Министерство науки и высшего образования Российской Федерации
Федеральное государственное автономное образовательное учреждение
высшего образования
«Уральский федеральный университет
имени первого Президента России Б.Н. Ельцина»

Институт Строительства и Архитектуры

Кафедра Информационное моделирование в строительстве

ДОПУСТИТЬ К ЗАЩИТЕ ПЕРЕД ГЭК

/Зав. кафедрой ИМС
_____ Зверева О.М.
(подпись)
«_07_» _июня_____2024 г.

ВЫПУСКНАЯ КВАЛИФИКАЦИОННАЯ РАБОТА

Разработка методики создания 3D-моделей металлоконструкций с использованием программного обеспечения Tekla Structures и Grasshopper

Научный руководитель: Сальников В.Б. _____ подпись
кан. техн. наук, доцент
Нормоконтролер: Варгина Т.А. _____ подпись
Студент группы СТМ-221001: Соколов М.А. _____ подпись

Екатеринбург
2024

Министерство науки и высшего образования Российской Федерации
Федеральное государственное автономное образовательное учреждение
высшего образования
«Уральский федеральный университет имени первого Президента России Б.Н. Ельцина»

Институт Строительства и Архитектуры
Кафедра Информационное моделирование в строительстве
Направление 08.04.01 – Строительство
Образовательная программа «Информационное моделирование зданий, сооружений и территорий»

УТВЕРЖДАЮ
/Зав. кафедрой ИМС
(подпись) Зверева О.М. (Ф.И.О.)
«07» июня 2024 г.

ЗАДАНИЕ
на выполнение выпускной квалификационной работы

студента Соколова Максима Анатольевича группы СТМ-221001
(фамилия, имя, отчество)

1. Тема ВКР: «Разработка методики создания 3D-моделей металлоконструкций с использованием программного обеспечения Tekla Structures и Grasshopper»
Утверждена распоряжением по институту от «02» октября 2023г. № 33-03-05/4а(02)

2. Руководитель Сальников В.Б., доцент, кан. техн. наук
(Ф.И.О., должность, ученое звание, ученая степень)

3. Исходные данные к работе

Документация на конструкцию модульного укрытия верхней части ДК.

4. Содержание пояснительной записки (перечень подлежащих разработке вопросов)

Введение

Раздел 1. Обзор программного обеспечения Grasshopper и его возможностей в связке с Tekla Structures

Раздел 2. Методика создания геометрии каркаса металлоконструкций, используя Grasshopper

Раздел 3. Методика создания алгоритма параметризации 3D-модели конструкций и изменения профилей каркаса

Раздел 4. Разработка и детализация узлов модульного укрытия с помощью связки Tekla Structures – Grasshopper

Заключение

Список использованных источников

5. Перечень демонстрационных материалов
1. Слайды, описывающие актуальность, цель, задачи работы, научную новизну работы.

2. Слайды, демонстрирующие интерфейс, возможности и принципы работы Grasshopper.
3. Слайды, демонстрирующие моделирование основных частей каркаса в Grasshopper.
4. Слайды, демонстрирующие управление габаритами здания.
5. Слайды, демонстрирующие создание компонентов в Tekla Structures, и установка их в необходимые места через Grasshopper.
6.Слайды, демонстрирующие вывод.

6. Консультанты по проекту (работе) с указанием относящихся к ним разделов проекта

Раздел	Консультант		Подпись, дата	
			задание выдал	задание принял
Обзор программного обеспечения Grasshopper и его возможностей в связке с Tekla Structures	Сальников В.Б.			
Методика создания геометрии каркаса металлоконструкций, используя Grasshopper	Сальников В.Б.			
Методика создания алгоритма параметризации 3D-модели конструкций	Сальников В.Б.			
Разработка и детализация узлов модульного укрытия с помощью связки Tekla Structures – Grasshopper	Сальников В.Б.			

7. Календарный план

Наименование этапов выполнения работы	Срок выполнения этапов работы	Отметка о выполнении
Обзор программного обеспечения Grasshopper и его возможностей в связке с Tekla Structures	20.05.2024	
Разработка методики создания каркаса металлоконструкций с использованием ПО Grasshopper	30.05.2024	
Создание возможности изменения профилей элементов каркаса здания через Grasshopper.	05.06.2024	
Разработка и детализация узлов фермы модульного укрытия с помощью связки Tekla – Grasshopper	10.06.2024	

Руководитель _____ Сальников В.Б.
(подпись) Ф.И.О.

Задание принял к исполнению _____
(подпись)

8. Выпускная работа закончена «_10_» __июня_____ 2024 г.

Считаю возможным допустить Соколова Максима Анатольевича
к защите его выпускной квалификационной работы в экзаменационной комиссии.

Руководитель _____

9. Допустить Соколова Максима Анатольевича к защите выпускной квалификационной работы в экзаменационной комиссии (протокол заседания кафедры № _5_ от «_07_» _июня_ 2024 г.)

/Зав. кафедрой _____

РЕФЕРАТ

Отчет 61 с., 34 рис., 16 источников.

Ключевые слова: автоматизация, алгоритм, метод, трудозатраты, компонент, массив.

Объект ВКР − автоматизация процессов моделирования 3D-модели металлоконструкций.

Цель работы − автоматизация процессов для достижения сокращения трудозатрат на моделирование типовых объектов из металлоконструкций в Tekla Structures.

Методы исследования − эмпирические, а именно: эксперимент и тестирование; теоретические, такие как анализ, синтез и моделирование.

Результат работы − программный код в Grasshopper 3D, работающий совместно с Tekla Structures, отстраивающий основную геометрию строения из металлоконструкций и позволяющий изменять основные геометрические параметры объекта строительства, а также сечения элементов и узлы крепления элементов металлических конструкций.

Область применения полученных результатов – конструкторские бюро и проектные организации, занимающиеся разработкой комплектов КМД.

Практическая значимость работы состоит в возможности использования программного кода в ООО «КБ «Техно-Парк».

Научная новизна работы состоит в разработке алгоритма, создающего не отдельный элемент строения, а всего объекта из металлоконструкций с возможностью изменять геометрических характеристик, сечений и узлов элементов.

СОДЕРЖАНИЕ

ТЕРМИНЫ И ОПРЕДЕЛЕНИЯ

В настоящей выпускной квалификационной работе применяются следующие термины с соответствующими определениями.

Технология информационного моделирования — это процесс формирования и корректировки данных о строительных объектах. Одним из основных результатов данного процесса является цифровая информационная модель строительного объекта, которая является цифровым представлением характеристик построенного объекта.

Цифровая информационная модель — электронный документ в составе информационной модели объекта капитального строительства (ИМ ОКС), представленный в цифровом объектно-пространственном виде [1, п. 3.1.6].

Коллизия — дефект, содержащийся в цифровой информационной модели и заключающийся в пространственном или ином пересечении двух или более элементов цифровой информационной модели [1, п. 3.1.8].

Автоматизация — направление научно-технического прогресса, которое использует различные технические средства и математические методы, которые позволяют выполнять задачи быстрее, продуктивнее и эффективнее, с целью полного освобождения человека от участия в научно-технологических процессах, либо существенного уменьшения степени этого участия или трудоемкости выполняемых задач [2].

Трудозатраты — это количество времени, которое тратит специалист на выполнение конкретной задачи.

Полилиния — это совокупность отрезков и/или дуг, которая программой рассматривается как единое целое [3].

Массив — это структура данных, которая хранит упорядоченный набор однотипных элементов [4].

ПЕРЕЧЕНЬ СОКРАЩЕНИЙ И ОБОЗНАЧЕНИЙ

В настоящей выпускной квалификационной работе применяются следующие сокращения и обозначения:

BIM — Building Information Model — информационная модель здания;

IFC — Industry Foundation Classes — открытый формат файлов;

БД — база данных;

ВП и НП — верхний пояс и нижний пояс (фермы), соответственно;

ДК — дом культуры;

КБ — конструкторское бюро;

КМД — конструкции металлические детализированные;

ОКС — объект капитального строительства;

ПО — программное обеспечение;

ТИМ — технология информационного моделирования;

ЦИМ — цифровая информационная модель.

ВВЕДЕНИЕ

В современных условиях строительная отрасль имеет высокие темпы роста, это связано как с освоением новых территорий страны, так и с развитием существующих городов. Именно поэтому, данная отрасль постоянно совершенствуется, путем создания новых методов моделирования. Одним из таких направлений, развивающихся стремительными темпами, является BIM-моделирование.

По мнению Уськова В. В. Технология-BIM – это современный подход к проектированию, строительству и эксплуатации. Она позволяет объединить различные программные продукты и инструменты, что позволяет проводить моделирование значительно дешевле, упрощает процессы визуализации будущего объекта [5].

И в нынешних реалиях одним из таких современных подходов к моделированию является использование программ визуального программирования с программными комплексами для моделирования строительных объектов разной степени сложности и детализации.

Программой способной создавать модель с высокой степенью детализации металлических конструкций является Tekla Structures. Однако построение модели со сложными геометрическими параметрами в данном программном комплексе требует больших трудозатрат. Для оптимизации работы можно использовать различные программные комплексы, которые упрощают процесс моделирования. Одним из подобных комплексов является Grasshopper, преимуществом которого выступает надстройка, позволяющая создать связь с Tekla Structures.

Grasshopper – это язык визуального программирования и среда, которая работает в рамках модульного приложения для автоматизированного проектирования Rhinoceros 3D [6].

Grasshopper, как инструмент визуального программирования, позволяет автоматически создавать геометрию каркасов зданий на основе заданных параметров, а также сложные алгоритмические модели, которые могут адаптироваться к изменениям проекта.

А Tekla Structures является программным обеспечением для информационного моделирования зданий и сооружений промышленного и гражданского строительства, которое способно моделировать конструкции из различных строительных материалов, включая сталь, бетон и дерево [7].

Создание 3д-модели строительного объекта из металлоконструкций с высокой степенью детализации – это долгий и трудозатратный процесс. В связи с этим компании сталкиваются с такими проблемами как: увеличение сроков моделирования, а также возникновения различных ошибок, связанных с наличием «человеческого фактора» в работе.

При условии работы с типовыми объектами становится возможным решение представленных выше проблем с помощью визуального языка программирования Grasshopper 3D, работающего совместно с программным комплексом Tekla Structures.

Актуальность работы заключается в автоматизации процессов моделирования, что позволит организациям сократить трудозатраты и минимизировать риск возникновения ошибок в модели.

Цель – автоматизация процессов для достижения сокращения трудозатрат на моделирование типовых объектов из металлоконструкций в Tekla Structures.

Достижение поставленной цели может быть реализовано путем решения следующий задач:

– изучить ПО Grasshopper и его возможности в связке с Tekla Structures;

– определить методику создания каркаса металлоконструкций в Tekla Structures, используя только Grasshopper;

– разработать алгоритмы построения и параметризации геометрии каркаса металлоконструкций в Grasshopper;

– разработать и детализировать узлы конструкций модульного укрытия с помощью связки Tekla Structures – Grasshopper;

– создать возможность изменения профилей каркаса здания через Grasshopper.

Предполагаемые результаты:

1) программный код в Grasshopper 3D, работающий совместно с Tekla Structures, отстраивающий основную геометрию строения из металлоконструкций и позволяющий изменять основные геометрические параметры объекта строительства, а также сечения элементов;

2) дополненный и оптимизированный исходный код, отстраивающий узлы крепления элементов металлических конструкций;

3) сокращение трудозатрат, касающихся процесса моделирования типовых объектов, на 70%.

Объектом исследования является автоматизация процессов моделирования 3D-модели металлоконструкций.

Практическая значимость работы состоит в возможности использования программного кода в ООО «КБ «Техно-Парк».

1 ОБЗОР ПРОГРАММНОГО ОБЕСПЕЧЕНИЯ GRASSHOPPER И ЕГО ВОЗМОЖНОСТЕЙ В СВЯЗКЕ С TEKLA STRUCTURES

1.1 Изучение ПО Tekla Structures

Tekla Structures – программное обеспечение для информационного моделирования зданий и сооружений промышленного и гражданского строительства, способное моделировать конструкции из различных строительных материалов, включая сталь, бетон, дерево и стекло.

Tekla Structures позволяет проектировщикам и инженерам строить конструкцию здания и его компоненты с помощью трёхмерного моделирования, создавать двухмерные чертежи и получать доступ к информации о здании или сооружении. ПО Tekla Structures ранее называлось Xsteel.

Переход Xsteel на Tekla Structures в 2004 году значительно расширил функциональность и совместимость 3D-моделей [7].

Следовало бы отметить, что основными объектами проектирования в Tekla Structures являются, стальные металлоконструкции промышленных и общественных зданий, так как данная программа разрабатывалась для данного вида объектов.

Инструменты моделирования позволяют сразу строить параметрические элементы конструкций: колонны, балки, связи, ригели и раскосы любого профиля с заданными свойствами. Это делает процесс создания трехмерной модели интуитивно понятным и быстрым.

Интерфейс ПО Tekla Structures представлен на рисунке 1.1. Данный интерфейс оснащен удобной функцией скрытия/отображения необходимых вкладок на панели, и подсказками при наведении курсора на значки.

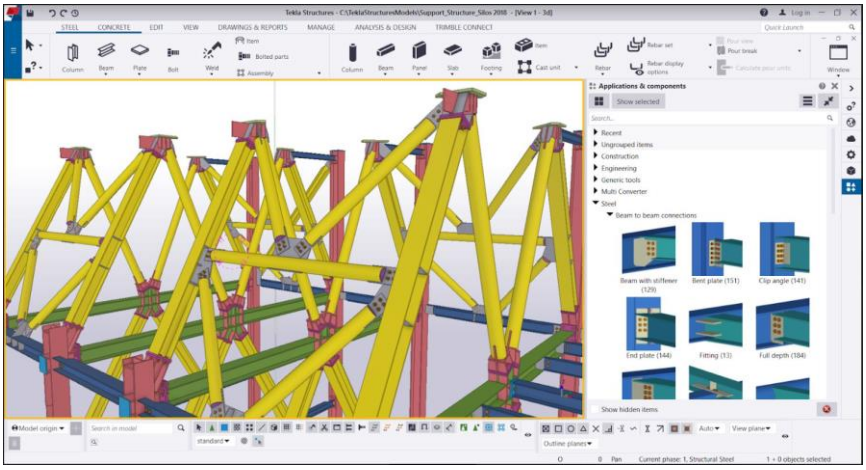

Рис. 1.1 – Интерфейс системы Tekla Structures

Tekla Structures часто используется вместе с Autodesk Revit и Navisworks — несущий каркас разрабатывается в Tekla и экспортируется в Revit с использованием форматов DWG/DXF.

В ПО также встроена система, которая генерирует файлы различных форматов для станков с ЧПУ – таких, как газокислородная и плазменная резка, станки для арматуры, пилы и сверла [7].

Выделим, что опорные модели различных форматов, таких как IFC, IFC4, IFCzip, IFCxml, tcZIP, 3DD, DXF, DWG, DGN, XML, LandXML, STP, IGS, SKP и PDF, преобразовываются механизмом TrimBimConverter в формат TrimBIM (.trb) в момент вставки опорной модели. Помимо этого, есть возможность совместной работы с ПК ЛИРА и облаками точек.

Создаваемые узлы и элементы конструкции – параметрические. Помимо обширной базы существующих узлов, у пользователя есть возможность создавать свои параметрические узлы.

Основные преимущества использования Tekla Structures:

– высокая итоговая производительность;

– возможность одновременной работы большого количества инженеров над одним проектом – многопользовательский режим;

– автоматический поиск одинаковых деталей. Автоматическая нумерация деталей и отправочных марок (сборок) КМД;

– автоматическая маркировка сборок (отправочных марок) на чертежах монтажных схем (КМД);

– автоматическая генерация чертежей отдельных деталей с размерами;

– автоматическая генерация сборочных чертежей (КМД);

– автоматическая генерация чертежей железобетонных конструкций;

– автоматическая генерация любых табличных данных на чертежах;

– автоматическая генерация любых отчетов: как текстовых, так и в форматах EXCEL;

– возможность пакетной печати всего проекта как на принтере, так и в разнообразные форматы файлов;

– возможность обмена данными с другими программами, используя выгрузки в БД. Удобная связь с 1С;

– возможность включать в состав проекта 3D-модели из огромного списка сторонних программ.

Тем не менее у данного ПО есть ряд недостатков. Например, привязка в Tekla Structures уступает AutoCad – при образмеривании деталей грани «не хватаются», или «хватаются» не там, возникают проблемы с привязками при образмеривании деталей [8].

Удобно создаются разрезы, но следует проявить внимательность при установке глубины вида – разрез может рассекать марку не там, где стоит обозначение разреза на чертеже.

Следующим минусом можно отметить необходимость повышение квалификации инженеров. Для эффективной работы в Tekla Structures необходим достаточный уровень знаний и навыков.

Далее рассмотрим следующее программное обеспечение, необходимое для достижения поставленной цели.

1.2 Изучение ПО Grasshopper

В начале исследования стоит изучить программные комплексы с помощью которых будет достигаться поставленная цель. Начнем изучение с Grasshopper.

Grasshopper представляет собой инструмент визуального программирования. Данное ПО интегрировано в модульное приложение Rhinoceros 3D, которое позволяет визуализировать геометрию элементов, сгенерированную алгоритмами Grasshopper.

Основное назначение Grasshopper заключается в разработке инновационных генеративных алгоритмов, которые позволяют создавать уникальные и высокоэффективные проекты. Однако его применение не ограничивается только этим: расширенные области применения включают параметрическое моделирование конструкций, архитектурных форм и процессов, а также проведение анализа эффективности освещения и энергопотребления зданий, обеспечивая более устойчивую и инновационную архитектуру.

Первая версия Grasshopper была выпущена в сентябре 2007 года, и с тех пор эта платформа стала важным инструментом для множества отраслей, включая параметрическую мебель, биоморфную моду, 3D-принтинг, ювелирное дело и генеративное искусство [6].

Поскольку Grasshopper 3D представляет собой популярный инструмент для проектирования, выявим его преимущества и недостатки.

Преимущества Grasshopper 3D:

– визуальное программирование: Grasshopper позволяет создавать алгоритмы и моделировать объекты с помощью визуального интерфейса, что делает процесс проектирования более интуитивным и доступным;

– гибкость и адаптивность: Grasshopper позволяет быстро изменять параметры и настройки модели, что делает его идеальным инструментом для исследования различных вариантов проекта и оптимизации дизайна;

– интеграция с Rhino 3D: Grasshopper полностью интегрирован с программой Rhino 3D, что позволяет обмениваться данными и моделями между двумя приложениями без потери качества;

– большое сообщество пользователей: Grasshopper имеет активное сообщество пользователей, которые делятся своими знаниями и опытом, что делает процесс изучения и использования программы более эффективным.

Недостатки Grasshopper 3D:

– изучение требует времени: для полноценного освоения Grasshopper требуется время и терпение, особенно для пользователей без опыта в программировании или архитектурном проектировании;

– ограниченные возможности в сравнении с программированием: хотя Grasshopper предоставляет множество инструментов для создания алгоритмов, его возможности могут быть ограничены по сравнению с традиционным программированием на языках, таких как Python или C#;

– зависимость от Rhino 3D: Grasshopper является плагином для Rhino 3D, поэтому его использование требует наличия и знания этой программы;

– производительность: некоторые сложные модели в Grasshopper могут быть ресурсоемкими и могут снизить производительность компьютера.

В целом, несмотря на некоторые ограничения, Grasshopper 3D остается одним из самых мощных инструментов для архитектурного проектирования и алгоритмического моделирования. [9]

Далее рассмотрим более детально принцип работы в Grasshopper.

Принцип работы в Grasshopper очень похож на работу в Dynamo (см. рисунок 1.2): программы создаются путем «перетаскивания» компонентов на холст. Затем выходы этих компонентов подключаются к входам последующих компонентов, образуя алгоритм.

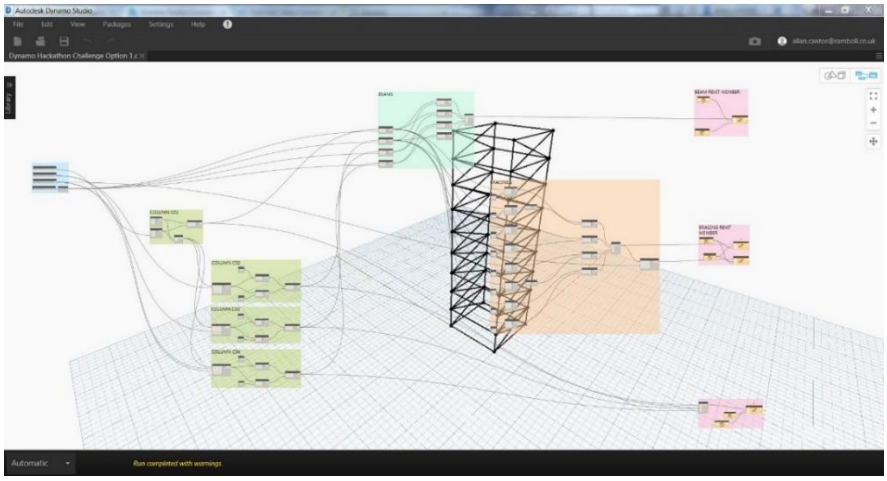

Рисунок 1.2 – Интерфейсы программы Dynamo

Вся геометрия, сгенерированная с помощью компонентов Grasshopper, будет по умолчанию отображаться во окне Rhino (см. рисунок 1.3).

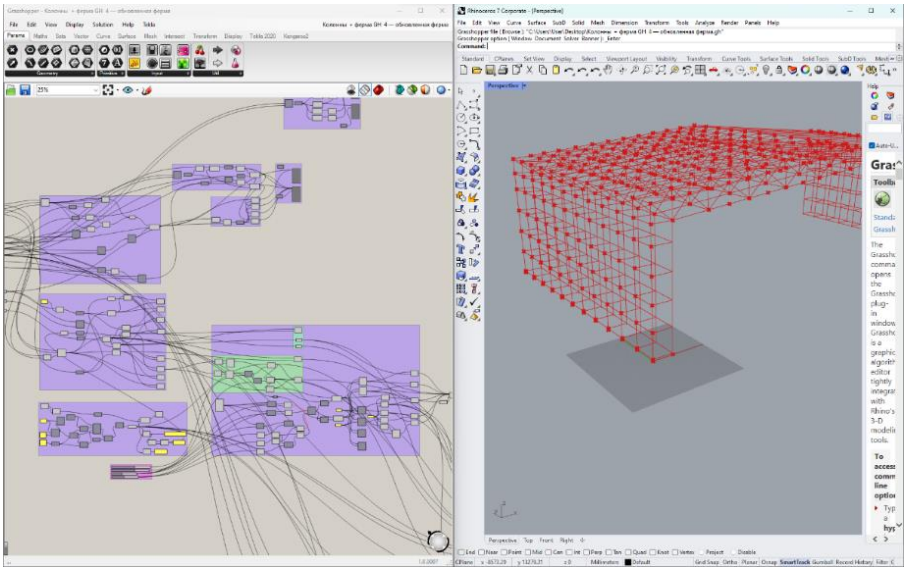

Рисунок 1.3 – Интерфейсы программ Grasshopper и Rhinoceros

Далее рассмотрим более детально интерфейс и принципы работы нодов в Grasshopper.

Как уже упоминалось ранее, визуальное программирование – это удобная концепция компьютерного программирования, в рамках которой пользователь манипулирует логическими элементами графически, а не в формате текстового алгоритма.

На рисунке 1.4 пользовательский интерфейс Grasshopper описан более детально.

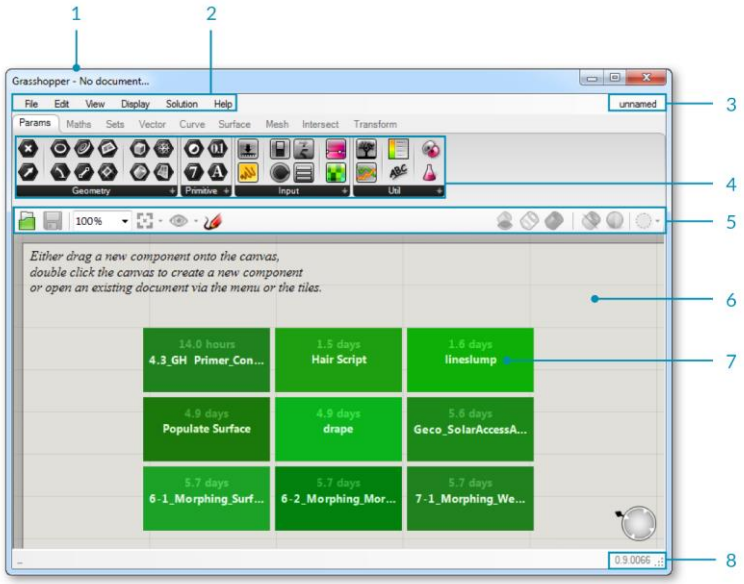

Рисунок 1.4 – Пользовательский интерфейс Grasshopper (детально): 1) строка заголовка Windows; 2) строка главного меню; 3) управление файловым браузером; 4) палитра компонентов; 5) панель инструментов; 6) холст; 7) сетка недавно использованных файлов; 8) версия программы

Grasshopper состоит из двух основных типов пользовательских объектов: параметров и компонентов. Параметры хранят данные, в то время как компоненты выполняют действия, которые приводят к данным.

Наиболее часто используемыми компонентами являются: точка, кривая и слайдер (см. рисунок 1.5).

Рисунок 1.5 – Компоненты Grasshopper: точка, кривая и слайдер (слева направо)

Компоненты — это объекты, которые пользователь размещает на холсте и соединяет вместе с помощью линий связи, образуя визуальную программу. Компоненты имеют входы и выходы в то время, как параметры могут иметь только выход (например, Слайдер).

Двойным щелчком левой кнопки мыши вызывается поиск по ключевому слову для конкретного компонента – это удобная функция, однако она требует знания названия основных параметров и компонентов (см. рисунок 1.6 – а).

Радиальное меню также может значительно увеличить скорость создания алгоритмов (см. рисунок 1.6 – б).

а) б)

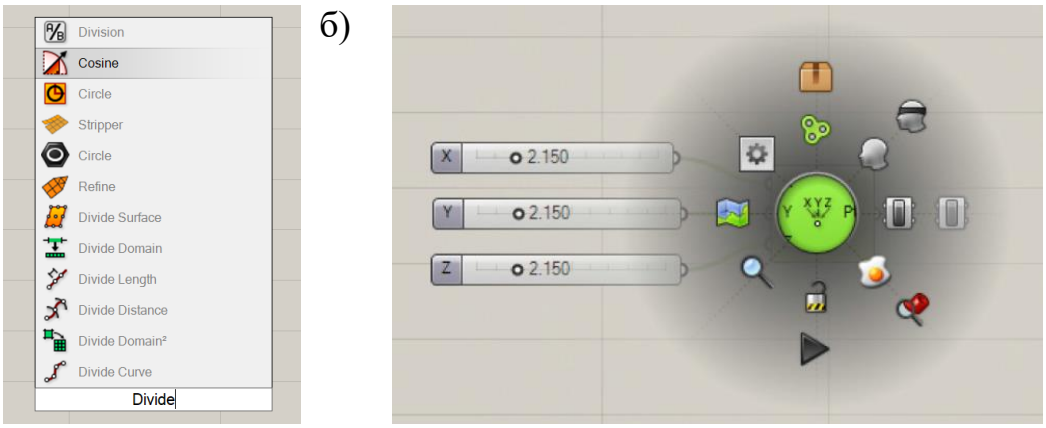

Рисунок 1.6 – а) меню поиска по ключевому слову; б) радиальное меню

Подводя итог, выделим, что Grasshopper – это многофункциональный инструмент, используемый в различных видах деятельности. Данное ПО содержит в себе множество компонентов и функций, благодаря которым, становится возможным создание сложной геометрии с высокой степенью точности, а также автоматизация многих процессов, затрагивающих моделирование, на чем и будет строиться данная работа.

1.3 Настройка связки программ Tekla Structures - Grasshopper

Обратим внимание, что рассмотренные выше программные обеспечения имеют возможность совместной работы.

Связка Tekla Structures с Grasshopper представляет собой набор компонентов, которые могут создавать объекты и взаимодействовать между собой в реальном времени в Tekla Structures.

Для настройки связки между программами необходимо скачать соответствующую надстройку с сайта Tekla Warehouse и установить на свой компьютер.

Для корректной работы связки программ необходимо первым делом запускать Tekla, а уже после неё – Grasshopper.

Если не соблюдать это условие, связка не образуется и Grasshopper будет выдавать ошибку работы компонентов [10].

Совместная работа Tekla и Grasshopper обеспечивается посредством компонентов Grasshopper, которые строят такие конструктивные элементы, как балки, колонны и пластины, а также оперируют их свойствами в модели Tekla.

Используя язык программирования Visual C# или Python, пользователь также может создавать собственные подобные компоненты для расширения функциональных возможностей существующей связки [11].

Автором было изучено, что именно такая комбинация программ позволяет обеспечить автоматизацию процесса моделирования металлоконструкций путем построения алгоритма в Grasshopper с отображением изменений в Tekla Structures в реальном времени.

Подробное описание процесса создания алгоритма в Grasshopper будет рассмотрено в следующих главах данной работы.

2 МЕТОДИКА СОЗДАНИЯ ГЕОМЕТРИИ КАРКАСА МЕТАЛЛОКОНСТРУКЦИЙ, ИСПОЛЬЗУЯ GRASSHOPPER

2.1 Определение входных параметров для построения алгоритма

На начальном этапе работы необходимо определить входные параметры объекта и произвести построение зависимости между элементами, чтобы пользователь мог изменять шаг как самих колонн и ферм, так и шаг конструкций, входящих в колонны и фермы, высоту, ширину и длину строения, угол наклона фермы и т.п.

Основой для разработки алгоритма послужила документация на конструкции модульного укрытия верхней части ДК.

В ходе работы были определены следующие входные параметры алгоритма построения основной геометрии ОКС:

– координаты X, Y и Z для начальной точки построения геометрии каркаса;

– высота составной колонны, на которую опирается ферма;

– расстояние (вдоль оси X) между стойками колонны;

– шаг стоек составной колонны (вдоль оси Y);

– количество шагов стоек составной колонны;

– смещение горизонтальных элементов колонны от нижней точки стойки по высоте (вдоль оси Z);

– шаг горизонтальных элементов колонны (вдоль оси Z);

– расстояние от начальной точки до конька (вдоль оси X);

– высота подъема конька (вдоль оси Z);

– высота фермы;

– шаг стоек фермы;

– количество шагов стоек фермы.

Начальной точкой моделирования с использованием алгоритма определим нижнюю точку крайней стойки колонны проектируемого строения.

Входные параметры алгоритма построения основной геометрии проиллюстрированы на рисунке 2.1.

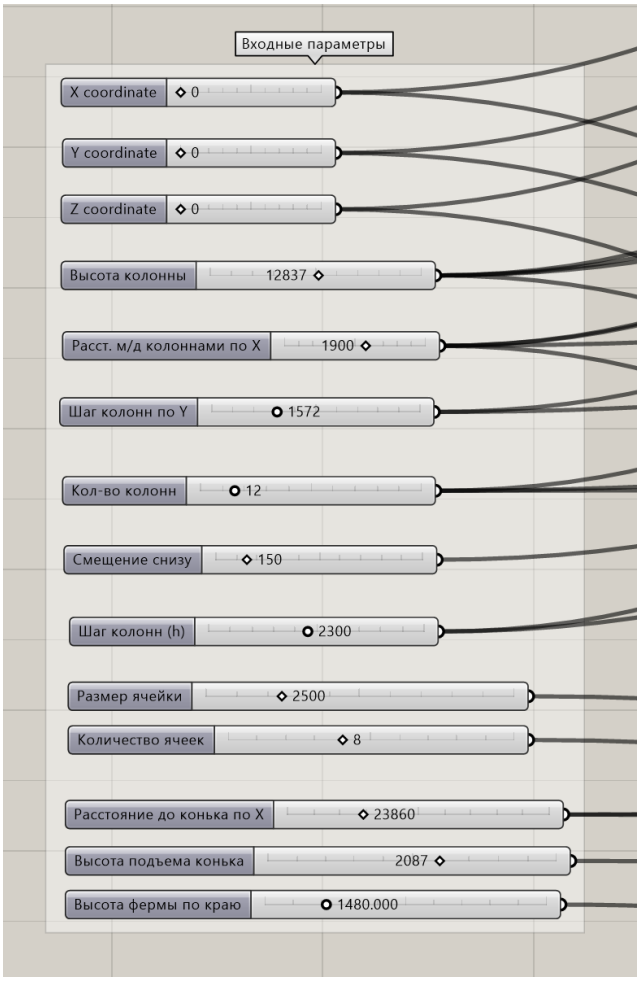

Рисунок 2.1 – Входные параметры для построения основной геометрии строения

После определения входных параметров можем приступить к следующему этапу работы.

2.2 Составление алгоритмов построения геометрии

Так как строение имеет ось симметрии, на первом этапе геометрия будет выстроена по одну сторону от неё, т.е. слева, а на втором – все элементы будут «отзеркалены» относительно оси симметрии.

Следует обратить внимание на то, что на данном этапе работы речь будет идти о построении линий, определяющих положение основных элементов каркаса, так как преобразование данных линий в конкретные профили будет осуществлено на другом этапе работы.

Поскольку начальной точкой была определена нижняя точка крайней стойки колонны строения, то относительно неё будет начинаться составление алгоритма построения геометрии.

2.3 Описание алгоритма построения геометрии стоек колонны

Опишем алгоритм построения стоек левой составной колонны.

Заданные координаты начальной точки являются входными параметрами для компонента «Construct Point», строящего эту точку. Таким образом задается нижняя точка крайней стойки колонны строения. Чтобы построить следующую точку, т.е. верхнюю точку этой же стойки необходимо использовать компонент «Move», входными параметрами для которого является точка, описанная выше, и высота колонны, заданная через вектор, направленный вдоль оси Z. После проделанных манипуляций строится линия, компонент «Line», входными параметрами для которой являются найденные ранее точки стойки.

Соседняя стойка колонны строится похожим образом: используя компонент «Move», строится точка, определяющая ширину колонны, входящими данными является начальная точка и ширина колонны, заданная через смещение по оси X. Чтобы построить верхнюю точку, аналогично предыдущей стойке, используется компонент «Move». После чего строится линия, компонент «Line», объединяющая точки второй стойки. На данном этапе имеются две прямые, определяющие высоту и ширину колонны.

Следующим шагом будет определена длина колонны. Для этого необходимо определить массив значений координат вдоль оси Y, это осуществляется компонентом «Series», входными параметрами для которого

являются шаг стоек и количество шагов стоек. После чего значения данного массива переводятся в векторный формат компонентом «Unit Y», таким образом, что на выходе получается массив векторов, в котором ненулевыми будут являться только значения, соответствующие направлению вдоль оси Y.

Заключительным этапом построения геометрии стоек колонны является копирование с заданным шагом двух линий, определяющих высоту и ширину колонны, для этого используется компонент «Move», входные данные: определенные ранее линии стоек и массив векторов.

Итог описанного выше алгоритма проиллюстрирован на рисунке 2.2.

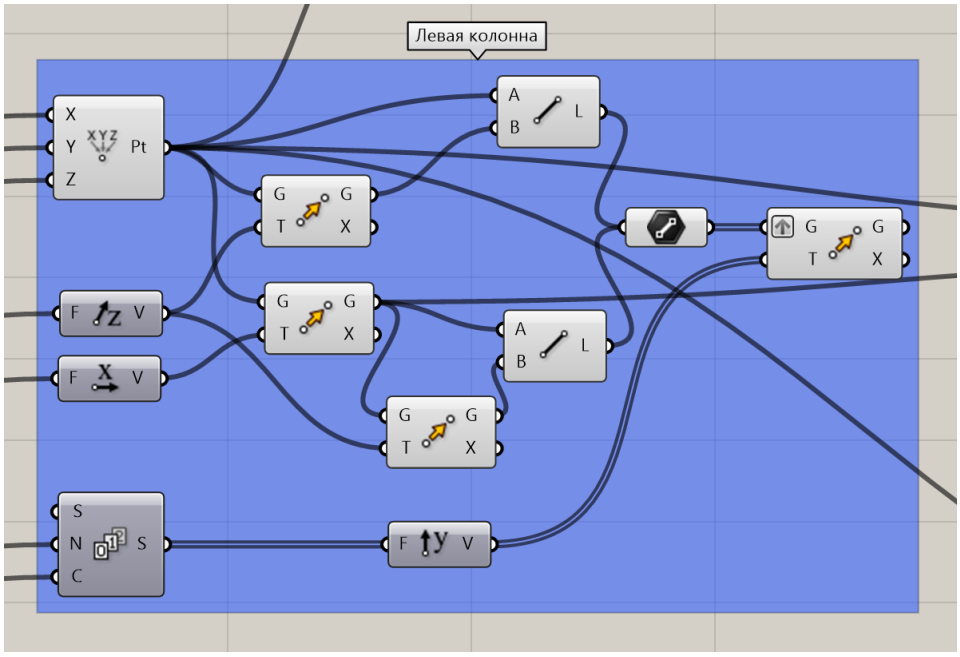

Рисунок 2.2 – Алгоритм построения стоек левой составной колонны

Следующим этапом опишем алгоритм построения горизонтальных связей левой составной колонны.

2.4 Описание алгоритма построения геометрии горизонтальных связей колонны

Чтобы построить горизонтальные связи вдоль оси Y необходимо определить начальные точки элементов, для этого необходим массив, созданный ранее для определения длинны колонны. Значения данного

массива, как и начальная точка, будут являться входными данными для компонента «Move». Таким образом будет определен массив начальных точек горизонтальных связей вдоль оси Y.

Чтобы определить количество горизонтальных связей по вертикали обратимся к математическим компонентам. Необходимо разделить значение высоты колонны на шаг горизонтальных элементов колонны вдоль оси Z, используя компонент «Integer Division», и прибавить единицу, используя компонент «Addition». Полученное значение будет использоваться в компоненте «Series», как количество шагов вдоль оси Z, также в этот компонент войдут шаг и смещение горизонтальных элементов колонны от нижней точки стойки по высоте, которые задаются пользователем.

После чего полученному массиву значений необходимо задать направление вдоль оси Z, и использовать его для копирования массива начальных точек горизонтальных связей вдоль оси Y, используя «Move». Таким образом, если соединить точки массива полилинией, то построение будет идти вдоль оси Z, а нам необходимо построение вдоль оси Y. Поэтому требуется переопределить направление построения точек, используя компонент «Flip Matrix». Далее значения полученного массива вносятся в компонент «Polyline», с последующим разбиением на сегменты компонентом «Explode». После чего компонентом «Move» строятся горизонтальные связи по другую сторону колонны, входными данными являются линии, полученные компонентом «Explode» и смещение вдоль оси X на значение ширины колонны.

Горизонтальные связи вдоль оси X строятся следующим образом. Для начала строится линия горизонтальной связи с начальной точкой модели компонентом «Line», после чего, используя созданный ранее массив значений вдоль оси Z, через «Move» строятся связи на всю высоту колонны с заданным шагом. На заключительном этапе, используя «Move» и массив определяющий шаг стоек вдоль оси Y, строятся все горизонтальные связи вдоль оси X

Описанный выше алгоритма проиллюстрирован на рисунке 2.3.

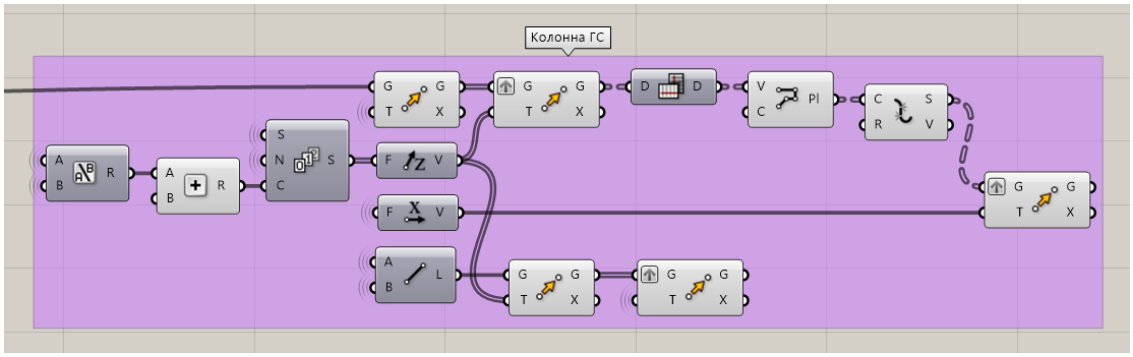

Рисунок 2.3 – Алгоритм построения горизонтальных связей составной колонны

Далее опишем алгоритм для построения геометрии вертикальных раскосов составной колонны.

2.5 Описание алгоритма построения геометрии вертикальных раскосов колонны

Введем новую переменную, т.е. новый входной параметр, задаваемый пользователем: «Расцентровка по раскосам колонны». Чтобы построить раскосы, исходящие от соседних стоек колонны и сходящиеся друг к другу на значение, регулируемое расцентровкой, необходимо использовать математические компоненты. Таким образом, используя «Division» определяется середина между стойками, а компонентами «Subtraction» и «Addition» задается расцентровка от найденной середины. Полученным значениям задается направление по оси X. Основой данного алгоритма являются точки горизонтальных связей, расположенные на крайней стойке колонны, данным точкам задаются смещения вдоль оси X, определенные ранее.

Для построения вертикальных раскосов понадобятся не все найденные точки, поэтому лишние из них необходимо исключить, для этого подойдет компонент «Cull Index». Для точек, расположенных вдоль стоек, используя указанный компонент, удаляются верхние точки, а для точек, определенных расцентровкой, удаляются нижние. После чего, полученные массивы точек

попарно объединяются компонентом «Line», образуя геометрию вертикальных раскосов.

Верхний раскос колонны геометрически представляет собой наклонную прямую, поэтому опишем его построение. В компонент «List Item» необходимо задать массив точек горизонтальных связей, расположенных на крайней стойке и оставить только верхнюю точку, задав параметр «-1». Далее строим линию, объединяющую найденную точку с верхней точкой соседней стойки. На заключительном этапе построения раскосов, всю полученную геометрию копируем вдоль оси Y с шагом колонн, используя «Move».

Результат описанного алгоритма проиллюстрирован на рисунке 2.4.

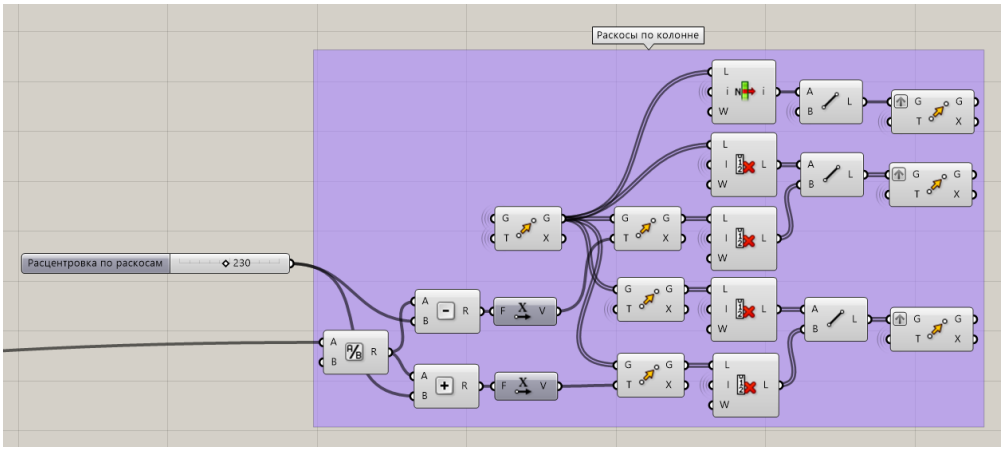

Рисунок 2.4 – Алгоритм построения вертикальных раскосов составной колонны

Следующим этапом опишем алгоритм построения вертикальных связей составной колонны, направленных вдоль оси Y.

2.6 Описание алгоритма построения геометрии вертикальных связей колонны

Основой данного алгоритма так же являются точки горизонтальных связей, расположенные на крайней стойке колонны, данным точкам задаются смещения вдоль оси Y и фильтрация. Так как данный вид связей располагается только вначале и в конце колонны, то для начала будут выстроены связи с одной стороны колонны, а в конец они будут скопированы.

Начальный массив точек следует скопировать в направлении Y с заданным шагом колонн: на 1 и 2 шага. После чего имеем три массива точек, расположенных вдоль трех стоек. К массиву точек, расположенному посередине между двумя оставшимися, будут сходится вертикальные связи, поэтому нижнюю точку из него нужно удалить компонентом «Cull Index», задав индекс «0». Далее из полученного массива удаляется каждая вторая точка компонентом «Cull Nth» с входным индексом «2». Таким образом получены точки схождения раскосов.

Найдем точки, из которых будут выстроены раскосы, это будет 4 массива, так как необходимо уже на данном этапе осуществить параметризацию, т.е. логика построений не должна быть нарушена при изменении количества вертикальных шагов колонны. Из двух оставшихся соседних массивов точек, расположенных вдоль стоек, необходимо удалить каждую вторую точку компонентом «Cull Nth» с входным индексом «2». После чего построим первые два массива вертикальных связей. В два компонента «Line» вводятся массивы точек с крайних стоек и массив с центральной стойки. Два этих массива образуют геометрию, связей в которой средняя точка находится выше крайних точек. Поэтому необходимо построить ещё два массива, когда средняя точка будет находиться ниже крайних точек. Для этого из среднего массива удаляется верхняя точка, а из крайних – нижняя. Попарное их объединение через компонент «Line», дает нам искомые два массива.

На данном этапе имеется геометрия вертикальных связей колонны вдоль одной её грани, поэтому, используя «Move», необходимо скопировать полученную геометрию вдоль оси X на значение равное ширине колонны. Так как с другой стороны колонны необходимы те же самые построения, геометрия связей должна быть скопирована, используя «Move», в конец колоны. Направление задается вдоль оси Y, а значение перемещения вычисляется математически, в зависимости от количества шагов стоек колонны вдоль оси Y. Так как шагов должно быть на 3 единицы меньше,

используя «Substruction», необходимо вычесть из заданного шага стоек это значение, после чего, используя «Multiplication», определить перемещение связей к другому концу колонны, задав определенное ранее количество шагов и значение шага вдоль оси Y. Итог продемонстрирован на рисунке 2.5.

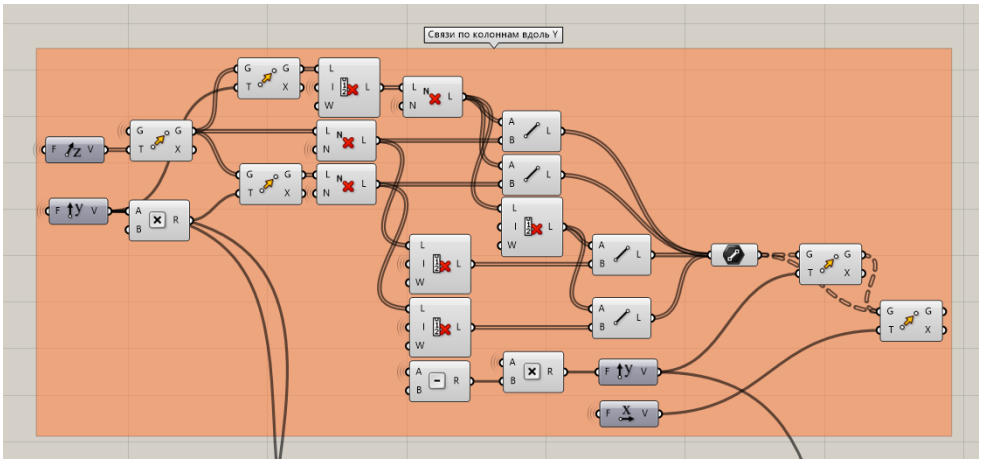

Рисунок 2.5 – Алгоритм построения вертикальных связей составной колонны

Итогом работы на данном этапе является составная колонна, состоящая из стоек с вертикальными связями и распорками, а также горизонтальными связями. Представим вид получившейся колонны в окне ПО Rhinoceros 3D на рисунке 2.6.

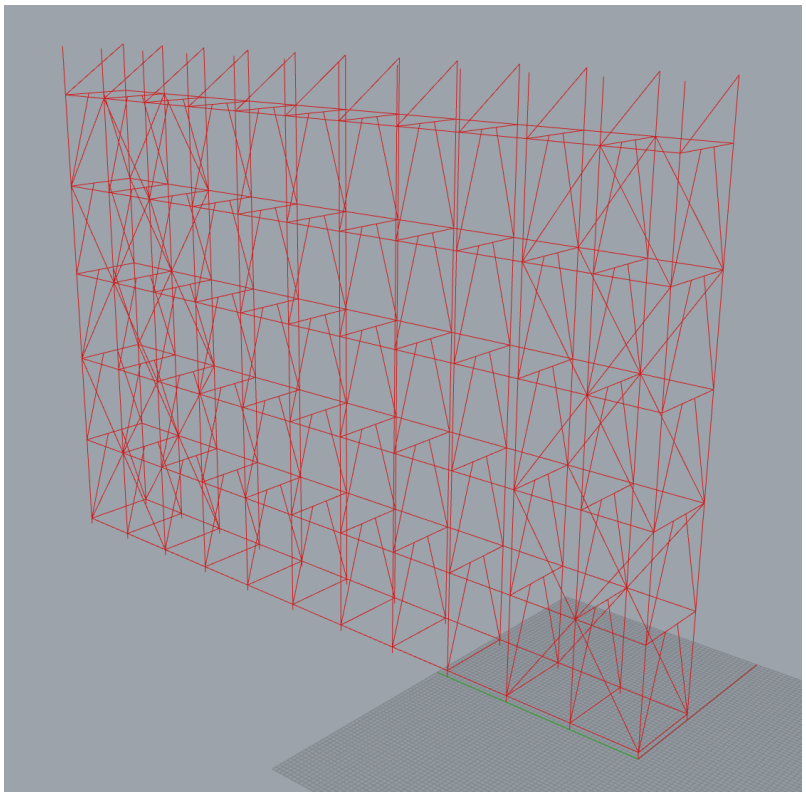

Рисунок 2.6 – Вид составной колонны в окне ПО Rhinoceros 3D

Это не вся геометрия колонны, оставшаяся геометрия будет описана далее, так как она начинает граничить с фермой и какие-то элементы можно считать принадлежащими как ферме, так и колонне. Также поводом для разделения описания алгоритмов построения геометрии послужила разница в сложности этих алгоритмов. В другой главе данной работы будут описаны более сложные методы построения, необходимость которых была вызвана усложненной параметризацией геометрии.

Так как на метод построения основной геометрии колонны уже был описан, перейдем к описанию алгоритма построения фермы строения.

2.7 Описание алгоритма построения геометрии контура фермы

В нашем случае метод построения геометрии контура фермы описывает алгоритм для левой половины фермы, так как вторая половина будет создана отзеркаливанием.

Начнем описывать построение с верхней крайней точки стойки колонны, далее – исходная точка, ведь именно в этом месте начинается геометрия фермы. Чтобы построить крайнюю стойку, необходимо скопировать исходную точку по оси Z на значение высоты фермы. Значению высоты задается направления через компонент «Unit Z», после чего данная точка копируется на указанную высоту компонентом «Move». Геометрия стойки образуется компонентом «Line», входными параметрами для которого являются две точки, описанные ранее.

Чтобы построить горизонтальную часть нижнего пояса фермы, опирающуюся на колонну, компонентом «Line» следует объединить две верхние точки торцевых стоек колонны.

Чтобы построить наклонные элементы верхнего и нижнего пояса фермы, необходимо определить координаты X и Z конька фермы, а они задаются пользователем. Следовательно, построить точку конькового узла можно, используя компонент «Vector XYZ», который задаст перемещение по X и Z

для верхней точки крайней стойки фермы. После чего определяется верхняя точка нижнего пояса фермы, она находится ниже точки конька, на значение равное высоте фермы. Поэтому, используя компонент «Move», строится эта точка. Чтобы получить геометрию верхнего пояса фермы, используя компонент «Line», необходимо объединить точку конька фермы и верхнюю точку крайней стойки фермы. Для НП используется аналогичный метод, но в данном случае объединяются определенные ранее крайние точки нижнего пояса.

Алгоритм описанный выше представлен на рисунке 2.7.

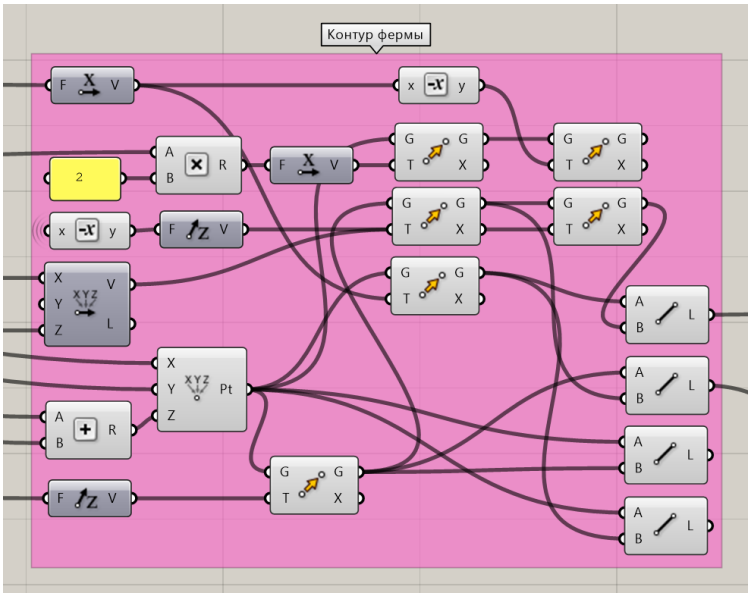

Рисунок 2.7 – Алгоритм построения геометрии контура фермы

Далее опишем алгоритм построения решетки фермы, аналогично только для ее половины.

2.8 Описание алгоритма построения геометрии решетки фермы

Для крайних раскосов фермы, граничащих с колонной, характерен особый шаг расстановки, это известно из исходной документации. Следовательно, необходимо ввести две новые переменные с возможностью их изменения пользователем, задаются через компонент «Slider». После того, как

величины первого и второго отступа для раскоса заданы, можно начать определять их исходные точки.

Первый раскос фермы находится прямо над колонной, поэтому для его построения уже есть исходные данные: соединив верхнюю точку крайней стойки колонны с предпоследней точкой соседней колонны компонентом «Line» получим геометрию первого раскоса.

Автором было определено ранее, что следующие два раскоса имеют нестандартный шаг, опишем их построение. Чтобы это сделать, будет использоваться метод копирования существующих точек стойки колонны по направлению ВП и НП фермы. Чтобы определить вектор копирования для ВП линию верхнего пояса, построенную ранее, необходимо разложить на точки конца и начала компонентом «End Points», после чего они помещаются в компонент «Vector 2Pt», образуя вектор направления. Далее, чтобы вдоль указанного вектора стало возможным перемещение точки на указанное расстояние, используется компонент «Amplitude» с найденным вектором и заданным шагом. Выход из компонента «Amplitude» необходимо объединить с «Move» с заданной точкой. Таким образом была определена верхняя точка схождения двух раскосов первого сегмента фермы. Чтобы определить нижнюю точку второго раскоса используем «Amplitude» с тем же вектором, но другим шагом и переместим найденную ранее точку. Так как искомая точка находится на НП фермы, спроецируем найденную точку на нижний пояс. Для этого используется компонент «Pull Point», на вход для которого определяется исходная точка и линия НП. После того как точка спроецирована можно построить геометрию раскосов первого (нестандартного) сегмента фермы, объединив необходимые точки, компонентом «Line».

Далее опишем алгоритм построения оставшихся сегментов фермы, включая стойки и раскосы, но в данном построении необходимо учитывать, что количество шагов, а также их размер задается пользователем, что осложняет построение геометрии.

Для начала с помощью логических переменных следует определить: шаг сегмента получается больше или меньше заданного пользователем, для этого используются «Larger Than» и «Smaller Than». Фильтрация этих значений происходит через «Cull Pattern». После, используя те же логические выражения и математические компоненты определяется шаг сегментов фермы. Выстроив определенную логику между компонентами деления исходных величин на значения шага и определив граничные значения количества шагов, имеется возможность создать массив шагов сегментов фермы, используя «Series» с найденным ранее значением шага и количеством его повторений. Чтобы найти все точки раскосов и стоек по НП и ВП следует данному массиву задать перемещение вдоль поясов компонентом «Amplitude». После чего, используя крайние точки первого (нестандартного) сегмента, значение из компонента «Amplitude» и компонент «Move», определяются массивы точек раскосов и стоек для НП. Аналогичным образом – для ВП фермы.

Чтобы построить стойки необходимо удалить каждую вторую точку массивов по НП и ВП, используя «Cull Nth», после чего, объединить полученные массивы компонентом «Line». Геометрия стоек, кроме коньковой – готова.

Ранее была описана методика построения раскосов, в данном случае будет использована похожая логика, но также будут использованы логические операторы, позволяющие отбросить лишние раскосы, которые могли бы появиться при изменении входных параметров. Таким же образом с помощью «Cull Nth», «Cull Index» и «List Item» вначале определяется положение раскосов с одним направлением, а потом с другим, но коньковый раскос строится отдельно, так как зачастую не вписывается в шаг, заданный пользователем. Далее логические переменные позволяют определить количество раскосов в крайнем сегменте: один или два, если два, то сегмент целый, если один, то визуально строится половина сегмента. Данная

особенность существует, так как заранее не всегда можно предугадать, какая по итогу получится геометрия раскосов и стоек.

Стойка конька строится с использованием конечных точек НП и ВП фермы, объединенных компонентом «Line».

Результат описанной методики построения геометрии решетки фермы проиллюстрирован на рисунке 2.8.

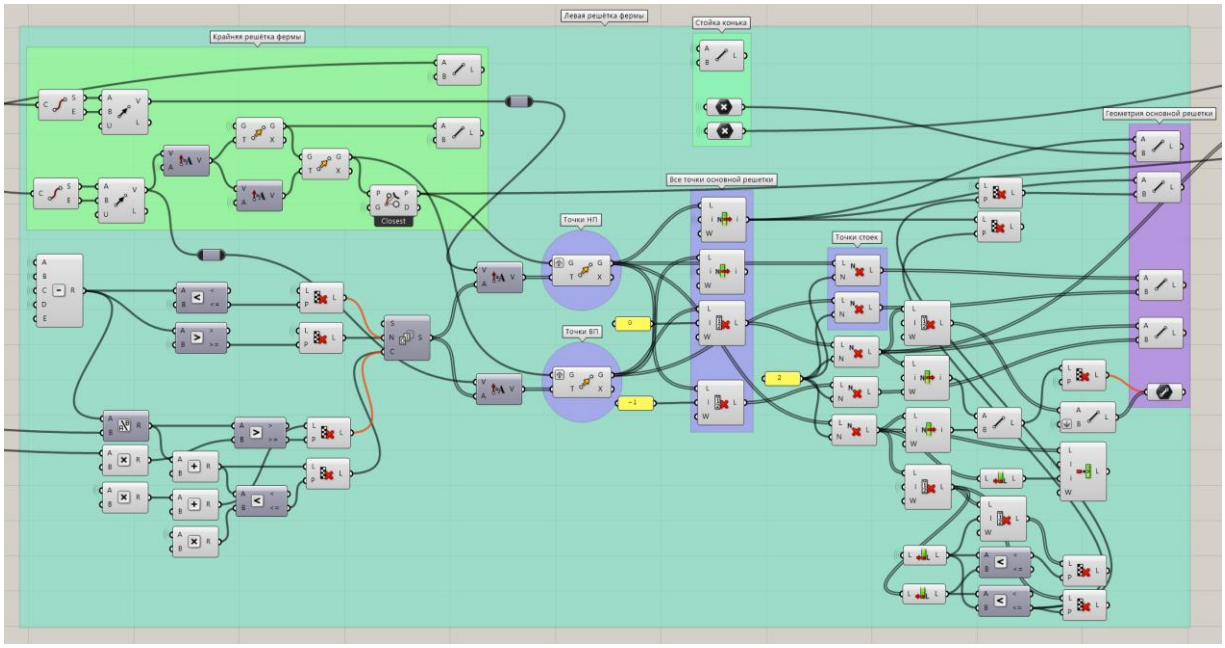

Рисунок 2.8 – Алгоритм построения геометрии решетки фермы

Так как описанная геометрия является формообразующей, то как и в случае с колонной, её следует скопировать по направлению Y с шагом стоек. Так мы получим контур фермы, но без горизонтальных связей.

Далее опишем алгоритм построения горизонтальных связей по ВП фермы, для тех же построений на НП, будет использовано копирование с верхнего пояса.

2.9 Описание алгоритма построения геометрии горизонтальных связей фермы

Как и для построения вертикальных связей вдоль колонны, используется алгоритм трех массивов, основой данного алгоритма являются точки раскосов

по ВП, точкам задаются смещения вдоль оси Y и фильтрация. Так как данный вид связей располагается только вначале и в конце фермы, то для начала будут выстроены связи с одной стороны фермы, а в конец они будут скопированы.

Начальный массив точек следует скопировать в направлении Y с заданным шагом колонн. После чего имеем три массива точек, расположенных вдоль верхних поясов. К массиву точек, расположенному посередине между двумя оставшимися, будут сходится горизонтальные связи. Так как некоторые точки не были массивом, например, точки первого сегмента фермы, то горизонтальные связи в таких местах задаются отдельно для каждого элемента, путем выбора конкретных точек, их перемещения компонентом «Move» и объединения компонентом «Line».

Однако, стоит отметить, что из-за параметризируемой геометрии, количество горизонтальных связей не является постоянным значением, а значит необходимо вводить компоненты, учитывающие это при построении геометрии. Поэтому был введен компонент «Shortest List», который приводит два списка к одной длине, выбрасывая лишние значения из более длинного списка. Таким образом, изменение количества сегментов фермы или их длины, никак не повлияет на точность построения геометрии горизонтальных связей.

Алгоритм описанный выше представлен на рисунке 2.9.

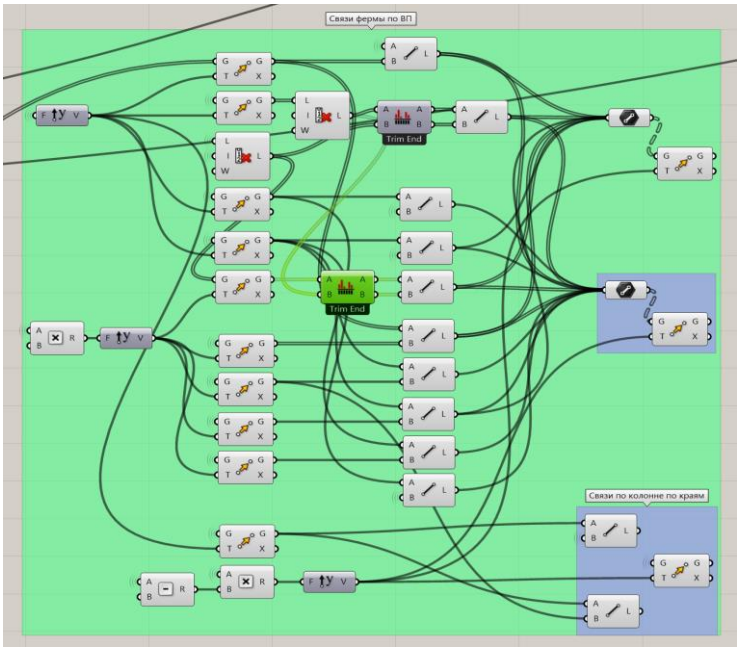

Рисунок 2.9 – Алгоритм построения геометрии горизонтальных связей фермы

Так же в указанном методе описано копирование сложившейся геометрии на другой конец фермы вдоль оси Y и последующее копирование, всей вышеупомянутой геометрии горизонтальных связей на нижний пояс фермы.

Итогом работы на данном этапе является левая часть фермы, состоящая из стоек с горизонтальными связями и распорками, а также стойками. Продемонстрируем вид получившейся фермы в окне ПО Rhinoceros 3D на рисунке 2.10.

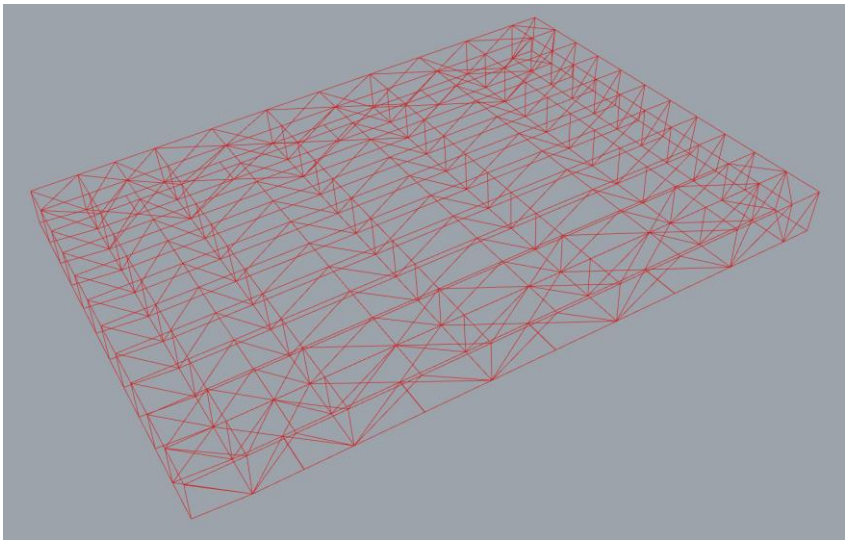

Рисунок 2.10 – Вид левой части фермы в окне ПО Rhinoceros 3D

Левая часть геометрии объекта имеет следующий вид (см. рисунок 2.11).

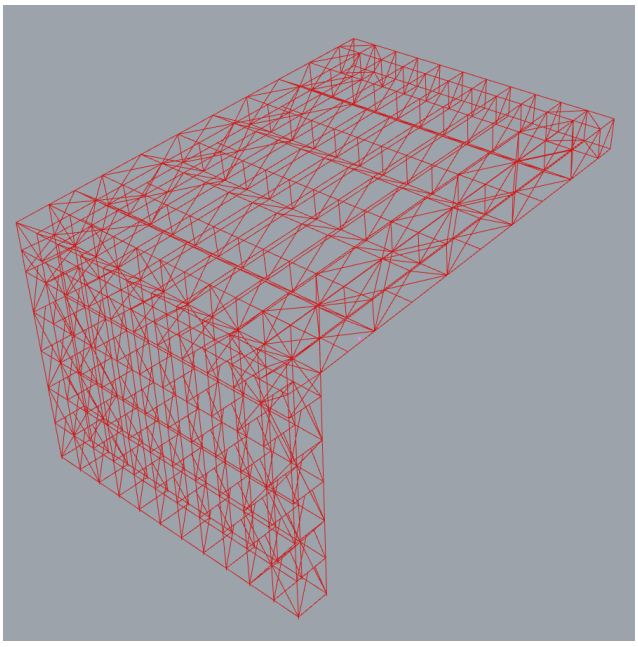

Рисунок 2.11 – Вид левой части строения в окне ПО Rhinoceros 3D

Так как задача состоит в создании алгоритма для всего объекта строительства, опишем создание правой части геометрии проектируемых конструкций.

Чтобы произвести отзеркаливание конструкций, необходимо задать плоскость, относительно которой будет произведено копирование существующей геометрии. Для этого задается плоскость вдоль осей YZ, находящаяся в коньковом узле фермы. Используя компонент «YZ Plane» и верхнюю точку ферм, получаем плоскость отзеркаливания геометрии. После чего, каждый вид конструкций по отдельности копируется относительно заданной плоскости, для этого используется компонент «Mirror». На входе у каждого такого компонента находится центральная плоскость отзеркаливания и геометрия определенного типа конструкций (см. рисунок 2.12).

Необходимость отзеркаливания каждого типа конструкций отдельно, заключается в том, что каждому типу присваивается свой профиль при визуализации в ПО Tekla Structures.

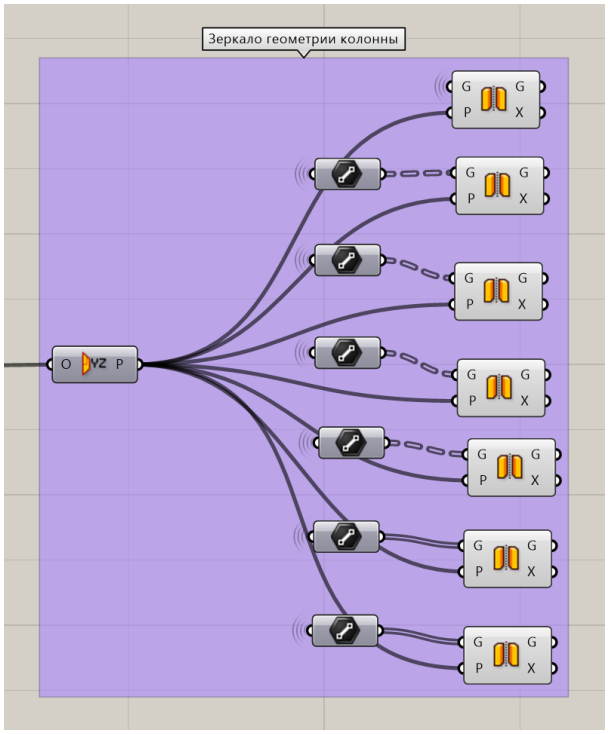

Рисунок 2.12 – Алгоритм построения правой половины геометрии строения

Итог алгоритма, описывающего построение основной геометрии с возможностью изменения основных геометрических параметров, проиллюстрирован на рисунке 2.13.

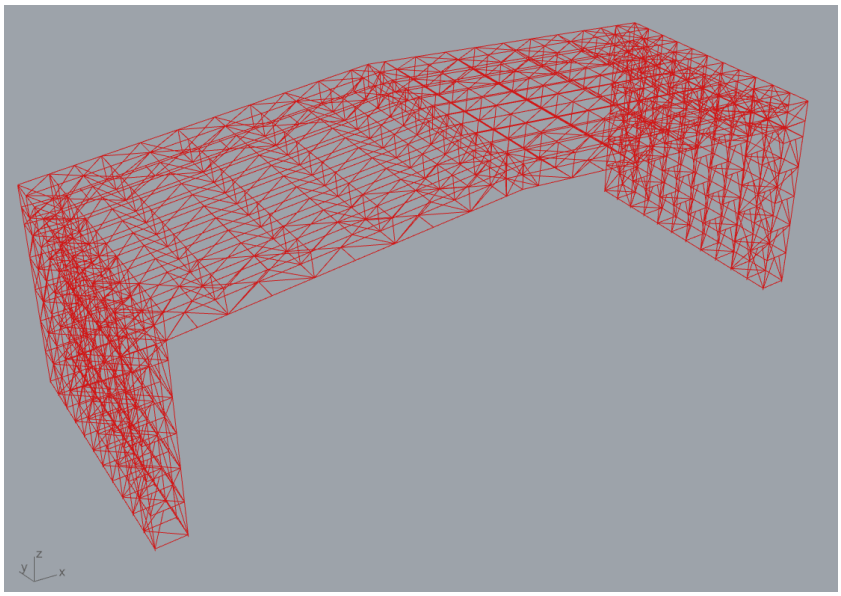

Рисунок 2.13 – Вид всей геометрии строения в окне ПО Rhinoceros 3D

Таким образом, имея основную геометрию объекта проектирования, можно перейти к этапу параметризации и присвоения элементам конструкций конкретных профилей.

3 МЕТОДИКА СОЗДАНИЯ АЛГОРИТМА ПАРАМЕТРИЗАЦИИ 3D-МОДЕЛИ КОНСТРУКЦИЙ И ИЗМЕНЕНИЯ ПРОФИЛЕЙ КАРКАСА

3.1 Описание алгоритма построения геометрии связей колонны

Параметризация — это метод моделирования, который использует параметры элементов модели и взаимосвязи между ними. Этот подход позволяет быстро изменять различные конструктивные схемы, корректируя параметры или геометрические соотношения, что помогает избежать ключевых ошибок в проектировании.

Параметрическое моделирование кардинально отличается от традиционных методов двумерного черчения и трёхмерного моделирования. При параметрическом проектировании создаётся математическая модель объектов с параметрами, изменение которых ведёт к изменению конфигурации детали или взаимного расположения деталей в сборке и т. д.

Геометрическая параметризация представляет собой тип параметрического моделирования, при котором геометрия каждого объекта пересчитывается в зависимости от положения родительских объектов, его параметров и переменных.

В рамках геометрической параметризации параметрическая модель включает в себя элементы построения и элементы изображения. Элементы построения (конструкторские линии) задают параметрические связи, а элементы изображения содержат линии, обводящие конструкторские линии, а также элементы оформления (размеры, надписи, штриховки и т. д.).

Некоторые элементы построения могут зависеть от других. Эти элементы также могут содержать параметры, например, радиус окружности или угол наклона прямой. При изменении одного элемента модели все

зависимые элементы автоматически перестраиваются в соответствии с заданными параметрами и их значениями [12].

Так как построение геометрии зависит от количества шагов стоек колонн вдоль пролета строения, опишем алгоритм параметризации геометрии в этом направлении. Необходимо это из-за того, что изменение количество шагов стоек вдоль пролета, приводит к изменению построения связей вдоль оси Y, т.е. если количество шагов четное, то строится одна геометрия связей, а если нечетное, то – немного другая.

Данное ухищрение обусловлено тем, что связи по колоннам должны быть продолжением связей по фермам, чтобы образовать единую пространственно-неизменяемую конструкцию, поэтому при определённой компоновке и входных параметрах, в связях по колонне появляется дублирование направления центральной связи.

Далее опишем вышеописанную задумку более подробно. Исходными данными является шаг колонн и его количество, для начала к заданному шагу прибавляется единица, а далее путем деления количества шагов на 2, компонент «Modulus» определяет остаток от деления. Таким образом, из-за того, что мы изначально прибавили единицу к исходному шагу, то если значение, выдаваемое компонентом «Modulus» равно 1 – значит шагов четное количество, а если значение рано 0, – то нечетное.

Подчеркнем, что для языка Grasshopper, как и для большинства языков программирования: единица – означает истинность выражения, а нулевое значение – ложность выражения [13]. Следовательно, используя «Cull Pattern» мы можем отсеять из алгоритма случай с нечетным количеством шагов, т.е. когда значение шагов будет нечетным, то оставшаяся цепочка алгоритма не будет отстраивать геометрию связей, а когда будет четным – будет. Поэтому также необходимо построить алгоритм для нечетного количества связей.

Через компонент «Boolean» числовые значения преобразовываются в логические: 1 – в «True», а 0 – в «False». После чего создается массив значений координат элементов с учетом заданных шагов и их количества, через «Series».

Используя комбинацию компонентов «Move», «Cull Index» и «Cull Nth» строится геометрия связей. Первая цепочка компонентов строит геометрия верхних связей от начала колонны к центру, вторая цепочка – те же связи от конца колонны к центру, третья – нижние связи от начала колонны к центру и четвертая цепочка – тоже нижние связи, но от конца колонны к ее центру.

Похожим образом, используя те же компоненты, но уже последовательно для всей колонны, необходимо создать цепочку компонентов, образующих алгоритм для нечетного количества шагов стоек. Результат алгоритма для четного количества шагов проиллюстрирован на рисунке 3.1.

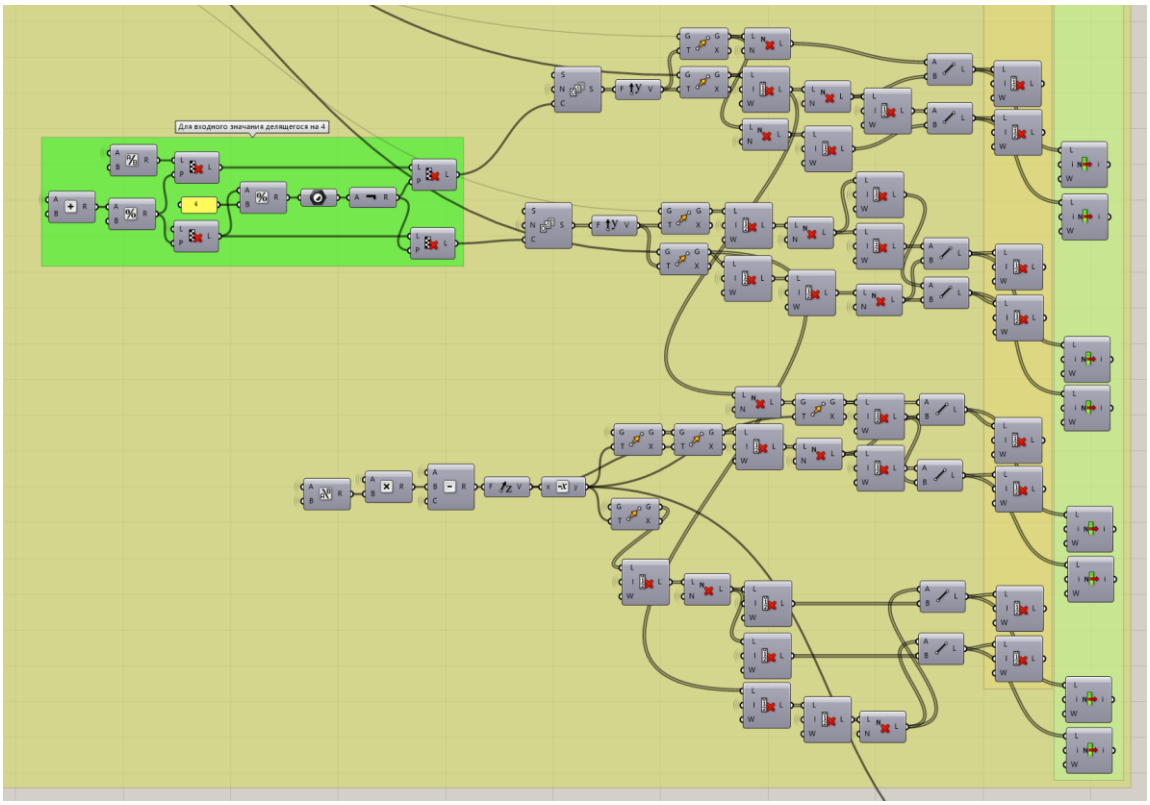

Рисунок 3.1 – Алгоритм параметризации вертикальных связей по колоннам

Горизонтальные связи по колонне строятся аналогичным образом, разница лишь в ориентации связей в пространстве. Для упрощения алгоритма была использована функция объединения нескольких компонентов в один «Cluster» (далее кластер). Кластеры можно использовать, когда необходимо визуально сократить алгоритм, состоящий из одних и тех же компонентов.

Алгоритм построения горизонтальных связей проиллюстрирован на рисунке 3.2.

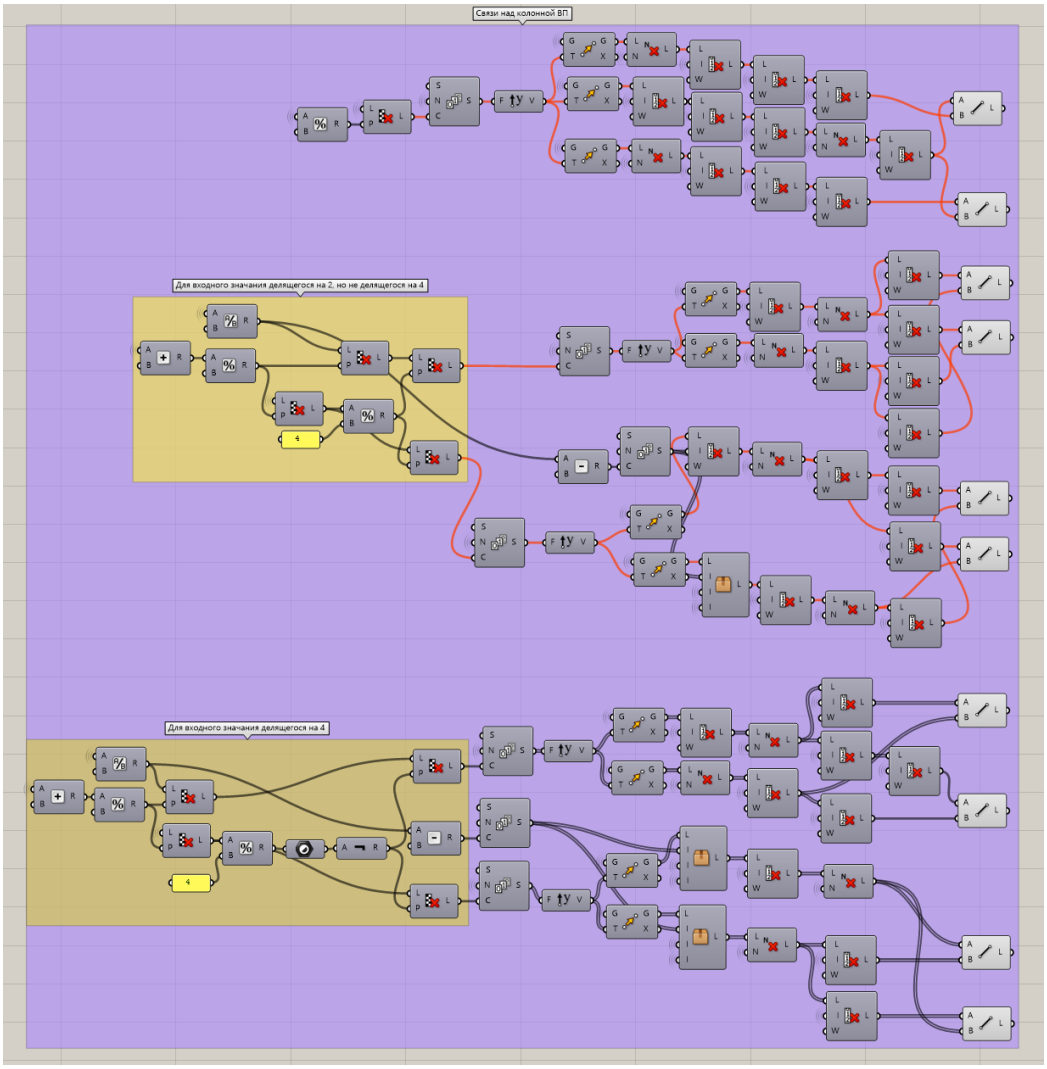

Рисунок 3.2 – Алгоритм параметризации горизонтальных связей по колоннам

Таким образом построение геометрии основного каркаса было описано алгоритмом, который учитывает изменение входных данных и «подстраивается» под них, перестраивая геометрию верным способом.

Так как геометрия каркаса ранее была описана только компонентами точек и прямых, а нашей задачей является ее построение в Tekla Structures, следует присвоить элементам геометрии конкретные профили.

3.2 Описание алгоритма присвоения и изменения профилей каркаса

Исходные данные для присвоения профилей каркаса были взяты из документации на конструкции укрытия ДК.

Автором было упомянуто ранее: чтобы связка Tekla Structures – Grasshopper работала корректно и все присвоения профилей, построение и их изменения, осуществляемые в Grasshopper, можно было проследить в Tekla Structures, для начала следует открыть модель в Tekla Structures, а уже потом окно Grasshopper.

Присвоение профиля элементу конструкции каркаса осуществляется компонентом «Beam», который, как и все компоненты для построений в Tekla Structures находится на вкладке «Tekla 2020» (вкладка соответствует версии Tekla). Чтобы профиль был присвоен конкретным элементам в первых вход компонента «Beam» необходимо определить линии геометрии, определяющие определенный вид конструкций. Второй вход определяет профиль, задаваемый данным конструкциям, который задается компонентом «Profile Catalog». Этот компонент подгружает настройки каталога профилей из открытой модели в Tekla Structures.

Далее, необходимо определить «Класс» детали. «Класс» — назначение категории. Для каждой категории определен свой цвет. Для назначения класса используются числа от 0 до 14, после значения 14 классы начинают повторяться [14]. Удобно определять класс по профилю: если ширина профиля – 80 мм, а толщина стенки – 4 мм, то класс можно задать значением 804. Данное значение необходимо поместить в соответствующих вход компонента «Part Attributes», который соединить с соответствующим входом с компонентов «Beam».

Итак, после присвоения «Класса», детали задается положение центральной линии и, по необходимости, поворот балки вокруг собственной оси компонентом «Position», который объединяется с одноименным входом в «Beam». Девять квадратов в данном компоненте определяют положение оси профиля вдоль поперечного сечения элемента, выбранное положение подсвечивается синим цветом. Поворот балки вокруг определенной ранее оси задается через вход «Rotation». Это значение определяется в градусах, поэтому, используя математические компоненты, необходимо определить

значение поворота. Например, чтобы определить угол наклона НП и ВП фермы к горизонту, для построения связей фермы, необходимо найти значение угла в радианах используя два вектора (вдоль НП фермы и вдоль горизонта) и компонент «Angle». Далее, зная значение угла в радианах, необходимо используя компоненты «Multiplication» и «Division» умножить полученное значение на 180 и разделить на число «Пи». Так определяется значение угла в градусах, полученному значению при необходимости можно задать отрицательное значение компонентом «Negative».

Компонент «Deforming» используется для изменения расстояния между концами элемента. Данная функция используется, чтобы не было конфликта между элементами, так как изначально вся геометрия пересекается между собой, следовательно, и профили тоже пересекаются.

Изменять профиль, «Класс», положение оси, поворот профиля элемента и положение концов вдоль элемента пользователь может в любой момент и все эти изменения в режиме реального времени будут отображаться в окне Tekla Structures.

Алгоритм присвоения и изменения профилей каркаса продемонстрирован на рисунке 3.3.

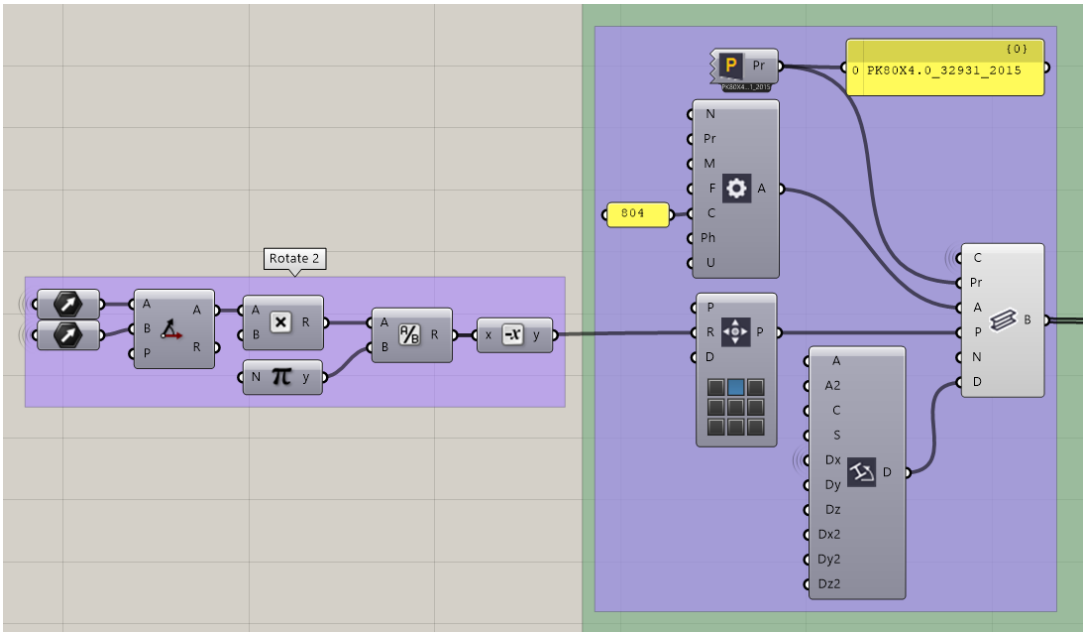

Рисунок 3.3 – Алгоритм присвоения и изменения профилей каркаса

Подобный алгоритм используется для всех элементов каркаса, но не для каждого из них необходим поворот вокруг собственной оси и изменение положения конечных точек.

Результат присвоения профилей каркаса продемонстрирован на рисунке 3.4.

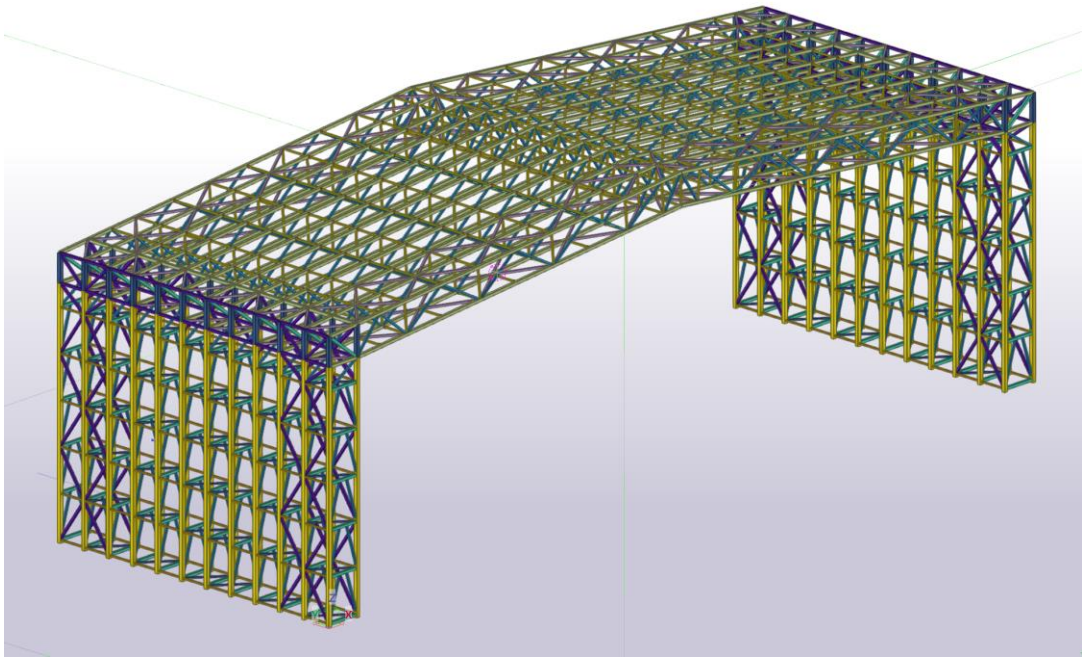

Рисунок 3.4 – Результат присвоения профилей каркаса через Grasshopper в окне ПО Tekla Structures

На данном этапе работы основная геометрия каркаса выстроена профилями в Tekla Structures, поэтому следующим шагом является разработка и детализация узлов конструкций строения.

4 РАЗРАБОТКА И ДЕТАЛИЗАЦИЯ УЗЛОВ МОДУЛЬНОГО УКРЫТИЯ С ПОМОЩЬЮ СВЯЗКИ TEKLA STRUCTURES – GRASSHOPPER

4.1 Описание алгоритма создания пользовательских компонентов в Tekla Structures

В ПО Tekla Structures уже разработана большая часть пользовательских стандартных компонентов, но они не настолько универсальны, чтобы подойти под все задачи проекта, поэтому нередко приходится собственноручно создавать компонент узла какого-либо соединения.

Чтобы перейти к разработке узлов, используя параметризируемые компоненты, разберемся, как создаются простые пользовательские компоненты в Tekla Structures.

Чтобы создать простой компонент, необходимо создать все элементы этого компонента в модели, а также заранее определить главную и второстепенную деталь компонента соединения. Главной деталью компонента соединения называется та деталь, к которой крепится второстепенная и наоборот: второстепенной деталью называется та деталь, которая крепится к главной.

После того как узел соединения есть в модели, из него можно сделать компонент, и использовать многократно на подобных соединениях деталей. Чтобы создать компонент необходимо открыть вкладку «Приложения и компоненты», далее «Доступ к расширенным функциям» и в выпадающем окне выбрать «Создать пользовательский компонент».

Далее в окне «Мастер пользовательских компонентов» необходимо выбрать тип компонента, но, чтобы это сделать необходимо понимать, что он из себя представляет. Так, например, компонент типа «Соединение» – объединяет две или более детали выбранными элементами компонента,

которые могут представлять собой балки, пластины, болты, сварные швы, срезы и т.п. Компонент типа «Стык» – используется, в основном, когда две детали необходимо соединить между собой, а линия, вдоль которой соединяются две детали, задается двумя точками в этом компоненте. Компонент типа «Узел» – создается на одной детали, т.е. для него нет второстепенных деталей, для него задается только главная деталь и точка расположения компонента. Самыми распространенными примерами такого узла могут являться ребра жесткости и заглушки в прокатных профилях [15].

На примере компонента типа «Соединение» разберемся, как создавать пользовательские компоненты. После выбора типа компонента, необходимо задать имя, иначе нельзя перейти к следующему шагу (см. рисунок 4.1).

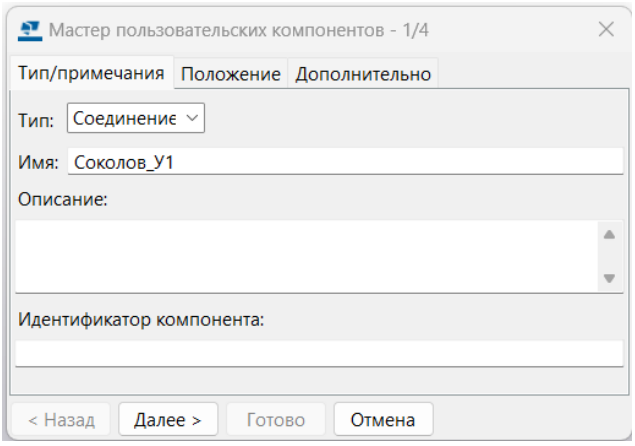

Рисунок 4.1 – Окно мастера пользовательских компонентов, первый шаг

После чего, нажав кнопку «Далее», появляется окно, требующее выделить все объекты, из которых состоит компонент (см. рисунок 4.2).

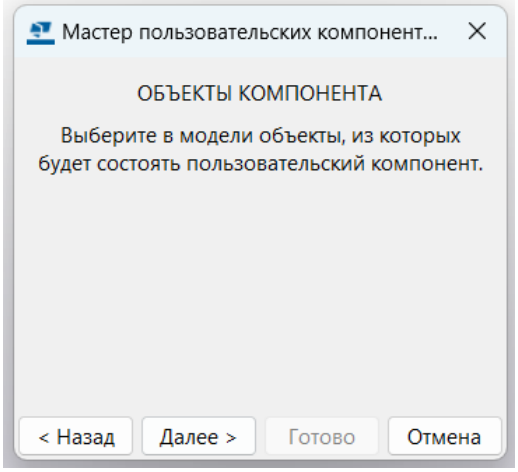

Рисунок 4.2 – Окно мастера пользовательских компонентов, второй шаг

Важно в этот момент понимать, что главную и второстепенную детали соединения нельзя выделять, они будут задаваться позже. После выбора всех элементов переходим к следующему шагу – выбор главной детали, далее – к выбору второстепенной детали, после чего, нажав кнопку «Готово» создается пользовательский компонент. Теперь компонент с заданным именем можно найти в галерее компонентов [16].

Однако данный компонент не учитывает изменения условий ввода, он применим к очень узкому кругу соединений, так как данный компонент разработан для определенных профилей и для определенного положения соединяемых элементов. Значит такой вид компонента не подошел бы под наши задачи. В нашей работе необходимы пользовательские параметризируемые компоненты способные «адаптироваться» под разные условия соединения.

4.2 Описание алгоритма разработки параметризируемых компонентов в Tekla Structures

В Tekla Structures параметризация компонентов производится в окне редактирования компонента, чтобы его вызвать необходимо найти его в галерее компонентов и нажатием правой кнопкой мыши выбрать пункт «Редактировать компонент…». После чего откроется новое рабочее окно с видом на компонент: его элемента, главную и второстепенную деталь.

Для работы с элементами компонента в маленьком всплывающем окне «Редактор пользовательских компонентов» необходимо вызвать «Обозреватель пользовательского компонента». В данном окне построчно отображаются все элементы, из которых состоит компонент, а их свойства определены иерархической структурой. Обозреватель дает возможность обратиться практически к любому свойству элемента: для фасок можно задать их вид и размер, для сварных швов – катет, притупление, обработку, стандарт и т.д.

Однако отметим, что эти свойства элементов компонента, можно задать переменными, которые сможет определять пользователь в любой момент работы с компонентом, а также их можно задать зависимостями свойств одних элементов от свойств других элементов и логическими или математическими формулами и алгоритмами. Для этого используется окно «Переменные», которое так же вызывается при помощи окна «Редактор пользовательских компонентов». В окне «Переменные» имеется таблица, в которой столбец «Имя» определяет наименование переменной, данный параметр необходим для идентификации формул или переменных. Параметр «Имя» используется в окне Обозревателя, чтобы задавать конкретные значения свойствам элементов.

Окно редактирования пользовательского компонента продемонстрировано на рисунке 4.3.

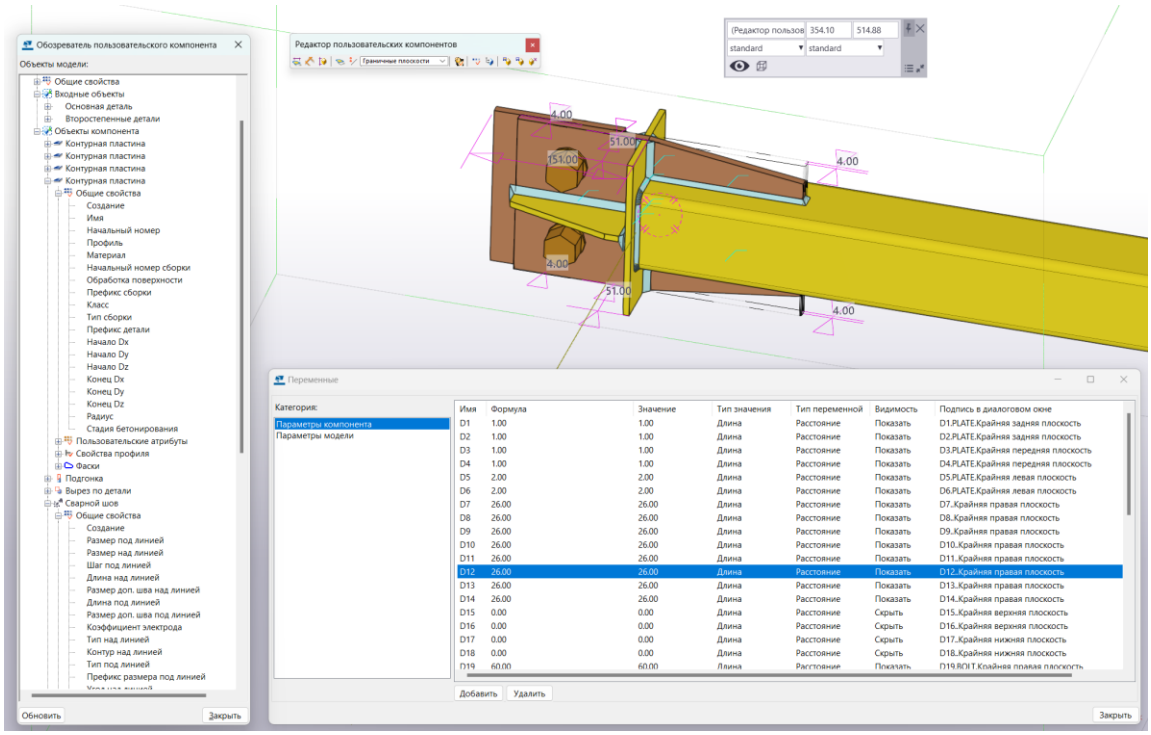

Рисунок 4.3 – Окно редактирования пользовательского компонента

Столбец «Формула» определяет способ задания переменной, если в него вписать какое-либо значение, то оно дублируется в столбце «Значение», если же ввести какую-либо формулу или логический алгоритм, то в столбце «Значения» будет выведен итог, полученный из соответствующего пункта,

столбца «Формула». Тип значения определяет, каким будет тип данных: численным, текстовым или «да/нет». «Тип переменной» определяет тип переменной.

С помощью столбцов «Видимость» и «Подпись в диалоговом окне», настраивается необходимость отображения переменных в диалоговом окне, а также их наименование в нем.

Окно «Переменные», заполненное для параметризируемого компонента, продемонстрировано на рисунке 4.4.

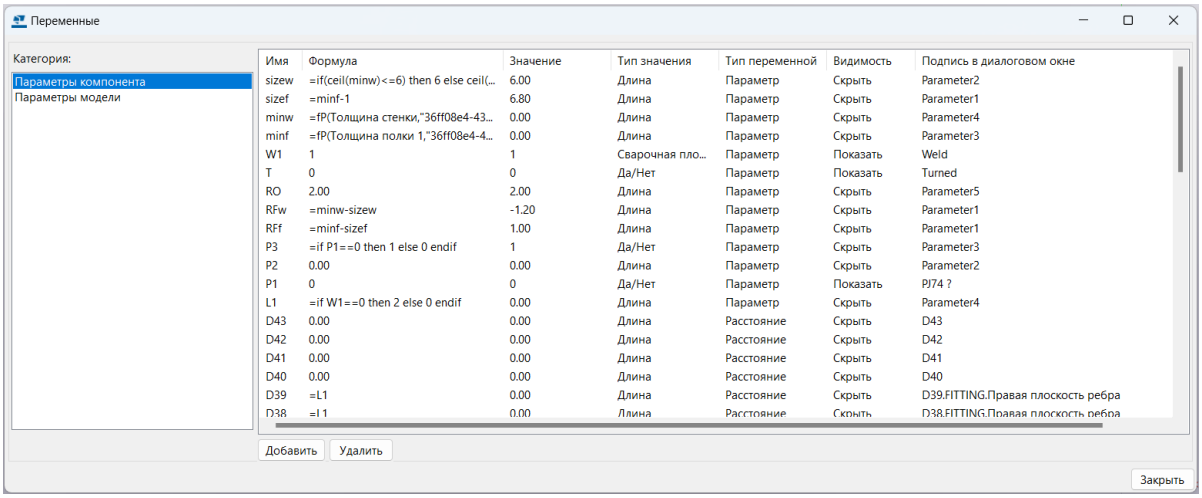

Рисунок 4.4 – Окно «Переменные» в редакторе пользовательского компонента

Таким образом, пользователь имеет возможность, настроить компонент таким образом, чтобы каждый раз, обращаясь к нему, он имел возможность ввести переменную, которая повлияет на вид узла. Например, можно ввести переменную, которая позволяет включать и убирать какой-либо элемент или элементы из узла компонента. Переменная так же может определять количественные данные: размеры профиля, количество болтов или элементов, расстояние между ними и т.п. Переменная может задаваться формулой или алгоритмом и быть скрыта от пользователя в момент обращения к диалоговому окну компонента во время моделирования. Помимо этого, компонент может состоять из других компонентов.

Во время работы с пользовательским компонентом важно знать какая деталь является главной, а какая второстепенной, иначе, при неверном

назначении деталей в модели, компонент установится неверным образом. Если с компонентом будет работать другой пользователь, то для него необходимо сделать картинку, определяющую последовательность назначения деталей и прикрепить ее в галерею компонентов.

Рассмотрим компонент крепления горизонтальной связи к поясу фермы: главной деталью является профиль связи, а второстепенной – приемная фасонка, созданная средствами Grasshopper и моделирование которой описано в следующем подразделе. Для начала определяется подрезка связи, т.е. ее укорачивание, данное значение привязывается к грани приемной фасонки и по необходимости может быть запараметризировано. После чего моделируется передающая усилие фасонка связи, ее ширина, толщина, длина и привязка основных точек вдоль оси фасонки задается инструментами, описанными выше. Прорезь для этой пластины связи моделируется вырезом по многоугольнику, в глубине прорези обязательно предусматриваются скругления, для снижения концентрации напряжений. Ширину прорези необходимо привязать к граням вставной пластины, прибавив по 1 мм с каждой стороны, для возможности создания сборки на заводе. Заглушающие пластины моделируются привязкой к граням связи, а ребро жесткости привязкой к вставной пластине по центральной оси связи. Таким образом моделируется основная геометрия узла.

Болты и сварные швы моделируются и привязываются похожим образом, каждый сварной шов, разбитый на линии, необходимо привязать в 3 плоскостях к конкретным местам стыка, а болтам задается положение, количество, ГОСТ, размер и расстояние между ними с возможностью редактирования, как и у катетов сварных швов.

После того, как все необходимые компоненты разработаны в Tekla Structures, можем перейти к описанию их расстановки средствами Grasshopper. Так как многие узлы крепления элементов в рассматриваемом объекте, моделируются через приемную пластину, опишем алгоритм её построения в Grasshopper.

4.3 Описание алгоритма построения элементов узла в Tekla Structures средствами Grasshopper

Чтобы построить приемную фасонку для установки узла связи, необходимо в Grasshopper найти 4 точки этой приемной фасонки, так как она моделируется, используя элемент «Пластина», которая строится по точкам, определяющим её контур. Но для этого, необходимо знать высоту пластины и её длину. Параметры длины и толщины фасонки – задаются пользователем, а её высота определяется высотой профиля пояса фермы.

Итак, чтобы определить высоту приемной пластины необходимо использовать компонент «Get Property» в который задается балка пояса фермы и параметр, определяющий необходимое свойство объекта, в нашем случае это высота, т.е. необходимо ввести имя параметра «HEIGHT». Так мы получили высоту профиля, но точки фасонки будут строится от центральной линии связи, поэтому полученное значение необходимо поделить на два, используя «Division».

Создадим возможность задавать длину фасонки, используя «Number Slider». Теперь, чтобы расставить точки приемных фасонок для каждой связи, соединяющейся с поясом фермы, необходимо преобразовать построенные связи обратно в линии, для этого используется компонент «Deconstruct Beam». Необходимо это для того, чтобы определить направление, копирования точек от конца связи к ее центру. Используя компонент «List Item», выделим одну прямую, определяющую направление связи. Далее с помощью «End Points» определяются ее конечные точки, а по ним задается направление вектора перемещения компонентом «Vector 2Pt». Так как профиль поясов фермы квадратный, то перемещение точек от центра пояса до грани профиля определяется направлением вектора и значением высоты, полученным ранее, с последующим использованием этих параметров в компоненте «Amplitude». Чтобы определить эти точки для всех связей, используется «End Points» для всех этих связей, а далее все точки конца (выход «End») смещаются на

толщину приемной фасонки, чтобы передающая пластина, которая врезается в связь могла устанавливаться по центру связи, а приемная фасонка имела смещение, равное ее толщине. После чего, возвращаясь к смещению точек вдоль связи, определим алгоритм перемещения точек.

Каждой концевой точке связей, ранее смещенной на толщину пластины, задается перемещение вдоль профиля связи на значение равное в первом случае значению высоты профиля, а во втором суммарному значению длины фасонки и высоты профиля. Чтобы найти 4 точки, определяющие контур пластины, необходимо, две точки, полученные ранее, скопировать вдоль направления стоек фермы.

Расстановка точек фасонки по высоте балки осуществляется с использованием «Vector 2Pt», входными данными для которого являются две конечные точки стойки. Далее перемещение вдоль этого вектора определяется через «Amplitude». И чтобы получить 4 точки контура пластин связей, необходимо лишь две точки, найденные ранее, скопировать по направлению стойки в обе стороны.

Чтобы построить приемную пластину в каждой связи необходимо скопировать полученные точки используя компонент «Mirror», так как геометрия всей фермы является зеркальной. После чего полученные точки задают контур фасонки в компоненте «Plate».

Алгоритм построения приемной фасонки представлен на рисунке 4.4.

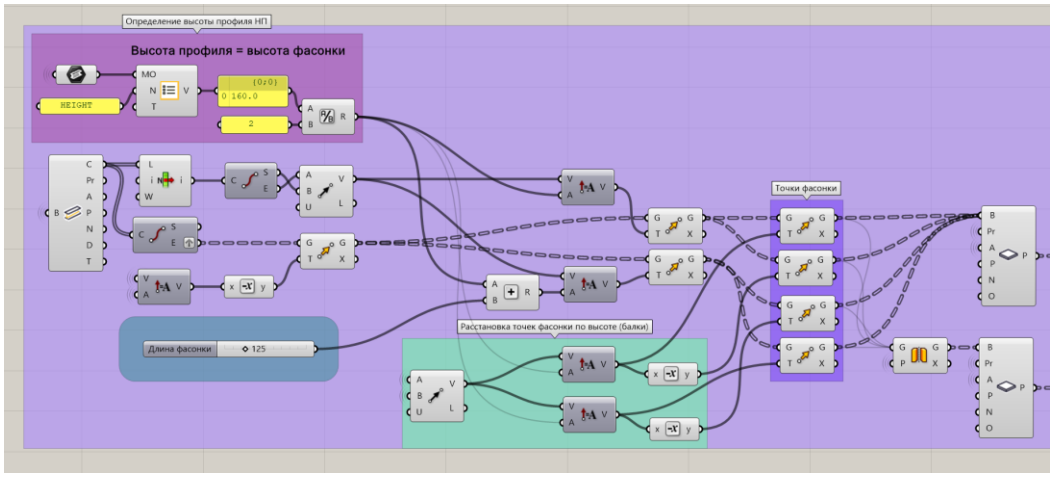

Рисунок 4.5 – Алгоритм построения приемной фасонки

Чтобы фасонка имела профиль, задаваемый пользователем и у нее, автоматически определялся Класс (цвет) в модели, необходимо создать этим значения возможность параметризации и определить их в ранее созданный компонент «Plate».

Профиль пластине задается компонентом «Concatenate», который объединяет части текстовых данных в одну строку, таким образом, в нашем случае толщина фасонки задается пользователем, используя «Number Slider», а начало текстового обозначения профиля описано в компоненте «Panel». На выходе получаем профиль и объединяем его с «Plate». Чтобы задать Класс (цвет) профиля в модели Tekla Structures через Grasshopper, используется компонент «Part Attributes», в его входной параметр «Class» задается толщина фасонки, а выход так же объединяется с «Plate».

Алгоритм задания профиля фасонке продемонстрирован на рисунке 4.6.

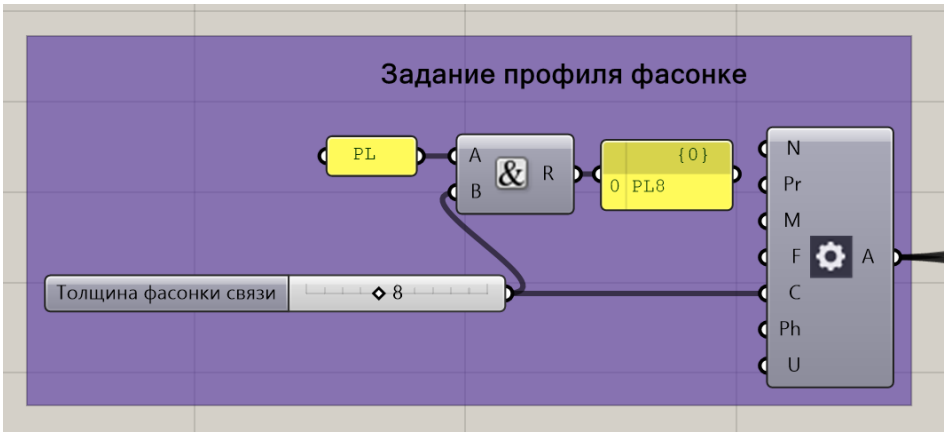

Рисунок 4.6 – Алгоритм задания профиля фасонке

На данном этапе пользователь имеет представление о том, как расставляются пластины приемных фасонок в Tekla Structures средствами Grasshopper. Заключительным этапом создания узлов соединения элементов является разработка алгоритма для расстановки компонентов.

4.4 Описание алгоритма расстановки компонентов Tekla Structures через Grasshopper

Алгоритм расстановки компонентов в Tekla Structures через Grasshopper применим к любым компонентам.

Этот алгоритм может использоваться для создания узлов между любыми элементами конструкций. Чтобы создать компонент узла между конструкциями связей и приемных фасонок можно использовать компонент Grasshopper «Component». Входными данными для которого, является название компонента из Tekla Structures и входные параметры, определяющие главные и второстепенные детали компонента. Чтобы определить имя компонента, используется «Component Catalog», который подгружает каталог компонентов из Tekla Structures, таким образом, кликнув на него, пользователь способен выбрать любой компонент, подходящий под узел крепления.

Входные параметры, определяющие главные и второстепенные детали компонента, задаются через «Plugin Input», в который первым подключаемым модулем является главная деталь, а во втором – второстепенные, если их несколько, но может быть и одна, как в нашем случае. Соединив все соответствующие связи и приемные фасонки в «Plugin Input», следует объединить полученные входные параметры в «Component». После чего, перейдя в окно Tekla Structures, пользователь может наблюдать, что все компоненты в соответствующих местах крепления расставлены алгоритмом.

Алгоритм расстановки компонентов Tekla Structures через Grasshopper продемонстрирован на рисунке 4.7.

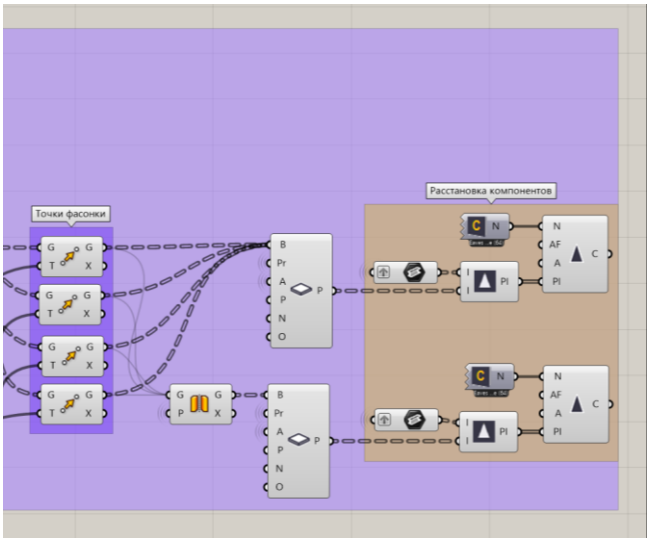

Рисунок 4.7 – Алгоритм расстановки компонентов Tekla Structures через Grasshopper

Результат алгоритма, описанного ранее, представлен в окне Tekla Structures на рисунке 4.8.

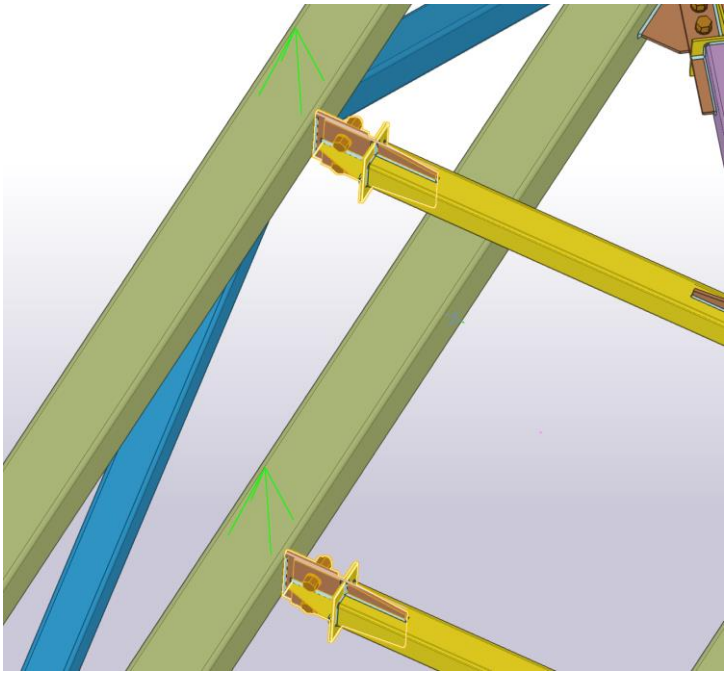

Рисунок 4.7 – Результат расстановки компонентов в Tekla Structures через Grasshopper

Чтобы построить оставшиеся основные узлы конструкций рассматриваемого строения, был использован описанный ранее метод создания и расстановки узлов конструкций.

ЗАКЛЮЧЕНИЕ

В начале процесса моделирования металлоконструкций инженер-конструктор сталкивается со сложностью построения геометрии элемента, которая должна соответствовать определенным требованиям, то есть – удовлетворять монтажным требованиям стыкующихся элементов, выполняя условия крепления под различными углами, а также другим пространственным условиям, например, расположение основных точек элемента в одной плоскости. Эти и многие другие условия повышают трудозатраты на выполнение основных задач инженера-конструктора.

Разработанный автором в рамках данной работы метод упрощает задачу моделирования основной геометрии строения из металлоконструкций. Итогом работы является алгоритм в Grasshopper 3D, отстраивающий основную геометрию строения из металлоконструкций и позволяющий изменять основные геометрические параметры объекта строительства, а также сечения элементов (см. Приложение А).

Помимо этого, в описанном автором алгоритме присутствует функция моделирования узлов соединения конструкций, расставляющая компоненты узлов в Tekla Structures через Grasshopper.

Достижение поставленной цели было реализовано автором данной работы, путем выполнения поставленных задач, а именно: было изучено ПО Grasshopper и его возможности в связке с Tekla Structures, определена методика создания каркаса металлоконструкций в Tekla Structures, используя только Grasshopper, разработан алгоритм построения и параметризации геометрии каркаса металлоконструкций в Grasshopper, разработаны и детализированы узлы конструкций модульного укрытия с помощью связки Tekla Structures – Grasshopper, создана возможность изменения профилей каркаса здания через Grasshopper.

По завершению работы автором были достигнуты предполагаемые ранее результаты: создан программный код в Grasshopper 3D, работающий совместно с Tekla Structures, отстраивающий основную геометрию строения из металлоконструкций и позволяющий изменять основные геометрические параметры объекта строительства и отстраивающий узлы крепления элементов металлических конструкций, сокращены трудозатраты, касающихся процесса моделирования типовых объектов, на 70% (см. Приложение Б).

Автором было изучено, что на выполнение данной работы по моделированию в среднем необходимо затратить 160 часов, данное значение получено методом хронометража рабочего времени инженера-конструктора. После внедрения разработанного алгоритма это время составит около 50 часов.

Автоматизация и параметризация построения основной геометрии, а также узлов конструкций позволит ускорить процесс создания моделей, повысить их качество и улучшить производительность труда.

Автор имеет публикацию в журнале ВАК «Природные и техногенные риски. Безопасность сооружений» и сертификат участия за выступление во Всероссийской студенческой научно-практической конференции «IV Ярмарка ТИМ талантов 2023» в рамках X Международного форума и выставки 100+ TechnoBuild.

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1. СП 333.1325800.2020. Информационное моделирование в строительстве. Правила формирования информационной модели объектов на различных стадиях жизненного цикла. Утвержден приказом Министерства строительства и жилищно-коммунального хозяйства Российской Федерации от 31 декабря 2020 г. N 928/пр и введен в действие с 1 июля 2021 г. 226 с.
2. Автоматизация / [Электронный ресурс] // Знание.Вики: [сайт]. – URL: https://znanierussia.ru/articles/Автоматизация#:~:text=Автоматизация%20—%20направление%20научно-технического%20прогресса%2C,участия%20или%20трудоемкости%20выполняемых%20задач (дата обращения: 05.05.2024).
3. Создать полилинию и плоскую область / [Электронный ресурс] // К3-Редактор: [сайт]. – URL: https://wiki.k3info.ru /index.php/Создать_полилинию_и_плоскую_область (дата обращения: 05.05.2024).
4. Что такое массив и как он устроен / [Электронный ресурс] // Skillbox: [сайт]. – URL: https://skillbox.ru/media/code/chto-takoe-massiv-i-kak-on-ustroen/ (дата обращения: 05.05.2024).
5. Уськов, В. В. Компьютерные технологии в подготовке и управлении строительных объектов. / учебное пособие. – М.: Инфра-Инженерия, 2013. – 320 с.Кр

6. Grasshopper 3D / [Электронный ресурс] // Альфапедия: [сайт]. — URL: https://alphapedia.ru/w/Grasshopper_3D (дата обращения: 05.05.2024).
7. Tekla Structures / [Электронный ресурс] // Wikipedia: [сайт]. – URL: https://ru.wikipedia.org/wiki/Tekla_Structures (дата обращения: 05.05.2024).
8. Импорт и экспорт в Tekla Structures / [Электронный ресурс] // Tekla User Assistance: [сайт]. – URL: https://support.tekla.com/ru/doc/tekla-structures/2022/int_importing_to_and_exporting_from_ts (дата обращения: 05.05.2024).
9. Обзор программы Grasshopper / [Электронный ресурс] // Junior: [сайт]. — URL: https://junior3d.ru/article/grasshopper.html (дата обращения: 09.05.2024).
10. Grasshopper-Tekla Live Link / [Электронный ресурс] // Tekla User Assistance: [сайт]. — URL: https://support.tekla.com/help/tekla-structures/not-version-specific/grasshopperteklalink (дата обращения: 09.05.2024).
11. Евгений Лещенко Параметрическое проектирование и высокотехнологичное информационное моделирование строительных конструкций на основе программного решения Tekla и Grasshopper [Текст] / Евгений Лещенко // САПР и Графика. — 2017. — № 8. — С. 25-28.3
12. Параметрическое моделирование / [Электронный ресурс] // Wikipedia: [сайт]. – URL: https://ru.wikipedia.org/wiki/ Параметрическое моделирование (дата обращения: 17.05.2024).
13. Истинные и ложные значения в Python: подробное введение / [Электронный ресурс] // FreeCodeCamp: [сайт]. – URL: https://www.freecodecamp.org/news/truthy-and-falsy-values-in-python/ (дата обращения: 17.05.2024).

14. Свойства колонн в Tekla Structures / [Электронный ресурс] // Сайт инженера-проектировщика: [сайт]. – URL: https://saitinpro.ru/tekla-structures/tekla-structures-uroki/metallicheskie-konstruktsii-v-tekla-structures/svojstva-kolonn-v-tekla-structures/ (дата обращения: 17.05.2024).
15. Компоненты / [Электронный ресурс] // Tekla User Assistance: [сайт]. – URL: https://support.tekla.com/ru/doc/tekla-structures/2024/det_getting_started_overview (дата обращения: 21.05.2024).
16. Создание компонентов пользователя в Tekla Structures / [Электронный ресурс] // Сайт инженера-проектировщика: [сайт]. – URL: https://saitinpro.ru/tekla-structures/tekla-structures-uroki/komponenti-v-tekla-structures/sozdanie-komponentov-polzovatelya-v-tekla-structures/ (дата обращения: 21.05.2024).

ПРИЛОЖЕНИЕ А
(справочное)

Алгоритм автоматизации процесса моделирования

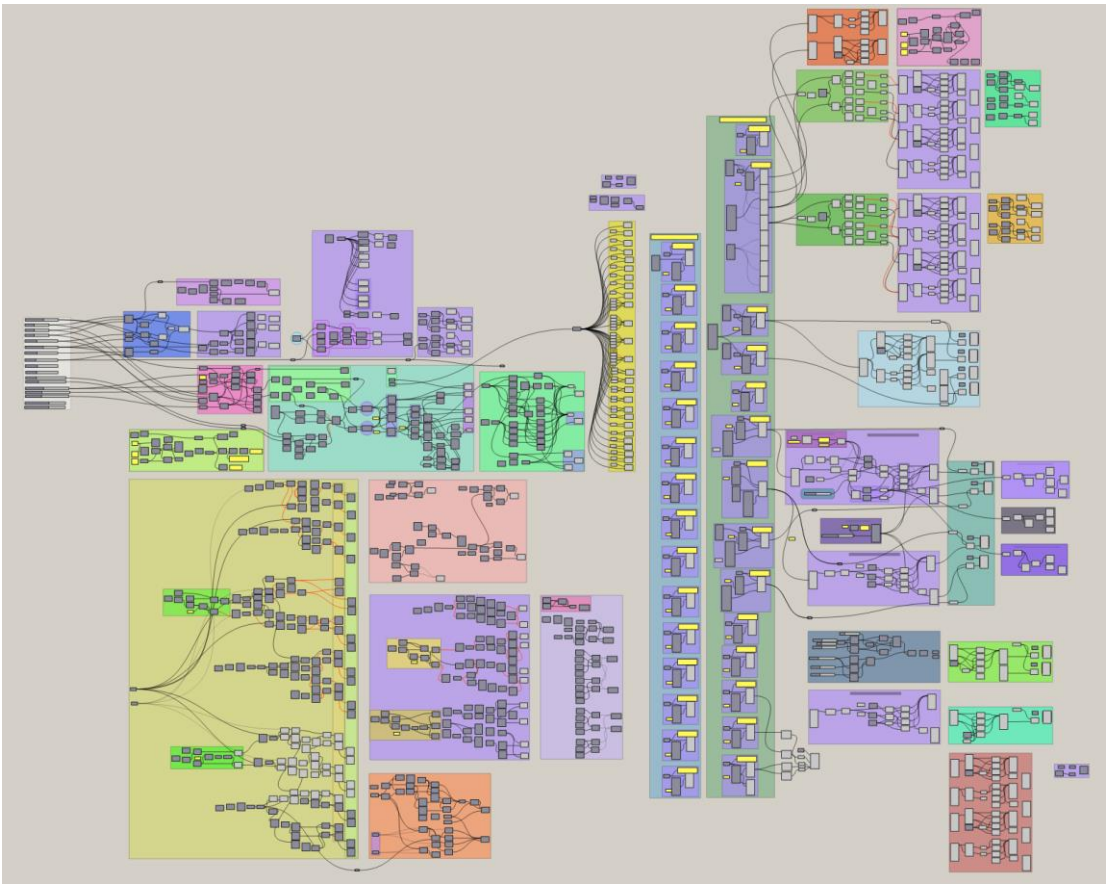

Рисунок А.1 – Алгоритм автоматизации процесса моделирования

ПРИЛОЖЕНИЕ Б
(справочное)

Итог работы алгоритма автоматизации процесса моделирования в окне ПО Tekla Structures

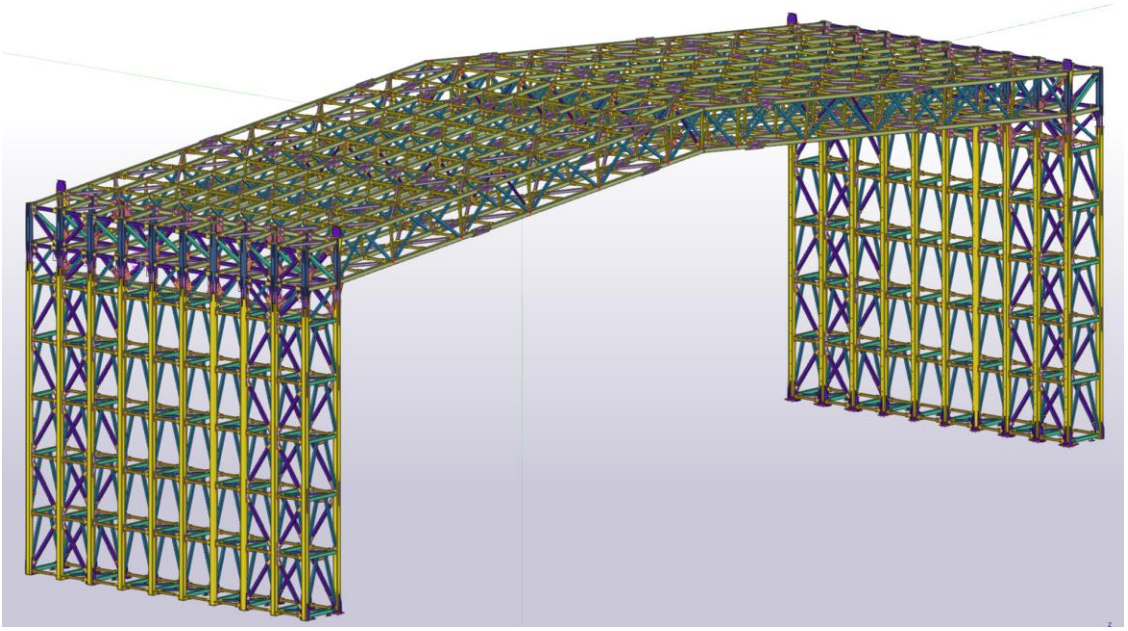

Рисунок Б.1 – Общий вид модели строения в окне ПО Tekla Structures

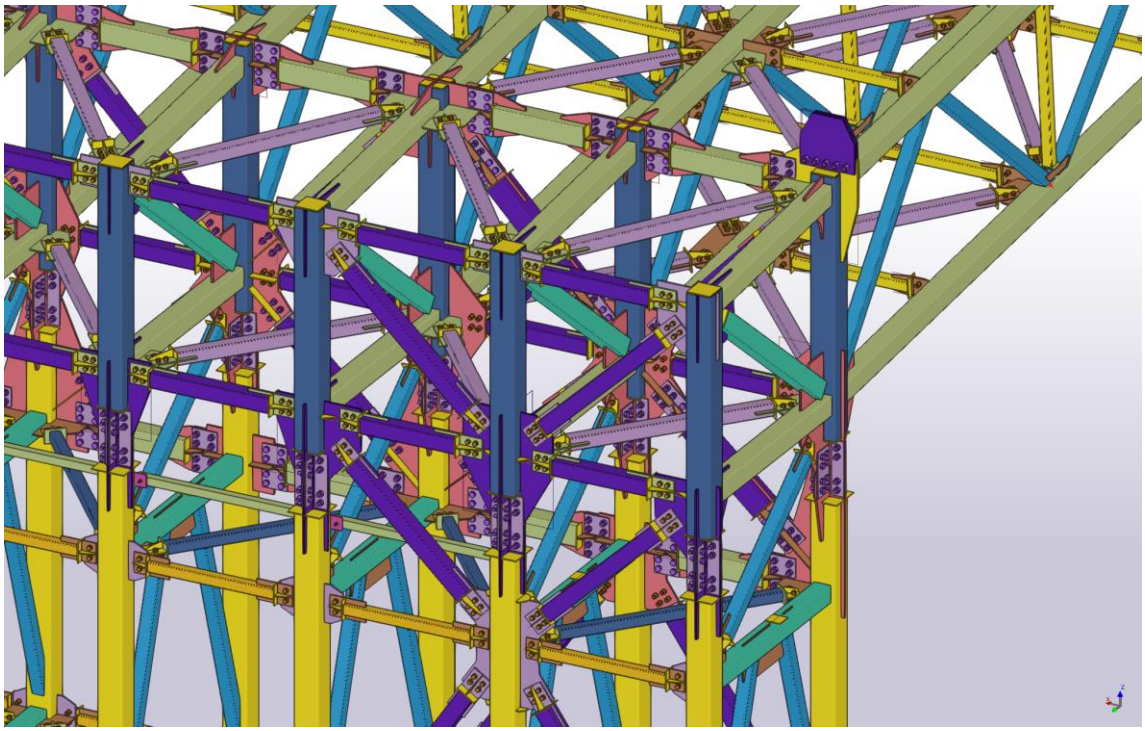

Рисунок Б.2 – Детализированный вид модели строения в окне ПО Tekla Structures