                dopr = buff_str.Mid(start,arg);
```

```
                    };
                };
            }
            else if (com_trans[i].GetLength() > 7 &&
com_trans[i].Left(7).MakeLower() == "dopr1->")
            {
                int z(0);
                int start(0);
                int arg(0);
                CString buff_str(com_trans[i]);
                for (int j = 0; j < buff_str.GetLength(); j++)
                {
                    if (buff_str[j] == ';')
                    {break;};
                    if (buff_str[j] == '>')
                    {z++; start = j+1; arg = 0;};
                    if (z == 1 && buff_str[j] != '>')
                    {
                        arg++;
                        dopr1 = buff_str.Mid(start,arg);
                    };
                };
            }
            else if (com_trans[i].GetLength() > 5 &&
com_trans[i].Left(5).MakeLower() == "vyt->")
            {
                int z(0);
                int start(0);
                int arg(0);
                CString buff_str(com_trans[i]);
```

```
                    for (int j = 0; j < buff_str.GetLength(); j++)
                    {
                        if (buff_str[j] == ';')
                        {break;};
                        if (buff_str[j] == '>')
                        {z++; start = j+1; arg = 0;};
                        if (z == 1 && buff_str[j] != '>')
                        {
                            arg++;
                            vyt = buff_str.Mid(start,arg);
                        };
                    };
                }
            };
            };
            // Формирование имени
            for (int j = 0; j < buff_inp.GetLength(); j++)
            {
                if (buff_inp[j] == ':')
                {
                    aname = buff_inp.Mid(0, j);
                    buff_com = buff_inp.Mid(j+1, buff_inp.GetLength()-j);
                    break;
               };
            };
            // КОМАНДЫ
            if (buff_com.Left(4).MakeLower() == "aprx" &&
buff_com.Left(5).MakeLower() != "aprxy"
                || !aprx.IsEmpty() && aprx.GetLength() < buff_com.GetLength()
&& buff_com.Left(aprx.GetLength()).MakeLower() == aprx)
```

```
{
        str = strn;
        name = aname;
        sx2 = "10000";
        sx3 = "1";
        int z(0);
        int start(0);
        int arg(0);

                for (int i = 0; ; i++)
                {
                        if (buff_com[i] == ';')
                        { break; };
                        if (z == 0)
                        { fun += buff_com[i]; };
                        if (buff_com[i] == '>' || buff_com[i] == ',')
                        {z++; start = i+1; arg = 0;};
                        if (z == 1 && buff_com[i] != '>') // Первый аргумент
                        {
                                arg++;
                                sx0 = buff_com.Mid(start,arg);
                        }
                        else if (z == 2 && buff_com[i] != ',')
                        {
                                arg++;
                                sx1 = buff_com.Mid(start,arg);
                        }
                        else if (z == 3 && buff_com[i] != ',')
                        {
                                arg++;
```

```
                    sx2 = buff_com.Mid(start,arg);
                }
                else if (z == 4 && buff_com[i] != ',')
                {
                    arg++;
                    sx3 = buff_com.Mid(start,arg);
                }
            };

            if (sx2.Left(1) == "!" && sx2[1] != '!')
            {
                st = 130;
                sx2 = sx2.Right(sx2.GetLength()-1);
            }
            else if (sx2.Left(2) == "!!")
            {
                st = 132;
                sx2 = sx2.Right(sx2.GetLength()-2);
            }
            else
            {
                st = 101;
            };
            a = converter(sx0, rz, vars);
            b = converter(sx1, rz, vars);
            rapapr = converter(sx3, rz, vars);
            maxsb = converter(sx2, rz, vars);
        }
        else if (buff_com.Left(5).MakeLower() == "aprxy"
```

```
                    || !aprxy.IsEmpty() && aprxy.GetLength() < buff_com.GetLength() && buff_com.Left(aprxy.GetLength()).MakeLower() == aprxy)
                    {
                        str = strn;
                        name = aname;
                        sx2 = "10000";
                        sx3 = "1";
                        int z(0);
                        int start(0);
                        int arg(0);
                        for (int i = 0; ; i++)
                        {
                            if (buff_com[i] == ';')
                            { break; };
                            if (z == 0)
                            { fun += buff_com[i]; };
                            if (buff_com[i] == '>' || buff_com[i] == ',')
                            {z++; start = i+1; arg = 0;};
                            if (z == 1 && buff_com[i] != '>') // Первый аргумент
                            {
                                arg++;
                                sy1 = buff_com.Mid(start,arg);
                            }
                            else if (z == 2 && buff_com[i] != ',')
                            {
                                arg++;
                                sy2 = buff_com.Mid(start,arg);
                            }
                            else if (z == 3 && buff_com[i] != ',') // Первый аргумент
```

```
        {
            arg++;
            sx0 = buff_com.Mid(start,arg);
        }
        else if (z == 4 && buff_com[i] != ',')
        {
            arg++;
            sx1 = buff_com.Mid(start,arg);
        }
        else if (z == 5 && buff_com[i] != ',')
        {
            arg++;
            sx2 = buff_com.Mid(start,arg);
        }
        else if (z == 6 && buff_com[i] != ',')
        {
            arg++;
            sx3 = buff_com.Mid(start,arg);
        }
};
if (sx2.Left(1) == "!" && sx2[1] != '!')
{
        st = 131;
        sx2 = sx2.Right(sx2.GetLength()-1);
}
else if (sx2.Left(2) == "!!")
{
        st = 133;
        sx2 = sx2.Right(sx2.GetLength()-2);
}
```

```
                else
                {
                        st = 102;
                };
                a = converter(sx0, rz, vars);
                b = converter(sx1, rz, vars);
                y1 = converter(sy1, rz, vars);
                y2 = converter(sy2, rz, vars);
                maxsb = converter(sx2, rz, vars);
                rapapr = converter(sx3, rz, vars);

        }
        // Формирование параметров построения объектов
    // ТОЧКА
        else if (buff_com.Left(6).MakeLower() == "point(" ||
buff_com.Left(6).MakeLower() == "point>"
                || !point.IsEmpty() && point.GetLength() <
buff_com.GetLength() && buff_com.Left(point.GetLength()).MakeLower() == point)
        {
                str = strn;
                name = aname;
                p = 1;
                st = 1000;
                int z(0);
                int start(0);
                int arg(0);
                for (int i = 0; ; i++)
                {
                        if (buff_com[i] == ';')
                        { break; };
```

```
                    if (z == 0)
                    { fun += buff_com[i]; };
                    if (buff_com[i] == '>' || buff_com[i] == ',')
                    {z++; start = i+1; arg = 0;};
                    if (z == 1 && buff_com[i] != '>') // Первый аргумент
                    {
                        arg++;
                        sx0 = buff_com.Mid(start,arg);
                    }
                    else if (z == 2 && buff_com[i] != ',')
                    {
                        arg++;
                        sy0 = buff_com.Mid(start,arg);
                    };
                };
                x0 = converter(sx0, rz, vars);
                y0 = converter(sy0, rz, vars);
            }
            else if (buff_com.Left(9).MakeLower() == "point_cc(" ||
buff_com.Left(9).MakeLower() == "point_cc>"
                || !point_cc.IsEmpty() && point_cc.GetLength() <
buff_com.GetLength() && buff_com.Left(point_cc.GetLength()).MakeLower() ==
point_cc)
            {
                str = strn;
                name = aname;
                p = 1;
                st = 1000;
                int z(0);
                int start(0);
```

```
int arg(0);
for (int i = 0; ; i++)
{
        if (buff_com[i] == ';')
        { break; };
        if (z == 0)
        { fun += buff_com[i]; };
        if (buff_com[i] == '>' || buff_com[i] == ',')
        {z++; start = i+1; arg = 0;};
        if (z == 1 && buff_com[i] != '>') // Первый аргумент
        {
                arg++;
                sx1 = buff_com.Mid(start,arg);
        }
        else if (z == 2 && buff_com[i] != ',')
        {
                arg++;
                sy1 = buff_com.Mid(start,arg);
        }
        else if (z == 3 && buff_com[i] != ',')
        {
                arg++;
                sa1 = buff_com.Mid(start,arg);
        }
        else if (z == 4 && buff_com[i] != ',')
        {
                arg++;
                sx2 = buff_com.Mid(start,arg);
        }
        else if (z == 5 && buff_com[i] != ',')
```

```
                {
                        arg++;
                        sy2 = buff_com.Mid(start,arg);
                }
                else if (z == 6 && buff_com[i] != ',')
                {
                        arg++;
                        sa2 = buff_com.Mid(start,arg);
                }
                else if (z == 7 && buff_com[i] != ',')
                {
                        arg++;
                        st1 = buff_com.Mid(start,arg);
                };
        };
        x1 = converter(sx1, rz, vars); // ц о1
        y1 = converter(sy1, rz, vars); // ц о1
        x2 = converter(sx2, rz, vars); // ц о2
        y2 = converter(sy2, rz, vars); // ц о2
        a1 = converter(sa1, rz, vars); // р о1
        a2 = converter(sa2, rz, vars); // р о2
        t1 = converter(st1, rz, vars); // т о1о2
        double prov(0);
        prov = sqrt((x1-x2)*(x1-x2)+(y1-y2)*(y1-y2));
        if (prov > a1+a2 && prov < abs(a1-a2))
        {
                AfxMessageBox(CString("Нет пересечений!"),
MB_OK);
                x0 = 0;
                y0 = 0;
```

```
                }
                else
                {
                        double asd(0), hsd(0);
                        asd = (a1*a1-a2*a2+prov*prov)/(2*prov);
                        hsd = sqrt(a1*a1-asd*asd);
                        x0 = x2+t1*hsd*(y2-y1)/prov;
                        y0 = y2-t1*hsd*(x2-x1)/prov;
                };
        }
        else if (buff_com.Left(9).MakeLower() == "point_to(" ||
buff_com.Left(9).MakeLower() == "point_to>"
                || !point_to.IsEmpty() && point_to.GetLength() <
buff_com.GetLength() && buff_com.Left(point_to.GetLength()).MakeLower() ==
point_to)
        {
                str = strn;
                name = aname;
                p = 1;
                st = 1001;
                int z(0);
                sx1 = "p0";
                int start(0);
                int arg(0);
                for (int i = 0; ; i++)
                {
                        if (buff_com[i] == ';')
                        { break; };
                        if (z == 0)
                        { fun += buff_com[i]; };
```

```
                if (buff_com[i] == '>' || buff_com[i] == ',')
                {z++; start = i+1; arg = 0;};
                if (z == 1 && buff_com[i] != '>') // Первый аргумент
                {
                        arg++;
                        sx2 = buff_com.Mid(start,arg);
                }
                else if (z == 2 && buff_com[i] != ',')
                {
                        arg++;
                        sy2 = buff_com.Mid(start,arg);
                }
                else if (z == 3 && buff_com[i] != ',')
                {
                        arg++;
                        sa1 = buff_com.Mid(start,arg).MakeLower();
                }
        };
        for (int j = 0; j < vars.size(); j++)
        {
                if (sx2 == vars[j].name)
                {
                        if (vars[j].st == 1)
                        {
                                double line(0);
                                line = sqrt((vars[j].x0 -
vars[j].x1)*(vars[j].x0 - vars[j].x1)+(vars[j].y0 - vars[j].y1)
                                        *(vars[j].y0 - vars[j].y1));
                                if (sa1 == "p1")
                                { t = 1 - converter(sy2, rz, vars) / line;}
```

```
                else
                {t = converter(sy2, rz, vars) / line;};
                if (t > 1 || t < 0)
                {

AfxMessageBox(CString("Откладываемая длина больше длины отрезка!"), MB_OK);
                        t = 0.5;
                };
                x0 = (1. - t) * vars[j].x0 + t * vars[j].x1;
                y0 = (1. - t) * vars[j].y0 + t * vars[j].y1;
                break;
        }
        else if (vars[j].st == 2)
        {
                double line(0);
                if (sa1 == "p2")
                {
                        a1 = vars[j].x2;
                        a3 = vars[j].y2;
                        a2 = a4 = 0;
                        for (double tt = 0.999; tt >= 0; tt -= 0.001)
                        {
                                a2 = (1.-tt)*(1.-tt)*vars[j].x0+2*tt*(1.-tt)*vars[j].x1+tt*tt*vars[j].x2;
                                a4 = (1.-tt)*(1.-tt)*vars[j].y0+2*tt*(1.-tt)*vars[j].y1+tt*tt*vars[j].y2;
                                line += sqrt((a2-a1)*(a2-a1)+(a4-a3)*(a4-a3));
```

```
                    a1 = a2;
                    a3 = a4;
                    if
((ceil(line*100)/100)/(ceil(converter(sy2, rz, vars)*100)/100) == 1)
                    {
                        x0 = a2;
                        y0 = a4;
                        break;
                    };
                };
            }
            else
            {
                a1 = vars[j].x0;
                a3 = vars[j].y0;
                a2 = a4 = 0;
                for (double tt = 0.001; tt <= 1; tt +=
0.001)
                {
                    a2 = (1.-tt)*(1.-
tt)*vars[j].x0+2*tt*(1.-tt)*vars[j].x1+tt*tt*vars[j].x2;
                    a4 = (1.-tt)*(1.-
tt)*vars[j].y0+2*tt*(1.-tt)*vars[j].y1+tt*tt*vars[j].y2;
                    line += sqrt((a2-a1)*(a2-
a1)+(a4-a3)*(a4-a3));
                    a1 = a2;
                    a3 = a4;
                    if
((ceil(line*100)/100)/(ceil(converter(sy2, rz, vars)*100)/100) == 1)
                    {
```

```
                                x0 = a2;
                                y0 = a4;
                                break;
                            };
                        };
                    };

                    break;
                }
                else if (vars[j].st == 3)
                {
                    double line(0);

                    if (sa1 == "p3")
                    {
                        a1 = vars[j].x3;
                        a3 = vars[j].y3;
                        a2 = a4 = 0;
                        for (double tt = 0.999; tt >= 0; tt -=
0.001)
                        {
                            a2 = (1-tt)*(1-tt)*(1-
tt)*vars[j].x0+3*tt*(1-tt)*(1-tt)*vars[j].x1
                                +3*tt*tt*(1-
tt)*vars[j].x2+tt*tt*tt*vars[j].x3;
                            a4 = (1-tt)*(1-tt)*(1-
tt)*vars[j].y0+3*tt*(1-tt)*(1-tt)*vars[j].y1
                                +3*tt*tt*(1-
tt)*vars[j].y2+tt*tt*tt*vars[j].y3;
```

```
                line += sqrt((a2-a1)*(a2-a1)+(a4-a3)*(a4-a3));
                a1 = a2;
                a3 = a4;
                if (floor(line)/floor(converter(sy2, rz, vars)) == 1 )
                {
                    x0 = a2;
                    y0 = a4;
                    break;
                };
            };
        }
        else
        {
            a1 = vars[j].x0;
            a3 = vars[j].y0;
            a2 = a4 = 0;
            for (double tt = 0.001; tt <= 1; tt += 0.001)
            {
                a2 = (1-tt)*(1-tt)*(1-tt)*vars[j].x0+3*tt*(1-tt)*(1-tt)*vars[j].x1
                    +3*tt*tt*(1-tt)*vars[j].x2+tt*tt*tt*vars[j].x3;
                a4 = (1-tt)*(1-tt)*(1-tt)*vars[j].y0+3*tt*(1-tt)*(1-tt)*vars[j].y1
                    +3*tt*tt*(1-tt)*vars[j].y2+tt*tt*tt*vars[j].y3;
```

```
                                                    line += sqrt((a2-a1)*(a2-
a1)+(a4-a3)*(a4-a3));
                                                    a1 = a2;
                                                    a3 = a4;
                                                    if
(floor(line)/floor(converter(sy2, rz, vars)) == 1)
                                                    {
                                                        x0 = a2;
                                                        y0 = a4;
                                                        break;
                                                    };
                                                };
                                            };
                                            break;
                                        };
                                    };
                                };
                            }
                        // ОТРЕЗОК
                            else if (buff_com.Left(5).MakeLower() == "line(" ||
buff_com.Left(5).MakeLower() == "line>"
                                || !line.IsEmpty() && line.GetLength() <
buff_com.GetLength() && buff_com.Left(line.GetLength()).MakeLower() == line)
                            {
                                str = strn;
                                name = aname;
                                p = 1;
                                st = 1;
                                int z(0);
```

```
int start(0);
int arg(0);
x3 = 1;
for (int i = 0; ; i++)
{
	if (buff_com[i] == ';')
	{ break; };
	if (z == 0)
	{ fun += buff_com[i]; };
	if (buff_com[i] == '>' || buff_com[i] == ',')
	{z++; start = i+1; arg = 0;};
	if (z == 1 && buff_com[i] != '>') // Первый аргумент
	{
		arg++;
		sx0 = buff_com.Mid(start,arg);
	}
	else if (z == 2 && buff_com[i] != ',')
	{
		arg++;
		sy0 = buff_com.Mid(start,arg);
	}
	else if (z == 3 && buff_com[i] != ',')
	{
		arg++;
		sx1 = buff_com.Mid(start,arg);
	}
	else if (z == 4 && buff_com[i] != ',')
	{
		arg++;
		sy1 = buff_com.Mid(start,arg);
```

```
                }
            };
            x0 = converter(sx0, rz, vars);
            x1 = converter(sx1, rz, vars);
            y0 = converter(sy0, rz, vars);
            y1 = converter(sy1, rz, vars);
            if (y0 < y1)  {p = -1;}
            else if (y0 > y1) {p = 1;}
            else {p = 0;};
        }
    // НЕВИДИМЫЙ ОТРЕЗОК
        else if (buff_com.Left(4).MakeLower() == "long"
            || !longline.IsEmpty() && longline.GetLength() < buff_com.GetLength() && buff_com.Left(longline.GetLength()).MakeLower() == longline)
        {
            str = strn;
            name = aname;
            p = 1;
            st = 4;
            int z(0);
            int start(0);
            int arg(0);
            for (int i = 0; ; i++)
            {
                if (buff_com[i] == ';')
                { break; };
                if (z == 0)
                { fun += buff_com[i]; };
                if (buff_com[i] == '>' || buff_com[i] == ',')
```

```
                {z++; start = i+1; arg = 0;};
                if (z == 1 && buff_com[i] != '>') // Первый аргумент
                {
                        arg++;
                        sx0 = buff_com.Mid(start,arg);
                }
                else if (z == 2 && buff_com[i] != ',')
                {
                        arg++;
                        sy0 = buff_com.Mid(start,arg);
                }
                else if (z == 3 && buff_com[i] != ',')
                {
                        arg++;
                        sx1 = buff_com.Mid(start,arg);
                }
                else if (z == 4 && buff_com[i] != ',')
                {
                        arg++;
                        sy1 = buff_com.Mid(start,arg);
                }
        };
        x0 = converter(sx0, rz, vars);
        x1 = converter(sx1, rz, vars);
        y0 = converter(sy0, rz, vars);
        y1 = converter(sy1, rz, vars);
        if (y0 < y1)  {p = -1;}
        else if (y0 > y1) {p = 1;}
        else {p = 0;};
}
```

```
// Кривая Безье 2-ой степени (чистое построение)
else if (buff_com.Left(5).MakeLower() == "bzr2(" ||
buff_com.Left(5).MakeLower() == "bzr2>"
|| !bzr2.IsEmpty() && bzr2.GetLength() <
buff_com.GetLength() && buff_com.Left(bzr2.GetLength()).MakeLower() == bzr2)
{
str = strn;
name = aname;
p = 1;
st = 2;
int z(0);
int start(0);
int arg(0);
for (int i = 0; ; i++)
{
if (buff_com[i] == ';')
{ break; };

if (z == 0)
{ fun += buff_com[i]; };
if (buff_com[i] == '>' || buff_com[i] == ',')
{z++; start = i+1; arg = 0;};
if (z == 1 && buff_com[i] != '>') // Первый аргумент
{
arg++;
sx0 = buff_com.Mid(start,arg);
}
else if (z == 2 && buff_com[i] != ',')
{
arg++;
```

```
                sy0 = buff_com.Mid(start,arg);
        }
        else if (z == 3 && buff_com[i] != ',')
        {
                arg++;
                sx1 = buff_com.Mid(start,arg);
        }
        else if (z == 4 && buff_com[i] != ',')
        {
                arg++;
                sy1 = buff_com.Mid(start,arg);
        }
        else if (z == 5 && buff_com[i] != ',')
        {
                arg++;
                sx2 = buff_com.Mid(start,arg);
        }
        else if (z == 6 && buff_com[i] != ',')
        {
                arg++;
                sy2 = buff_com.Mid(start,arg);
        };
};
x0 = converter(sx0, rz, vars);
y0 = converter(sy0, rz, vars);
x1 = converter(sx1, rz, vars);
y1 = converter(sy1, rz, vars);
x2 = converter(sx2, rz, vars);
y2 = converter(sy2, rz, vars);
if (y0 < y2)  {p = -1;}
```

```
                else if (y0 > y2) {p = 1;}
                else {p = 0;};
            }
            // Кривая Безье 2-ой степени (интерполирование)
            else if (buff_com.Left(6).MakeLower() == "bzr2i(" ||
buff_com.Left(6).MakeLower() == "bzr2i>"
                || !bzr2i.IsEmpty() && bzr2i.GetLength() <
buff_com.GetLength() && buff_com.Left(bzr2i.GetLength()).MakeLower() == bzr2i)
            {
                str = strn;
                name = aname;
                p = 1;
                st = 2;
                int z(0);
                int start(0);
                int arg(0);
                for (int i = 0; ; i++)
                {
                    if (buff_com[i] == ';')
                    { break; };
                    if (z == 0)
                    { fun += buff_com[i]; };
                    if (buff_com[i] == '>' || buff_com[i] == ',')
                    {z++; start = i+1; arg = 0;};
                    if (z == 1 && buff_com[i] != '>') // Первый аргумент
                    {
                        arg++;
                        sx0 = buff_com.Mid(start,arg);
                    }
                    else if (z == 2 && buff_com[i] != ',')
```

```
        {
            arg++;
            sy0 = buff_com.Mid(start,arg);
        }
        else if (z == 3 && buff_com[i] != ',')
        {
            arg++;
            sx2 = buff_com.Mid(start,arg);
        }
        else if (z == 4 && buff_com[i] != ',')
        {
            arg++;
            sy2 = buff_com.Mid(start,arg);
        }
        else if (z == 5 && buff_com[i] != ',')
        {
            arg++;
            sa1 = buff_com.Mid(start,arg);
        }
        else if (z == 6 && buff_com[i] != ',')
        {
            arg++;
            sa2 = buff_com.Mid(start,arg);
        }
        else if (z == 7 && buff_com[i] != ',')
        {
            arg++;
            st1 = buff_com.Mid(start,arg);
        };
};
```

```
x0 = converter(sx0, rz, vars);
y0 = converter(sy0, rz, vars);
a1 = converter(sa1, rz, vars);
a2 = converter(sa2, rz, vars);
x2 = converter(sx2, rz, vars);
y2 = converter(sy2, rz, vars);
t = converter(st1, rz, vars);
x1 = (a1 - (1 - t) * (1 - t) * x0 - t * t * x2) / (2 * t * (1 - t));
y1 = (a2 - (1 - t) * (1 - t) * y0 - t * t * y2) / (2 * t * (1 - t));
sx1.Format(_T("%.2f"),x1);
sy1.Format(_T("%.2f"),y1);
if (y0 < y2)  {p = -1;}
else if (y0 > y2) {p = 1;}
else {p = 0;};
}


// Кривая Безье 3-й степени (чистое построение)
else if (buff_com.Left(5).MakeLower() == "bzr3(" ||
buff_com.Left(5).MakeLower() == "bzr3>"
|| !bzr3.IsEmpty() && bzr3.GetLength() <
buff_com.GetLength() && buff_com.Left(bzr3.GetLength()).MakeLower() == bzr3)
{
str = strn;
name = aname;
p = 1;
st = 3;
int z(0);
int start(0);
```

```
int arg(0);
for (int i = 0; ; i++)
{
	if (buff_com[i] == ';')
	{ break; };
	if (z == 0)
	{ fun += buff_com[i]; };
	if (buff_com[i] == '>' || buff_com[i] == ',')
	{z++; start = i+1; arg = 0;};
	if (z == 1 && buff_com[i] != '>') // Первый аргумент
	{
		arg++;
		sx0 = buff_com.Mid(start,arg);
	}
	else if (z == 2 && buff_com[i] != ',')
	{
		arg++;
		sy0 = buff_com.Mid(start,arg);
	}
	else if (z == 3 && buff_com[i] != ',')
	{
		arg++;
		sx1 = buff_com.Mid(start,arg);
	}
	else if (z == 4 && buff_com[i] != ',')
	{
		arg++;
		sy1 = buff_com.Mid(start,arg);
	}
	else if (z == 5 && buff_com[i] != ',')
```

```
        {
            arg++;
            sx2 = buff_com.Mid(start,arg);
        }
        else if (z == 6 && buff_com[i] != ',')
        {
            arg++;
            sy2 = buff_com.Mid(start,arg);
        }
        else if (z == 7 && buff_com[i] != ',')
        {
            arg++;
            sx3 = buff_com.Mid(start,arg);
        }
        else if (z == 8 && buff_com[i] != ',')
        {
            arg++;
            sy3 = buff_com.Mid(start,arg);
        };
};
x0 = converter(sx0, rz, vars);
y0 = converter(sy0, rz, vars);
x1 = converter(sx1, rz, vars);
y1 = converter(sy1, rz, vars);
x2 = converter(sx2, rz, vars);
y2 = converter(sy2, rz, vars);
x3 = converter(sx3, rz, vars);
y3 = converter(sy3, rz, vars);
if (y0 < y3)  {p = -1;}
else if (y0 > y3) {p = 1;}
```

```
                else {p = 0;};
            }
            // Кривая Безье 3-й степени (интерполирование)
            else if (buff_com.Left(6).MakeLower() == "bzr3i(" ||
buff_com.Left(6).MakeLower() == "bzr3i>"
                || !bzr3i.IsEmpty() && bzr3i.GetLength() <
buff_com.GetLength() && buff_com.Left(bzr3i.GetLength()).MakeLower() == bzr3i)
            {
                str = strn;
                name = aname;
                p = 1;
                st = 3;
                int z(0);
                int start(0);
                int arg(0);
                for (int i = 0; ; i++)
                {
                    if (buff_com[i] == ';')
                    { break; };
                    if (z == 0)
                    { fun += buff_com[i]; };
                    if (buff_com[i] == '>' || buff_com[i] == ',')
                    {z++; start = i+1; arg = 0;};
                    if (z == 1 && buff_com[i] != '>') // Первый аргумент
                    {
                        arg++;
                        sx0 = buff_com.Mid(start,arg);
                    }
                    else if (z == 2 && buff_com[i] != ',')
                    {
```

```
            arg++;
            sy0 = buff_com.Mid(start,arg);
        }
        else if (z == 3 && buff_com[i] != ',')
        {
            arg++;
            sx3 = buff_com.Mid(start,arg);
        }
        else if (z == 4 && buff_com[i] != ',')
        {
            arg++;
            sy3 = buff_com.Mid(start,arg);
        }
        else if (z == 5 && buff_com[i] != ',')
        {
            arg++;
            sa1 = buff_com.Mid(start,arg);
        }
        else if (z == 6 && buff_com[i] != ',')
        {
            arg++;
            sa2 = buff_com.Mid(start,arg);
        }
        else if (z == 7 && buff_com[i] != ',')
        {
            arg++;
            sa3 = buff_com.Mid(start,arg);
        }
        else if (z == 8 && buff_com[i] != ',')
        {
```

```
                arg++;
                sa4 = buff_com.Mid(start,arg);
            }
            else if (z == 9 && buff_com[i] != ',')
            {
                arg++;
                st1 = buff_com.Mid(start,arg);
            }
            else if (z == 10 && buff_com[i] != ',')
            {
                arg++;
                st2 = buff_com.Mid(start,arg);
            };
        };
        x0 = converter(sx0, rz, vars);
        y0 = converter(sy0, rz, vars);
        a1 = converter(sa1, rz, vars);
        a2 = converter(sa2, rz, vars);
        a3 = converter(sa3, rz, vars);
        a4 = converter(sa4, rz, vars);
        x3 = converter(sx3, rz, vars);
        y3 = converter(sy3, rz, vars);
        t = converter(st1, rz, vars);
        t1 = converter(st2, rz, vars);
    double xa(a1), ya(a2), xb(a3), yb(a4), ta(t), tb(t1);
    double aax(0), bax(0), cax(0);
    double abx(0), bbx(0), cbx(0);
    double detx(0), detx1(0), detx2(0);
    double aay(0), bay(0), cay(0);
    double aby(0), bby(0), cby(0);
```

```
double dety(0), dety1(0), dety2(0);
// Расчет координат промежуточных опорных вершин
aax = xa-(1.-ta)*(1.-ta)*(1.-ta)*x0-ta*ta*ta*x3;
bax = 3.*ta*(1.-ta)*(1.-ta);
cax = 3.*ta*ta*(1.-ta);
abx = xb-(1.-tb)*(1.-tb)*(1.-tb)*x0-tb*tb*tb*x3;
bbx = 3.*tb*(1.-tb)*(1.-tb);
cbx = 3.*tb*tb*(1.-tb);
detx=bax*cbx-cax*bbx; // по правилу Крамера
detx1=aax*cbx-abx*cax;
detx2=bax*abx-aax*bbx;
x1=detx1/detx; // абсциссы
x2=detx2/detx;
aay = ya-(1.-ta)*(1.-ta)*(1.-ta)*y0-ta*ta*ta*y3;
bay = 3.*ta*(1.-ta)*(1.-ta);
cay = 3.*ta*ta*(1.-ta);
aby = yb-(1.-tb)*(1.-tb)*(1.-tb)*y0-tb*tb*tb*y3;
bby = 3.*tb*(1.-tb)*(1.-tb);
cby = 3.*tb*tb*(1.-tb);
dety=bay*cby-cay*bby; // по правилу Крамера
dety1=aay*cby-aby*cay;
dety2=bay*aby-aay*bby;
y1=dety1/dety; // ординаты
y2=dety2/dety;
sx1.Format(_T("%.2f"),x1);
sy1.Format(_T("%.2f"),y1);
sx2.Format(_T("%.2f"),x2);
sy2.Format(_T("%.2f"),y2);
if (y0 < y3)  {p = -1;}
else if (y0 > y3) {p = 1;}
```

```
else {p = 0;};
}
// КОМАНДЫ
// Перемещение объекта
//по вектору, пущенному из начальной опорной вершины
else if (buff_com.Left(4).MakeLower() == "move"
    || !move.IsEmpty() && move.GetLength() < buff_com.GetLength() && buff_com.Left(move.GetLength()).MakeLower() == move)
{
    str = strn;
    name = aname;
    st = 100; // Цифровой ключ команды
    int z(0);
    int start(0);
    int arg(0);
    for (int i = 0; ; i++)
    {
        if (buff_com[i] == ';')
        { break; };
        if (z == 0)
        { fun += buff_com[i]; };
        if (buff_com[i] == '>' || buff_com[i] == ',')
        {z++; start = i+1; arg = 0;};
        if (z == 1 && buff_com[i] != '>') // Первый аргумент
        {
            arg++;
            sx0 = buff_com.Mid(start,arg);
        }
        else if (z == 2 && buff_com[i] != ',')
        {
```

```
                    arg++;
                    sy0 = buff_com.Mid(start,arg);
                };
            };
            double ugol(0), dlina(0);
            x0 = converter(sx0, rz, vars);
            y0 = converter(sy0, rz, vars);
        }
        else if (buff_com.Left(4).MakeLower() == "turn"
            || !turn.IsEmpty() && turn.GetLength() < buff_com.GetLength() && buff_com.Left(turn.GetLength()).MakeLower() == turn)
        {
            str = strn;
            name = aname;
            st = 190; // Цифровой ключ команды
            int z(0);
            int start(0);
            int arg(0);
            sx0 = "0"; sy0 = "0";
            for (int i = 0; ; i++)
            {
                if (buff_com[i] == ';')
                { break; };
                if (buff_com[i] == '>' || buff_com[i] == ',')
                {z++; start = i+1; arg = 0;};
                if (z == 1 && buff_com[i] != '>') // Первый аргумент
                {
                    arg++;
                    sa1 = buff_com.Mid(start,arg);
                }
```

```
                else if (z == 2 && buff_com[i] != ',')
                {
                        arg++;
                        sx0 = buff_com.Mid(start,arg);
                }
                else if (z == 3 && buff_com[i] != ',')
                {
                        arg++;
                        sy0 = buff_com.Mid(start,arg);
                };
        };
        x0 = converter(sx0, rz, vars); // центр
        y0 = converter(sy0, rz, vars);
        a1 = converter(sa1, rz, vars);
}
else if (buff_com.Left(1).MakeLower() == "="
        || !ravno.IsEmpty() && ravno.GetLength() < buff_com.GetLength() && buff_com.Left(ravno.GetLength()).MakeLower() == ravno)
{
        str = strn;
        name = aname;
        st = 5; // Цифровой ключ команды
        int z(0);
        int start(0);
        int arg(0);
        for (int i = 0; ; i++)
        {
                if (buff_com[i] == ';')
                { break; };
                if (z == 0)
```

```
                { fun += buff_com[i]; };
                if (buff_com[i] == '=' || buff_com[i] == ',')
                {z++; start = i+1; arg = 0;};
                if (z == 1 && buff_com[i] != '=') // Первый аргумент
                {
                        arg++;
                        sx0 = buff_com.Mid(start,arg);
                }
        };
        x0 = converter(sx0, rz, vars);
}
else if (buff_com.Left(8).MakeLower() == "line_to(" || buff_com.Left(8).MakeLower() == "line_to>"
        || !line_to.IsEmpty() && line_to.GetLength() < buff_com.GetLength() && buff_com.Left(line_to.GetLength()).MakeLower() == line_to)
{
        x3 = 2;
        str = strn;
        name = aname;
        p = 1;
        st = 1;
        int z(0);
        int start(0);
        int arg(0);
        for (int i = 0; ; i++)
        {
                if (buff_com[i] == ';')
                { break; };
                if (z == 0)
```

```
        { fun += buff_com[i];};
        if (buff_com[i] == '>' || buff_com[i] == ',')
        {z++; start = i+1; arg = 0;};
        if (z == 1 && buff_com[i] != '>') // Первый аргумент
        {
                arg++;
                sx0 = buff_com.Mid(start,arg);
        }
        else if (z == 2 && buff_com[i] != ',')
        {
                arg++;
                sy0 = buff_com.Mid(start,arg);
        }
        else if (z == 3 && buff_com[i] != ',')
        {
                arg++;
                sx2 = buff_com.Mid(start,arg);
        }
        else if (z == 4 && buff_com[i] != ',')
        {
                arg++;
                sy2 = buff_com.Mid(start,arg);
        }
};
double ugol(0), dlina(0);
x0 = converter(sx0, rz, vars);
y0 = converter(sy0, rz, vars);
ugol = converter(sx2, rz, vars);
dlina = converter(sy2, rz, vars);
if (ugol < 0)
```

```
{
	x1 = dlina*cos(ugol*3.1415/180)+x0;
	y1 = -dlina*sin(ugol*3.1415/180)+y0;
}
else
{
	x1 = dlina*cos(ugol*3.1415/180)+x0;
	y1 = dlina*sin(ugol*3.1415/180)+y0;
};
if (y0 < y1)  {p = -1;}
else if (y0 > y1) {p = 1;}
else {p = 0;};
}
else if (buff_com.Left(4).MakeLower() == "dopr"
	|| !dopr.IsEmpty() && dopr.GetLength() < buff_com.GetLength() && buff_com.Left(dopr.GetLength()).MakeLower() == dopr)
{
	str = strn;
	name = aname;
	p = 1;
	st = 222;
	int z(0);
	int start(0);
	int arg(0);
	if (buff_com.Left(5).MakeLower() == "dopr1"
		|| !dopr1.IsEmpty() && dopr1.GetLength() < buff_com.GetLength() && buff_com.Left(dopr1.GetLength()).MakeLower() == dopr1)
	{
		nit = "1";
	}
```

```
else
{
        nit = "2";
};
for (int i = 0; ; i++)
{
        if (buff_com[i] == ';')
        { break; };
        if (z == 0)
        { fun += buff_com[i]; };
        if (buff_com[i] == '>' || buff_com[i] == ',')
        {z++; start = i+1; arg = 0;};
        if (z == 1 && buff_com[i] != '>') // Первый аргумент
        {
                arg++;
                sa1 = buff_com.Mid(start,arg); // кол-во доп
рядов

        }
        else if (z == 2 && buff_com[i] != ',')
        {
                arg++;
                sa2 = buff_com.Mid(start,arg); // распеределение
        }
        else if (z == 3 && buff_com[i] != ',')
        {
                arg++;
                sa4 = buff_com.Mid(start,arg); // сторона
        }
        else if (z == 4 && buff_com[i] != ',')
```

```
            {
                arg++;
                sa3 = buff_com.Mid(start,arg); // высота ряда
            };
        };
        a1 = converter(sa1, rz, vars);
        a2 = converter(sa2, rz, vars);
        a3 = converter(sa4, rz, vars);
        b = converter(sa3, rz, vars);
        // Длина
        double dlina1(0), NR1(0);
        double dlina2(0);
        for (int j = 0; j < vars.size(); j++)
        {
            if (vars[j].name == name && vars[j].st != 22
                && vars[j].st != 101 && vars[j].st != 102)
            {
                x0 = vars[j].x0;
                sx0 = vars[j].sx0;
                y0 = vars[j].y0;
                sy0 = vars[j].sy0;
                x2 = vars[j].x1;
                sx2 = vars[j].sx1;
                y2 = vars[j].y1;
                sy2 = vars[j].sy1;
                break;
            };
        };
    dlina1 = sqrt((x2-x0)*(x2-x0)+(y2-y0)*(y2-y0));
```

```
NR1 = floor(abs(y2-y0)/b+0.5); // Первоначальное количество рядов
dlina2 = dlina1 + a1*b;
double line(0);
double xa(0), ya(0), aa3(0), aa4(0), xv(0), yv(0);
    // позиция начальной точки
xa = (1 - a2)*x0 + a2*x2;
ya = (1 - a2)*y0 + a2*y2;
xv = xa;
double k(0);
    if (x2 - x0 == 0)
    {
        yv = ya;
    }
    else
    {
        k = (y2-y0)/(x2-x0);
    };
    for (;;)
    {
        if (x2 - x0 == 0)
        {
            xv += a3*b*0.01;
        }
        else
        {
            xv += a3*b*0.01;
            yv = -(xv - xa)/k + ya;
        };
        aa3 = x0; aa4 = y0;
```

```
                t = t1 = 0;
                for (double tt = 0.0001; tt <= 1.0; tt += 0.0001)
                {
                    t = (1.-tt)*(1.-tt)*x0 + 2.*tt*(1.-tt)*xv + tt*tt*x2;
                    t1 = (1.-tt)*(1.-tt)*y0 + 2.*tt*(1.-tt)*yv + tt*tt*y2;
                    line += sqrt((aa3-t)*(aa3-t)+(aa4-t1)*(aa4-t1));
                    aa3 = t;
                    aa4 = t1;
                };
                if (line >= dlina2)
                {
                    x1 = xv;
                    y1 = yv;
                    break;
                };

                line = 0;
            };
        }
        else if (buff_com.Left(3).MakeLower() == "vyt"
            || !vyt.IsEmpty() && vyt.GetLength() < buff_com.GetLength()
&& buff_com.Left(vyt.GetLength()).MakeLower() == vyt)
        {
            str = strn;
            name = aname;
            st = 223;
            p = 1;
            int z(0), zz(0);
            int start(0);
            int arg(0);
```

```
vysota_ryada = "0.1";
for (int i = 0; ; i++)
{
	if (buff_com[i] == ';')
	{
		if (!sa1.IsEmpty() && zz == 1)
		{
			right.push_back(sa1);
		};

		break;
	};
	if (z == 0)
	{ fun += buff_com[i]; };
	if (buff_com[i] == '|')
	{
		zz++;
		start = i+1; arg = 0;
		if (!sa1.IsEmpty() && zz == 1)
		{
			left.push_back(sa1);
		}
		else if (!sa1.IsEmpty() && zz == 2)
		{
			right.push_back(sa1);
		};
	};
	if (buff_com[i] == '>' || buff_com[i] == ',')
	{
		z++; start = i+1; arg = 0;
```

```
            if (zz == 0 && !sa1.IsEmpty())
            {
                left.push_back(sa1);
            }
            else if (zz == 1 && !sa1.IsEmpty())
            {
                right.push_back(sa1);
            };
        };
        if(zz == 0)
        {
            if (z == 1 && buff_com[i] != '>')
            {
                arg++;
                sa1 = buff_com.Mid(start,arg);
            }
            else if (z == 2 && buff_com[i] != ',')
            {
                arg++;
                sa1 = buff_com.Mid(start,arg);
            }
            else if (z == 3 && buff_com[i] != ',')
            {
                arg++;
                sa1 = buff_com.Mid(start,arg);
                z = 2;
            };
        }
        else if (zz == 1)
        {
```

```
        if (z == 1 && buff_com[i] != '|')
        {
                arg++;
                sa1 = buff_com.Mid(start,arg);
        }
        else if (z == 2 && buff_com[i] != ',')
        {
                arg++;
                sa1 = buff_com.Mid(start,arg);
        }
        else if (z == 3 && buff_com[i] != ',')
        {
                arg++;
                sa1 = buff_com.Mid(start,arg);
                z = 2;
        };
}
else if (zz == 2)
{
        if (z == 2 && buff_com[i] != '|')
        {
                arg++;
                end_left = buff_com.Mid(start,arg);
        }
        else if (z == 3 && buff_com[i] != ',')
        {
                arg++;
                end_right = buff_com.Mid(start,arg);
        };
}
```

```
                else if (zz == 3)
                {
                        if (buff_com[i] != '|')
                        {
                                arg++;
                                vysota_ryada = buff_com.Mid(start,arg);
                        };
                }
                else if (zz == 4)
                {
                        if (buff_com[i] != '|')
                        {
                        arg++;
                        p = converter(buff_com.Mid(start,arg+1), rz,
vars);
                        };
                };
        };
        CString proverka;
        for (int i = 0; i < left.size(); i++)
        {
                for (int j = 0; j < left[i].GetLength(); j++)
                {
                        if (left[i][j] != '>' && left[i][j] != ',' && left[i][j]
!= '|')
                        {
                                proverka += left[i][j];
                        }
                };
                left[i] = proverka;
```

```
                proverka.Empty();
            };
            for (int i = 0; i < right.size(); i++)
            {
                for (int j = 0; j < right[i].GetLength(); j++)
                {
                if (right[i][j] != '>' && right[i][j] != ',' && right[i][j] !=
'|')
                        {
                            proverka += right[i][j];
                        }
                };
                right[i] = proverka;
                proverka.Empty();
            };
            for (int i = 0; i < end_left.GetLength(); i++)
            {
                if (end_left[i] != '>' && end_left[i] != ',' && end_left[i]
!= '|')
                {
                    proverka += end_left[i];
                }
            };
            end_left = proverka;
            proverka.Empty();
            for (int i = 0; i < end_right.GetLength(); i++)
            {
            if (end_right[i] != '>' && end_right[i] != ',' && end_right[i] !=
'|')
                {
```

```
                proverka += end_right[i];
            }
        };
        end_right = proverka;
        proverka.Empty();
        for (int i = 0; i < vysota_ryada.GetLength(); i++)
        {
            if (vysota_ryada[i] != '>' && vysota_ryada[i] != ',' &&
vysota_ryada[i] != '|')
            {
                proverka += vysota_ryada[i];
            }
        };
        vysota_ryada = proverka;
        proverka.Empty();
        if (end_left.IsEmpty())
        {
            end_left = "4";
        };
        if (end_right.IsEmpty())
        {
            end_right = "4";
        };
        vyt_l = left[0];
        vyt_r = right[0];

        double p0xl(0), p0yl(0), p0xr(0), p0yr(0); // основание
        double tl(0), tr(0), ty(0), ty2(0);
        // Построение вытачки
            // left
```

```
for (int i = 0; i < vars.size(); i++)
{
	if (vars[i].name == left[0] && p == 1 && vars[i].st == 1)
	{
		if (vars[i].y0 < vars[i].y1)
		{
			p0xl = vars[i].x1;
			p0yl = vars[i].y1;
			tl = vars[i].x0;
			ty = vars[i].y0;
		}
		else
		{
			p0xl = vars[i].x0;
			p0yl = vars[i].y0;
			tl = vars[i].x1;
			ty = vars[i].y1;
		};

		left_line = sqrt((vars[i].x0 - vars[i].x1)*(vars[i].x0 - vars[i].x1)+(vars[i].y0 - vars[i].y1)*(vars[i].y0 - vars[i].y1));
		break;
	}
	else if (vars[i].name == left[0] && p == 3 && vars[i].st == 1)
	{
		if (vars[i].y0 > vars[i].y1)
		{
			p0xl = vars[i].x1;
```

```
                    p0yl = vars[i].y1;
                    tl = vars[i].x0;
                    ty = vars[i].y0;
                }
                else
                {
                    p0xl = vars[i].x0;
                    p0yl = vars[i].y0;
                    tl = vars[i].x1;
                    ty = vars[i].y1;
                };
                left_line = sqrt((vars[i].x0 - vars[i].x1)*(vars[i].x0
- vars[i].x1)+(vars[i].y0 - vars[i].y1)*(vars[i].y0 - vars[i].y1));
                break;
            }
            else if (vars[i].name == left[0] && p == 2 && vars[i].st
== 1)
            {
                if (vars[i].x0 > vars[i].x1)
                {
                    p0xl = vars[i].x1;
                    p0yl = vars[i].y1;
                    tl = vars[i].x0;
                    ty = vars[i].y0;
                }
                else
                {
                    p0xl = vars[i].x0;
                    p0yl = vars[i].y0;
                    tl = vars[i].x1;
```

```
                    ty = vars[i].y1;
                };
                left_line = sqrt((vars[i].x0 - vars[i].x1)*(vars[i].x0
- vars[i].x1)+(vars[i].y0 - vars[i].y1)*(vars[i].y0 - vars[i].y1));
                break;
            }
            else if (vars[i].name == left[0] && p == 4 && vars[i].st
== 1)
            {
                if (vars[i].x0 < vars[i].x1)
                {
                    p0xl = vars[i].x1;
                    p0yl = vars[i].y1;
                    tl = vars[i].x0;
                    ty = vars[i].y0;
                }
                else
                {
                    p0xl = vars[i].x0;
                    p0yl = vars[i].y0;
                    tl = vars[i].x1;
                    ty = vars[i].y1;
                };
                left_line = sqrt((vars[i].x0 - vars[i].x1)*(vars[i].x0
- vars[i].x1)+(vars[i].y0 - vars[i].y1)*(vars[i].y0 - vars[i].y1));
                break;
            }
        };

            // right
```

```
for (int i = 0; i < vars.size(); i++)
{
    if (vars[i].name == right[0] && p == 1 && vars[i].st == 1)
    {
        if (vars[i].y0 < vars[i].y1)
        {
            p0xr = vars[i].x1;
            p0yr = vars[i].y1;
            tr = vars[i].x0;
            ty2 = vars[i].y0;
        }
        else
        {
            p0xr = vars[i].x0;
            p0yr = vars[i].y0;
            tr = vars[i].x1;
            ty2 = vars[i].y1;
        };
        right_line = sqrt((vars[i].x0 - vars[i].x1)*(vars[i].x0 - vars[i].x1)+(vars[i].y0 - vars[i].y1)*(vars[i].y0 - vars[i].y1));
        break;
    }
    else if (vars[i].name == right[0] && p == 3 && vars[i].st == 1)
    {
        if (vars[i].y0 > vars[i].y1)
        {
            p0xr = vars[i].x1;
            p0yr = vars[i].y1;
```

```
                tr = vars[i].x0;
                ty2 = vars[i].y0;
            }
            else
            {
                p0xr = vars[i].x0;
                p0yr = vars[i].y0;
                tr = vars[i].x1;
                ty2 = vars[i].y1;
            };
            right_line = sqrt((vars[i].x0 - vars[i].x1)*(vars[i].x0 - vars[i].x1)+(vars[i].y0 - vars[i].y1)*(vars[i].y0 - vars[i].y1));
            break;
        }
    else if (vars[i].name == right[0] && p == 2 && vars[i].st == 1)
        {
            if (vars[i].x0 > vars[i].x1)
            {
                p0xr = vars[i].x1;
                p0yr = vars[i].y1;
                tr = vars[i].x0;
                ty2 = vars[i].y0;
            }
            else
            {
                p0xr = vars[i].x0;
                p0yr = vars[i].y0;
                tr = vars[i].x1;
                ty2 = vars[i].y1;
```

```
                    };
                    right_line = sqrt((vars[i].x0 - vars[i].x1)*(vars[i].x0 - vars[i].x1)+(vars[i].y0 - vars[i].y1)*(vars[i].y0 - vars[i].y1));
                    break;
                }
            else if (vars[i].name == right[0] && p == 4 && vars[i].st == 1)
                {
                    if (vars[i].x0 < vars[i].x1)
                    {
                        p0xr = vars[i].x1;
                        p0yr = vars[i].y1;
                        tr = vars[i].x0;
                        ty2 = vars[i].y0;
                    }
                    else
                    {
                        p0xr = vars[i].x0;
                        p0yr = vars[i].y0;
                        tr = vars[i].x1;
                        ty2 = vars[i].y1;
                    };
                    right_line = sqrt((vars[i].x0 - vars[i].x1)*(vars[i].x0 - vars[i].x1)+(vars[i].y0 - vars[i].y1)*(vars[i].y0 - vars[i].y1));
                    break;
                }
            };
            // N доп рядов
            double rastvor(0);
            int nryad(0);
```

```
double hr(0);
hr = converter(vysota_ryada, rz,vars);
rastvor = sqrt((tr-tl)*(tr-tl)+(ty-ty2)*(ty-ty2));
if (p == 1 || p == 3)
{
    nryad = floor(2*sqrt(rastvor*rastvor/4 + right_line*right_line)/hr + 0.5);
}
else
{
nryad = floor(sqrt(rastvor*rastvor/4 + right_line*right_line)/hr + 0.5);
};
n_ryadov.Format(_T("%i"), nryad);
// точка закрытия
if (p == 1)
{
    pzx = tl + (tr-tl)/2;
    pzy = p0yl - left_line;
}
else if (p == 3)
{
    pzx = tl + (tr-tl)/2;
    pzy = p0yl + left_line;
}
else if (p == 2)
{
    pzy = p0yl - rastvor/2;
    pzx = p0xl - left_line;
}
```

```
else if (p == 4)
{
        pzy = p0yl - rastvor/2;
        pzx = p0xl + left_line;
}
if (p == 1 || p == 3)
{
        //левое изменение
        a1 = pzx - tl; // dx
        a2 = pzy - ty; // dy
        // правое изменеие
        a3 = pzx - tr; // dx
        a4 = pzy - ty2; // dy
}
else if (p == 2)
{
        a1 = p0xl + left_line - tl;
        a2 = p0yl - ty;
        a3 = p0xr + left_line - tr;
        a4 = p0yr - ty2;
}
else if (p == 4)
{
        a1 = p0xl - left_line - tl;
        a2 = p0yl - ty;
        a3 = p0xr - left_line - tr;
        a4 = p0yr - ty2;
};
// точка основания вытачки
osnx = p0xl;
```

```
                osny = p0yl;
                ras = rastvor;
            };
    }
    double CObjects::converter(CString argum, std::vector<CRazmer> rz, std::vector
<CObjects> vars)
    {
        double argument (0);
        std::vector<CString> temp1;
        std::vector<CString> temp2;
        CString aName;
        int z(1);
        int start(0);
        int arg(0);
        int pr(0);
        int v2_count(0);
        std::vector <CString> v1; // выходная строка
        std::vector <CString> v2; // знаки
        std::vector <double> result;
        double n1(0), n2(0), res(0);
        if (argum[0] == '-')
        {
            argum = CString("0.00")+argum;
        }
        else
        {
            argum = CString("0.00+")+argum;
        };
        for (int i = 0; i < argum.GetLength() ; i++)
        {
```

```
                    if (argum[i] == '+' || argum[i] == '-' || argum[i] == '*' || argum[i]
== '/'
                        || argum[i] == '(' || argum[i] == ')')
                    {
                        if (argum[i] == '-' && argum[i-1] == '(')
                        {
                            z++; start = i+1; arg = 0;

                            temp1.push_back(CString("-1.00"));

                            temp1.push_back(CString("*"));
                        }
                        else
                        {
                            z++; start = i+1; arg = 0;
                            temp1.push_back(aName);
                            temp1.push_back(CString(argum[i]));
                        };
                    };
                    if (z == 1)
                    {
                        arg++;
                        aName = argum.Mid(start,arg);
                    }
                    else if (z == 2)
                    {
                        arg++;
                        aName = argum.Mid(start,arg-1);
                    }
```

```
                else if (z == 3)
                {
                        arg++;
                        aName = argum.Mid(start,arg-1);
                        z = 2;
                };
        };
        temp1.push_back(aName);
        for (int i = 0; i < temp1.size(); i++)
        {
                for (int j = 0; j < temp1[i].GetLength(); j++)
                {
                        if (temp1[i][j] == '.')
                        {
                                for (int h = 0; h < vars.size(); h++)
                                {
                                        if (vars[h].name == temp1[i].Left(j))
                                        {
                                        if     (temp1[i].Right(2).MakeLower()     ==
"x0")
                                                {
                                                        CString bbb;
                                                        bbb.Format(_T("%.2f"),
vars[h].x0);
                                                        temp1[i] = bbb;
                                                }
                                        else  if  (temp1[i].Right(2).MakeLower() ==
"y0")
                                                {
                                                        CString bbb;
```

```
                bbb.Format(_T("%.2f"),
vars[h].y0);
                temp1[i] = bbb;
            }
        else if (temp1[i].Right(2).MakeLower() ==
"x1")
            {
                CString bbb;
                bbb.Format(_T("%.2f"),
vars[h].x1);
                temp1[i] = bbb;
            }
        else if (temp1[i].Right(2).MakeLower() ==
"y1")
            {
                CString bbb;
                bbb.Format(_T("%.2f"),
vars[h].y1);
                temp1[i] = bbb;
            }
        else if (temp1[i].Right(2).MakeLower() ==
"x2")
            {
                CString bbb;
                bbb.Format(_T("%.2f"),
vars[h].x2);
                temp1[i] = bbb;
            }
        else if (temp1[i].Right(2).MakeLower() ==
"y2")
```

```
                {
                    CString bbb;
                    bbb.Format(_T("%.2f"),
vars[h].y2);
                    temp1[i] = bbb;
                }
            else if (temp1[i].Right(2).MakeLower() ==
"x3")
                {
                    CString bbb;
                    bbb.Format(_T("%.2f"),
vars[h].x3);
                    temp1[i] = bbb;
                }
            else if (temp1[i].Right(2).MakeLower() ==
"y3")
                {
                    CString bbb;
                    bbb.Format(_T("%.2f"),
vars[h].y3);
                    temp1[i] = bbb;
                }
            else if (temp1[i].Right(1).MakeLower() ==
"l")
                {
                    double line(0);
                    if (vars[h].st == 1)
                    {
                        line = sqrt((vars[h].x1 -
vars[h].x0)*(vars[h].x1 - vars[h].x0)
```

```
                                        +(vars[h].y1 -
vars[h].y0)*(vars[h].y1 - vars[h].y0));
                                    }
                                    else if (vars[h].st == 2)
                                    {
                                        double a1(vars[h].x0),
b1(vars[h].y0), a2(0), b2(0);
                                        for (double t = 0.001; t
<= 1; t += 0.001)
                                        {
                                            a2 = (1-t)*(1-
t)*vars[h].x0+2*t*(1-t)*vars[h].x1+t*t*vars[h].x2;
                                            b2 = (1-t)*(1-
t)*vars[h].y0+2*t*(1-t)*vars[h].y1+t*t*vars[h].y2;
                                            line += sqrt((a1-
a2)*(a1-a2)+(b1-b2)*(b1-b2));
                                            a1 = a2;
                                            b1 = b2;
                                        };
                                    }
                                    else if (vars[h].st == 3)
                                    {
                                        double a1(vars[h].x0),
b1(vars[h].y0), a2(0), b2(0);
                                        for (double t = 0.001; t
<= 1; t += 0.001)
                                        {
                                            a2 = (1-t)*(1-
t)*(1-t)*vars[h].x0+3*t*(1-t)*(1-t)*vars[h].x1+3*t*t*(1-t)*vars[h].x2+t*t*t*vars[h].x3;
```

```
                                                                b2 = (1-t)*(1-t)*(1-t)*vars[h].y0+3*t*(1-t)*(1-t)*vars[h].y1+3*t*t*(1-t)*vars[h].y2+t*t*t*vars[h].y3;
                                                                line += sqrt((a1-a2)*(a1-a2)+(b1-b2)*(b1-b2));
                                                                a1 = a2;
                                                                b1 = b2;
                                                            };
                                                        }

                                                        CString bbb;
                                                        bbb.Format(_T("%.2f"), line);
                                                        temp1[i] = bbb;
                                                    };
                                                };
                                            };
                                            break;
                                        };
                                    };
                                    for (int j = 0; j < rz.size(); j++)
                                    {
                                        if (rz[j].name.MakeLower() == temp1[i].MakeLower())
                                        {
                                            CString bbb;
                                            bbb.Format(_T("%.2f"), rz[j].rz);
                                            temp1[i] = bbb;
                                        }
                                    };
                                    for (int j = 0; j < vars.size(); j++)
                                    {
                                        if (vars[j].name == temp1[i])
```

```
                        {
                            CString bbb;
                            bbb.Format(_T("%.2f"), vars[j].x0);
                            temp1[i] = bbb;
                        };
                    };
                };
                for (int i = 0; i < temp1.size(); i++)
                {
                    if (!temp1[i].IsEmpty())
                    {
                        temp2.push_back(temp1[i]);
                    };
                };
                temp1.clear();
                for (int i = 0; i < temp2.size(); i++)
                {
                    if (temp2[i] == "+" || temp2[i] == "-" || temp2[i] == "/" || temp2[i] ==
"*"
                        || temp2[i] == "(" || temp2[i] == ")")
                    {
                        pr = prior(temp2[i]);
                        v2.push_back(temp2[i]);

                        if (temp2[i] == ")")
                        {
                            for (int j = v2.size()-1; ; j--)
                            {
                                v2_count++;
                                if (v2[j] == "(")
```

```
                    {
                        v2.pop_back();
                        break;
                    };
                    v1.push_back(v2[j]);
                    if (v1.back() == "(")
                    {
                        v1.pop_back();
                    };
                    if (v1.back() == ")")
                    {
                        v1.pop_back();
                    };
                };
                for (int j = 0; j < v2_count-1; j++)
                {
                    v2.pop_back();
                };
                v2_count = 0;
            };
        }
        else
        {
            v1.push_back(temp2[i]);
            if (pr == 1)
            {
                v1.push_back(v2.back());
                v2.pop_back();
            };
        };
```

```
};
for (int j = v2.size()-1; j >= 0; j--)
{
        if(v2[j] != "(")
        {v1.push_back(v2[j]);};
};
std::stack<double> st1;
for (int i = 0; i < v1.size(); i++)
{
                if (v1[i] == "+")
                {
                        n1 = st1.top();
                        st1.pop();
                        n2 = st1.top();
                        st1.pop();
                        res = n2+n1;
                        st1.push(res);
                }
                else if (v1[i] == "-")
                {
                        n1 = st1.top();
                        st1.pop();
                        n2 = st1.top();
                        st1.pop();
                        res = n2-n1;
                        st1.push(res);
                }
                else if (v1[i] == "/")
                {
                        n1 = st1.top();
```

```
                    st1.pop();
                    n2 = st1.top();
                    st1.pop();
                    if (n1 == 0.000)
                    {
AfxMessageBox(CString("Внимание! Деление на ноль!"), MB_OK);
                        break;
                    }
                    else
                    {
                        res = n2/n1;
                        st1.push(res);
                    };
                }
                else if (v1[i] == "*")
                {
                    n1 = st1.top();
                    st1.pop();
                    n2 = st1.top();
                    st1.pop();
                    res = n2*n1;
                    st1.push(res);
                }
                else
                {
                    st1.push(_tstof(v1[i]));
                };
            };
            return floor(st1.top()*100 + 0.5)/100;
```

```
};
int CObjects::prior(CString zn)
{
	if (zn == '+' || zn == '-' || zn == '(' || zn == ')')
	{return 0;}
	else if (zn == '*' || zn == '/')
	{return 1;}
	else
	{return 2;};
};
```

Файл “Razmer.cpp”

```
#include "Razmer.h"
CRazmer::CRazmer(void)
{
}
CRazmer::~CRazmer(void)
{
}
CRazmer::CRazmer(CString buff, CString tit)
{
	title = tit;
	CString argum;
	if (buff.Left(4).MakeLower() == "dts=")
	{
		name = "dts";
		int z(0);
		int start(0);
		int arg(0);
		for (int i = 0; ; i++)
```

```
        {
                if (buff[i] == ';')
                {break;};
                if (buff[i] == '=')
                {z++; start = i+1; arg = 0;};
                if (z == 1 && buff[i] != '=') // Первый аргумент
                {
                        arg++;
                        argum = buff.Mid(start,arg);
                        rz = converter(argum);
                }
        };
}
else if (buff.Left(2).MakeLower() == "p=")
{
        name = "p";
        int z(0);
        int start(0);
        int arg(0);
        for (int i = 0; ; i++)
        {
                if (buff[i] == ';')
                {break;};
                if (buff[i] == '=')
                {z++; start = i+1; arg = 0;};
                if (z == 1 && buff[i] != '=') // Первый аргумент
                {
                        arg++;
                        argum = buff.Mid(start,arg);
                        rz = converter(argum);
```

```
                }
        };
}
else if (buff.Left(5).MakeLower() == "vtos=")
{
        name= "vtos";
        int z(0);
        int start(0);
        int arg(0);
        for (int i = 0; ; i++)
        {
                if (buff[i] == ';')
                {break;};
                if (buff[i] == '=')
                {z++; start = i+1; arg = 0;};
                if (z == 1 && buff[i] != '=') // Первый аргумент
                {
                        arg++;
                        argum = buff.Mid(start,arg);
                        rz = converter(argum);
                }
        };
}
else if (buff.Left(4).MakeLower() == "vpt=")
{
        name= "vpt";
        int z(0);
        int start(0);
        int arg(0);
        for (int i = 0; ; i++)
```

```
        {
                if (buff[i] == ';')
                {break;};
                if (buff[i] == '=')
                {z++; start = i+1; arg = 0;};
                if (z == 1 && buff[i] != '=') // Первый аргумент
                {
                        arg++;
                        argum = buff.Mid(start,arg);
                        rz = converter(argum);
                }
        };
}
else if (buff.Left(4).MakeLower() == "vlt=")
{
        name= "vlt";
        int z(0);
        int start(0);
        int arg(0);
        for (int i = 0; ; i++)
        {
                if (buff[i] == ';')
                {break;};
                if (buff[i] == '=')
                {z++; start = i+1; arg = 0;};
                if (z == 1 && buff[i] != '=') // Первый аргумент
                {
                        arg++;
                        argum = buff.Mid(start,arg);
                        rz = converter(argum);
```

```
                }
        };
}
else if (buff.Left(3).MakeLower() == "vk=")
{
        name= "vk";
        int z(0);
        int start(0);
        int arg(0);
        for (int i = 0; ; i++)
        {
                if (buff[i] == ';')
                {break;};
                if (buff[i] == '=')
                {z++; start = i+1; arg = 0;};
                if (z == 1 && buff[i] != '=') // Первый аргумент
                {
                        arg++;
                        argum = buff.Mid(start,arg);
                        rz = converter(argum);
                }
        };
}
else if (buff.Left(5).MakeLower() == "vsht=")
{
        name= "vsht";
        int z(0);
        int start(0);
        int arg(0);
        for (int i = 0; ; i++)
```

```
        {
                if (buff[i] == ';')
                {break;};
                if (buff[i] == '=')
                {z++; start = i+1; arg = 0;};
                if (z == 1 && buff[i] != '=') // Первый аргумент
                {
                        arg++;
                        argum = buff.Mid(start,arg);
                        rz = converter(argum);
                }
        };
}
else if (buff.Left(4).MakeLower() == "vzu=")
{
        name= "vzu";
        int z(0);
        int start(0);
        int arg(0);
        for (int i = 0; ; i++)
        {
                if (buff[i] == ';')
                {break;};
                if (buff[i] == '=')
                {z++; start = i+1; arg = 0;};
                if (z == 1 && buff[i] != '=') // Первый аргумент
                {
                        arg++;
                        argum = buff.Mid(start,arg);
                        rz = converter(argum);
```

```
                }
        };
}
else if (buff.Left(5).MakeLower() == "vlop=")
{
        name= "vlop";
        int z(0);
        int start(0);
        int arg(0);
        for (int i = 0; ; i++)
        {
                if (buff[i] == ';')
                {break;};
                if (buff[i] == '=')
                {z++; start = i+1; arg = 0;};
                if (z == 1 && buff[i] != '=') // Первый аргумент
                {
                        arg++;
                        argum = buff.Mid(start,arg);
                        rz = converter(argum);
                }
        };
}
else if (buff.Left(4).MakeLower() == "vps=")
{
        name= "vps";
        int z(0);
        int start(0);
        int arg(0);
        for (int i = 0; ; i++)
```

```
        {
                if (buff[i] == ';')
                {break;};
                if (buff[i] == '=')
                {z++; start = i+1; arg = 0;};
                if (z == 1 && buff[i] != '=') // Первый аргумент
                {
                        arg++;
                        argum = buff.Mid(start,arg);
                        rz = converter(argum);
                }
        };
}
else if (buff.Left(4).MakeLower() == "osh=")
{
        name= "osh";
        int z(0);
        int start(0);
        int arg(0);
        for (int i = 0; ; i++)
        {
                if (buff[i] == ';')
                {break;};
                if (buff[i] == '=')
                {z++; start = i+1; arg = 0;};
                if (z == 1 && buff[i] != '=') // Первый аргумент
                {
                        arg++;
                        argum = buff.Mid(start,arg);
                        rz = converter(argum);
```

```
                }
        };
}
else if (buff.Left(4).MakeLower() == "og1=")
{
        name= "og1";
        int z(0);
        int start(0);
        int arg(0);
        for (int i = 0; ; i++)
        {
                if (buff[i] == ';')
                {break;};
                if (buff[i] == '=')
                {z++; start = i+1; arg = 0;};
                if (z == 1 && buff[i] != '=') // Первый аргумент
                {
                        arg++;
                        argum = buff.Mid(start,arg);
                        rz = converter(argum);
                }
        };
}
else if (buff.Left(4).MakeLower() == "og2=")
{
        name= "og2";
        int z(0);
        int start(0);
        int arg(0);
        for (int i = 0; ; i++)
```

```
        {
                if (buff[i] == ';')
                {break;};
                if (buff[i] == '=')
                {z++; start = i+1; arg = 0;};
                if (z == 1 && buff[i] != '=') // Первый аргумент
                {
                        arg++;
                        argum = buff.Mid(start,arg);
                        rz = converter(argum);
                }
        };
}
else if (buff.Left(4).MakeLower() == "og3=")
{
        name= "og3";
        int z(0);
        int start(0);
        int arg(0);
        for (int i = 0; ; i++)
        {
                if (buff[i] == ';')
                {break;};
                if (buff[i] == '=')
                {z++; start = i+1; arg = 0;};
                if (z == 1 && buff[i] != '=') // Первый аргумент
                {
                        arg++;
                        argum = buff.Mid(start,arg);
                        rz = converter(argum);
```

```
                }
        };
}
else if (buff.Left(4).MakeLower() == "og4=")
{
        name= "og4";
        int z(0);
        int start(0);
        int arg(0);
        for (int i = 0; ; i++)
        {
                if (buff[i] == ';')
                {break;};
                if (buff[i] == '=')
                {z++; start = i+1; arg = 0;};
                if (z == 1 && buff[i] != '=') // Первый аргумент
                {
                        arg++;
                        argum = buff.Mid(start,arg);
                        rz = converter(argum);
                }
        };
}
else if (buff.Left(3).MakeLower() == "ot=")
{
        name= "ot";
        int z(0);
        int start(0);
        int arg(0);
        for (int i = 0; ; i++)
```

```
        {
                if (buff[i] == ';')
                {break;};
                if (buff[i] == '=')
                {z++; start = i+1; arg = 0;};
                if (z == 1 && buff[i] != '=') // Первый аргумент
                {
                        arg++;
                        argum = buff.Mid(start,arg);
                        rz = converter(argum);
                }
        };
}
else if (buff.Left(3).MakeLower() == "ob=")
{
        name= "ob";
        int z(0);
        int start(0);
        int arg(0);
        for (int i = 0; ; i++)
        {
                if (buff[i] == ';')
                {break;};
                if (buff[i] == '=')
                {z++; start = i+1; arg = 0;};
                if (z == 1 && buff[i] != '=') // Первый аргумент
                {
                        arg++;
                        argum = buff.Mid(start,arg);
                        rz = converter(argum);
```

```
                }
        };
}
else if (buff.Left(4).MakeLower() == "ob1=")
{
        name= "ob1";
        int z(0);
        int start(0);
        int arg(0);
        for (int i = 0; ; i++)
        {
                if (buff[i] == ';')
                {break;};
                if (buff[i] == '=')
                {z++; start = i+1; arg = 0;};
                if (z == 1 && buff[i] != '=') // Первый аргумент
                {
                        arg++;
                        argum = buff.Mid(start,arg);
                        rz = converter(argum);
                }
        };
}
else if (buff.Left(5).MakeLower() == "obed=")
{
        name= "obed";
        int z(0);
        int start(0);
        int arg(0);
        for (int i = 0; ; i++)
```

```
        {
                if (buff[i] == ';')
                {break;};
                if (buff[i] == '=')
                {z++; start = i+1; arg = 0;};
                if (z == 1 && buff[i] != '=') // Первый аргумент
                {
                        arg++;
                        argum = buff.Mid(start,arg);
                        rz = converter(argum);
                }
        };
}
else if (buff.Left(3).MakeLower() == "ok=")
{
        name= "ok";
        int z(0);
        int start(0);
        int arg(0);
        for (int i = 0; ; i++)
        {
                if (buff[i] == ';')
                {break;};
                if (buff[i] == '=')
                {z++; start = i+1; arg = 0;};
                if (z == 1 && buff[i] != '=') // Первый аргумент
                {
                        arg++;
                        argum = buff.Mid(start,arg);
                        rz = converter(argum);
```

```
                }
        };
}
else if (buff.Left(3).MakeLower() == "oi=")
{
        name= "oi";
        int z(0);
        int start(0);
        int arg(0);
        for (int i = 0; ; i++)
        {
                if (buff[i] == ';')
                {break;};
                if (buff[i] == '=')
                {z++; start = i+1; arg = 0;};
                if (z == 1 && buff[i] != '=') // Первый аргумент
                {
                        arg++;
                        argum = buff.Mid(start,arg);
                        rz = converter(argum);
                }
        };
}
else if (buff.Left(5).MakeLower() == "osch=")
{
        name= "osch";
        int z(0);
        int start(0);
        int arg(0);
        for (int i = 0; ; i++)
```

```
        {
                if (buff[i] == ';')
                {break;};
                if (buff[i] == '=')
                {z++; start = i+1; arg = 0;};
                if (z == 1 && buff[i] != '=') // Первый аргумент
                {
                        arg++;
                        argum = buff.Mid(start,arg);
                        rz = converter(argum);
                }
        };
}
else if (buff.Left(3).MakeLower() == "os=")
{
        name= "os";
        int z(0);
        int start(0);
        int arg(0);
        for (int i = 0; ; i++)
        {
                if (buff[i] == ';')
                {break;};
                if (buff[i] == '=')
                {z++; start = i+1; arg = 0;};
                if (z == 1 && buff[i] != '=') // Первый аргумент
                {
                        arg++;
                        argum = buff.Mid(start,arg);
                        rz = converter(argum);
```

```
                }
        };
}
else if (buff.Left(4).MakeLower() == "dsb=")
{
        name= "dsb";
        int z(0);
        int start(0);
        int arg(0);
        for (int i = 0; ; i++)
        {
                if (buff[i] == ';')
                {break;};
                if (buff[i] == '=')
                {z++; start = i+1; arg = 0;};
                if (z == 1 && buff[i] != '=') // Первый аргумент
                {
                        arg++;
                        argum = buff.Mid(start,arg);
                        rz = converter(argum);
                }
        };
}
else if (buff.Left(4).MakeLower() == "dsp=")
{
        name= "dsp";
        int z(0);
        int start(0);
        int arg(0);
        for (int i = 0; ; i++)
```

```
        {
                if (buff[i] == ';')
                {break;};
                if (buff[i] == '=')
                {z++; start = i+1; arg = 0;};
                if (z == 1 && buff[i] != '=') // Первый аргумент
                {
                        arg++;
                        argum = buff.Mid(start,arg);
                        rz = converter(argum);
                }
        };
}
else if (buff.Left(3).MakeLower() == "dn=")
{
        name= "dn";
        int z(0);
        int start(0);
        int arg(0);
        for (int i = 0; ; i++)
        {
                if (buff[i] == ';')
                {break;};
                if (buff[i] == '=')
                {z++; start = i+1; arg = 0;};
                if (z == 1 && buff[i] != '=') // Первый аргумент
                {
                        arg++;
                        argum = buff.Mid(start,arg);
                        rz = converter(argum);
```

```
                }
        };
}
else if (buff.Left(4).MakeLower() == "dps=")
{
        name= "dps";
        int z(0);
        int start(0);
        int arg(0);
        for (int i = 0; ; i++)
        {
                if (buff[i] == ';')
                {break;};
                if (buff[i] == '=')
                {z++; start = i+1; arg = 0;};
                if (z == 1 && buff[i] != '=') // Первый аргумент
                {
                        arg++;
                        argum = buff.Mid(start,arg);
                        rz = converter(argum);
                }
        };
}
else if (buff.Left(5).MakeLower() == "dpob=")
{
        name= "dpob";
        int z(0);
        int start(0);
        int arg(0);
        for (int i = 0; ; i++)
```

```
        {
                if (buff[i] == ';')
                {break;};
                if (buff[i] == '=')
                {z++; start = i+1; arg = 0;};
                if (z == 1 && buff[i] != '=') // Первый аргумент
                {
                        arg++;
                        argum = buff.Mid(start,arg);
                        rz = converter(argum);
                }
        };
}
else if (buff.Left(3).MakeLower() == "ds=")
{
        name= "ds";
        int z(0);
        int start(0);
        int arg(0);
        for (int i = 0; ; i++)
        {
                if (buff[i] == ';')
                {break;};
                if (buff[i] == '=')
                {z++; start = i+1; arg = 0;};
                if (z == 1 && buff[i] != '=') // Первый аргумент
                {
                        arg++;
                        argum = buff.Mid(start,arg);
                        rz = converter(argum);
```

```
                }
        };
}
else if (buff.Left(3).MakeLower() == "op=")
{
        name= "op";
        int z(0);
        int start(0);
        int arg(0);
        for (int i = 0; ; i++)
        {
                if (buff[i] == ';')
                {break;};
                if (buff[i] == '=')
                {z++; start = i+1; arg = 0;};
                if (z == 1 && buff[i] != '=') // Первый аргумент
                {
                        arg++;
                        argum = buff.Mid(start,arg);
                        rz = converter(argum);
                }
        };
}
else if (buff.Left(3).MakeLower() == "oz=")
{
        name= "oz";
        int z(0);
        int start(0);
        int arg(0);
        for (int i = 0; ; i++)
```

```
        {
                if (buff[i] == ';')
                {break;};
                if (buff[i] == '=')
                {z++; start = i+1; arg = 0;};
                if (z == 1 && buff[i] != '=') // Первый аргумент
                {
                        arg++;
                        argum = buff.Mid(start,arg);
                        rz = converter(argum);
                }
        };
}
else if (buff.Left(5).MakeLower() == "okis=")
{
        name= "okis";
        int z(0);
        int start(0);
        int arg(0);
        for (int i = 0; ; i++)
        {
                if (buff[i] == ';')
                {break;};
                if (buff[i] == '=')
                {z++; start = i+1; arg = 0;};
                if (z == 1 && buff[i] != '=') // Первый аргумент
                {
                        arg++;
                        argum = buff.Mid(start,arg);
                        rz = converter(argum);
```

```
                }
        };
}
else if (buff.Left(4).MakeLower() == "shp=")
{
        name= "shp";
        int z(0);
        int start(0);
        int arg(0);
        for (int i = 0; ; i++)
        {
                if (buff[i] == ';')
                {break;};
                if (buff[i] == '=')
                {z++; start = i+1; arg = 0;};
                if (z == 1 && buff[i] != '=') // Первый аргумент
                {
                        arg++;
                        argum = buff.Mid(start,arg);
                        rz = converter(argum);
                }
        };
}
else if (buff.Left(6).MakeLower() == "dluch=")
{
        name= "dluch";
        int z(0);
        int start(0);
        int arg(0);
        for (int i = 0; ; i++)
```

```
        {
                if (buff[i] == ';')
                {break;};
                if (buff[i] == '=')
                {z++; start = i+1; arg = 0;};
                if (z == 1 && buff[i] != '=') // Первый аргумент
                {
                        arg++;
                        argum = buff.Mid(start,arg);
                        rz = converter(argum);
                }
        };
}
else if (buff.Left(5).MakeLower() == "dzap=")
{
        name= "dzap";
        int z(0);
        int start(0);
        int arg(0);
        for (int i = 0; ; i++)
        {
                if (buff[i] == ';')
                {break;};
                if (buff[i] == '=')
                {z++; start = i+1; arg = 0;};
                if (z == 1 && buff[i] != '=') // Первый аргумент
                {
                        arg++;
                        argum = buff.Mid(start,arg);
                        rz = converter(argum);
```

```
                }
        };
}
else if (buff.Left(4).MakeLower() == "d3p=")
{
        name= "d3p";
        int z(0);
        int start(0);
        int arg(0);
        for (int i = 0; ; i++)
        {
                if (buff[i] == ';')
                {break;};
                if (buff[i] == '=')
                {z++; start = i+1; arg = 0;};
                if (z == 1 && buff[i] != '=') // Первый аргумент
                {
                        arg++;
                        argum = buff.Mid(start,arg);
                        rz = converter(argum);
                }
        };
}
else if (buff.Left(4).MakeLower() == "vpr=")
{
        name= "vpr";
        int z(0);
        int start(0);
        int arg(0);
        for (int i = 0; ; i++)
```

```
        {
                if (buff[i] == ';')
                {break;};
                if (buff[i] == '=')
                {z++; start = i+1; arg = 0;};
                if (z == 1 && buff[i] != '=') // Первый аргумент
                {
                        arg++;
                        argum = buff.Mid(start,arg);
                        rz = converter(argum);
                }
        };
}
else if (buff.Left(2).MakeLower() == "vg=")
{
        name= "vg";
        int z(0);
        int start(0);
        int arg(0);
        for (int i = 0; ; i++)
        {
                if (buff[i] == ';')
                {break;};
                if (buff[i] == '=')
                {z++; start = i+1; arg = 0;};
                if (z == 1 && buff[i] != '=') // Первый аргумент
                {
                        arg++;
                        argum = buff.Mid(start,arg);
                        rz = converter(argum);
```

```
                }
        };
}
else if (buff.Left(4).MakeLower() == "dtp=")
{
        name= "dtp";
        int z(0);
        int start(0);
        int arg(0);
        for (int i = 0; ; i++)
        {
                if (buff[i] == ';')
                {break;};
                if (buff[i] == '=')
                {z++; start = i+1; arg = 0;};
                if (z == 1 && buff[i] != '=') // Первый аргумент
                {
                        arg++;
                        argum = buff.Mid(start,arg);
                        rz = converter(argum);
                }
        };
}
else if (buff.Left(3).MakeLower() == "dp=")
{
        name= "dp";
        int z(0);
        int start(0);
        int arg(0);
        for (int i = 0; ; i++)
```

```
        {
                if (buff[i] == ';')
                {break;};
                if (buff[i] == '=')
                {z++; start = i+1; arg = 0;};
                if (z == 1 && buff[i] != '=') // Первый аргумент
                {
                        arg++;
                        argum = buff.Mid(start,arg);
                        rz = converter(argum);
                }
        };
}
else if (buff.Left(5).MakeLower() == "vprz=")
{
        name= "vprz";
        int z(0);
        int start(0);
        int arg(0);
        for (int i = 0; ; i++)
        {
                if (buff[i] == ';')
                {break;};
                if (buff[i] == '=')
                {z++; start = i+1; arg = 0;};
                if (z == 1 && buff[i] != '=') // Первый аргумент
                {
                        arg++;
                        argum = buff.Mid(start,arg);
                        rz = converter(argum);
```

```
                }
        };
}
else if (buff.Left(5).MakeLower() == "dts1=")
{
        name= "dts1";
        int z(0);
        int start(0);
        int arg(0);
        for (int i = 0; ; i++)
        {
                if (buff[i] == ';')
                {break;};
                if (buff[i] == '=')
                {z++; start = i+1; arg = 0;};
                if (z == 1 && buff[i] != '=') // Первый аргумент
                {
                        arg++;
                        argum = buff.Mid(start,arg);
                        rz = converter(argum);
                }
        };
}
else if (buff.Left(6).MakeLower() == "dvcht=")
{
        name= "dvcht";
        int z(0);
        int start(0);
        int arg(0);
        for (int i = 0; ; i++)
```

```
        {
                if (buff[i] == ';')
                {break;};
                if (buff[i] == '=')
                {z++; start = i+1; arg = 0;};
                if (z == 1 && buff[i] != '=') // Первый аргумент
                {
                        arg++;
                        argum = buff.Mid(start,arg);
                        rz = converter(argum);
                }
        };
}
else if (buff.Left(4).MakeLower() == "shg=")
{
        name= "shg";
        int z(0);
        int start(0);
        int arg(0);
        for (int i = 0; ; i++)
        {
                if (buff[i] == ';')
                {break;};
                if (buff[i] == '=')
                {z++; start = i+1; arg = 0;};
                if (z == 1 && buff[i] != '=') // Первый аргумент
                {
                        arg++;
                        argum = buff.Mid(start,arg);
                        rz = converter(argum);
```

```
                }
        };
}
else if (buff.Left(3).MakeLower() == "cg=")
{
        name= "cg";
        int z(0);
        int start(0);
        int arg(0);
        for (int i = 0; ; i++)
        {
                if (buff[i] == ';')
                {break;};
                if (buff[i] == '=')
                {z++; start = i+1; arg = 0;};
                if (z == 1 && buff[i] != '=') // Первый аргумент
                {
                        arg++;
                        argum = buff.Mid(start,arg);
                        rz = converter(argum);
                }
        };
}
else if (buff.Left(4).MakeLower() == "shs=")
{
        name= "shs";
        int z(0);
        int start(0);
        int arg(0);
        for (int i = 0; ; i++)
```

```
        {
                if (buff[i] == ';')
                {break;};
                if (buff[i] == '=')
                {z++; start = i+1; arg = 0;};
                if (z == 1 && buff[i] != '=') // Первый аргумент
                {
                        arg++;
                        argum = buff.Mid(start,arg);
                        rz = converter(argum);
                }
        };
}
else if (buff.Left(5).MakeLower() == "dpzr=")
{
        name= "dpzr";
        int z(0);
        int start(0);
        int arg(0);
        for (int i = 0; ; i++)
        {
                if (buff[i] == ';')
                {break;};
                if (buff[i] == '=')
                {z++; start = i+1; arg = 0;};
                if (z == 1 && buff[i] != '=') // Первый аргумент
                {
                        arg++;
                        argum = buff.Mid(start,arg);
                        rz = converter(argum);
```

```
                }
            };
        }
        else if (buff.Left(5).MakeLower() == "ogol=")
        {
            name= "ogol";
            int z(0);
            int start(0);
            int arg(0);
            for (int i = 0; ; i++)
            {
                if (buff[i] == ';')
                {break;};
                if (buff[i] == '=')
                {z++; start = i+1; arg = 0;};
                if (z == 1 && buff[i] != '=') // Первый аргумент
                {
                    arg++;
                    argum = buff.Mid(start,arg);
                    rz = converter(argum);
                }
            };
        };
}

double CRazmer::converter(CString argum)
{
    double argument(_tstof(argum));
    return argument;
};
```

Файл “ReportM1.cpp”

```
#include "ReportM1.h"
CReportM1::CReportM1(void)
{
}
CReportM1::~CReportM1(void)
{
}
CReportM1::CReportM1(std::vector<CAprObj> input)
{
	CAprObj rep;
	rep.ax = input[0].sbav;
	rep.ay = input[0].rapapr;
	int match(0);
	for (int i = 0; i < input.size(); i++)
	{
		match += input[i].sbav;
	};
	if (match == 0)
	{
		rep.ay = 0;
		rep.ax = 0;
		for (int i = 0; i < input.size(); i++)
		{
			rep.ay += input[i].rapapr;
		};
		sbavki.push_back(rep);
	}
	else
```

```
{
for (int i = 1; i < input.size(); i++)
{
    if(input[i].sbav == input[i-1].sbav && input[i].rapapr == input[i-1].rapapr)
    {
        rep.ax += input[i].sbav;
        rep.ay += input[i].rapapr;
    }
    else
    {
        rep.pol = input[i-1].pol*(-input[i-1].side_pol);
        rep.side_pol = input[i-1].side_pol;
        rep.maxsb = input[i-1].maxsb;
        rep.rapapr = input[i-1].rapapr;

        sbavki.push_back(rep);

        if (sbavki.back().ax == 0 && sbavki.back().ay == 0)
        {sbavki.pop_back();};

        rep.ax = input[i].sbav;
        rep.ay = input[i].rapapr;
    };
};

if (sbavki.back().ax == 0 && sbavki.back().ay == 0)
{sbavki.pop_back();};
rep.side_pol = input.back().side_pol;
rep.pol = input.back().pol*(-input.back().side_pol);
```

```
sbavki.push_back(rep);
std::vector<CAprObj> sbavki2;
int exit(0);
for (;;)
{
	rep.ax = sbavki[0].ax;
	rep.ay = sbavki[0].ay;
	rep.pol = sbavki[0].pol;
	for (int i = 1; i < sbavki.size(); i++)
	{
		if (sbavki[i].ay == sbavki[i-1].ay && sbavki[i-1].ax == 0 || sbavki[i-1].ax == 0 && sbavki[i].ax == 0
			|| sbavki[i].ay == 1 && sbavki[i-1].ax == 0 || sbavki[i].ay == sbavki[i-1].ay && sbavki[i-1].ax == sbavki[i].ax)
		{
			rep.ay += sbavki[i].ay;
			rep.ax += sbavki[i].ax;
			rep.pol = sbavki[i].pol;
			exit++;
		}
		else
		{
			sbavki2.push_back(rep);

			rep.ay = sbavki[i].ay;
			rep.ax = sbavki[i].ax;
			rep.pol = sbavki[i].pol;
		};
	};
```

```
                    if (sbavki.back().ay == sbavki[sbavki.size() - 2].ay &&
sbavki[sbavki.size() - 2].ax == 0)
                    {
                        rep.ay += sbavki.back().ay;
                        rep.ax = sbavki.back().ax;
                        rep.pol = sbavki.back().pol;
                        sbavki2.push_back(rep);
                    }
                    else
                    {
                        sbavki2.push_back(sbavki.back());
                    };
                    sbavki.clear();
                    for (int i = 0; i < sbavki2.size(); i++)
                    {
                        sbavki.push_back(sbavki2[i]);
                    };
                    sbavki2.clear();
                    if (exit == 0)
                    {break;}
                    exit = 0;
                };
            };
        }
```

Файл “ReverseObj.cpp”

```
#include "ReverseObj.h"
CReverseObj::CReverseObj(void)
{
```

```
}
CReverseObj::~CReverseObj(void)
{
}
CReverseObj::CReverseObj(CObjects Obj)
{
	if (Obj.st == 1)
	{
		if (Obj.y1 > Obj.y0)
		{
			x0 = Obj.x1;
			y0 = Obj.y1;
			x1 = Obj.x0;
			y1 = Obj.y0;
			st = 1;
			p = 1;
			name = Obj.name;
		}
		else if (Obj.y1 < Obj.y0)
		{
			x0 = Obj.x0;
			y0 = Obj.y0;
			x1 = Obj.x1;
			y1 = Obj.y1;
			st = 1;
			p = 1;
			name = Obj.name;
		}
		else
		{
```

```
                x0 = Obj.x0;
                y0 = Obj.y0;
                x1 = Obj.x1;
                y1 = Obj.y1;
                st = 1;
                p = 1;
                name = Obj.name;
        };
}
else if (Obj.st == 2 || Obj.st == 222)
{
        if (Obj.y0 < Obj.y2)
        {
                x0 = Obj.x2;
                y0 = Obj.y2;
                x1 = Obj.x1;
                y1 = Obj.y1;
                x2 = Obj.x0;
                y2 = Obj.y0;
                st = 2;
                p = 1;
                name = Obj.name;
        }
        else if (Obj.y0 > Obj.y2 || Obj.y2 == Obj.y0)
        {
                x0 = Obj.x0;
                y0 = Obj.y0;
                x1 = Obj.x1;
                y1 = Obj.y1;
                x2 = Obj.x2;
```

```
                y2 = Obj.y2;
                st = 2;
                p = 1;
                name = Obj.name;
        }
        else
        {
                x0 = Obj.x0;
                y0 = Obj.y0;
                x1 = Obj.x1;
                y1 = Obj.y1;
                x2 = Obj.x2;
                y2 = Obj.y2;
                st = 2;
                p = 1;
                name = Obj.name;
        }
}
else if (Obj.st == 3)
{
        if (Obj.y0 < Obj.y3)
        {
                x0 = Obj.x3;
                y0 = Obj.y3;
                x1 = Obj.x2;
                y1 = Obj.y2;

                x2 = Obj.x1;
                y2 = Obj.y1;
                x3 = Obj.x0;
```

```
        y3 = Obj.y0;
        st = Obj.st;
        p = 1;
        name = Obj.name;
}
else if (Obj.y0 > Obj.y3 || Obj.y3 == Obj.y0)
{
        x0 = Obj.x0;
        y0 = Obj.y0;
        x1 = Obj.x1;
        y1 = Obj.y1;
        x2 = Obj.x2;
        y2 = Obj.y2;
        x3 = Obj.x3;
        y3 = Obj.y3;
        st = Obj.st;
        p = 1;
        name = Obj.name;
}
else
{
        x0 = Obj.x0;
        y0 = Obj.y0;
        x1 = Obj.x1;
        y1 = Obj.y1;

        x2 = Obj.x2;
        y2 = Obj.y2;
        x3 = Obj.x3;
        y3 = Obj.y3;
```

```
                    st = Obj.st;
                    p = 1;
                    name = Obj.name;
                };
        };
}
```

Файл “RzR.cpp”

```
#include "Designer k-wear.h"
#include "RzR.h"
#include "afxdialogex.h"
// диалоговое окно CRzR
IMPLEMENT_DYNAMIC(CRzR, CDialog)
CRzR::CRzR(CWnd* pParent /*=NULL*/)
        : CDialog(CRzR::IDD, pParent)
{
}
CRzR::~CRzR()
{
}
void CRzR::DoDataExchange(CDataExchange* pDX)
{
        CDialog::DoDataExchange(pDX);
}
BEGIN_MESSAGE_MAP(CRzR, CDialog)
END_MESSAGE_MAP()
```

Файл “Setup.cpp”

```
#include "Designer k-wear.h"
#include "Setup.h"
#include "afxdialogex.h"
```

```
// диалоговое окно CSetup
IMPLEMENT_DYNAMIC(CSetup, CDialog)
CSetup::CSetup(CWnd* pParent /*=NULL*/)
	: CDialog(CSetup::IDD, pParent)
{
}
CSetup::~CSetup()
{
}
void CSetup::DoDataExchange(CDataExchange* pDX)
{
	CDialog::DoDataExchange(pDX);
}
BEGIN_MESSAGE_MAP(CSetup, CDialog)
END_MESSAGE_MAP()
// обработчики сообщений CSetup
void CSetup::OnCbnSelchangeCombo1()
{
}
```

Файл “ZoomWindow.cpp”

```
#include "ZoomWindow.h"
CZoomWindow::CZoomWindow()
{
 win=CSize(1,1);                          // начальные установки
 view=CSize(1,1);
}
CZoomWindow::~CZoomWindow()
{
}
```

```
void CZoomWindow::ZoomIn()                          // больше
{
 if (win==CSize(1,1)) view+=CSize(1,1);             // если размер окно достиг минимума увеличивать разрешение
 else win-=CSize(1,1);                              // иначе менять размер окна

}
void CZoomWindow::ZoomOut()                         // меньше
{
 if (view==CSize(1,1)) win+=CSize(1,1);             // если разрешение достигло минимума менять размер окна
 else view-=CSize(1,1);                             // иначе уменьшать разрешение
}
CSize CZoomWindow::GetWin()
{
 return win;                                        // вернуть размер окна
}
CSize CZoomWindow::GetView()
{
 return view;                                       // вернуть разрешение
};
```

Федеральное государственное бюджетное образовательное учреждение
высшего профессионального образования
«Московский государственный университет дизайна и технологии»

На правах рукописи

Ланшаков Денис Евгеньевич

Разработка технологий и конструкций сложных цельновязаных изделий на базе комплексной автоматизированной системы

Специальность 05.19.02 – Технология и первичная обработка текстильных материалов и сырья

ПРИЛОЖЕНИЕ 2
диссертации на соискание ученой степени
кандидата технических наук

Научный руководитель –
доктор технических наук
профессор Колесникова Елена Николаевна

Москва – 2014 г.

ОГЛАВЛЕНИЕ

ВВЕДЕНИЕ

В настоящее время на рынке промышленного программного обеспечения, используемого в текстильном производстве, осуществляется основной упор на узкоспециализированную область применения, например, выполнение конструирования и последующего вывода лекал, при использовании мощных инструментов конструирования (швейное производство) с помощью графопостроителя (плоттера) или проектирования технологии вязания (трикотажное производство) с модулем ввода лекал без возможности полноценного конструирования и т.д. Однако совместное использование систем проектирования из разных типов производств допускается, но представляет собой слишком затратное мероприятие, в стоимость которого войдет не только оплата высококвалифицированного программиста (например, для создания конвертера), но и покупка двух систем проектирования. Причем полная функциональная эксплуатация этих систем будет невозможна из-за ненадобности некоторых функций и модулей, например, оформление уголков в лекалах, что необходимо для последующей качественной швейной обработки на швейном производстве, нецелесообразно использовать при проектировании цельновязаного изделия или изделия с малой плотностью трикотажа на трикотажном производстве.

Программное обеспечение “DESIGNER K-WEAR” является дополнительным техническим решением, позволяющим расширить возможности поставляемых производителями вязального оборудования систем проектирования.

1 ЭЛЕМЕНТЫ УПРАВЛЕНИЯ И РАБОЧАЯ ОБЛАСТЬ ПОСТРОЕНИЯ В “DESIGNER K-WEAR”

На рисунках П2.1, П2.2, П2.3 представлены основные элементы управления, а также область построения (конструирования).

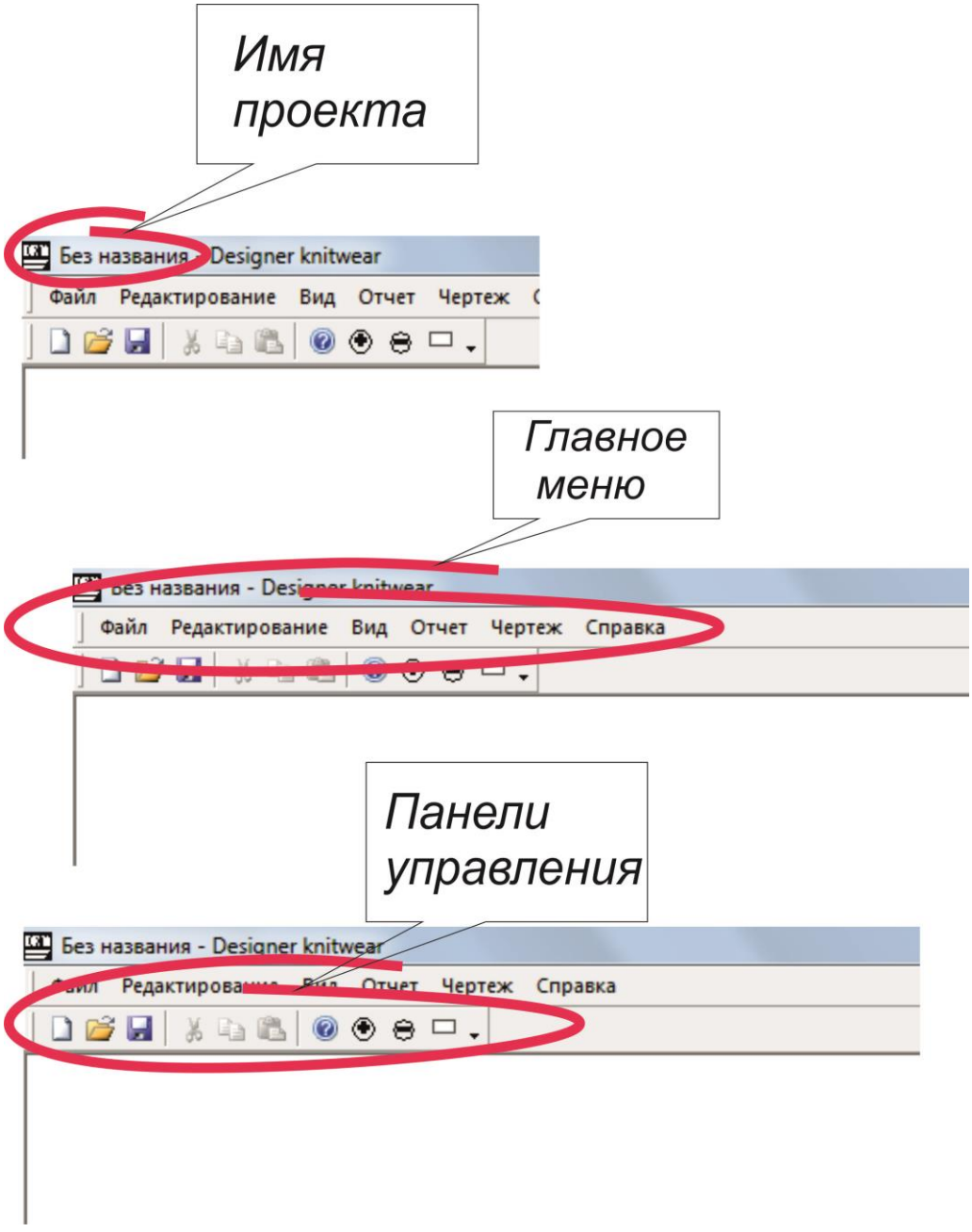


Рисунок П2.1 – Главные элементы управления

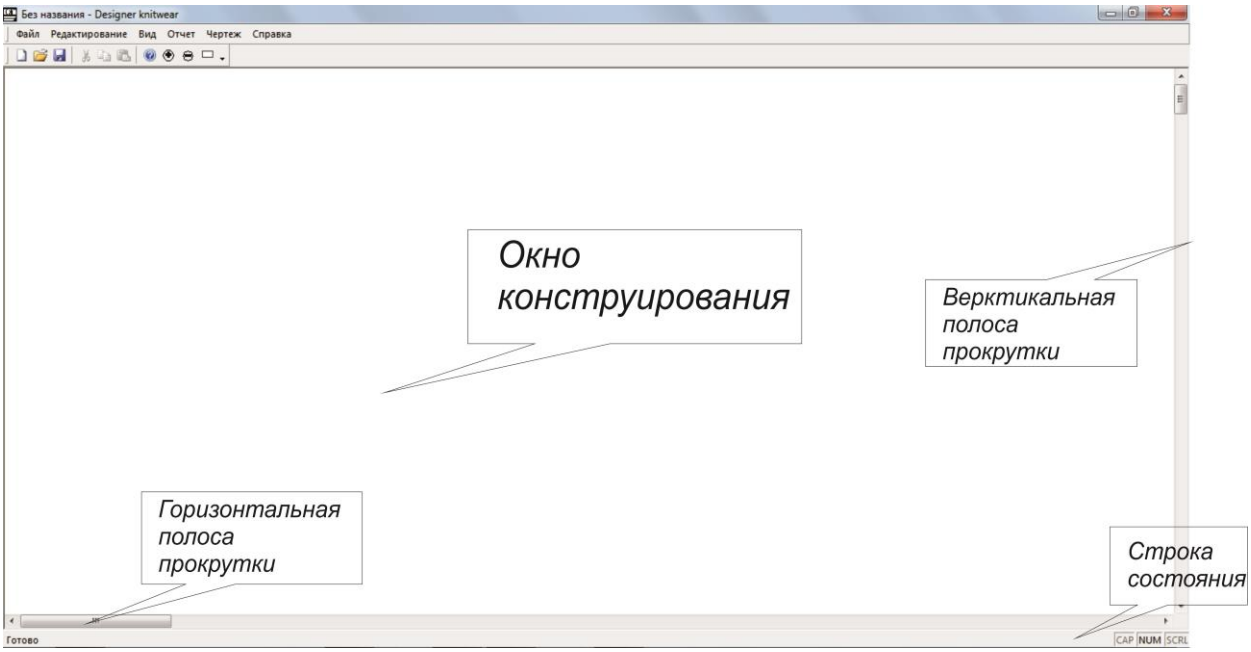


Рисунок П2.2 – Элементы отображения графики и состояния

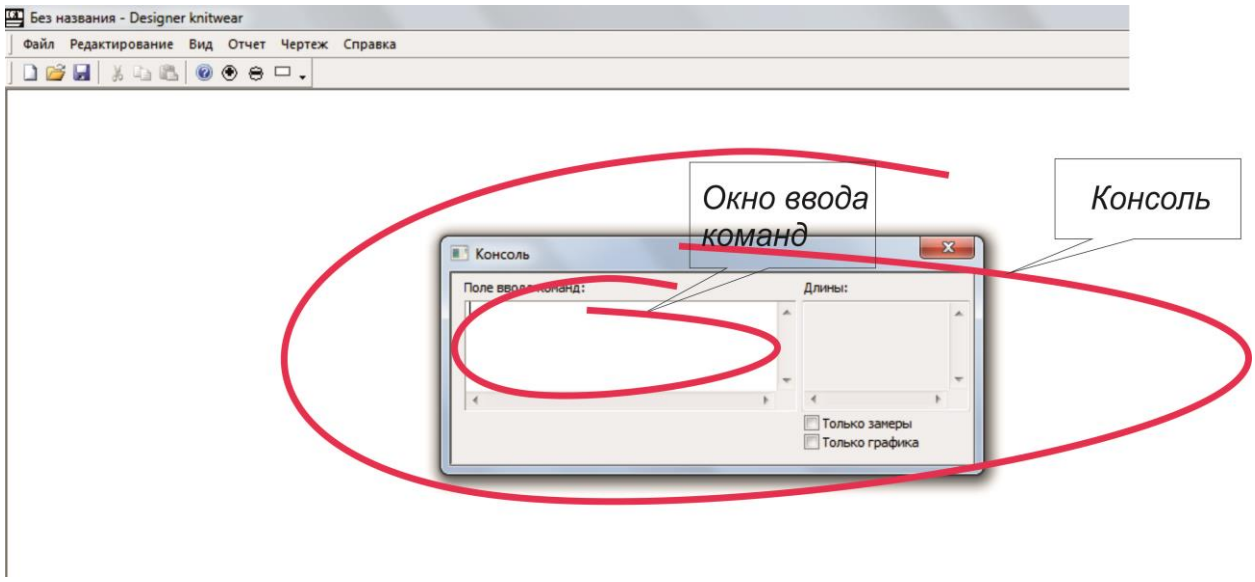


Рисунок П2.3 – Консоль и поле ввода команд с клавиатуры

Описание остальных функциональных элементов представлено в таблице П2.1.

Таблица П2.1 – Описание функциональных элементов “DESIGNER K-WEAR”

№	Название элемента	Доступ к элементу	Описание	Дополнительно
1	ФАЙЛ	ФАЙЛ	Включает в себя элементы по созданию, сохранению и т.д. файлов.	Пример: ФАЙЛ > Создать (создает новый файл “Designer k-wear”)
2	Создать	ФАЙЛ > Создать	Создает новый файл “Designer k-wear”	----------
3	Открыть	ФАЙЛ > Открыть	Открывает заранее созданный файл “Designer k-wear”	----------
4	Сохранить	ФАЙЛ > Сохранить	Сохраняет текущий файл “Designer k-wear”	----------
5	Сохранить как…	ФАЙЛ > Сохранить как…	Позволяет сохранить файл “Designer k-wear” под новым именем	----------
6	Выход	ФАЙЛ > Выход	Выполняет выход в ОС Windows из “Designer k-wear”	----------
7	Редактирование	Редактирование	Включает в себя элементы по сглаживанию сбавок и конвертированию кромки	----------

Продолжение таблицы П2.1

№	Название элемента	Доступ к элементу	Описание	Дополнительно
8	Сглаживание сбавок	Редактирование > Сглаживание сбавок	Сглаживает сбавки (прибавки) при незначительном изменении. Например, при совершении сбавки и следующей прибавки на прямом участке кромки.	Опция включена по умолчанию.
9	M1(report): side(R) → side(L)	Редактирование > M1(report): side(R)→side(L)	Позволяет аналитически преобразовать правую кромку в левую. Например, при переносе результатов аппроксимации правой кромки в САПР М1 компании STOLL при этом вводя симметричные лекала.	Опция выключена по умолчанию.
10	Вид	Вид	Включает в себя элементы по отображению данных проектирования их некоторых инструментов	---------
11	Строка состояния	Вид > Строка состояния	Показывает (скрывает) строку состояния	---------
12	Консоль	Вид > Консоль	Отображает консоль клавиатурного ввода команд	Окно Консоль содержит в себе элемент отображения длин и величин, используемых объектов в алгоритме; элемент «Только графика» отображает только замеры графических объектов, элемент «Только замеры» - только используемые величины (например, переменные)
13	Представление	Вид > Представление	Включает в себя режимы отображения данных	Подменю

Продолжение таблицы П2.1

14	Графика	Вид > Представление > Графика	Режим отображения только линий для конструирования	Режим включен по умолчанию
15	Петли	Вид > Представление > Петли	Режим отображения только результатов аппроксимации	
16	Графика+Петли	Вид > Представление > Графика+Петли	Комбинированный режим отображения линий конструирования и результатов аппроксимации	Позволяет выполнить сравнение кривых (первоначальной с аппроксимированной)
17	Информация	Вид > Информация	Включает в себя режимы отображения вспомогательной информации	Подменю
18	Петли	Вид > Информация > Петли	Режим отображения результатов аппроксимации в окне конструирования	
19	Вершины	Вид > Информация > Вершины	Режим отображения опорных вершин, построенных кривых Безье	
20	Отчет	Отчет	Включает в себя виды отчетов	Необходима предустановленная версия MS Office 2007 или выше
21	Отчет полный	Отчет > Отчет полный	Выводит отчет об аппроксимации в таблицу	
22	Отчет краткий	Отчет > Отчет краткий	Выводит отчет об аппроксимации в таблицу	Возможен последующий импорт данных в САПР STOLL M1 для построения лекал
23	Чертеж	Чертеж	Включает в себя элементы по работе с конструированием	
24	Построить	Чертеж > Построить	Выполняет построение согласно, заранее созданному алгоритму	
25	Масштаб	Чертеж > Масштаб	Включает в себя элементы увеличения и уменьшения масштаба	Подменю

Продолжение таблицы П2.1

26	Увеличить	Чертеж > Масштаб > Увеличить	Увеличивает масштаб объектов в окне конструирования	
27	Уменьшить	Чертеж > Масштаб > Уменьшить	Уменьшает масштаб объектов в окне конструирования	
28	Единицы измерения	Чертеж > Единицы измерения	Включает в себя режимы конструирования по типу метрических единиц	Подменю
29	Сантиметры	Чертеж > Единицы измерения > Сантиметры	Конструирование осуществляется в сантиметровой сетке	Режим включен по умолчанию
30	Миллиметры	Чертеж > Единицы измерения > Миллиметры	Конструирование осуществляется в миллиметровой сетке	
31	Размерные признаки > Загрузить	Чертеж > Размерные признаки > Загрузить	Позволяет загрузить заранее сохраненный файл с размерными признаками	
32	Вытачки	Чертеж > Вытачки	Включает в себя элементы по работе с вытачками	Подменю
33	Открыть	Чертеж > Вытачки > Открыть	Открывает вытачки	Режим включен по умолчанию
34	Закрыть	Чертеж > Вытачки > Закрыть	Закрывает вытачки	
35	Справка	Справка	Включает в себя информационные ресурсы	
36	О проекте Designer k-wear	Справка > О проекте Designer k-wear	Отображает в диалоговом окне данные об авторе	
37	Контекстовое меню окна конструирования	Щелчок правой кнопки мышки на области окна конструирования	Содержит элементы Консоль(12), Построить (24), Вытачки открыть (33), Вытачки закрыть (34), Загрузить РП (31)	

2 ОСНОВНЫЕ ПРИНЦИПЫ ПРОЕКТИРОВАНИЯ И РАБОТЫ В “DESIGNER K-WEAR”

Основными конструкторскими объектами в “DESIGNER K-WEAR”, используемыми для построения контуров лекал, кромок и т.д. являются кривые Безье. Эти кривые представляют собой реверсивные кривые (неважно в каком порядке осуществляется построение) с простой параметрической записью (координаты точки кривой будут зависеть от значения параметра). Причем аналитическая запись кривой содержит в себе координаты опорных вершин, от взаимного расположения в одной плоскости которых будет зависеть форма кривой Безье. При этом, чем сложнее будет кривая, тем более высокую степень она будет иметь и соответственно, большее количество опорных вершин.

Для проектирования конструкторской составляющей “DESIGNER K-WEAR” были проведены исследования, существующих систем проектирования швейных и трикотажных изделий и выполнен анализ известных методик конструирования одежды (Мюллер и сын, методика ЦНИИШП, единый метод ЦОТШЛ и т.д.). В результате было выявлено три типа часто используемых объектов графического построения необходимых для оформления боковых линий, проймы, оката рукава и т.д.

Данные типы кривых были получены с помощью кривых Безье и заложены в основу “DESIGNER K-WEAR” (рисунок П2.4).

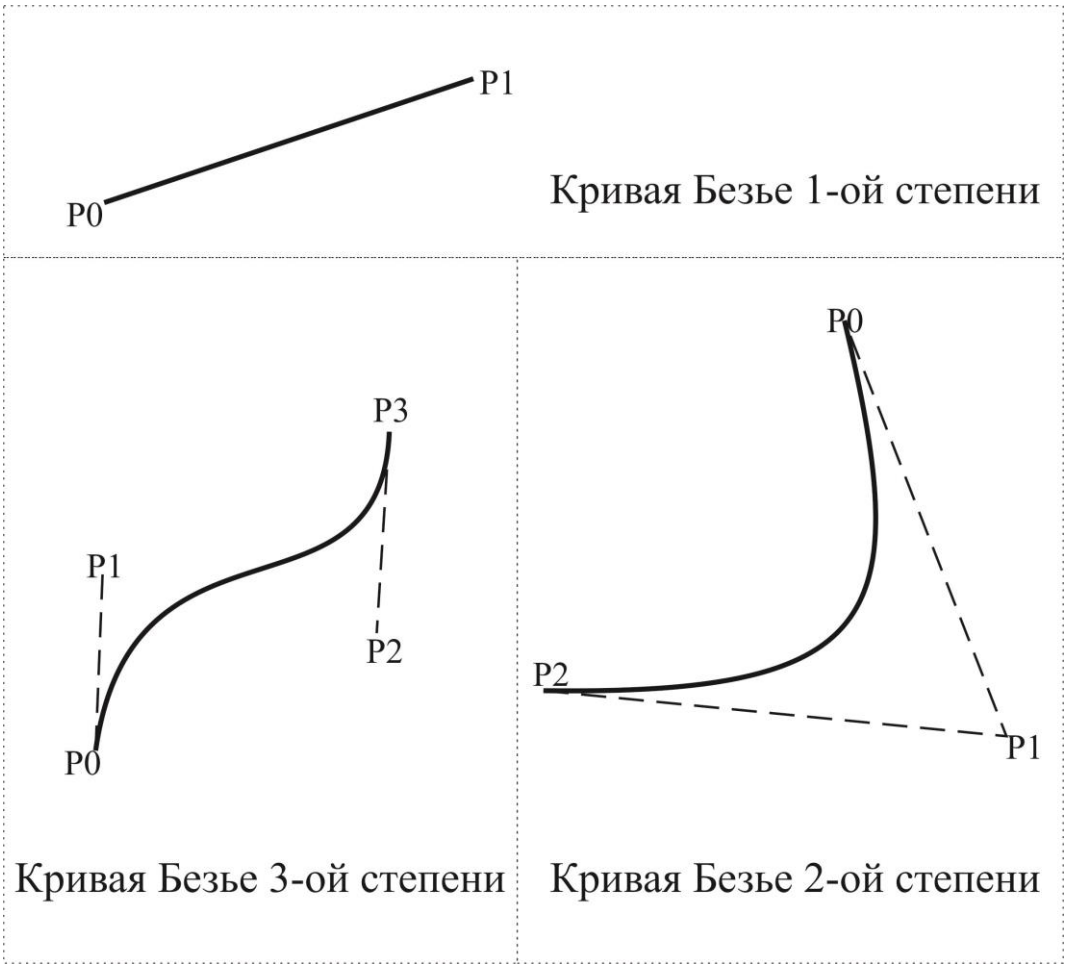


Рисунок П2.4 – Типы кривых Безье, используемых в Designer k-wear

Процесс проектирования в “DESIGNER K-WEAR” представляет собой конструирование и последующий расчет сбавок (прибавок), необходимых для вязания купона (рисунок П2.5). Причем результат выводится в формате электронной таблицы двух типов: подробный (выводятся все точки аппроксимации, сбавки, направления сбавок, точки, лежащие на кривой и т.д.) и сокращенный тип (выводятся значения аппроксимации в виде сформированных ступеней по петельным рядам и столбикам).

Конструирование и проектирование вязания в “DESIGNER K-WEAR” осуществляется при помощи составления параметрического алгоритма с использованием заранее известных команд построения объектов и их аппроксимации.

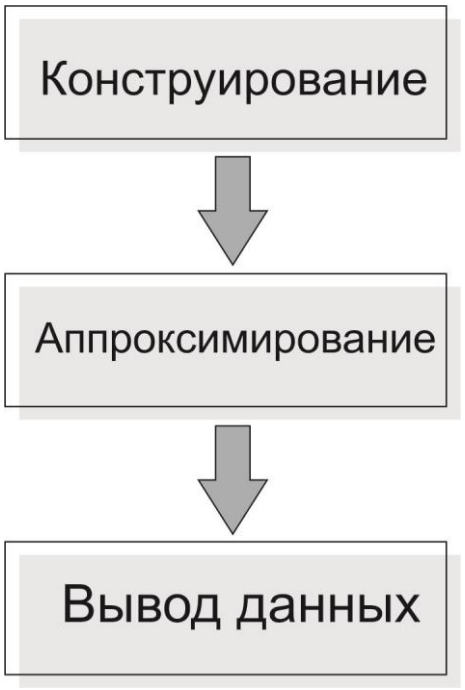


Рисунок П2.5 – Этапы проектирования в “DESIGNER K-WEAR”

Рассмотрим общий вид команды:

ИМЯ : ФУНКЦИЯ > ПАРАМЕТРЫ;

где *ИМЯ* – часть команды, при помощи которой происходит дальнейшая идентификация объектов внутри алгоритма (идентификатор);

ИМЯ может включать в себя любое количество символов в различных вариациях, за исключением имен, зарезервированных под специальные команды и некоторые символы.

Далее идет оператор присвоения объекту идентификатора – “:” .

ФУНКЦИЯ – часть команды, необходимая для создания графического объекта, команды аппроксимации и т.д.

“>” – оператор присвоения *ФУНКЦИИ* параметров.

ПАРАМЕТРЫ - числовые или текстовые данные, необходимые для создания графического объекта, команды аппроксимации и т.д.

“;” – оператор окончания команды. Без данного оператора команда будет считаться незавершенной, и её выполнение будет недопустимо.

Рассмотрим пример:

Line1:line>0,0,10,10;
Line1:aprx>0.3,0.23;

В данном примере приведена команда построения отрезка с именем Line1, с помощью функции line и параметрами, являющимися координатами начала и конца отрезка. А также дальнейшей аппроксимации объекта Line1 с помощью функции aprx.

Таким образом, команда построения объекта должна обязательно включать в себя имя (идентификатор) объекта, которое будет использоваться при дальнейшем редактировании или корректировании объекта, его аппроксимации и т.д., функцию (line или aprx), отвечающую за сам вид объекта и параметры функции, непосредственно определяющие характеристики объекта, характер аппроксимации и т.д.

3 ПОСТРОЕНИЕ ПРОСТЫХ ГРАФИЧЕСКИХ ОБЪЕКТОВ

Построение, как было установлено раннее, осуществляется при помощи ввода текстовых команд с помощью клавиатуры. Точка начала координат в “DESIGNER K-WEAR” находится в левом верхнем углу окна конструирования (рисунок П2.6). Положительные направления осей координат направлены слева направо (ось абсцисс) и сверху вниз (ось ординат). Точка О на рисунке П2.6 является точкой начала координат.

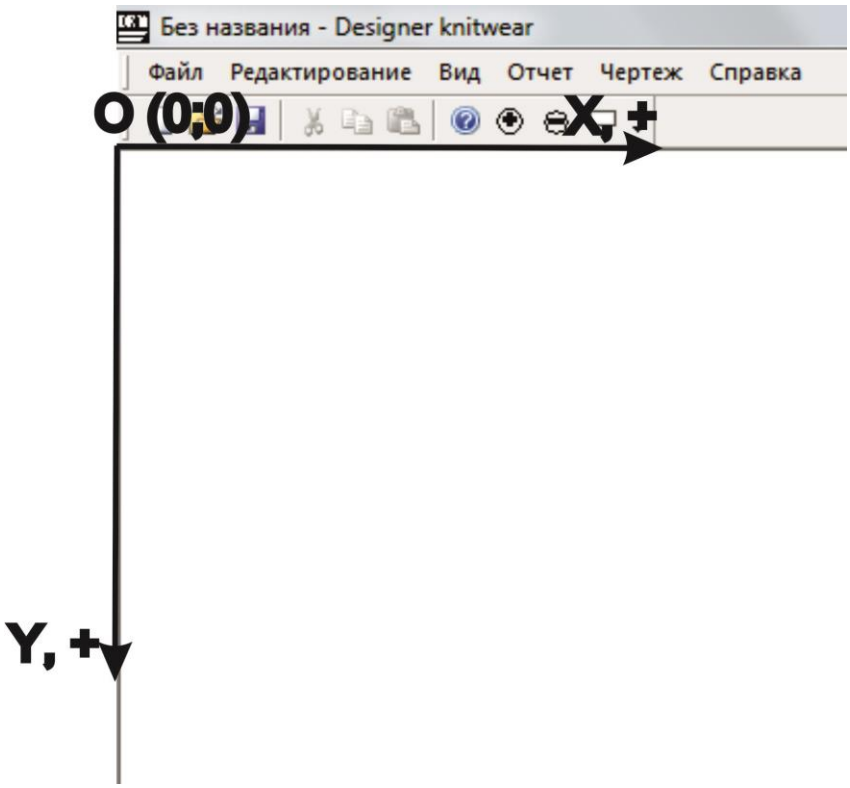


Рисунок П2.6 – Положение начала координат и направления осей относительно окна конструирования “DESIGNER K-WEAR”

Самым простым графическим объектом в “DESIGNER K-WEAR” является точка. Команда построения данного объекта включает в себя по принятой унификации команд идентификатор, функцию построения и параметры функции построения:

ИМЯ : point > x0, y0;

где *point* – функция построения точки;

x0, y0 – координаты положения точки.

Рассмотрим пример построения точки:

tochka : point >2,2;

В данном примере точка с именем tochka имеет координаты (2;2). построенная точка представлена на рисунке П2.7.

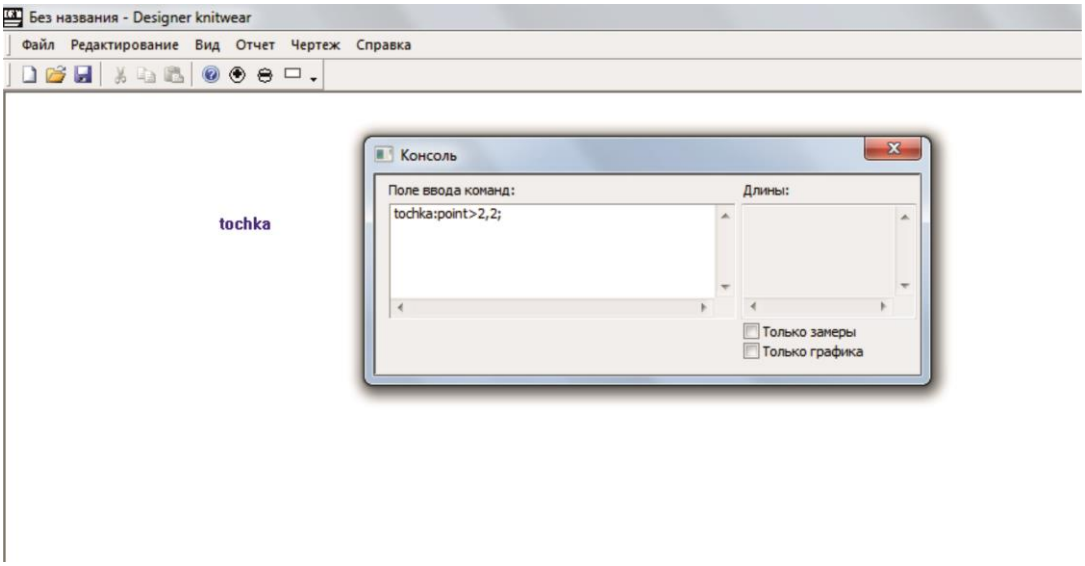


Рисунок П2.7 – Построение точки

Для *построения отрезка*, как и в случае с точкой, необходимо задать идентификатор, функцию построения и параметры функции построения:

ИМЯ : line > x0, y0, x1, y1;

где *line* – функция построения отрезка;

x0, y0 – координаты точки начала отрезка;

x1, y1 – координаты точки конца отрезка.

Рассмотрим пример построения отрезка:

Otrezok:line>2,2,3,5;

В данном примере отрезок с именем Otrezok имеет начало в точке с координатами (2;2) и конец в точке с координатами (3;5). Построенный отрезок представлен на рисунке П2.8.

Для того чтобы построить объект необходимо выбрать соответствующий пункт меню:

Чертеж -> Построить или щелкнуть на пустом месте окна конструирования левой кнопкой мыши или на пустом месте окна конструирования щелкнуть правой кнопкой мыши и выбрать в контекстовом меню пункт *Построить*.

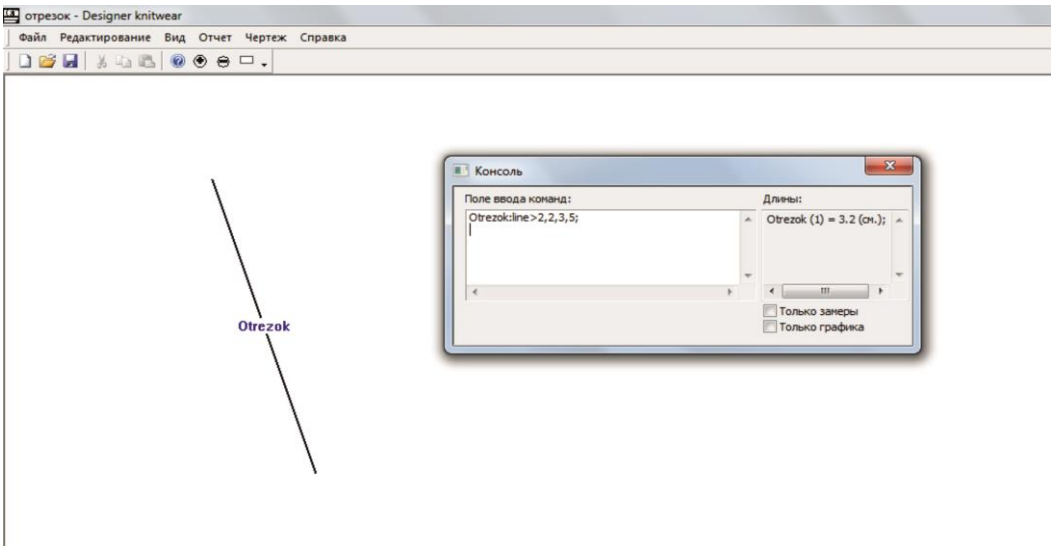


Рисунок П2.8 – Построение отрезка

<u>Построение кривой Безье второго порядка:</u>

ИМЯ : bzr2 > x0, y0, x1, y1, x2, y2;

где *bzr2* – функция построения кривой Безье 2-го порядка;

x0, *y0* – начальная вершина кривой;

x1, *y1* – промежуточная вершина кривой;

x2, *y2* – конечная вершина кривой.

Рассмотрим пример построения кривой Безье второго порядка:

Bezie:bzr2>2,2,5,6,4,4;

Построенная кривая Безье второго порядка представлена на рисунке П2.9.

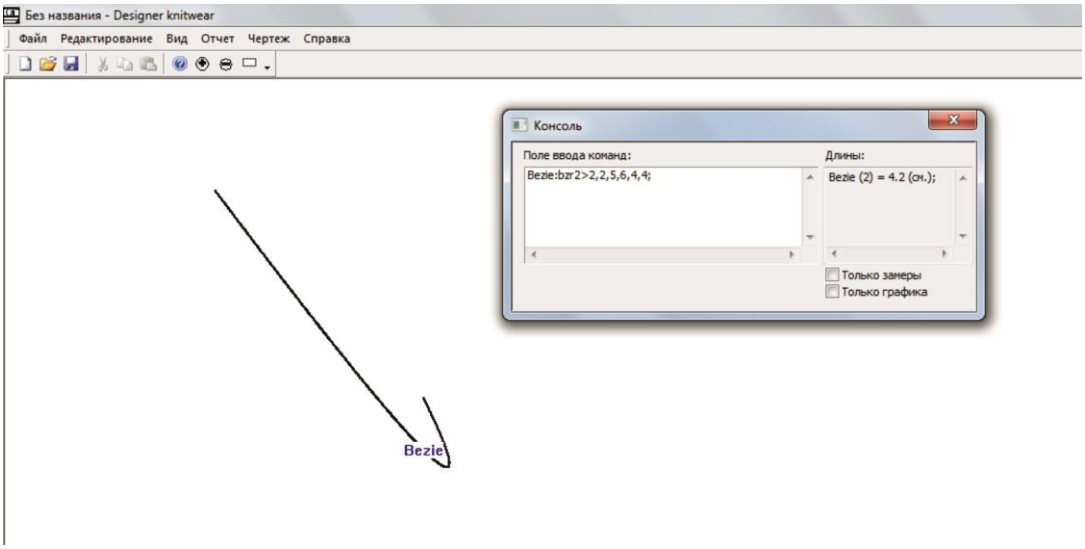


Рисунок 9 – Построение кривой Безье второго порядка

Построение кривой Безье 3-го порядка:

ИМЯ : bzr3 > x0, y0, x1, y1, x2, y2, x3, y3;

где *bzr3* – функция построения кривой Безье 3-го порядка;

x0, *y0* – начальная вершина кривой;

x1, *y1*, *x2*, *y2* – промежуточные вершины кривой;

x3, *y3* – конечная вершина кривой.

Рассмотрим пример построения кривой Безье третьего порядка:

Bezie:bzr3>2,2,5,6,4,4,1,8;

Построенная кривая Безье третьего порядка представлена на рисунке П2.10.

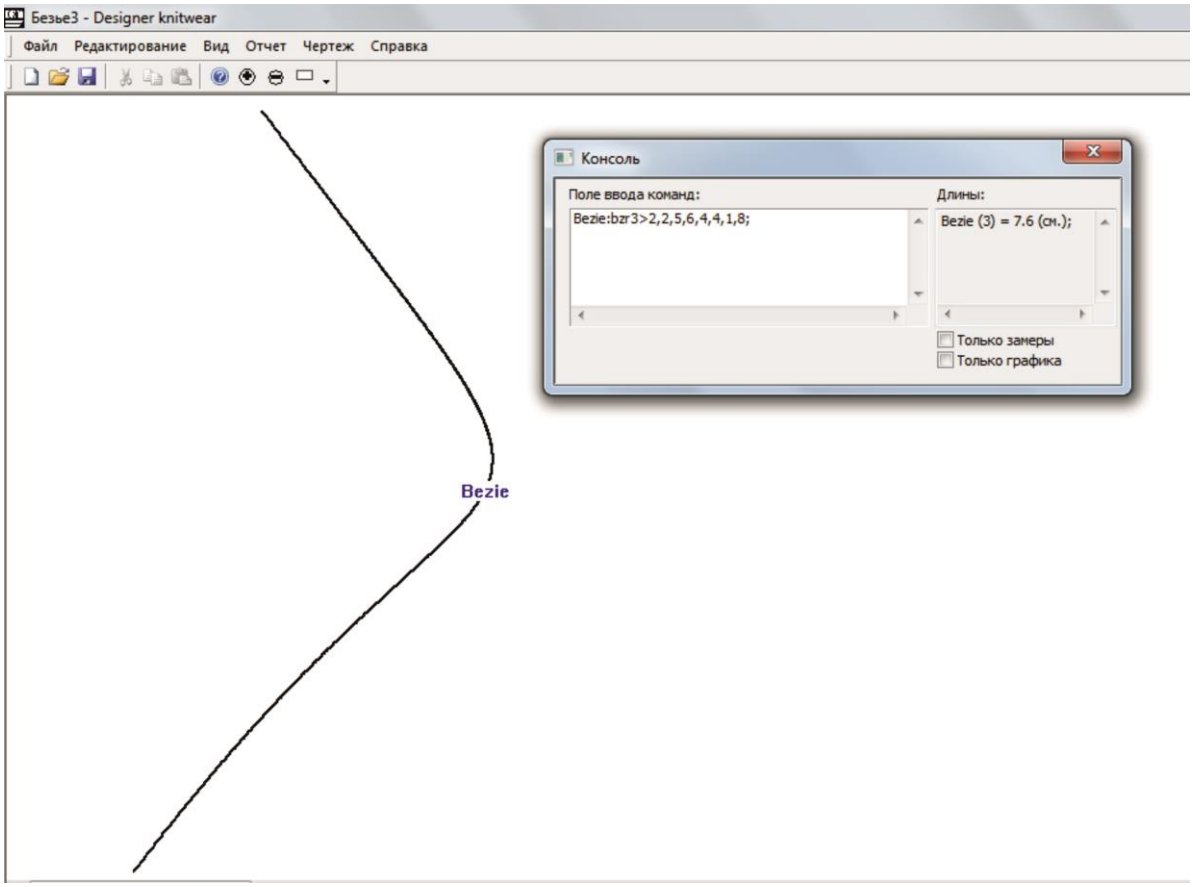


Рисунок 10 – Построение кривой Безье 3-го порядка

Построение точки за счет пересечения двух окружностей:

ИМЯ : point_cc >x1, y1, r1, x2, y2, r2, s;

где *point_cc* – функция построения точки за счет пересечения двух окружностей;

x1, *y1* – координаты точки центра первой окружности;

r1 – радиус первой окружности;

x2, *y2* – координаты точки центра второй окружности;

r2 – радиус второй окружности;

s – точка пересечения (так как результатом пересечения двух окружностей будут две точки, то параметр *s* определяет левую (при s = -1) или правую (при s = 1) точку пересечения).

Пример (рисунок П2.11):

Per_pravoe:point_cc>2,2,5,5,5,1,1;

Per_levoe:point_cc>2,2,5,5,5,1,-1;

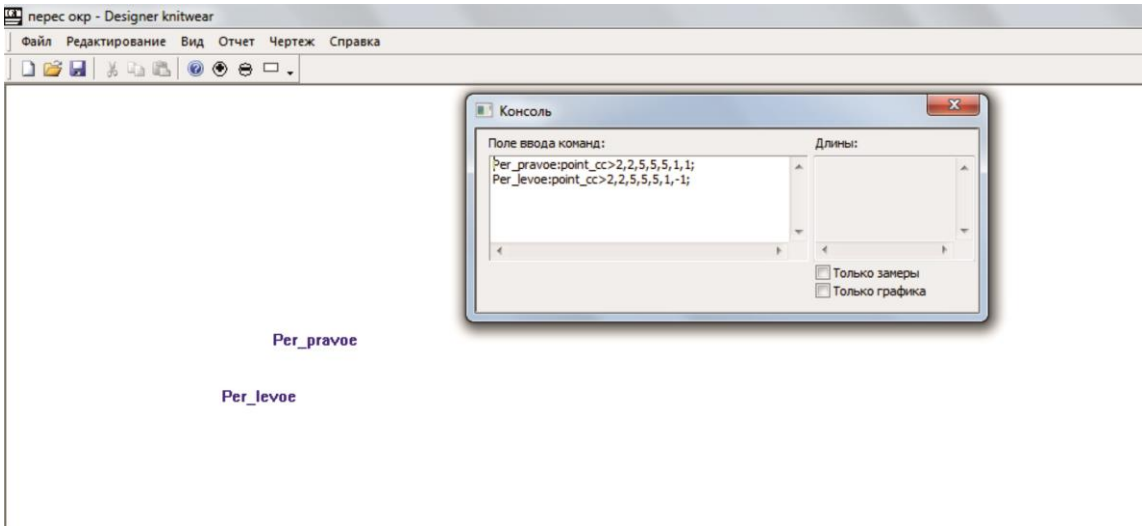


Рисунок П2.11 – Точки пересечения окружностей

Построение точки за счет её откладывания на каком-либо графическом объекте (например, отрезке, кривой Безье 3-го порядка и т.д.)

ИМЯ : point_to> ИМЯ2, long, start_point;

где *point_to* – функция построения точки за счет её откладывания на каком-либо графическом объекте;

ИМЯ2 – имя объекта на котором предстоит отложить точку;

long – откладываемая длина;

start_point – идентификатор точки от которой следует её отложить (т.е. указывает старт откладывания от начала построения объекта на котором она откладывается или от его конца).

start_point = p0 – откладывание от начала построения объекта (для всех объектов);

start_point = p1 – откладывание от конца построения отрезка;

start_point = p2 – откладывание от конца построения кривой Безье 2-го порядка;

start_point = p3 – откладывание от конца построения кривой Безье 3-го порядка;

Приведем пример с кривой Безье 2-го порядка, в котором построим данную кривую и отложим две точки с разных её концов построения (рисунок П2.12).

Пример:

Curve:bzr2>2,2,5,5,5,2;

Point1:point_to>Curve, 2, p0;

Point2:point_to>Curve, 1.5, p2;

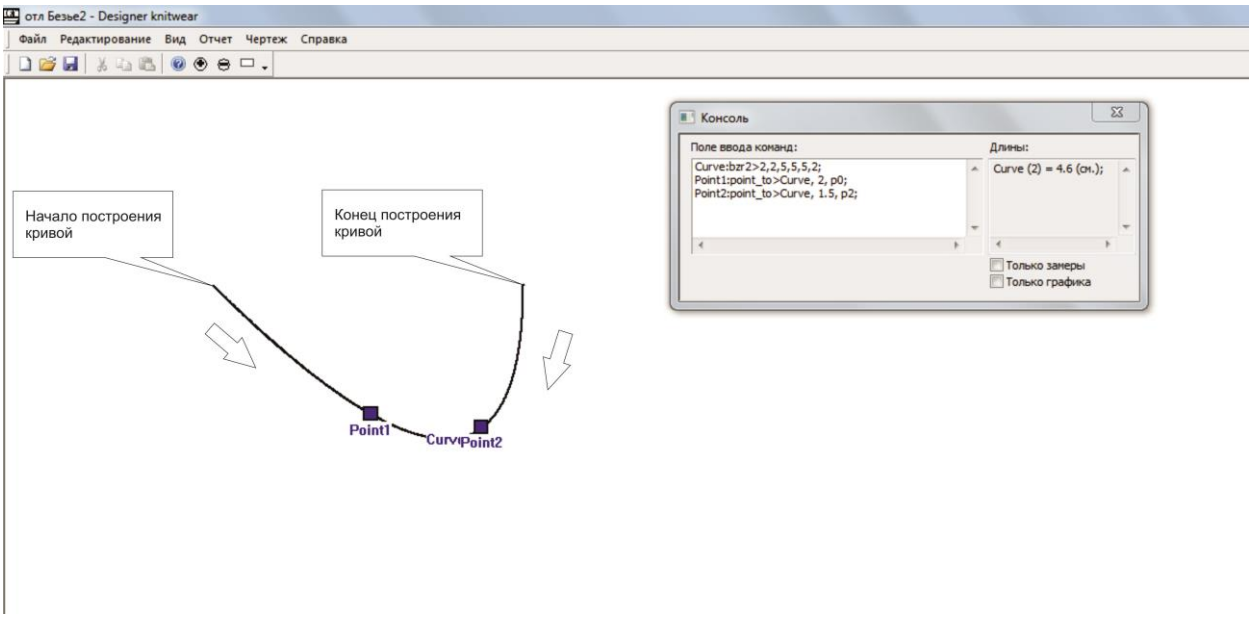


Рисунок П2.12 – Построение точки за счет её откладывания в объекте

Согласно рисунку П2.12 точка Point1 находится на кривой Curve в двух сантиметрах от точки начала построения кривой Curve по длине, точка Point2 – в полутора сантиметрах.

Построение отрезка как вектора:

ИМЯ : line_to> x0, y0, Ug, Lng;

где *line_to* – функция построения отрезка;

x0, *y0* – координаты точки начала вектора;

Ug – угол, под которым откладывается отрезок относительно оси абсцисс;

Lng –длина откладываемого отрезка.

Пример (рисунок П2.13):

Отрезок1:line_to>2,2,0,3;

Отрезок2:line_to>2,2,15,3

Отрезок3:line_to>2,2,30,3;

Отрезок4:line_to>2,2,38,3

Отрезок5:line_to>2,2,54,3;

Отрезок6:line_to>2,2,90,3;

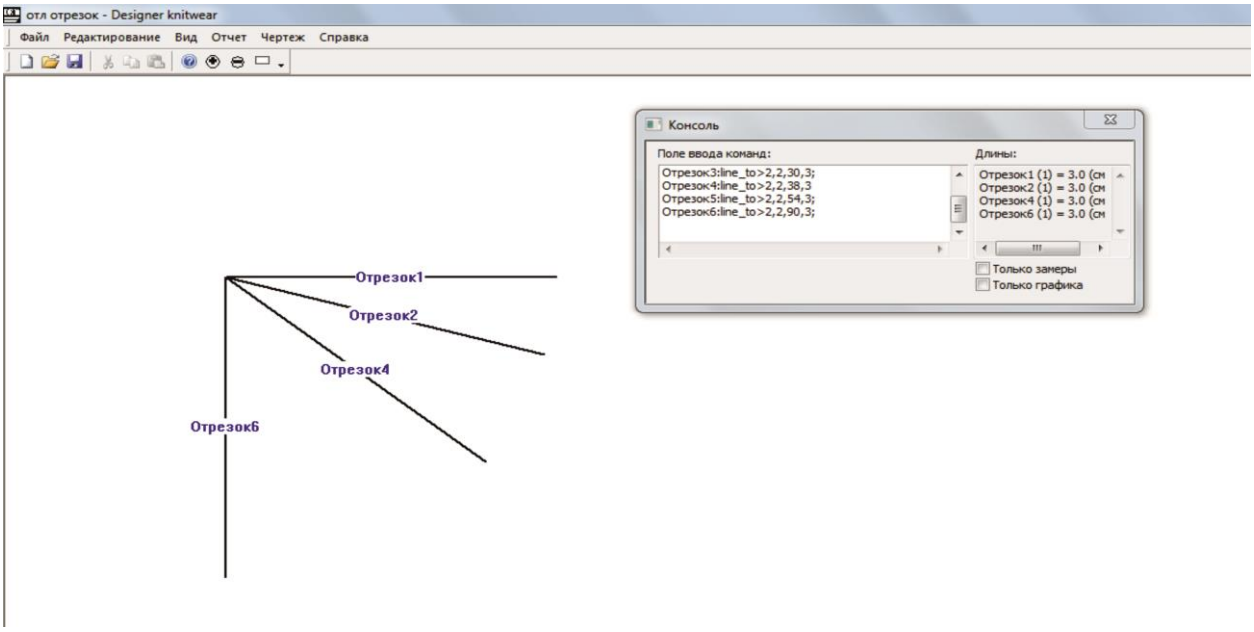


Рисунок П2.13 – Построение отрезков одинаковой длины под разным углом

Построение кривых Безье 2-го и 3-го порядков при оцифровывании лекал.

Данный метод построения будет полезен при оцифровывании аналоговых (картонных) лекал, готовых изделий и т.п. Для этого потребуется определить координаты ключевых точек кривой и её длину.

1. Определение координат ключевых точек

Чтобы построить кривую необходимо определить координаты начальной и конечной опорных вершин, а также координаты дополнительной точки, лежащей на самой кривой, причем количество дополнительных точек будет зависеть от степени кривой Безье (для кривой 2-го порядка необходима одна дополнительная точка, для кривой 3-го порядка – две дополнительные точки). Для этого устанавливается в любом удобном месте начало системы координат (рисунок П2.14). И уже относительно установленной системы расчитываются координаты вершин и дополнительных точек.

2. Определение параметров дополнительных точек

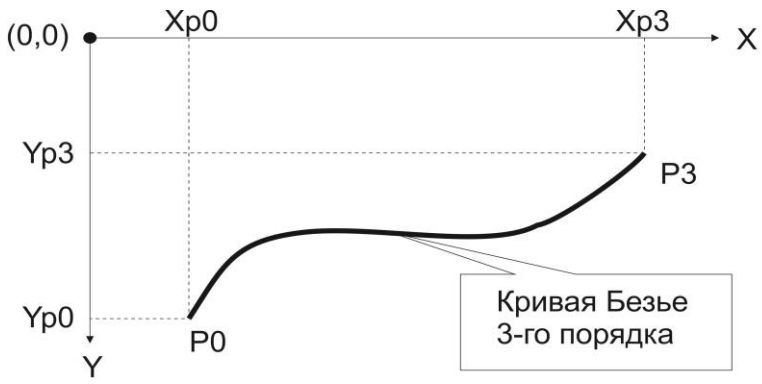
Однако, полученных координат опорных вершин и дополнительных точек для более точного построения кривой Безье будет недостаточно. Так как через эти точки может проходить бесконечное множество кривых, то необходимо введение дополнительного ограничения количества всевозможных вариантов кривых, сводящее множество кривых к одной необходимой. Данным ограничением будет являться параметр, входящий в аналитическую структуру записи кривой Безье, область значения которого находится в пределах от нуля до единицы включительно. Смысл данного параметра указывает на долевое соотношение длины участка строющейся кривой Безье к длине уже построенной, то есть определяет момент прохождения кривой через определенную точку (рисунок П2.14 В). Таким образом, чтобы рассчитать данный параметр необходимо измерить длину участка кривой от начальной опорной вершины до дополнительной точки и разделить её на общую длину кривой:

$$t_i = \frac{L_i}{L_{\text{общ.}}}, \qquad (\text{П2.1})$$

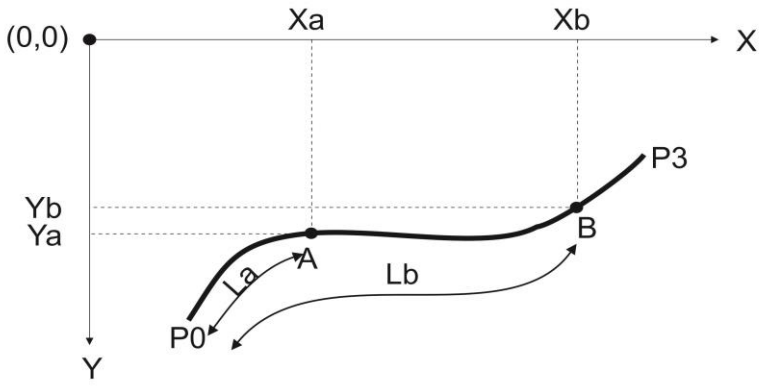
где t_i – параметр дополнительной i-ой точки кривой;

L_i – длина участка кривой от её начальной опорной вершины до i-ой дополнительной вершины;

$L_{общ.}$ – длина полностью построенной кривой Безье.



А. Определение координат начальной и конечной вершин.



B.Определение координат дополнительных точек и их параметров.

Рисунок П2.14 – Определение координат вершин кривой Безье, дополнительных точек и их параметров

В этом случае команды построения кривых будут иметь вид:

Для кривой Безье 2-ой степени

ИМЯ : bzr2i> x0, y0, x2, y2, xa, ya, ta;

где *bzr2i* – функция построения кривой;

x0, y0, x2, y2 – координаты начальной и конечной опорных вершин кривой Безье;

xa, ya – координаты дополнительной точки;

ta –параметр точки.

Для кривой Безье 3-ей степени

ИМЯ : bzr3i> x0, y0, x3, y3, xa, ya, xb, yb, ta, tb;

где *bzr3i* – функция построения кривой;

x0, y0, x3, y3 – координаты начальной и конечной опорных вершин кривой Безье;

xa, ya, xb, yb – координаты дополнительных точек;

ta, tb –параметры точек.

Рассмотрим пример построения кривой Безье 2-го порядка при оцифровывании кривой линии проймы готового изделия - мужского джемпера 54 размера с втачным типом рукава. Все измерения выполняем в сантиметровой сетке.

ШАГ 1. Определим положение начала осей координат относительно проймы изделия (рисунок П2.15).

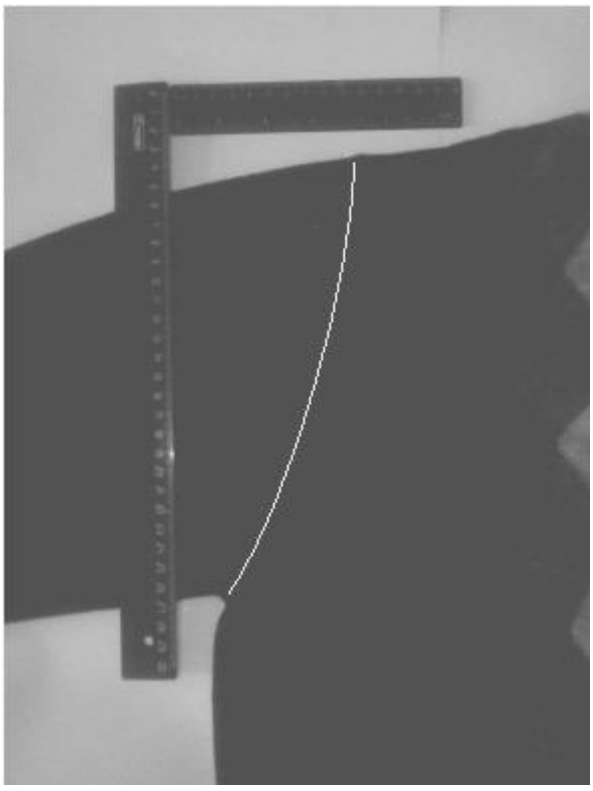


Рисунок П2.15 – Кривая проймы

ШАГ 2. Определим координаты начальной и конечной опорных вершин (рисунки П2.16 и П2.17).

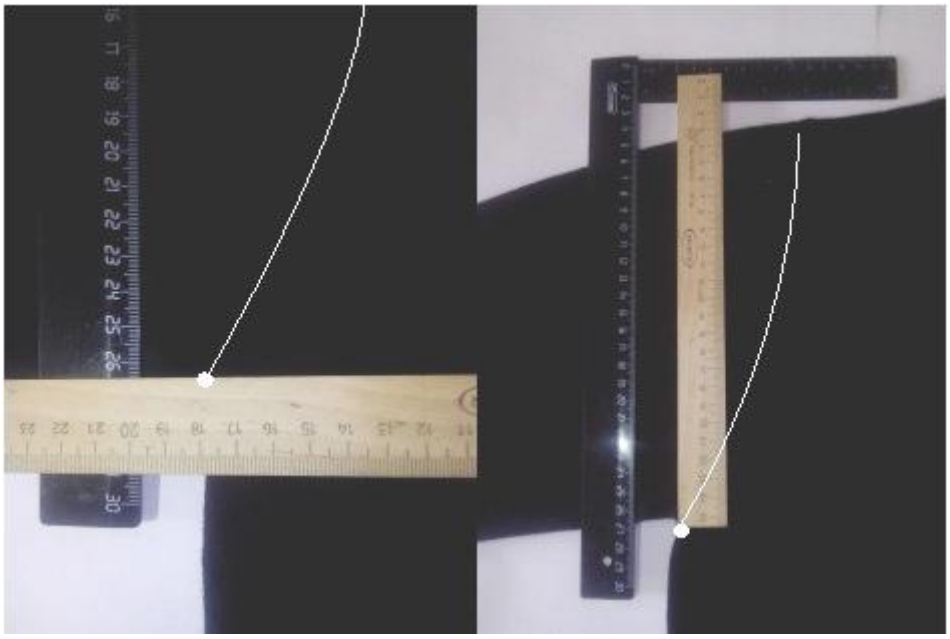


Рисунок П2.16 – Начальная опорная вершина

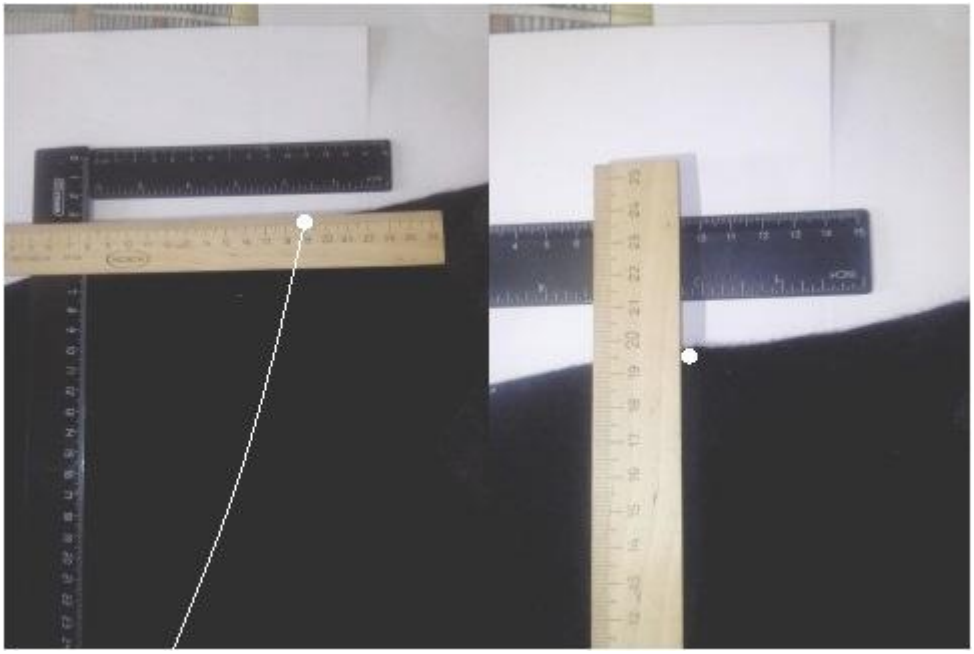


Рисунок П2.17 – Конечная опорная вершина

ШАГ 3. Установим дополнительную точку на кривой и определим её координаты (рисунок П2.18).

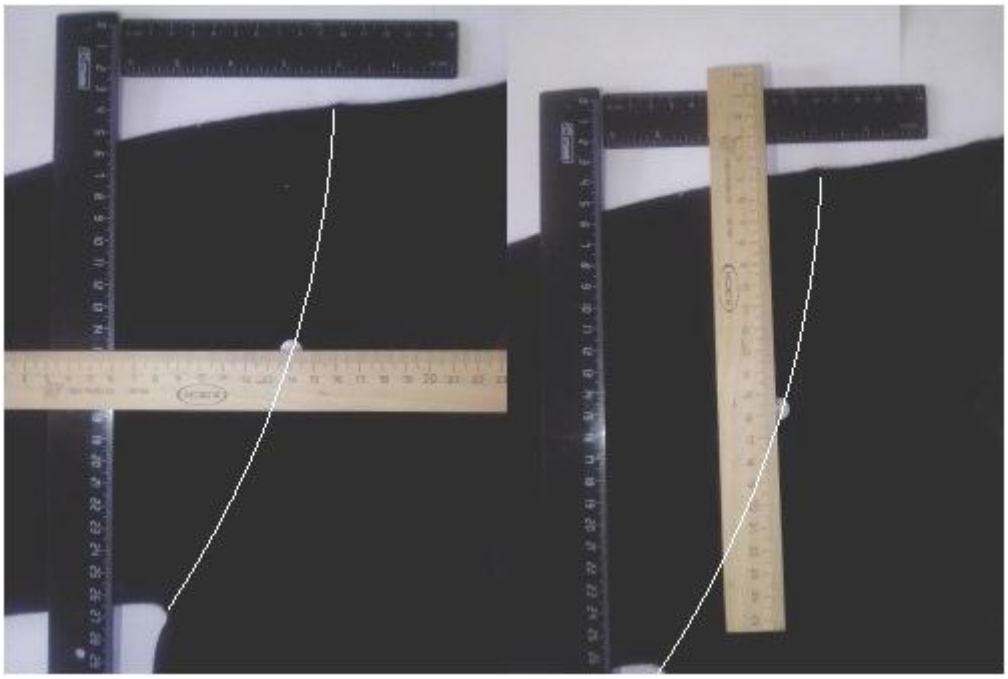


Рисунок П2.18 – Дополнительная точка

ШАГ 4. Определим длину участка кривой от начальной опорной вершины до дополнительной точки (рисунок П2.19).

Рисунок П2.19 – Длина участка

ШАГ 5. Определим общую длину кривой проймы (рисунок П2.20).

Рисунок П2.20 – Длина кривой проймы

ШАГ 6. Рассчитаем параметр кривой, воспользовавшись формулой (П2.1) и составим команду построения:

$$ta = 14/26 = 0.54$$

Полученные данные представим в виде сводной таблицы (таблица П2.2)

Таблица П2.2 – Сводная таблица параметров функции команды построения кривой

Параметр функции	Значение, см.
X0	2.0
Y0	26.5
X2	9.5
Y2	3.5
Xa	7.5
Ya	15.0
Ta	0.54

Составим команду построения:

proyma:bzr2i>2,26.5,9.5,3.5,7.5,15,0.54;

Построенная кривая Безье представлена на рисунке П2.21.

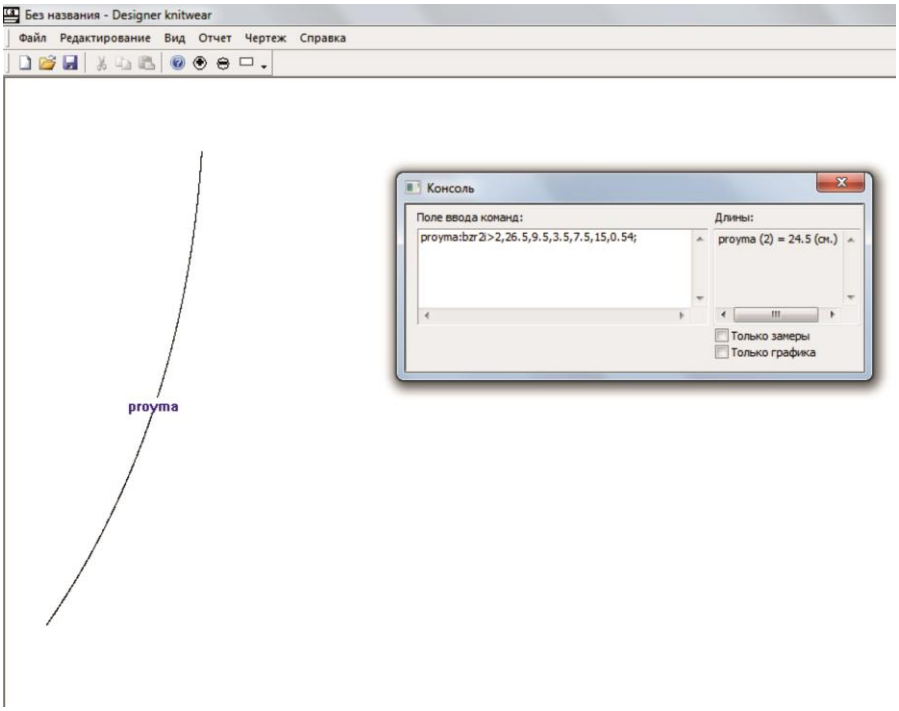


Рисунок П2.21 – Построенная пройма мужского джемпера 54 размера с втачным типом рукава

Визуально сравним результаты (рисунок П2.22).

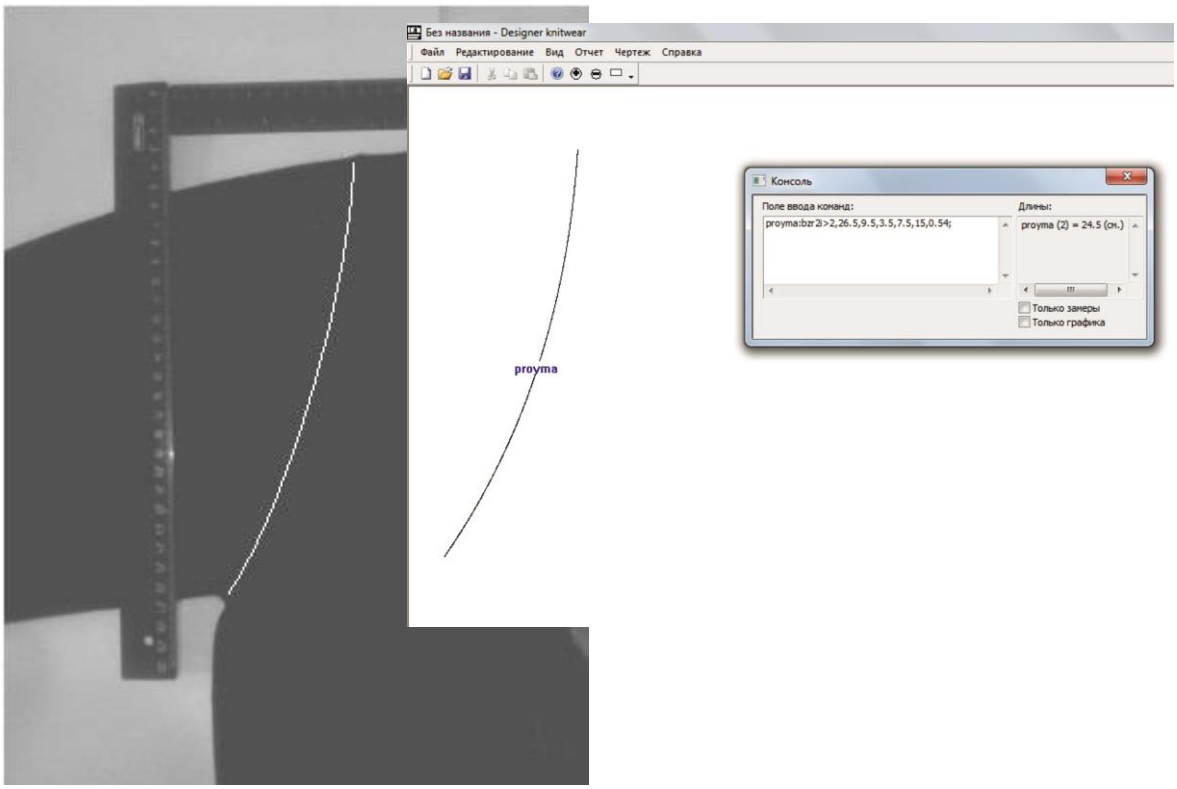


Рисунок П2.22 – Сравнение результатов

Таким образом, приведенный выше пример показывает, что в “DESIGNER K-WEAR” также возможно построение и за счет импорта оцифрованных кривых с лекал, готовых изделий и т.д. без использования каких-либо специальных дополнительных технических устройств (например, дигитайзера) или дорогостоящих приспособлений.

4 СВЕДЕНИЯ О ВНУТРЕННИХ ДАННЫХ, ПРИНЦИПАХ СОЗДАНИЯ ИХ СВЯЗЕЙ ВНУТРИ АЛГОРИТМА И ОБ ОПТИМАЛЬНОЙ СХЕМЕ ПОСТРОЕНИЯ АЛГОРИТМА В “DESIGNER K-WEAR”

В “DESIGNER K-WEAR” при проектировании трикотажного изделия возможно использование нескольких типов данных (данные, содержащие в себе информацию по части конструирования и данные по аппроксимации). Данные конструирования и аппроксимации могут иметь несколько форм:

1. Явно заданная форма, которая характеризуется постоянным значением какого-либо числа, иначе постоянная.
2. Неявно заданная форма, характеризующаяся значением какого-либо выражения с использованием явно заданных данных, неявно заданных данных, а также их комбинированного значения.
3. Исключения, характерные для данных аппроксимации, использующие в своей структуре дополнительные символы, меняющие их смысловую нагрузку.

Общая команда создания объекта данных имеет вид:

ИМЯ := ПАРАМЕТРЫ;

где *ИМЯ* – идентификатор объекта данных, необходимый для последующего обращения к нему внутри алгоритма (таким идентификатором, как и в случае с принятой унифицированной командой построения графических объектов, может являться любое сочетание символов, за исключением зарезервированных);

:= – оператор присвоения;

ПАРАМЕТРЫ – данные, которые необходимо присвоить к идентифицируемому объекту.

Создание связей между объектами данных внутри алгоритма возможно за счет использования их идентификаторов. Однако, определение объектов данных должно быть выполнено до того как они будут вызваны за счет своих идентификаторов в алгоритме. Это связано, прежде всего, с тем, что алгоритм

построения начинает читаться с первой верхней строки поля ввода в окне консоли, поэтому раннее использование уже определенных объектов данных при построении не допускается.

Приведем пример создания объектов данных в явной и неявной формах:

const:=100;
*var1:=const+20*2;*
var2:=var1+const;

Из примера:

const:=100; - объект данных с явной формой определения внутри всего алгоритма будет иметь постоянный тип, где const – идентификатор объекта данных;

*var1:=const+20*2;* - объект данных с неявной формой определения, использующий в своей структуре выражение, состоящее из объекта с явно заданной формой и чисел;

var2:=var1+const; - объект данных с неявной формой определения, использующий в своей структуре выражение, состоящее из объекта с явно заданной формой и объекта с неявно заданной формой определения.

Определение связей между объектами создано согласно внутреннему правилу.

Таким образом, при проектировании изделии в “DESIGNER K-WEAR” возможно не только выполнять построение конструкций по уже рассчитанной методике, но и создавать свои собственные методики построения с использованием вводных коэффициентов, собственных формул расчета и т.д.

Кроме того при создании графического объекта автоматически создаются объекты данных, содержащие в себе информацию о положении его опорных вершин и длине. Данные объекты обладают всеми теми же свойствами, описанными выше.

Способы к их доступу приведены в таблице П2.3.

Таблица П2.3 – Доступ к данным графических объектов

Объект	Значение	Идентификатор
X0, X1, X2, X3	Абсцисса опорной вершины графического объекта	ИМЯ.x0, ИМЯ.x1, ИМЯ.x2, ИМЯ.x3
Y0, Y1, Y2, Y3	Ордината опорной вершины графического объекта	ИМЯ.y0, ИМЯ.y1, ИМЯ.y2, ИМЯ.y3
Длина	Длина графического объекта	ИМЯ.l

Где ИМЯ – идентификатор графического объекта.

Список реализованных математических операторов:

+ – сложение;

- – вычитание;

* – умножение;

/ – деление;

( ) – обозначение приоритета действия.

Приведем пример построения графических объектов с зависимыми опорными вершинами:

```
otkl:=2.5;
otkl_curve:=3;
linia1:line>2,2,5,5+otkl;
curve1:bzr2>linia1.x0,linia1.y0,
linia1.x0+otkl_curve,curve.y0+otkl_curve,linia1.x1+2*otkl,linia1.y1;
```

*curve2:bzr2>linia1.x1,linia1.y1,(curve1.x2-linia1.x1)*2,(curve1.y2-linia1.y1)*2,curve1.x2,curve1.y2;*

Построение представлено на рисунке П2.23.

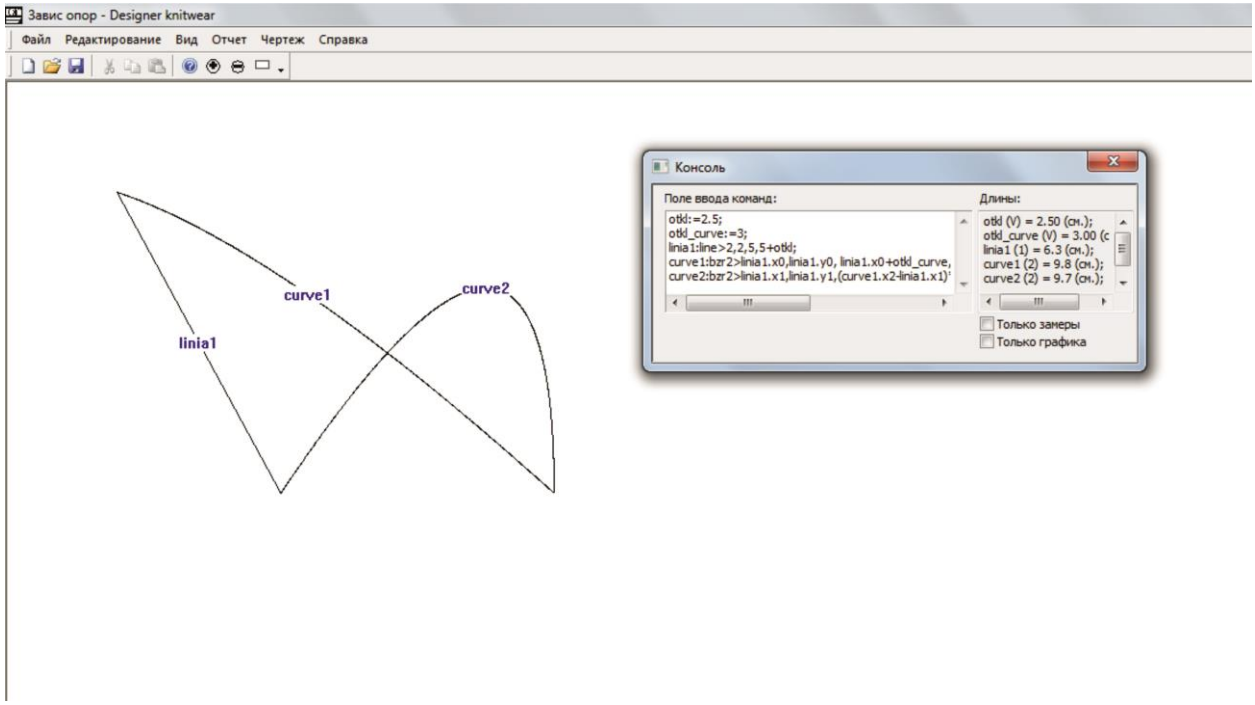


Рисунок П2.23 – Построение с использованием зависимости данных

Согласно рассмотренному примеру, отрезок с идентификатором linia1 построен с учетом смещения конечной опорной вершины на величину согласно объекту данных с идентификатором otkl. Кривые curve1 и curve2 построены с учетом зависимости их начальных и конечных опорных вершин от других объектов.

Таким образом, при составлении алгоритма построения в “DESIGNER K-WEAR”, прежде всего, необходимо учитывать последовательность определения объектов данных, построение графических объектов и создание их связей внутри алгоритма согласно, рассмотренным выше правилам.

Для повышения читабельности алгоритма, а также снижения возникновения ошибки нарушения внутренних связей между объектами необходимо воспользоваться следующими рекомендациями:

1. Создание алгоритма необходимо выполнять по схеме ***«Определение данных > Построение объектов > Формирование контура лекал > Аппроксимация»***. При такой схеме построения сокращается время поиска нужной команды в алгоритме, а также увеличивается его читабельность.
2. Присваивать наиболее понятные и простые идентификаторы объектам, полагаясь на смысловую доступность внутри производственного коллектива специалистов.
3. Давать исчерпывающие комментарии внутри функций построения, аппроксимации и т.д.:

   ИМЯ:ФУНЦКИЯ (комментарий) > ПАРАМЕТРЫ;

   Например, line1:line(боковая линия полочки)>40,40,40,60;

5 ОСНОВЫ РАБОТЫ С РАЗМЕРНЫМИ ПРИЗНАКАМИ, СПОСОБЫ СОЗДАНИЯ СОБСТВЕННОЙ БАЗЫ ДАННЫХ РАЗМЕРНЫХ ПРИЗНАКОВ

Размерные признаки в “DESIGNER K-WEAR” представляют собой массив заранее определенных объектов данных с зарезервированными идентификаторами. Доступ к этим объектам осуществляется, как и к любому другому объекту данных, а именно через его идентификатор. Однако прежде чем приступить к построению, данные о размерных признаках необходимо загрузить через диалоговое окно: *Чертеж* > *Размерные признаки* > *Загрузить...* В открывшемся диалоговом окне выбрать файл, содержащий необходимые размерные признаки (рисунок П2.24).

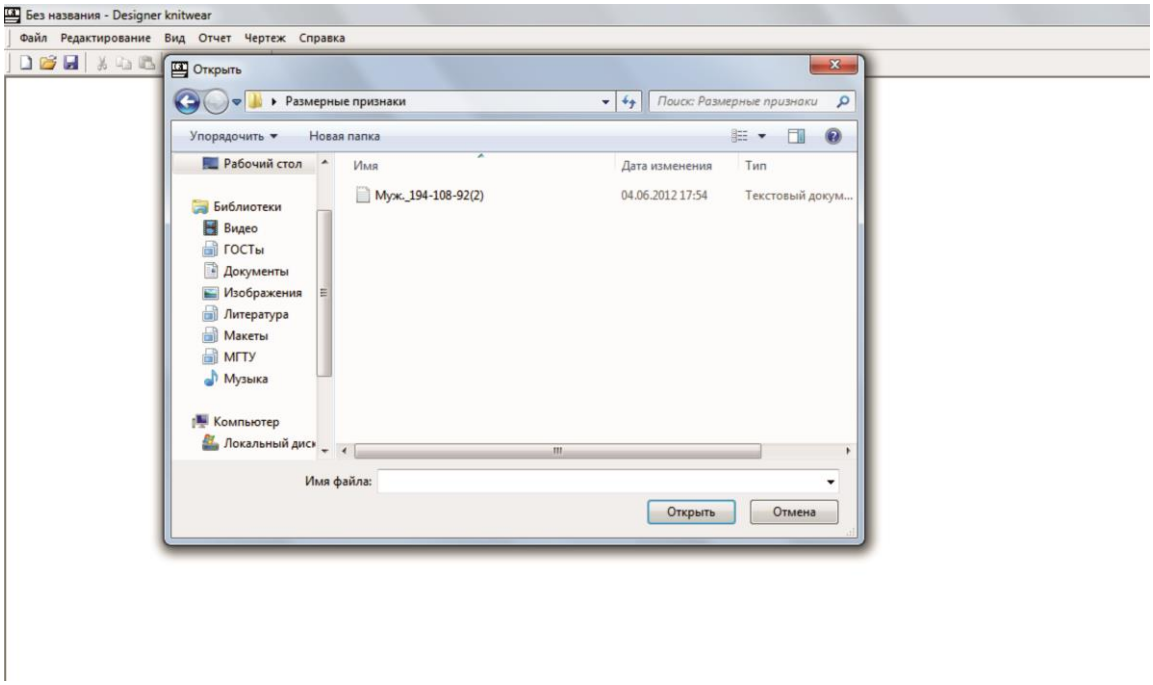


Рисунок 24 – Окно загрузки размерных признаков

Файл с размерными признаками является файлом Текстового документа (*.txt), поэтому при создании собственной базы данных необходимо воспользоваться специальной программой по созданию текстовых документов (например, программой «Блокнот», поставляемой в любой современной сборке ОС Windows от Microsoft).

Шаблон по созданию с описаниями приведен далее:

P= ; // Рост

Vtos= ; // Высота точки основания шеи спереди

Vtosh= ; // Высота точки основания шеи сбоку

Vpt= ; //Высота плечевой точки

Vlt= ; // Высота линии талии

Vst= ; // Высота сосковой точки

Vlt= ; // Высота линии талии

Vk= ; // Высота коленной точки

Vsht= ; // Высота точки основания шеи

Vzu= ; // Высота заднего угла подмышечной впадины

Vlop= ; // Высота лопаточной точки

Vps= ; // Высота подъягодичной складки

Osh= ; // Обхват шеи

Og1= ; // Обхват груди 1-й

Og2= ; // Обхват груди 2-й

Og3= ; // Обхват груди 3-й

Og4= ; // Обхват груди 4-й

Ot= ; // Обхват талии

Ob= ; // Обхват бедер с учетом выступа живота

Ob1= ; // Обхват бедер без учета выступа живота

Obed= ; // Обхват бедра

Ok= ; // Обхват колена

Oi= ; // Обхват икры

Osch= ; // Обхват щиколотки

Os= ; // Обхват подъема стопы

Dsb= ; // Расстояние от линии талии до пола сбоку

Dsp= ; // Расстояние от линии талии до пола спереди

Dn= ; // Длина ноги по внутренней поверхности

Dps= ; // Расстояние от линии талии до подъягодичной складки

Dpob= ; // Дуга через паховую область

Ds= ; // Расстояние от линии талии до плоскости сидения

Op= ; // Обхват плеча

Oz= ; // Обхват запястья

Okis= ; // Обхват кисти

SHp= ; // Длина плечевого ската

Dluch= ; // Расстояние от точки основания шеи сбоку до лучевой точки

Dzap= ; // Расстояние от точки основания шеи сбоку до линии обхвата запястья

D3p= ; // Расстояние от точки основания шеи сбоку до конца третьего пальца

Vpr= ; // Расстояние от точки основания шеи сбоку до линии обхвата груди 1 сбоку

Vg= ; // Расстояние от точки основания шеи сбоку до сосковой точки (высота груди)

Dtp= ; // Расстояние от точки основания шеи сбоку до линии талии спереди (длина талии спереди)

Dp= ; // Дуга через высшую точку плечевого сустава

Vprz= ; // Расстояние от точки основания шеи сзади до линии обхвата груди 1 и 2 с учетом выступа лопаток

Dts= ; // Длина спины до талии с учетом выступа лопаток

Dts1= ; // Расстояние от линии талии сзади до точки основания шеи сбоку

Dvcht= ; // Длина дуги верхней части туловища через точку основания шеи сбоку

SHg= ; // Ширина груди

Cg= ; // Расстояние между сосковыми точками (центр груди)

SHs= ; // Ширина спины

dpzr= ; // Передне-задний диаметр руки

Ogol= ; // Обхват головы

Приведем пример готового файла с мужскими размерными признаками 194-108-92 второй полнотной группы ГОСТ 52774-2007:

P= 194; // Рост

Vtos= 160.9; // Высота точки основания шеи спереди

Vtosh= 167.4; // Высота точки основания шеи сбоку

Vpt= 160.2; //Высота плечевой точки

Vlt= 123.1; // Высота линии талии

Vst= 0; // Высота сосковой точки

Vk= 56.9; // Высота коленной точки

Vsht= 169.1; // Высота точки основания шеи

Vzu= 148.1; // Высота заднего угла подмышечной впадины

Vlop= 147.0; // Высота лопаточной точки

Vps= 91.6; // Высота подъягодичной складки

Og1= 111.2; // Обхват груди 1-й

Og2= 112.2; // Обхват груди 2-й

Og3= 108; // Обхват груди 3-й

Ot= 92; // Обхват талии

Ob= 110; // Обхват бедер с учетом выступа живота

Ob1= 107.2; // Обхват бедер без учета выступа живота

Ok= 42.7; // Обхват колена

Dps= 34.9; // Расстояние от линии талии до подъягодичной складки

Op= 34.6; // Обхват плеча

Oz= 18.4; // Обхват запястья

Okis= 25.4; // Обхват кисти

SHp= 18; // Длина плечевого ската

Dzap= 84.2; // Расстояние от точки основания шеи сбоку до линии обхвата запястья

Dtp= 50.5; // Расстояние от точки основания шеи сбоку до линии талии спереди (длина талии спереди)

Vprz= 22; // Расстояние от точки основания шеи сзади до линии обхвата груди 1 и 2 с учетом выступа лопаток

Dts= 47.6; // Длина спины до талии с учетом выступа лопаток

Dts1= 50.6; // Расстояние от линии талии сзади до точки основания шеи сбоку

SHg= 42; // Ширина груди

Cg= 24.2; // Расстояние между сосковыми точками (центр груди)

SHs= 43.8; // Ширина спины

6 ФОРМИРОВАНИЕ КОНТУРА ЛЕКАЛ ИЛИ ПРИНЦИПЫ СОЗДАНИЯ СЛОЖНЫХ ГРАФИЧЕСКИХ ОБЪЕКТОВ

В “DESIGNER K-WEAR” этап формирования контура лекал (сложных объектов или д-объектов) является одним из заключительных этапов в проектировании трикотажного изделия.

Контур сложного объекта представляет собой совокупность графических объектов, записанных в определенной последовательности, где запись объектов всегда должна выполняться в последовательности «снизу-вверх» (ассоциативно для вязания изделия, детали и т.п.).

В “DESIGNER K-WEAR” имеется три типа сложных объектов:

- *форма* (объект, имеющий две замкнутые друг с другом кромки);
- *кромка* (объект, чаще всего использующийся в проектировании симметричной детали);
- *соединение* (является узлом соединения деталей в цельновязаном изделии);
- *внутренний объект* (сложный объект, содержащийся внутри другого сложного объекта, например, накладной карман, выпуклость и т.д.);

КРОМКА

Кромка является наиболее простым формированием графических объектов. Формирование кромки, согласно правилу, осуществляется снизу вверх с помощью команды:

ИМЯ : SIDE > список объектов | сторона | коэф. усадки по гориз., коэф. усадки по вертик.;

где *ИМЯ* – идентификатор сложного объекта, необходимый для его дальнейшего использования;

SIDE – функция построения кромки;

список объектов – список графических объектов, необходимых для формирования кромки;

| – разделитель;

сторона – указатель расположения кромки относительно полотна (если кромка находится слева, то LEFT, если справа – RIGHT);

коэф. усадки по гориз., коэф. усадки по вертик. – коэффициенты усадки по горизонтали и вертикали соответственно (могут быть получены расчетным путем).

Рассмотрим пример построения кромки:

bok:line>5,5,6,10;

niz:line>bok.x1,bok.y1,11,10;

proyma:bzr2>bok.x0,bok.y0,7,3,6,1;

plecho:line>proyma.x2,proyma.y2, niz.x1-2,proyma.y2-0.5;

kromka:side>niz,bok,proyma,plecho|left|1.1,1.1;

Построенный данный пример представлен на рисунке П2.25.

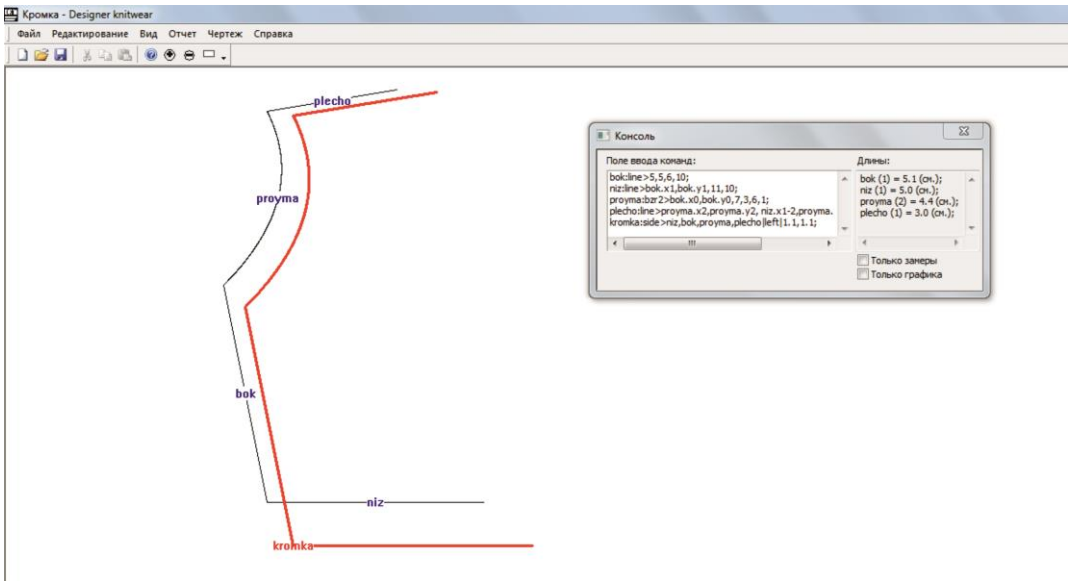


Рисунок П2.25 – Сформированная кромка

ФОРМА

Форма является сочетанием двух кромок и формируется с помощью следующей команды:

ИМЯ : FORM > список объектов слева | список объектов справа | коэф. усадки по гориз., коэф. усадки по вертик.;

где *ИМЯ* – идентификатор сложного объекта, необходимый для его дальнейшего использования;

FORM – функция построения формы;

список объектов слева, список объектов справа – список графических объектов, необходимых для формирования левой и правой кромок;

| – разделитель;

коэф. усадки по гориз., коэф. усадки по вертик. – коэффициенты усадки по горизонтали и вертикали соответственно (могут быть получены расчетным путем).

Рассмотрим пример построения формы:

```
bok:line>5,5,6,10;
niz:line>bok.x1,bok.y1,11,10;
proyma:bzr2>bok.x0,bok.y0,7,3,6,1;
plecho:line>proyma.x2,proyma.y2, niz.x1-2,proyma.y2-0.5;
bok2:line>niz.x1,niz.y1,niz.x1+0.5,niz.y1-6;
    gorlovina:bzr3>bok2.x1,bok2.y1,bok2.x1-0.5,bok2.y1-
3,plecho.x1+2,plech0.y1+3, plecho.x1,plecho.y1;
polochka_left:form>niz,bok,proyma,plecho|bok2,gorlovina|1,1;
```

Построенный данный пример представлен на рисунке П2.26.

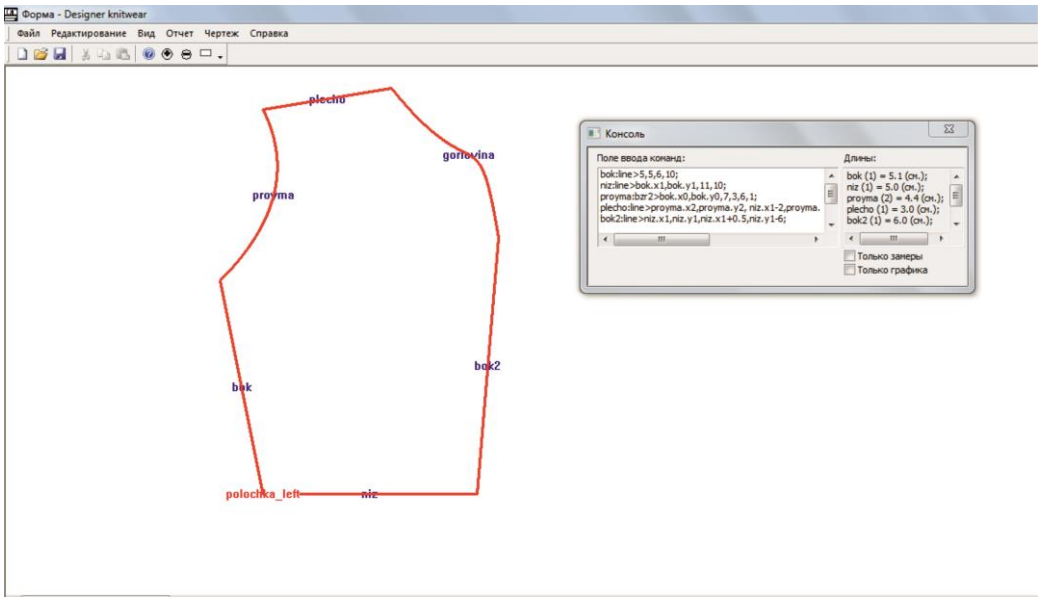


Рисунок П2.26 – Сформированная форма

СОЕДИНЕНИЕ

Соединение является объектом узла соединения деталей в цельновязаном изделии и формируется с помощью следующей команды:

ИМЯ : COMB > список объектов слева | список объектов справа | | коэф. усадки по гориз., коэф. усадки по вертик.;

где *ИМЯ* – идентификатор сложного объекта, необходимый для его дальнейшего использования;

COMB – функция формирования соединения;

список объектов слева, список объектов справа – список графических объектов, необходимых для формирования левой и правой кромок соединения;

| – разделитель;

коэф. усадки по гориз., коэф. усадки по вертик. – коэффициенты усадки по горизонтали и вертикали соответственно (могут быть получены расчетным путем).

Рассмотрим пример построения узла соединения:

proyma:bzr2>10,10,6,9,7,1.9;

okat:bzr3>10,10,14,9,13,3.5,16,3.2;

proyma_soed:comb>proyma|okat;

Построенный данный пример представлен на рисунке П2.27.

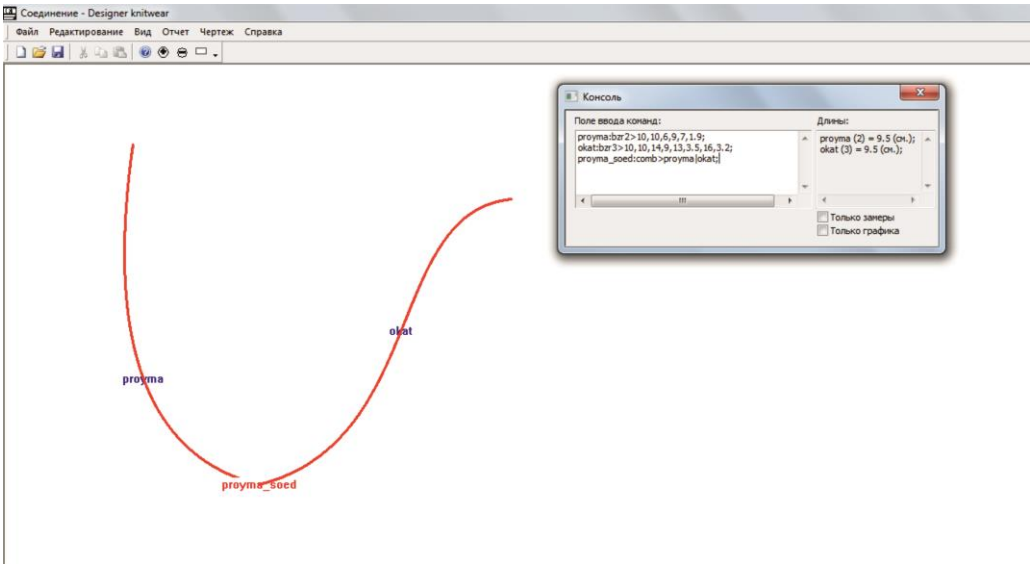


Рисунок П2.27 – Сформированное соединение

ВНУТРЕННИЙ ОБЪЕКТ

Внутренним объектом может являться любой сложный объект (форма, кромка, соединение).

Команда формирования внутреннего объекта для формы и кромки отличается от команд формирования данных объектов лишь идентификатором, где указывается дополнительный идентификатор сложного объекта, с которым необходимо будет выполнить сопряжение:

ИМЯ1$ИМЯ2:остальная команда…

ИМЯ1 – идентификатор внутреннего объекта;

ИМЯ2 – идентификатор главного объекта;

$ - символ сопряжения.

Рассмотрим пример построения внутреннего объекта на кромке:

bok:line>5,5,6,10;

niz:line>bok.x1,bok.y1,11,10;

proyma:bzr2>bok.x0,bok.y0,7,3,6,1;

plecho:line>proyma.x2,proyma.y2, niz.x1-2,proyma.y2-0.5;

linia1:line>niz.x1,niz.y1-1, niz.x1-3,niz.y1-1;

linia2:line>linia1.x1,linia1.y1, linia1.x1,linia1.y1-3;

linia3:line>linia2.x1,linia2.y1,linia1.x0,linia2.y1;

linia4:line>linia3.x1,linia3.y1,linia1.x0,linia1.y0;

kromka:side>niz,bok,proyma,plecho|left|1,1;

karman$kromka:form>linia1,linia2|linia4,linia3|1,1;

Построенный данный пример представлен на рисунке П2.28.

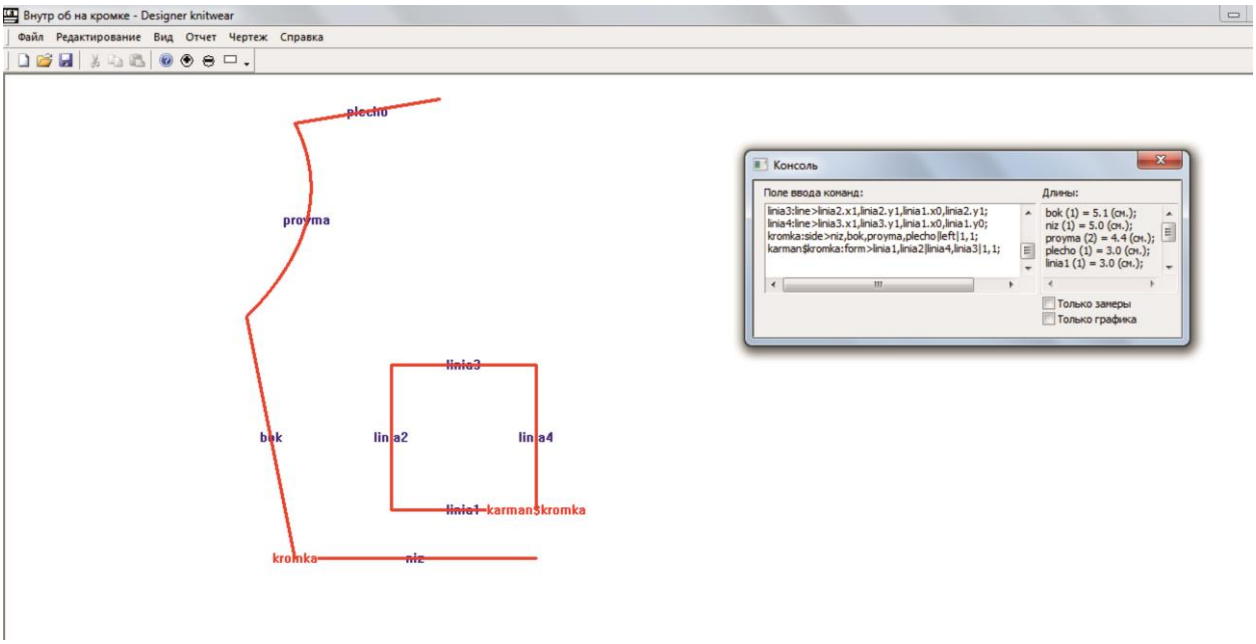


Рисунок П2.28 – Формирование кромки с сопряженным внутренним объектом

В идентификаторе, при формировании узла соединения, за счет сопряжения сложных объектов указываются идентификаторы сопрягаемых друг с другом сложных объектов.

ИМЯ1-ИМЯ2:COMB>остальная часть команды…

где *ИМЯ1* – идентификатор сложного объекта, находящегося слева от соединения;

ИМЯ2 – идентификатор сложного объекта, находящегося справа от соединения;

- – символ сопряжения при соединении.

При указании кромок соединения важно, чтобы графические объекты соединения также входили в состав соединяемых сложных объектов.

Рассмотрим пример соединения двух кромок.

Рассмотрим пример построения узла соединения за счет соединения сложных объектов:

bok:line>5,5,6,10;
niz:line>bok.x1,bok.y1,11,10;
proyma:bzr2>bok.x0,bok.y0,7,3,6,1;
plecho:line>proyma.x2,proyma.y2,niz.x1-2,proyma.y2-0.5;
bok_ruk:line>bok.x0-2,bok.y0,niz.x0-4,niz.y0-0.5;
okat:bzr3>bok_ruk.x0,bok_ruk.y0,bok_ruk.x0-2,bok_ruk.y0-2,2,2.10,0.35,1.64;
niz_ruk:line>bok_ruk.x1,bok_ruk.y1,okat.x3,bok_ruk.y1;
kromka:side>niz,bok,proyma,plecho|left|1,1;
rukav:side>niz_ruk,bok_ruk,okat|right|1,1;
rukav-kromka:comb>okat|proyma|1,1;

Построенный данный пример представлен на рисунке П2.29.

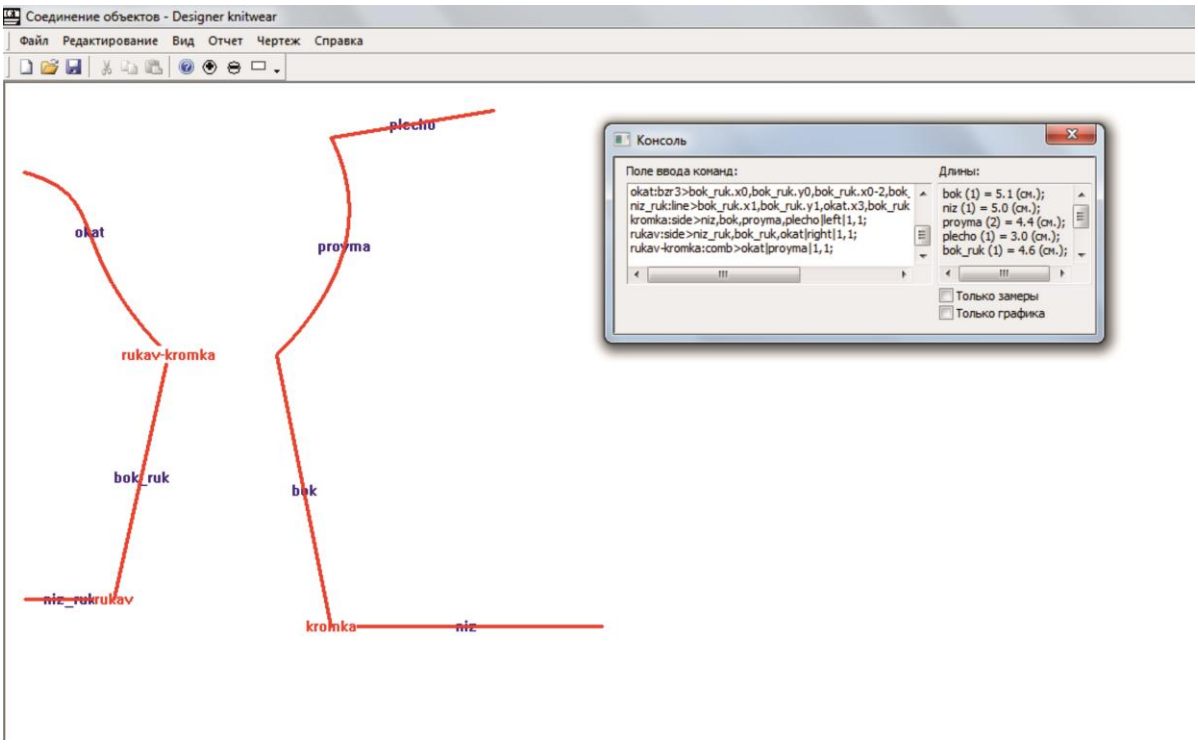


Рисунок 29 – Формирование сопряжения сложных объектов через соединение

7 ЗАКЛЮЧИТЕЛЬНЫЙ ЭТАП ПРОЕКТИРОВАНИЯ, АППРОКСИМАЦИЯ, ОСНОВЫ РАБОТЫ С ОТЧЕТАМИ

На заключительном этапе проектирования основную роль играет аппроксимация, сформированных сложных объектов. Аппроксимация – научный метод, состоящий в замене одних объектов другими, в том или ином смысле близкими к исходным, но более простыми. В “DESIGNER k_WEAR” аппроксимация графических объектов выполняется ячейками (см. глава 2).

Команда аппроксимации имеет вид:

ИМЯ:APRX>A,B,MAXSB,RAPAPR;

где *ИМЯ* – идентификатор графического объекта, который необходимо аппроксимировать (если необходимо выполнить одинаковую аппроксимацию для всех объектов, то ИМЯ = “ALL”);

APRX – функция аппроксимации;

A, B – ширина и высота петельного столбика и ряда соответственно;

MAXSB – максимально допустимая сбавка (прибавка) в петельном ряду (по умолчанию равна 1000);

если !MAXSB, то выполнение сбавки (прибавки) в ряду равной указанной или равной нулю (например, !3 – выполнение в ряду сбавки (прибавки) равной 3 или 0), если !!MAXSB – выполнение сбавки (прибавки) исключительно равной указанной (например, !!2 – выполнение сбавки (прибавки) исключительно равной 2);

RAPAPR – раппорт аппроксимации (количество пропускаемых рядов перед следующей операцией аппроксимации).

Команда аппроксимации на заданном участке имеет вид:

ИМЯ:APRXY>Y1,Y2,A,B,MAXSB,RAPAPR;

где *APRXY* – функция аппроксимации на определенном участке;

Y1 – ордината начала аппроксимации;

Y2 – ордината конца аппроксимации.

Рассмотрим пример аппроксимации:

bok:line>5,5,6,10;

niz:line>11,10,bok.x1,bok.y1;

proyma:bzr2>bok.x0,bok.y0,7,3,6,1;

plecho:line>proyma.x2,proyma.y2, niz.x0-2,proyma.y2-0.5;

kromka:side>niz,bok,proyma,plecho|left;

all:aprx>0.1,0.11;

proyma:aprx>0.1,0.11,2,3;

proyma:aprxy>proyma.y2+0.7,proyma.y2,0.1,0.11,!!1,1;

Построенный данный пример представлен на рисунке П2.30.

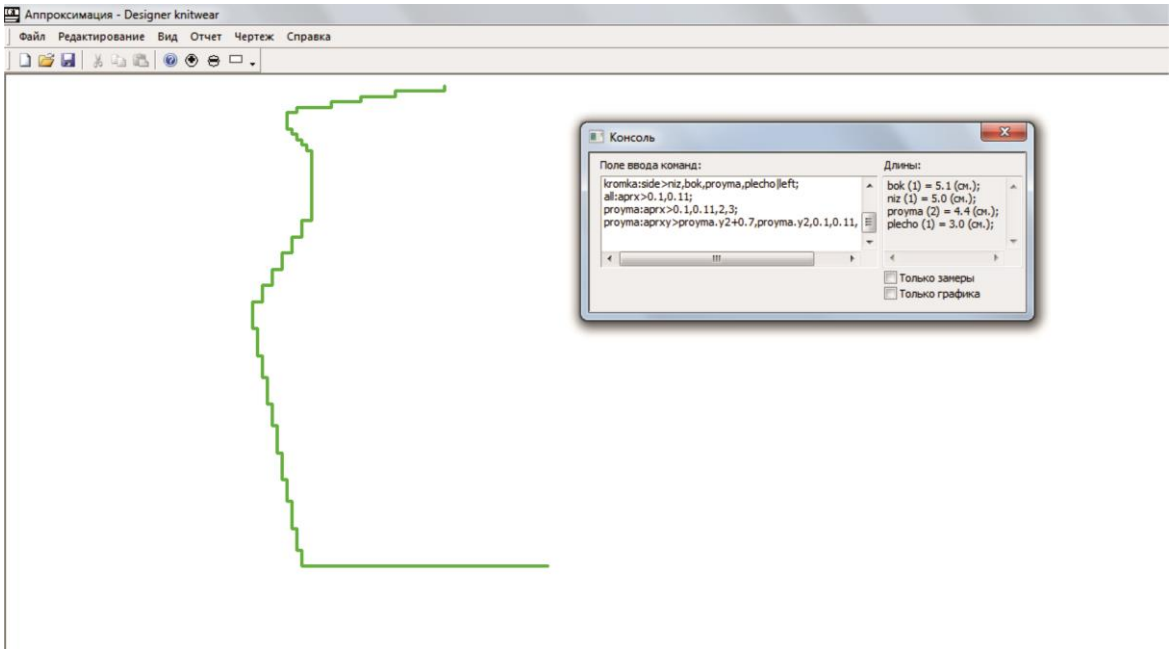


Рисунок П2.30 – Аппроксимация

При соединении аппроксимация выполняется с компенсацией рядов по кромкам соединяемых сложных объектов.

Результатом аппроксимации в “DESIGNER K-WEAR” является отчет, выводимый в таблицу MS Office Excel, содержащий в себе сведения об аппроксимации.

Полный тип отчета содержит в себе наиболее подробные сведения об аппроксимации, краткий тип отчета является сводной таблицей аппроксимации, сведенной к ступеням (также данные краткого типа отчета можно экспортировать в САПР М1 фирмы STOLL).

Выведем отчеты для рассмотренного примера (таблицы П2.4 и П2.5)

Таблица П2.4 – Полный тип отчета аппроксимации

Имя:	kromka					
ax	ay	t	xl	сбавка	Ряд	Направление
11	10	0	11	0	1	0
6	10	1	6	50	1	-1
6	9,89	0,02181	5,97819	0	2	1
6	9,78	0,04381	5,95619	0	3	1
5,9	9,67	0,06581	5,93419	1	4	-1
5,9	9,56	0,08781	5,91219	0	5	1
5,9	9,45	0,10982	5,89018	0	6	1
5,9	9,34	0,13182	5,86818	0	7	1
5,8	9,23	0,15382	5,84618	1	8	-1
5,8	9,12	0,17582	5,82418	0	9	1
5,8	9,01	0,19782	5,80218	0	10	1
5,8	8,9	0,21983	5,78017	0	11	1
5,8	8,79	0,24183	5,75817	0	12	1
5,7	8,68	0,26383	5,73617	1	13	-1
5,7	8,57	0,28583	5,71417	0	14	1
5,7	8,46	0,30784	5,69216	0	15	1
5,7	8,35	0,32984	5,67016	0	16	1
5,6	8,24	0,35184	5,64816	1	17	-1
5,6	8,13	0,37384	5,62616	0	18	1
5,6	8,02	0,39584	5,60416	0	19	1
5,6	7,91	0,41785	5,58215	0	20	1
5,6	7,8	0,43985	5,56015	0	21	1
5,5	7,69	0,46185	5,53815	1	22	-1
5,5	7,58	0,48385	5,51615	0	23	1
5,5	7,47	0,50586	5,49414	0	24	1
5,5	7,36	0,52786	5,47214	0	25	1
5,5	7,25	0,54986	5,45014	0	26	1
5,4	7,14	0,57186	5,42814	1	27	-1
5,4	7,03	0,59386	5,40614	0	28	1
5,4	6,92	0,61587	5,38413	0	29	1
5,4	6,81	0,63787	5,36213	0	30	1
5,3	6,7	0,65987	5,34013	1	31	-1

Продолжение таблицы П2.4

ax	ay	t	xl	сбавка	Ряд	Направление
5,3	6,59	0,68187	5,31813	0	32	1
5,3	6,48	0,70388	5,29612	0	33	1
5,3	6,37	0,72588	5,27412	0	34	1
5,3	6,26	0,74788	5,25212	0	35	1
5,2	6,15	0,76988	5,23012	1	36	-1
5,2	6,04	0,79188	5,20812	0	37	1
5,2	5,93	0,81389	5,18611	0	38	1
5,2	5,82	0,83589	5,16411	0	39	1
5,1	5,71	0,85789	5,14211	1	40	-1
5,1	5,6	0,87989	5,12011	0	41	1
5,1	5,49	0,9019	5,0981	0	42	1
5,1	5,38	0,9239	5,0761	0	43	1
5,1	5,27	0,9459	5,0541	0	44	1
5	5,16	0,9679	5,0321	1	45	-1
5	5,05	1	5	0	46	1
5	4,94	1	5	0	47	1
5,2	4,61	0,08239	5,309196	2	50	1
5,4	4,28	0,1649	5,578024	2	53	1
5,6	3,95	0,24741	5,806005	2	56	1
5,8	3,62	0,32991	5,993118	2	59	1
6	3,29	0,41242	6,139409	2	62	1
6,2	2,96	0,49493	6,244853	2	65	1
6,2	2,63	0,57744	6,309449	0	68	1
6,2	2,3	0,65995	6,333198	0	71	-1
6,2	1,97	0,74246	6,316099	0	74	1
6,2	1,64	0,82496	6,258163	0	77	-1
6,1	1,53	0,85247	6,229765	1	78	-1
6	1,42	0,87997	6,196838	1	79	-1
5,9	1,31	0,90747	6,159375	1	80	-1
5,8	1,2	0,93497	6,117373	1	81	-1
5,7	1,09	0,96248	6,070817	1	82	-1
5,9	0,76	1	6	2	85	1
6,6	0,65	0,21987	6,55961	7	86	1
7,2	0,54	0,4399	7,2197	6	87	1
7,9	0,43	0,65992	7,87976	7	88	1
8,9	0,32	1	8,9	10	89	1
8,9	0,21	1	8,9	0	90	1

Где из таблицы П2.4 *ax* – абсцисса, полученная в результате аппроксимации;

ay – ордината, полученная в результате аппроксимации;

t – текущий параметр графического объекта;

xl – абсцисса точки, лежащей на кривой с ординатой ay;

сбавка – сбавка (прибавка) в ряду;

ряд – значение ряда, полученного в результате аппроксимации при ординате ay;

направление – направление сбавки (прибавки); -1 – влево, 1 – вправо.

Таблица П2.5 – Краткий тип отчета аппроксимации

Имя:	kromka
Кромка	
Высота (ряды)	Ширина (петли)
0	-50
3	-1
4	-1
5	-1
4	-1
10	-2
4	-1
5	-1
4	-1
5	-1
2	0
18	12
12	0
5	-5
3	2
1	7
1	6
1	7
1	10
1	0

8 ПРОВЯЗЫВАНИЕ ДОПОЛНИТЕЛЬНЫХ РЯДОВ. ПОЛУЧЕНИЕ ВЫПУКЛЫХ УЧАСТКОВ ТРИКОТАЖА

Для получения выпуклого участка трикотажа существует несколько способов: создание вытачек, переход на частичное вязание несколькими нитеводителями и последующий их перенос, провязывание дополнительных петельных рядов. Однако закрытие вытачек и переход на частичное вязание может существенно повлиять на локальную структуру полотна, а именно допустить появление дополнительного шва (в случае с вытачкой швейного типа) или столбика со сдвоенными петлями, получаемого при частичном вязании. В результате чего это приведет к нарушению художественно-колористического оформления готового изделия.

В “DESIGNER K-WEAR” реализована возможность замены вытачек и получения необходимых выпуклостей за счет провязывания в полотне дополнительных рядов двумя или одним нитеводами.

Команды построения имеют вид:

ИМЯ:DOPR1>NR, RASPR, SIDE, B;

ИМЯ:DOPR2>NR, RASPR, SIDE, B;

где *ИМЯ* – идентификатор отрезка, на всей протяженности которого следует провязывать дополнительные петельные ряды;

DOPR1, *DOPR2* – провязывание дополнительных рядов при использовании одного или двух нитеводов соответственно;

NR – количество дополнительных рядов;

RASPR – указание центра распределения в значении параметра t графического объекта (распределение отсчитывается от начальной опорной вершины объекта);

SIDE – сторона распределения (-1 – слева, 1- справа);

B – высота петельного ряда.

Рассмотрим пример формирования выпуклого участка трикотажа за счет провязывания дополнительных петельных рядов:

linia:line>5,5,5,10;

s:side>linia|right;

linia:dopr2>6,0.3,1,0.5;

all:aprx>0.5,0.5;

Построенный данный пример представлен на рисунке П2.31.

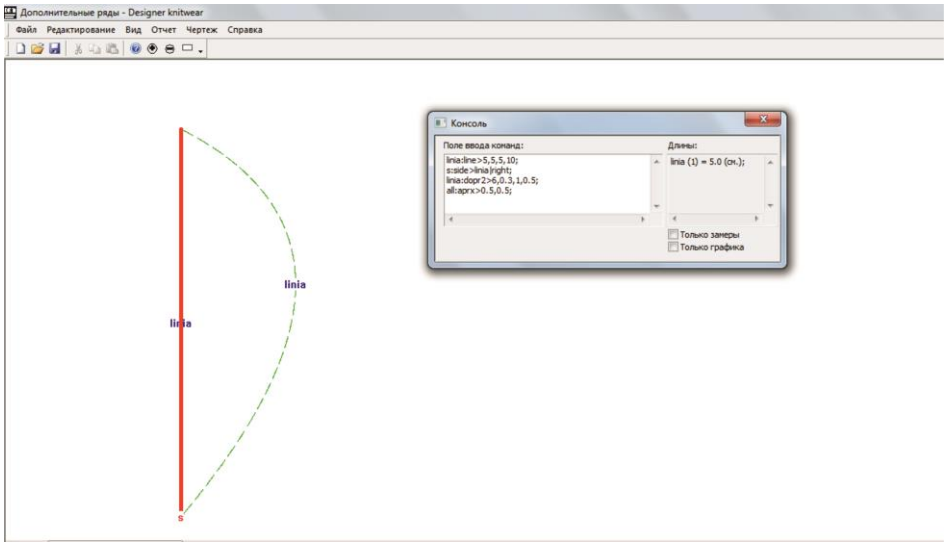


Рисунок П2.31 – Построение выпуклости за счет провязывания дополнительных рядов

На рисунке П2.31 штриховой линией указана максимально приближенная выпуклость, получаемая за счет провязывания дополнительных петельных рядов.

Выведем отчет полного типа по сформированной выпуклости (таблица П2.6).

Таблица П2.6 – Отчет полного типа по сформированной выпуклости

Имя:	s (linia)	
t	Ряд	Кол-во доп. рядов
0,19982	3	2
0,49985	6	1
0,69987	8	1
1	10	1
0,29983	4	1

В колонке Ряд (полный тип отчета) указывается номер ряда, после которого следует провязать рассчитанное количество дополнительных рядов, указанных в колонке справа. В данном примере указано получение выпуклости с помощью двух нитеводителей, где один нитеводитель провязывает основные петельные ряды, другой провязывает дополнительные петельные ряды.

В случае, когда необходим выпуклый участок трикотажа сложной формы можно сформировать края выпуклости при помощи кривых Безье и специальной команды:

ИМЯ:FORMD>левый край | правый край| центр;

где *ИМЯ* – идентификатор объекта выпуклого участка сложной формы;

FORMD – функция формирования объекта;

левый край – список графических объектов, формирующих левый край выпуклого участка трикотажа;

правый край – список графических объектов, формирующих правый край выпуклого участка трикотажа;

центр – идентификатор отрезка, на всей протяженности которого следует провязывать дополнительные петельные ряды.

Рассмотрим пример формирования выпуклого участка трикотажа сложной формы за счет провязывания дополнительных петельных рядов:

centr:line>10,2,10,14;
left:bzr3>centr.x1,centr.y1,3.45,13.38,3.57,3.57,7.38,1.95;
right:bzr3>centr.x1,centr.y1,16.41,13.80,17.79,6.03,12.69,1.92;
s:side>centr|right;
centr:dopr2>20,0.5,1,0.14;
s:formd>left|right|centr;
all:aprx>0.17,0.14;

Построенный данный пример представлен на рисунке П2.32.

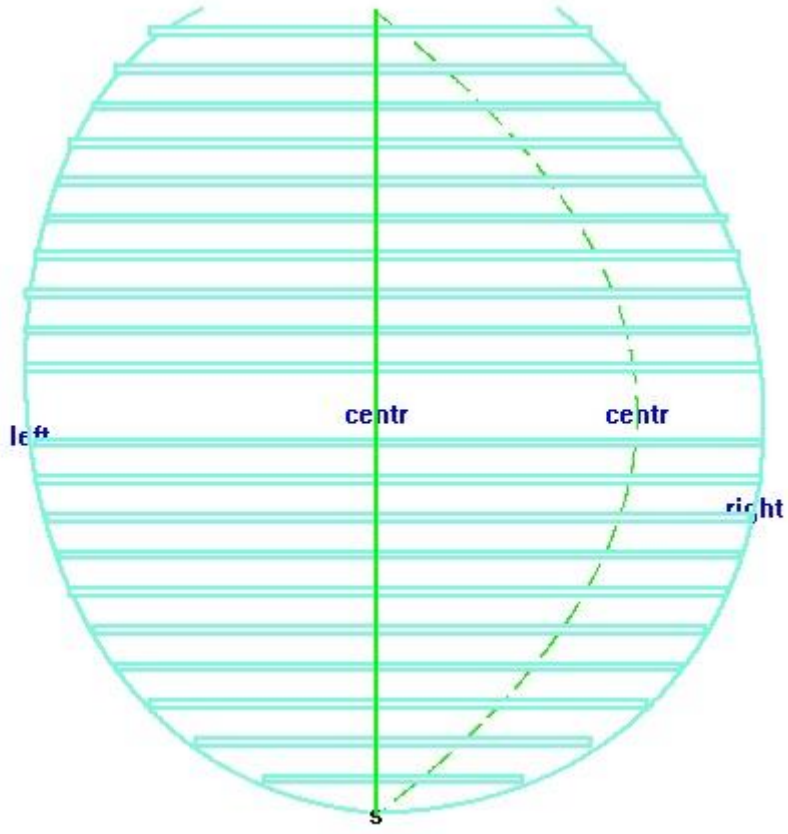


Рисунок П2.32 – Выпуклый участок трикотажа сложной формы

На рисунке П2.32 горизонтальные линии представляют собой провязанные дополнительные петельные ряды.

Выведем отчет полного типа по сформированной выпуклости (таблица П2.7).

Таблица П2.7 – Отчет полного типа для выпуклого участка трикотажа сложной формы

Имя:	s (centr)			
t	Ряд	Кол-во стол. слева	Кол-во доп. рядов	Кол-во стол. справа
0,03489	4	10	1	13
0,08156	8	16	1	19
0,12823	12	20	1	24
0,17491	16	23	1	27
0,22158	20	25	1	29
0,26825	24	27	1	31
0,31492	28	28	1	32
0,36159	32	29	1	33
0,40826	36	30	1	34
0,45493	40	30	1	34
0,96832	84	20	1	19
0,92165	80	23	1	22
0,87498	76	25	1	25
0,8283	72	27	1	27
0,78163	68	28	1	29
0,73496	64	29	1	31
0,68829	60	30	1	32
0,64162	56	31	1	33
0,59495	52	31	1	33
0,54828	48	31	1	34

В сравнении с отчетом предыдущего примера в этом указывается, какой ширины необходимо провязать тот или иной дополнительный петельный ряд.

9 ПОСТРОЕНИЕ ВЫТАЧЕК

В “DESIGNER K-WEAR” вытачки имеют несколько иную конечную форму, чем принято. В данной системе вытачка заменяется способом получения выпуклости за счет провязывания дополнительных петельных рядов (см. глава 3).

Команда построения вытачки имеет вид:

ИМЯ:VYT>список объектов слева | список объектов справа | контр. вершина слева, контр. вершина справа | высота пет. ряда | тип вытачки;

где *ИМЯ* – идентификатор вытачки;

список объектов слева, список объектов справа – графические объекты, которые должны поменять свои позиции при закрытии вытачки;

контр. вершина слева, контр. вершина справа – закрепленная опорная вершина конечного графического объекта: 0 – начальная опорная вершина, 1,2,3 – остальные опорные вершины, 4 – перемещаются все вершины.

высота пет. ряда – высота петельного ряда в области получения выпуклости;

тип вытачки – тип вытачки, определенный в “DESIGNER K-WEAR”: 1 – вертикальная вытачка с верхним закрытием, 2 – горизонтальная вытачка с правым закрытием, 3 – вертикальная вытачка с нижним закрытием, 4 – горизонтальная вытачка с левым закрытием.

Пример:

ver1:line>2,2,5,5;

ver2:line>5,5,8,2;

ver3:line>ver1.x0,ver1.y0,ver1.x1-4.5,ver1.y1-2;

ver4:line>ver2.x1,ver2.y1,ver2.x0+4.5,ver2.y0-2;

ver5:line>11,5,8,8;

ver6:line>11,5,14,8;

ver7:line>ver5.x1,ver5.y1,5.10,6.54;

ver8:line>ver6.x1,ver6.y1,16.72,6.22;

hor1:line>25,5,30,2;

hor2:line>hor1.x0,hor1.y0,hor1.x0+5,hor1.y0+3;

hor3:line>hor1.x1,hor1.y1,hor1.x1-1,hor1.y1-1.5;

hor4:line>hor2.x1,hor2.y1,hor2.x1-1,hor2.y1+1.5;

hor5:line>25,15,20,13;

hor6:line>hor5.x0,hor5.y0,hor5.x0-5,hor5.y0+3;

hor7:line>hor5.x1,hor5.y1,hor5.x1+1,hor5.y1-1.5;

hor8:line>hor6.x1,hor6.y1,hor6.x1+1,hor6.y1+1.5;

vyt1:vyt>ver1,ver3|ver2,ver4|1,1|1|1;

vyt3:vyt>ver5,ver7|ver6,ver8|1,1|1|3;

vyt2:vyt>hor1,hor3|hor2,hor4|1,1|1|2;

vyt4:vyt>hor5,hor7|hor6,hor8|1,1|1|4;

вытачка_тип_1:side>ver1+ver2|right;

вытачка_тип_2:side>hor1+hor2|right;

вытачка_тип_3:side>ver5+ver6|right;

вытачка_тип_4:side>hor5+hor6|right;

all:aprx>1,1;

На рисунке П2.32 представлены построенные вытачки в открытом состоянии, где цифрами обозначены их типы.

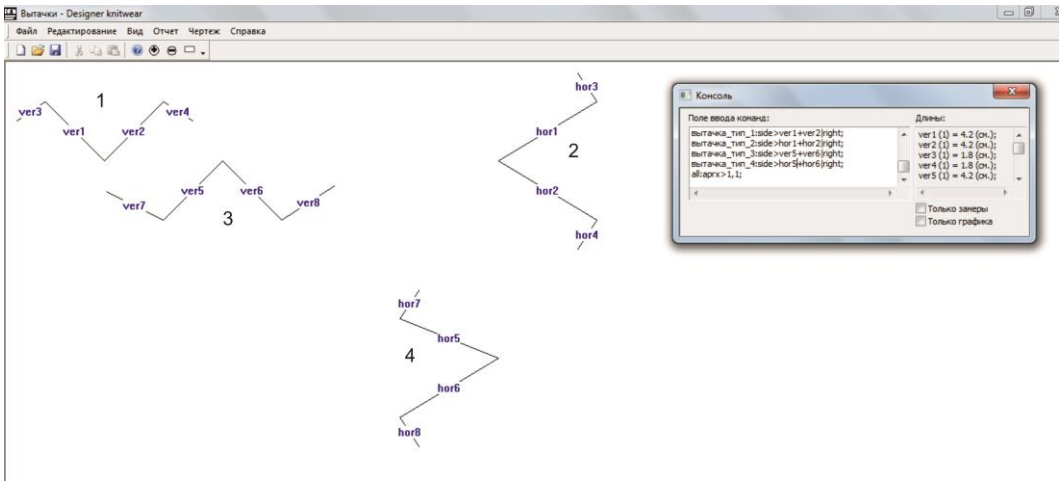


Рисунок П2.32 – Открытые вытачки

На рисунке П2.33 представлены вытачки в закрытом состоянии и с полученными, в результате закрытия, профилями выпуклых участков трикотажа.

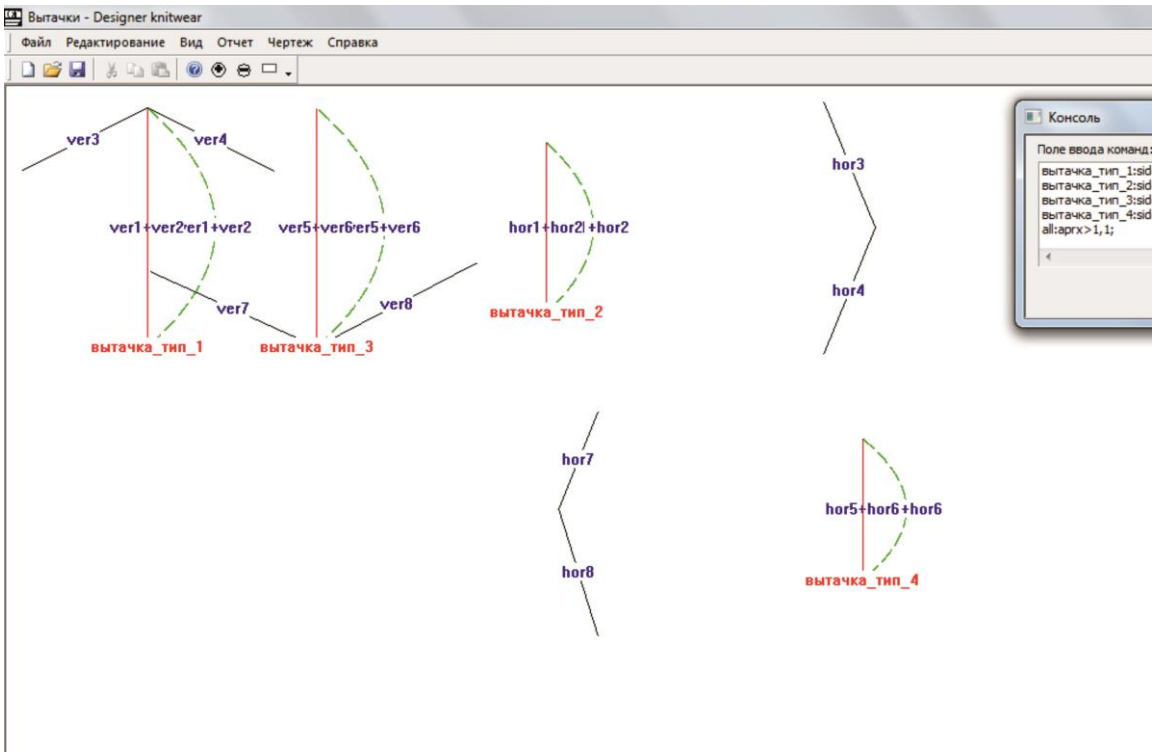


Рисунок П2.33 – Закрытые вытачки

Закрытие вытачек можно осуществить выполнив: *Чертеж* > *Вытачки* > *Закрыть* или *щелчок правой кнопкой мыши* в окне конструирования > в открывшемся контекстовом меню *Закрыть*.

Идентификатор выпуклости после закрытия вытачки генерируется за счет сложения двух идентификаторов графических объектов, оформляющих стенки вытачки. Согласно примеру идентификатор вытачки 1-го типа (рисунки П2.32 и П2.33) будет состоять из идентификаторов отрезков *ver1* и *ver2*, разделяющихся знаком «+», т.е. будет иметь вид: *ver1+ver2*. При помощи данного идентификатора вытачку в дальнейшем можно будет внести в структуру какого-либо сложного объекта (*вытачка_тип_1:side>ver1+ver2|right;*).

Важно, перед выводом отчета вытачки необходимо закрыть для их замены выпуклостями.

Выведем отчеты по дополнительным петельным рядам, сформированным в результате закрытия вытачек (таблицы П2.8, П2.9, П2.10, П2.11).

Таблица П2.8 – Отчет для вытачки типа 1

Имя:	вытачка_тип_1 (ver1+ver2)	
t	Ряд	Кол-во доп. рядов
0,23562	3	2
0,47135	5	4
0,82494	8	2
0,58921	6	2

Таблица П2.9 – Отчет для вытачки типа 2

Имя:	вытачка_тип_2 (hor1+hor2)	
t	Ряд	Кол-во доп. рядов
0,33324	3	4
1	6	2
0,49992	4	2

Таблица П2.10 – Отчет для вытачки типа 3

Имя:	вытачка_тип_3 (ver5+ver6)	
t	Ряд	Кол-во доп. рядов
0,23562	3	2
0,47135	5	4
0,82494	8	2
0,58921	6	2

Таблица П2.11 – Отчет для вытачки типа 4

Имя:	вытачка_тип_4 (hor5+hor6)	
t	Ряд	Кол-во доп. рядов
0,39969	3	4
0,59971	4	2

10 ДОПОЛНИТЕЛЬНЫЕ СВЕДЕНИЯ ОБ АППРОКСИМАЦИИ ПРИ ФОРМИРОВАНИИ УЗЛА СОЕДИНЕНИЯ В ЦЕЛЬНОВЯЗАНОМ ИЗДЕЛИИ

Для проектирования соединения деталей на цельновязаном изделии в “DESIGNER K-WEAR” заложено ряд дополнительных функций аппроксимации. Совокупность данных функций может не только существенно повысить качество цельновязаного изделия (улучшенный внешний вид узла соединения, улучшенная посадка изделия на теле человека и т.п.), но также сократить время вязания, расход сырья и т.д.

При аппроксимации кромок соединения в момент формирования узла соединения в команде по функции соединения необходимо указывать его тип.

Команда формирования соединения с типом аппроксимации по умолчанию (*тип 11*):

ИМЯ : COMB > список объектов слева | список объектов справа | | коэф. усадки по гориз., коэф. усадки по вертик.;

или

ИМЯ : COMB11 > список объектов слева | список объектов справа | | коэф. усадки по гориз., коэф. усадки по вертик.;

Данный тип аппроксимации выполняет балансирование элементов соединения по левой и правой кромках.

Тип 10:

ИМЯ : COMB10 > список объектов слева | список объектов справа | | коэф. усадки по гориз., коэф. усадки по вертик.;

При данном типе аппроксимации происходит сложение столбиков по правой кромке с рядами по левой кромке (например, при проектировании плечевого участка, когда петли стана остаются на иглах в момент провязывания следующего ряда на рукаве).

Тип 01:

ИМЯ : COMB01 > список объектов слева | список объектов справа | | коэф. усадки по гориз., коэф. усадки по вертик.;

Обратный тип 10.

Рассмотрим пример соединения типа 11:

left:bzr2>5.68,1.66,2.58,6.34,6.22,9.74;
right:bzr3>9.30,1.82,6.46,4.92,11.80,6.22,8.38,9.78;
соединение_11:comb11>left|right;
all:aprx>0.23,0.35;

Построенный пример представлен на рисунке П2.34.

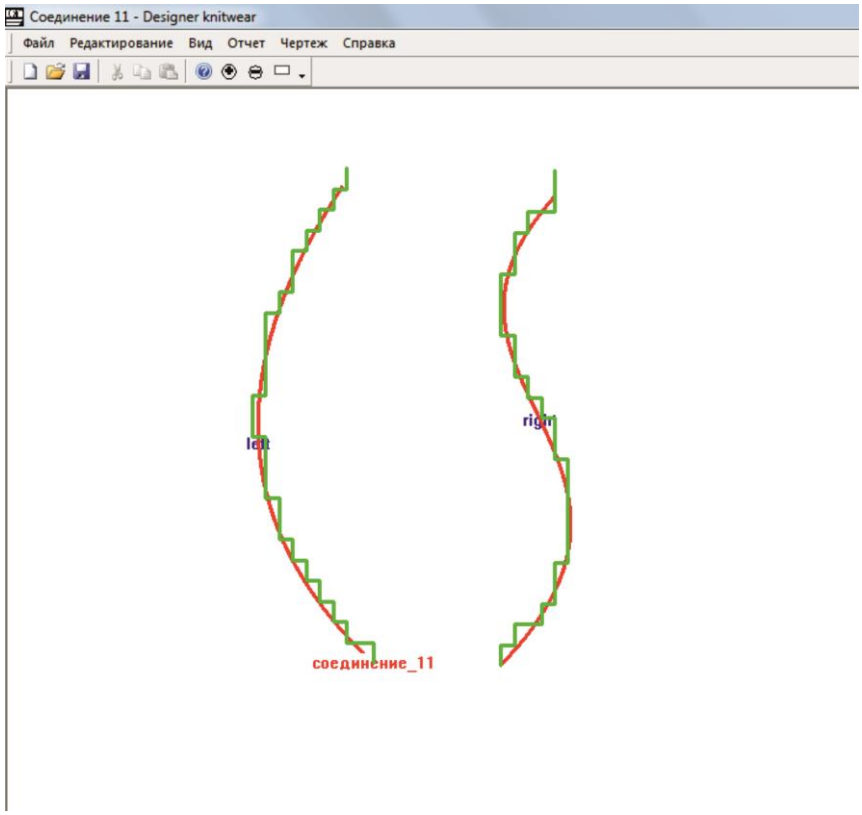


Рисунок П2.34 – Соединение типа 11

Выведем полный отчет по сформированному узлу соединения (таблицы П2.12, П2.13 и П2.14).

Таблица П2.12 – Отчет по узлу соединения для левой кромки соединения

Имя:	соединение_11 (Тип: 11)					
ax	ay	t	xl	сбавка	Ряд	Направление
6,22	9,74	0	6,22	0	1	1
5,76	9,39	0,05085	5,86724	2	2	-1
5,53	9,04	0,1009	5,554067	1	3	-1
5,3	8,69	0,15005	5,279387	1	4	-1
5,07	8,34	0,19836	5,041136	1	5	-1
4,84	7,99	0,24586	4,837553	1	6	-1
4,61	7,64	0,2926	4,666915	1	7	-1
4,61	7,29	0,33861	4,527706	0	8	1
4,38	6,94	0,38392	4,418502	1	9	-1
4,38	6,59	0,42857	4,337961	0	10	1
4,38	6,24	0,47258	4,284874	0	11	1
4,15	5,89	0,51598	4,258092	1	12	-1
4,15	5,54	0,55879	4,256549	0	13	1
4,38	5,19	0,60105	4,279256	1	14	1
4,38	4,84	0,64276	4,325274	0	15	1
4,38	4,49	0,68395	4,393732	0	16	1
4,38	4,14	0,72463	4,483791	0	17	1
4,61	3,79	0,76484	4,594732	1	18	1
4,84	3,44	0,80457	4,725754	1	19	1
4,84	3,09	0,84386	4,876251	0	20	1
5,07	2,74	0,88271	5,045524	1	21	1
5,3	2,39	0,92114	5,232983	1	22	1
5,53	2,04	0,95916	5,438034	1	23	1
5,76	1,69	1	5,68	1	24	1
5,76	1,34	1	5,68	0	25	1

Таблица П2.13 – Отчет по узлу соединения для правой кромки соединения

ax	ay	t	xl	сбавка	Ряд	Направление
8,38	9,78	0	8,38	0	1	1
8,61	9,43	0,03338	8,693827	1	2	1
9,07	9,08	0,0683	8,963562	2	3	1
9,3	8,73	0,10477	9,185953	1	4	1
9,3	8,38	0,14286	9,358786	0	5	1
9,53	8,03	0,18265	9,480483	1	6	1
9,53	7,68	0,22418	9,550197	0	7	1
9,53	7,33	0,26748	9,568309	0	8	1
9,53	6,98	0,31251	9,5368	0	9	1
9,53	6,63	0,35916	9,459802	0	10	1
9,3	6,28	0,40728	9,343884	1	11	-1
9,3	5,93	0,4566	9,198348	0	12	1
9,07	5,58	0,50677	9,035023	1	13	-1
8,84	5,23	0,55738	8,867621	1	14	-1
8,61	4,88	0,60799	8,710701	1	15	-1
8,61	4,53	0,65814	8,578503	0	16	1
8,38	4,18	0,70743	8,483628	1	17	-1
8,38	3,83	0,7555	8,436268	0	18	1
8,38	3,48	0,80212	8,443707	0	19	1
8,61	3,13	0,8471	8,510445	1	20	1
8,61	2,78	0,89034	8,638459	0	21	1
8,84	2,43	0,93183	8,827866	1	22	1
9,3	2,08	1	9,3	2	23	1
9,3	1,73	1	9,3	0	24	1
9,3	1,38	777	9,3	0	25	1

Если t = 777, то выполняется добавление компенсирующего петельного ряда.

Таблица П2.14 – Дополнительные сведения об узле соединения

Линия разностей (ряд)	
отсутствует	
Разность по рядам	
1	
Распределение доп. рядов	
через каждые	
24	рядов

Линия разностей или Линия 1 (см. глава 3) – линия после которой происходит расхождение по рядам между кромками соединения (указывается номером ряда).

Разность по рядам в данном случае означает, что в правой кромке на один ряд меньше, чем в левой (если «-1», то в левой меньше).

Пример соединения типа 01:

left:bzr2>5.68,1.66,2.58,6.34,6.22,9.74;
right:bzr3>9.30,1.82,6.46,4.92,11.80,6.22,8.38,9.78;
соединение_01:comb01>left|right;
all:aprx>0.23,0.35;

Построенный пример представлен на рисунке П2.35.

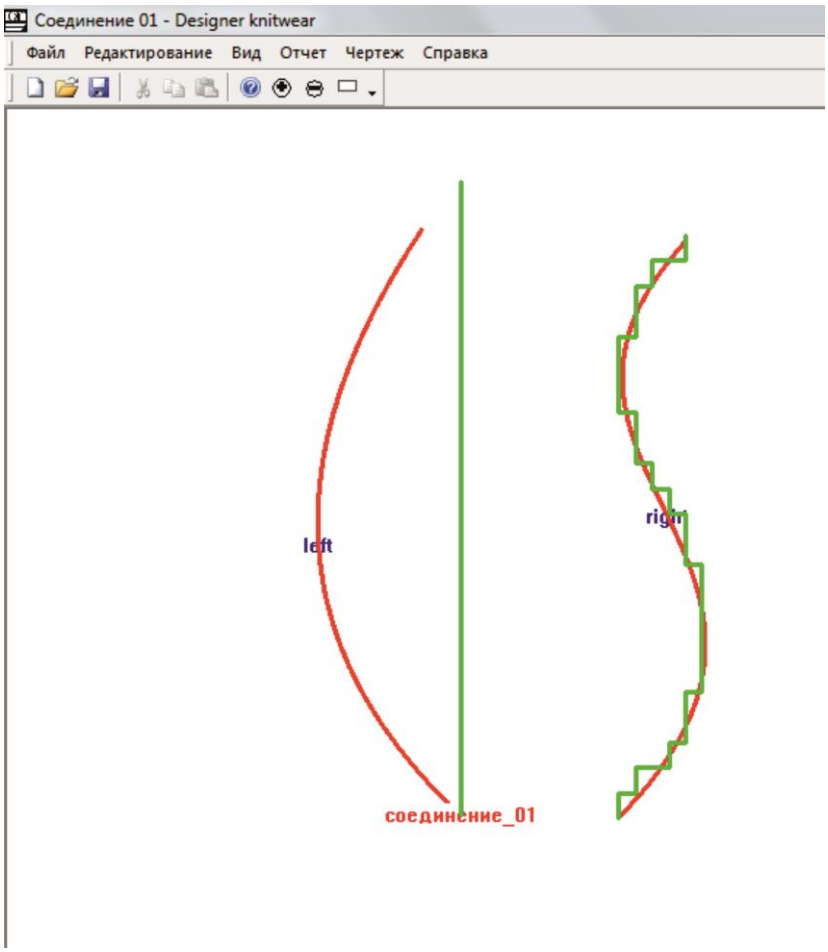


Рисунок П2.35 – Соединение типа 01

При более подробном проектировании участка соединения возможно комбинированное использование типов соединения по графическим объектам. Однако при данном способе проектирования необходимо сформировать сложные объекты (кромки, формы) и создать их сопряжение друг с другом.

Пример:

left:bzr2>5.68,1.66,2.58,6.34,6.22,9.74;

right:bzr3>9.30,1.82,6.46,4.92,11.80,6.22,8.38,9.78;

left2:bzr3>left.x2,left.y2,7.40,13.48,4.92,17.52,6.08,20.92;

right2:bzr3>right.x3,right.y3,10,10,14.52,18.36,9.68,20.68;

side_left:side>left2,left|right;

side_right:side>right2,right|left;

side_left-side_right:comb01>left2|right2;

side_left-side_right:comb11>left|right;

all:aprx>0.23,0.35;

Соединение осуществляется между двумя кромками, содержащими по несколько графических объектов (левая: left2, left; правая:right2, right).

Объекты кромок left2 и right2 соединяются между собой по типу 01, а объекты left и right - по типу 11 (рисунок П2.36).

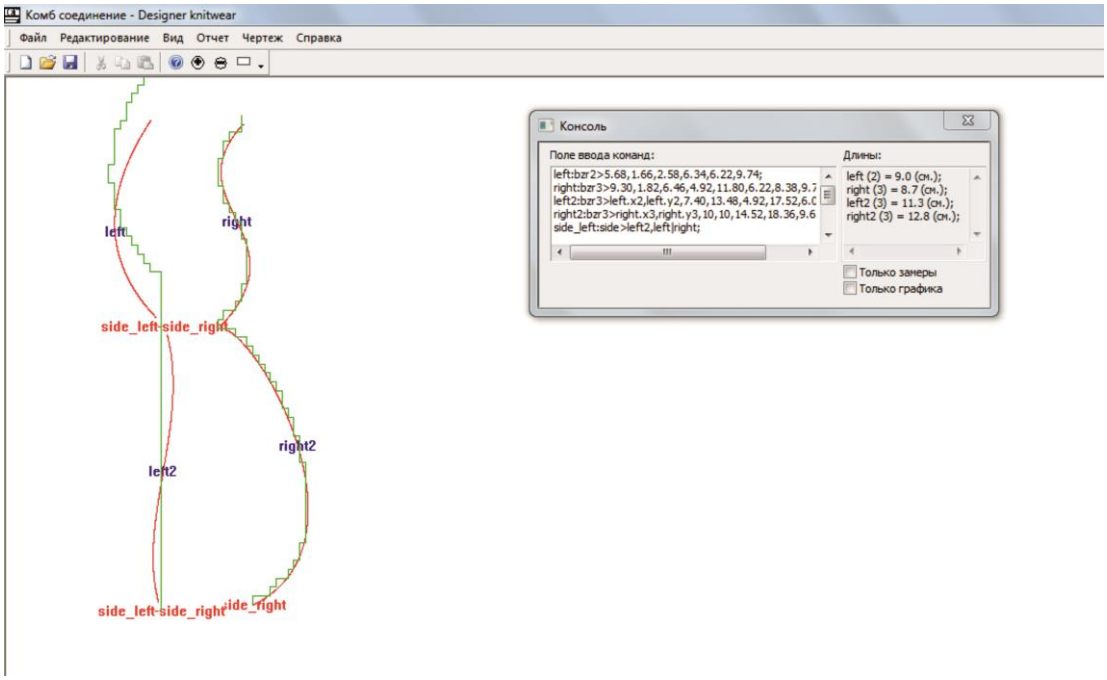


Рисунок П2.36 – Комбинированный способ проектирования соединения

ЗАКЛЮЧЕНИЕ

“DESIGNER K-WEAR” – это техническое решение совершенно нового уровня в подходе к проектированию трикотажных изделий. “DESIGNER K-WEAR”, объединяющий в себе возможности двух систем проектирования, взятых с двух различных производств (швейного и трикотажного), позволяет не только создавать конструкции проектируемых изделий, формировать на их основе лекала, рассчитывать необходимые сбавки (прибавки) и т.п., но и дает возможность для более подробной проработки мест соединения в цельновязаном изделии (за счет использования комбинированных типов соединения), получения выпуклостей за счет провязывания дополнительных рядов двумя способами вязания (одним нитеводом или двумя), а также позволяет заменить сшивание вытачки провязыванием дополнительных рядов, геометрически не меняя конечную выпуклость. При этом в “DESIGNER K-WEAR” реализована возможность динамического отслеживания геометрических отклонений аппроксимированного профиля лекала от исходного профиля, полученного при конструировании. С помощью чего можно сделать выбор не только в сторону повышения производительности оборудовании за счет частичного ухудшения посадки изделия или наоборот, но и прийти к компромиссу, включающему в себя получение качественной посадки и технологичного вязания, при этом, не снижая производительность оборудования.

Ведение собственной базы размерных признаков позволит существенно сократить время на разработку новой конструкции, а также её аппроксимацию.

Так как “DESIGNER K-WEAR” является дополнительным техническим решением, расширяющим возможности систем проектирования, поставляемых производителями вязального оборудования, то не исключена возможность параллельного использования одновременно с несколькими САПР трикотажа.