МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РОССИЙСКОЙ ФЕДЕРАЦИИ

Государственное образовательное учреждение высшего профессионального образования «Нижегородский государственный университет им. Н.И. Лобачевского»

Институт информационных технологий, математики и механики

Центр прикладных и информационных технологий  
Специальность «Прикладная информатика»  
Профиль прикладная информатика в информационных системах

Выпускная квалификационная работа

«Синтез 3D объектов по набору (их) 2D проекций»

Выполнил: студент группы 381507-3

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_Каткова Д.А.

Научный руководитель:

к.т.н. \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_Васин Д.Ю.

Нижний Новгород

2019

Общие замечания по тексту

Название сапров везде одинаковое и т.д

Kompas-3D

Оглавление

[Введение 3](#_Toc510727841)

[1. Теоретическая часть 13](#_Toc510727843)

[1.1. Обзор существующих методов решения 13](#_Toc510727844)

[1.1.1. Метод отображения границ 13](#_Toc510727844)

[1.1.2. Метод конструктивной сплошной геометрии 13](#_Toc510727844)

[1.1.3. Восстановление сплошныз тел по шести проекциям 13](#_Toc510727844)

[1.2. Технологические этапы формирования 3D объекта по набору их 2D проекций 13](#_Toc510727844)

[1.2.1. Извлечение 2d ортографической информации 13](#_Toc510727844)

[1.2.2. Разметка 2d граней и вершин 13](#_Toc510727844)

[1.2.3. Объединение граней и вершин 13](#_Toc510727844)

[1.2.4. Нахождение точек пересечения ребер 13](#_Toc510727844)

[1.2.4.1 Определение нахождения ребер на одной прямой 13](#_Toc510727844)

[1.2.4.2 Определение пересечения ребер 13](#_Toc510727844)

[1.2.5. Создание не ортогональных ребер 13](#_Toc510727844)

[1.2.6. Удаление лишних ребер 13](#_Toc510727844)

[1.3. Математическая модель 13](#_Toc510727844)

[1.4. Постановка задач 13](#_Toc510727844)

[2. Практическая часть 13](#_Toc510727843)

[2.1. Описание разработанных и используемых алгоритмов 13](#_Toc510727844)

[2.1.1. Алгоритм 13](#_Toc510727844)

[3. Информационное обеспечение системы 13](#_Toc510727843)

[3.1. Формат CDR 13](#_Toc510727844)

[3.2. Формат AI 13](#_Toc510727844)

[3.3. Формат DXF 13](#_Toc510727844)

[4. Условие разработки и функционирования ПО 13](#_Toc510727843)

[5. Детальное описание разрабатываемого ПО 13](#_Toc510727843)

[5.1. Назначение 13](#_Toc510727844)

[5.2. Входные/выходные данные 20](#_Toc510727844)

[5.3. Коды завершения 21](#_Toc510727845)

[5.4. Руководство оператора 20](#_Toc510727844)

[5.5. Действия оператора при сбое 20](#_Toc510727844)

[5.6. Результаты 21](#_Toc510727845)

[Заключение 27](#_Toc510727846)

[Список литературы 28](#_Toc510727853)

[Приложение 28](#_Toc510727853)

**Введение**

В настоящее время предметом активных исследований и развитий является проблема трёхмерного моделирования. Задача синтеза 3D объектов по набору (их) 2D проекций является одной из наиболее актуальных, наиболее сложных и наукоемких задач компьютерной графики. Для того чтобы построить объект, создается его чертеж. Чертежи распространяются в различных форматов файлов: это могут быть как отсканированные изображения, так и файлы программ компьютерного проектирования САПР или CAD. Для получения более широкого представления о том, что получится в итоге, создается 3D модель, даже если этот процесс занимает значительное количество времени. С появлением цифровых технологий проектирования задача упростилась, поскольку появилась возможность автоматизировать данный процесс. С этой целью были разработаны системы автоматизированного проектирования, в которых уже заложены готовые решения, находящиеся в библиотеке и существует возможность проверить и испытать готовый продукт с помощью математического моделирования.

На сегодняшний момент существует большое количество программ, восстанавливающих 3-мерную модель по проекциям, например, PTC CREO, однако каждая из них имеет ряд существенных недостатков(особенностей) – наиболее распространённым является неполная автоматизация восстановления. Универсальной системы, позволяющей без участия пользователя восстанавливать трёхмерную модель по проекциям для фигур любой степени сложности, на данный момент не существует. Возможность автоматического получения двухмерных проекций на основании трехмерной модели имеют практически все системы трехмерного моделирования, однако наличием обратного процесса, т.е. получения трехмерной параметрической модели на основании обычного двухмерного чертежа, характеризуются не все системы. В частности существуют системы автоматического перевода 2D в 3D, такие, как, например, система Xpanel 3D, или система фирмы Autodesk. Однако в этих системах отмечаются существенные отклонения сформированной трехмерной модели от исходных ортогональных проекций. Автоматизация построения чертежа в трехмерную модель очень актуальнаво многих отраслях науки и техники, а так же в сферах бизнеса и индустрии развлечений. Также решение данной проблемы является важным аспектом безопасности и контроля. Современные информационные разработки позволяют существенно облегчить доступ к необходимым системам машинной графики, графических редакторов и пакетов разных графических прикладных программ, позволяющих успешно использовать средства машинной графики практически во всех сферах человеческой деятельности.

Однако, долгое время, методология моделирования развивалась независимо отдельными науками. Отсутствовала единая система понятий и терминологии. Сейчас, с использование мощных процессоров и графических ускорителей трехмерная графика стала доступной реальностью для персональных компьютеров. Но в тоже время производители программного обеспечения столкнулись с проблемой отсутствия каких-либо стандартов, которые бы позволили писать программы, независимые от операционной системы и оборудования. Одним из первых таких стандартов, существующий до сих пор, является OpenGL. OpenGL - это графический стандарт в области компьютерной графики. Создание каждого нового приложения в данной области по-прежнему остается **актуальной** задачей и требует дополнительных исследований в связи со специфическими требованиями по разрешению, быстродействию, надежности распознавания и объему памяти. Методы компьютерного моделирования используются специалисты в области истории, математики, физике, медицины, машиностроении, строительстве, а так же в играх, фильмах и др. С помощью компьютерной графики можно спрогнозировать и имитировать явления, события или проектируемы предметы в заранее заданных параметрах.

Компьютерная графика позволяет осуществлять конструкторские разработки в двух направлениях. Первое направление базируется на двухмерной геометрической модели и использовании компьютера как особого средства, позволяющего значительно ускорять процесс конструирования и улучшать качество оформления конструкторских документов. Центральное место в этом подходе к конструированию занимает чертеж, который содержит всю необходимую графическую информацию для изготовления какого-либо изделия. В основе второго направления лежит пространственная геометрическая модель изделия, которая является более наглядным способом представления оригинала и более мощным и удобным инструментом решения геометрических задач. Чертеж в этих условиях играет вспомогательную роль, а способы его создания основаны на методах компьютерной графики. При использовании первого направления (традиционный процесс конструирования) обмен информацией осуществляется на основе конструкторской, нормативно-справочной и технологической документации; при использовании второго – на основе компьютерного представления геометрического объекта общей базы данных, что способствует эффективному функционированию программного обеспечения. Большинство разработанных систем использует восходящий подход моделирования, который начинается с идентификации основных 2D точек для последовательного восстановления 3D точек, линий и поверхностей с целью создания окончательной 3D модели. Хотя ортогональные проекции являются однозначными, синтезирующие процедуры, конструирующие трехмерные объекты, могут производить ложные 3D элементы, которые должны быть удалены для получения требуемой 3D модели.

Далее сформулируем цели, задачи и объект исследования в рамках данной работы.

**Цель** **исследования** заключается в разработке и реализации математического обеспечения решения задачи синтеза 3D объектов по набору (их) 2D проекций.

Для решения данной цели необходимо решить следующий **ряд задач**:

- изучить существующие форматы представления инженерных чертежей;

- изучить принципы и технологии моделирования двухмерных и трехмерных графических объектов;

- изучить принципы работы библиотеки OpenGL;

- рассмотреть основные подходы в восстановлении 3D объектов по 2D чертежам проекции;

- изучить существующие алгоритмы восстановления трехмерной модели по трем двухмерным ортогональным проекциям;

- разработать информационную систему, принимающую на вход из чертежа прямоугольное проецирование на три плоскости проекций: фронтальная, горизонтальная, профильная (прямоугольную изометрическую проекцию) и восстанавливающую корректную каркасную 3D модель заданного объекта на выходе.

**Объект исследования** - набор 2D чертежей, изначально представленных либо в бумажном представлении, либо в электронном виде, как результат выполнения проективных работ в системах САПР типа Kompas-3D, AutoCAD и др.

1. **Теоретическая часть**
   1. **Обзор существующих методов решения**

Важное место в современном документов обороте занимают графические документы со слабо формализованным описанием объектов (ГДСФ) [2]. К ним относятся: топографические карты и планы, различные чертежи, схемы и пр.(Рис.1).

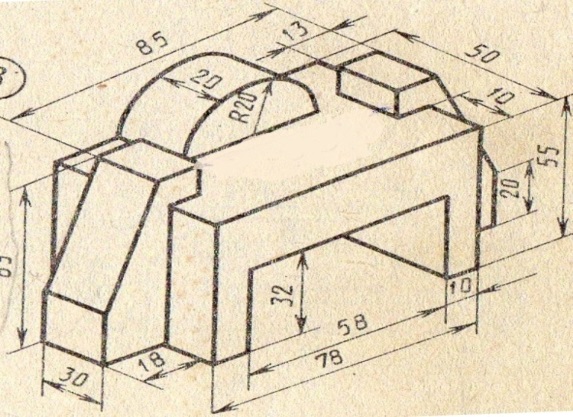
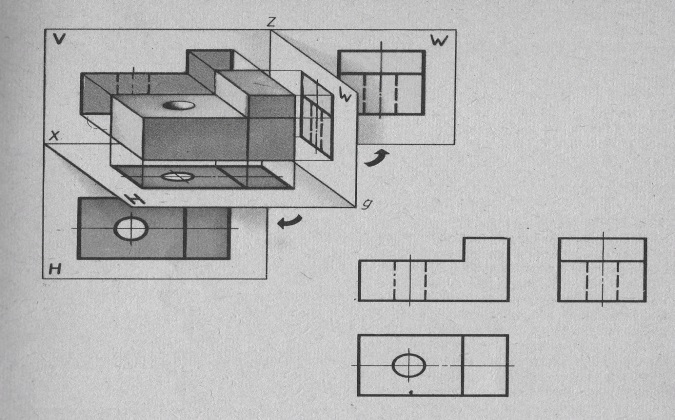
 

Рисунок 1. Примеры исходного графического документа

Важными особенностями данных документов являются:

• документы массового типа;

• документы исполнены на бумажных носителях среднего и плохого качества;

• на планах и схемах отсутствует жесткий стандарт и часто не соблюдены правила отрисовки отдельных графических знаков;

• в зависимости от качества носителя и правильности выбора параметров оцифровки, степень зашумленности изображения графического документа может варьироваться в широких пределах;

• метрическая точность графических знаков также варьируется широким диапазоном значений.

Характерные помехи, присутствующие на данном классе документов:

• слабо формализованные начертания объектов;

• произвольная ориентация;

• произвольные размеры символов;

• наличие большого количества меж объектных наложений;

• примыкания и пересечения.

Все это определяет набор требований к методам и алгоритмам обработки указанного класса документов. Перечислим основные из них:

• технологичность;

• согласованная с объектным наполнением графических данных их модельная структурированность;

• высокая емкостная и временная эффективность;

• естественная интегрируемость в общую схему информационного описания предметной области.

Как правило, в распоряжении пользователя, имеется только неполная геометрическая информация о таких объектах как, машиностроительный чертеж (бумажный носитель или эскиз с ортогональными проекциями) или электронный чертеж. Поэтому, необходим переход от неполной информации к трехмерному описанию всей сцены. Для этого необходимо решение следующих задач:

- распознавание геометрических примитивов чертежа;

- воссоздание векторного представления проекций;

- нахождение координат точек поверхности трехмерного объекта.

В работах Д.В. Никулина [] было проведено исследование повышения эффективности конструкторских САПР путем использования системы распознавания машиностроительных чертежей. Проблема заключалась в переводе машиностроительных чертежей с бумажного носителя в машинное представление, с возможностью дальнейшего использования и преобразования полученного результата на разных этапах автоматического проектирования. Был предложен эффективный метод восстановления информации на машиностроительных чертежах. Метод позволял не только восстановить информацию, но и улучшить качество восстановленного изображения.

В работе Ю. В. Литовка, В. В. Пэк, М. А. Попова, А. С. Попов - «Алгоритм формирования объемной геометрической модели детали из чертежа проекций», рассмотрен САПР гальванических процессов, важным этапом которого является ввод данных о геометрических параметрах обрабатываемой детали. Цель данной работы является, получение пространственных координат точек поверхности детали из их двумерного изображения на чертеже. Для решения задачи предложен алгоритм, включающий в себя следующие этапы: преобразование набора координат вершин в список отрезков с прослеживанием связности отрезков между собой; определение проекций; построение прямоугольников, описывающих каждый из трёх объектов; нахождение начала координат трёх проекций для использования этой точки в дальнейшем при склейке трёхмерной модели; построение трёхмерной модели на основе полученной двумерной. Для ускорения вычислений предложено распараллеливать вычисления.

Работы М.А. Гвирца [] посвящены разработке геометрических моделей формирования поверхностей деталей сложной формы. Целью создания таких систем является получение и обработка информации для контроля качества измеряемых деталей на предмет соответствия измеряемых поверхностей их теоретическим моделям. Особенно остро эта задача стоит применительно к изделиям, содержащим поверхности сложной формы. Такие поверхности обычно имеют место в изделиях базовых отраслей промышленности, одной из которых является авиадвигателестроение. Одной из задач, решаемых при этом, является построение методики измерения деталей сложной формы, обработки и анализа результатов измерения. Повышение точности восстановления поверхностей позволяет провести оптимизацию величины и распределения припуска по обрабатываемым поверхностям. Практическая значимость результатов данной работы, базирующихся на использовании сотовой сетки, заключается в том, что разработанные метод, методики и алгоритмы могут быть использованы при проведении обмеров и восстановления поверхностей деталей сложной формы с повышенной точностью. Это позволит снизить припуски, а вместе с ними массу и материалоемкость сложных деталей. Кроме того, открывается возможность принимать решения о годности заготовок в случаях, когда ранее это было невозможно ввиду влияния погрешности восстановления их поверхностей.

Среди зарубежных работ стоит отметить труды М. Ласло (США) [] о структурах геометрических данных и алгоритмах их обработки. Восстановлением объемного представления по эскизам ортогональных проекций занимался Дж. Гасслер (UK, Coventry University).

Наибольший прорыв в области восстановления 3D моделей совершили Wesley и Markowsky в своих работах [8, 11], предложив использовать каркасную модель как промежуточный шаг процесса восстановления. На сегодняшний день они являются одними из самых известных исследователей в этой области, а их подход используется при разработке практически любого нового алгоритма восстановления 3D моделей. Однако ни в одной работе не приведены методы, решающие все вышеперечисленные задачи в комплексе. Если использовать несколько различных продуктов, то возможно создать систему, удовлетворяющую требованиям и обрабатывающую все предлагаемые типы неполных исходных данных. Однако это потребует доработки программ и добавления некоторых функций, что представляет собой нетривиальную задачу ввиду несопоставимости форматов пакетов. Кроме того, комплекс программных продуктов имеет высокую стоимость. Поэтому решение данной проблемы по-прежнему остается актуальной.

Процесс восстановления, который включает в себя взаимодействие пользователя с системой, чаще всего занимает значительное количество времени, вследствие чего разрабатываются различные алгоритмы для автоматического восстановления 3D моделей. Эти алгоритмы могут быть разделены на два типа в зависимости от существующих подходов. На данных момент существует два основных похода в восстановление 3D моделей: метод отображения границ и метод конструктивной сплошной геометрии. Метод отображения границ состоит в работе с линиями (ребрами) и вершинами, в свою очередь метод конструктивной сплошной геометрии основан на работе с геометрическими примитивами, такими как куб или шар.

Можно выделить две основные стадии восстановления 3D объектов по проекциям: создание псевдокаркасной модели объекта (pseudo-wireframe) и создание объекта (solid). Псевдокаркасная модель состоит из набора вершин и ребер, соединяющих эти вершины. Она может содержать лишнюю, но в то же время более полную информацию об описываемом объекте. В отличие от каркасной модели, она может включать в себя «лишние» ребра, которые принадлежат описывающим ее проекциям (рис. 1). Одна псевдокаркасная модель может описывать большое количество 3D объектов (рис .2).

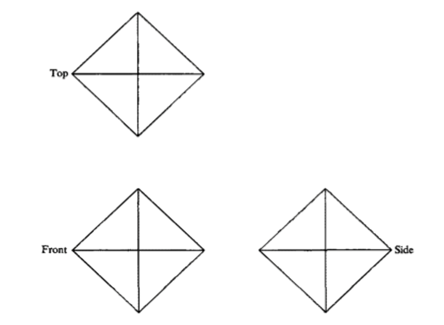
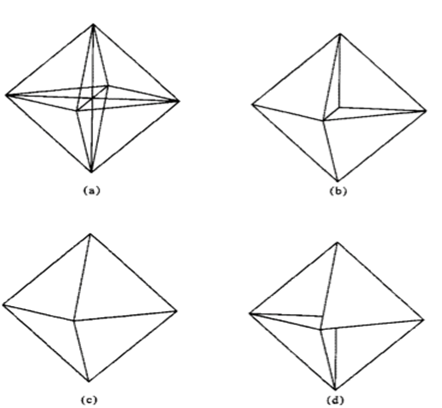
 

Рисунок 1. Три проекции октаэдра Рисунок 2 (a) – псевдокаркасная модель октаэдра,

(b, c, d) – возможные solid объекты, этой модели

в работе [] предлагает для сравнения САПРов, решающих данную задачу (Autodesk 3DS Max, Autodesk Autocad, Kompas-3D, Matlab, Blender), использовать следующие критерии (табл.1):

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  | Windows | OS X | Необходимость специальных знаний | Восстановление каркасной модели | Автоматическое выполнение функции | Лицензия |
| Приложения |
| 3DS Max | + | - | + | + | + | + |
| Autocad | + | + | + | + | - | + |
| Kompas-3D | + | - | + | - | - | + |
| Blender | + | - | + | - | - | - |
| Matlab | + | + | + | + | - | + |

Таблица 1. Сравнительная таблица существующих решений

Основным недостатком всех перечисленных систем за исключением последней является их платное распространение. Кроме этого Kompas-3D и Blender не имеют функционала восстановления 3D моделей из 2D проекций.

Autocad широко используется в сфере инженерного проектирования, однако реализованная в этой программе система восстановления 3D моделей из проекций основана на взаимодействии пользователя с системой.

3DS Max является практически самой распространенной и широко используемой системой для работы с трехмерной графикой. Программа предоставляет возможность восстанавливать 3D модель по проекциям, однако имеет довольно сложный интерфейс, а так же работает только в операционной системе Windows.

Matlab - кроссплатформенный пакет программ, который, как и Autocad широко используется в сфере инженерного проектирования. Кроме этого Matlab является высокоуровневым интерпретируемым языком программирования, поэтому для работы необходимы соответствующие знания языка.

Таким образом, очевидна актуальность разработки свободно распространяемой программы восстановления каркасных 3D объектов по трем 2D проекциям, которая была бы доступна для работы пользователю без специальной подготовки.

* + 1. **Метод отображения границ**

Одни из первых исследований в области восстановления 3D моделей по проекциям были проведены профессором Masanori Idesawa в 1973 году и были представлены в его работе «A system to generate a solid figure from three views» [6,7]. Idesawa предложил способы восстановления трехмерных вершин, ребер и лицевых поверхностей для многогранных объектов. Кроме того, так как некоторый набор вершин и ребер (каркасная модель) может описывать несколько объектов, он предложи способ, как удалять «призрачные» ребра для того чтобы в итоге получать уникальный объект. Сам предложенный алгоритм состоял в следующих шагах.

На вход программе подаются три проекции, которые представлены наборами вершин и ребер. Все вершины представляются в виде их координат x, y, z и порядковых номеров этих вершин. Пусть описывает вершину, тогда описывает набор вершин, которые соединяюются с ребрами. Если три или больше вершин находятся на одной прямой, они объединяются в группу. описывает группы, к которым принадлежит .

Точки в трехмерном пространстве описываются как:

, (1)

где - координаты i-го элемента списка вершин,

с – некоторая координата,

- количество вершин в ,

- количество вершин в ,

- количество вершин в .

Пусть () представляется комбинацией i, j, k элементов уравнения (1) так, что выполняются условия . Тогда () образуют матрицы трехмерных вершин . Эти операции применяются в (1) ко всем комбинациям i, j, k. Условия выполняются следующим образом:

Если , тогда

Если , тогда ,

Если , тогда

где – положительной число, обозначающее погрешность.

Пусть и – две разные вершины в . , , используются для обозначения i, j, k в . является элементом относительно ., , используются для обозначения i, j, k в . является элементом относительно . Далее обозначает k-ый элемент , который является элементом относительно. обозначает k-ый элемент , который является элементом относительно. Похожие переменные принимаются для плоскостей YZ и XZ.

Таким образом, результатом следующих булевых операций является определение того, соединяются ли вершины и ребром:

При восстановлении лицевых поверхностей используются следующие правила.

1. Существует k лицевых поверхностей, содержащих вершину пересечения k ребер;
2. Ребро представляет собой границу двух лицевых поверхностей, а в списках их границ вершины этого ребра идут в разном порядке;
3. Граница лицевой поверхности замкнута.

Для начала необходимо используя эти правила найти грань, которая может быть уникально определена. Затем нужно провести поиск всех ребер на границе грани и сохранить эти ребра в список и отсортировать, чтобы они удовлетворяли правилу (b). В конце нужно отсортировать вершины ребер против часовой стрелки для внешней лицевой поверхности и по часовой стрелке для внутренней поверхности грани.

Это был один из первых подходов к восстановлению 3D объектов. Впоследствии алгоритмы, в которых происходит работа с ребрами, получили общее название Boundary representation (Метод отображения границ). Модели в методе отображения границ создаются при помощи топологии и геометрических объектов. Основными топологическими единицами являются: faces (лицевые грани, или поверхности), edges (ребра) и vertices(вершины). Face – это ограниченная часть поверхности, относящаяся к некоторому ребру, ребро – сторона грани. В 1980 году George Markowky, профессор университета Мэна, и Michael A. Wesley из исследовательского института IBM Томаса Уотсона в своей работе «Fleshing out Wireframes» [8, 11] предложили использовать псевдокаркасную модель как промежуточную стадию процесса восстановления 3D объекта. Эта работа до сих пор является основополагающей для разработки новых алгоритмов в рамках метода отображения границ.

* + 1. **Метод конструктивной сплошной геометрии**

Первые исследования в рамках второго подхода к восстановлению 3D объектов по 2D проекциям были произведены Aldefel в 1983 году [1]. На входе в программу принимаются три проекции, из которых считываются данные о примитивах. Aldefel описывает возможные связи между примитивами. Одной из основных связей является CONTACT(n, m), которая обозначает что n и m связаны хотя бы одной общений точкой. Петля – замкнутая кривая без самопересечения.

Шаг 1: Установить все отношения между примитивами, которые связаны связью CONTACT.

Шаг 2: Найти, сохранить в списке все элементарные петли (петли, которые не содержат других петель) и отметить их как «открытые».

Шаг 3: Вычисление объема всех «открытых» петель и выбор петли P с максимальным объемом. Также случайным образом выбирается проекция , которая будет считаться базовым силуэтом.

Шаг 4: Предположим, что P описывает базовый силуэт одного или нескольких объектов. Для того чтобы доказать или опровергнуть это, необходимо выполнить следующий алгоритм:

* Случайным образом выбирается проекция и в ней ищутся все прямоугольные силуэты , которые совпадают с P. Если таких силуэтов не найдено, выйти из алгоритма.
* В оставшейся проекции найти для каждого прямоугольный силуэт , совпадающие как с , так и с P. Если таких силуэтов не найдено, выйти из алгоритма.
* Проанализировать P и найти все примитивы, которые требуют наличия линий в одной или обеих проекциях (базовая проекция не учитывается).
* Для каждой пары () найти полный список линий, которые описываются их примитивами. Если операция прошла успешно, итоговый объект представлен объединением и списка линий, полученных на этом шаге.

Шаг 5: «Увеличить» петлю P, создав все петли, которые включают в себя петли, прилежащие к P, если их еще не существует. Добавить их в список, отметить новые зависимости, отметить новые петли как «открытые», а P пометить как «закрытую».

Шаг 6: Спроецировать полученные объекты на плоскости, чтобы определить, описываются ли они входными проекциями. Если это так, то на этом работа алгоритма завершается, в противном случае перейти к шагу 3.

Описанный алгоритм работает только для объектов равномерной толщины. Подход носит название Конструктивная сплошная геометрия (Constructive solid geometry | CSG), и его основной концепцией является возможность математического описания любых сложных объектов при помощи более простых. Простейшими телами в CSG являются примитивы – тела простой формы, такие как куб, сфера, цилиндр, призма. Более сложные объекты создаются при помощи применения булевых операций (объединение, пересечение, разность) к некоторому набору примитивов.

* + 1. **Восстановление сплошных тел по шести проекциям**

Впоследствии Shum предложил алгоритм, в котором для восстановления 3D модели используются шесть проекций [4,11]. Алгоритм не обрабатывает пунктирные линии на проекциях. На проекциях допускаются прямые линии и круги, а также многогранные объекты с гранями, ортогональными хотя бы к одной оси координат, и ортогональные цилиндры.

В начале работы проекции делятся на три группы смежных проекций и располагаются соответствующим образом друг относительно друга. В процессе инкрементального выдавливания один из контуров становится образующим, а второй – направляющим. Из шести проекций образуются три пары. Для того чтобы образующий контур и направляющий не были параллельны, следующие операции выдавливания не выполняются:, , , и т.д., где f - front, R - rear, t - top, b - bottom, r - right, l - left. Кроме того, существует только шесть комбинаций пар проекций для процесса инкрементального выдавливания:

Инкрементальное выдавливание делится на три шага: сегментация образующих контуров в сегментные контуры, выдавливание толщины в направляющей проекции, совершение выдавливания.

В процессе инкрементального выдавливания учитываются только образующие контуры, внешние линии образующей грани. В начале, находятся все точки поворота в направляющем контуре, затем они проецируются перпендикулярно на общую грань проекции. Эти точки используются для разделения образующего контура не несколько сегментных контуров. Результатом является матрица SGS:

],

где - i-ый сегментный контур,

n – количество сегментных контуров для выдавливания.

Расстояния между точками поворота вдоль направления выдавливания определяют толщину выдавливания: , где m – количество инкрементальных выдавливаний.

В процессе самого выдавливания все сегментные контуры соединяются с толщинами выдавливания для того чтобы сформировать примитивные solid-объекты. Булевая операция объединения примитивных solid-объектов затем создаст выдавленный solid.

В итоге для каждой пары проекций объединение всех примитивов создаст два выдавленных solid-объекта. К ним применяется операция пересечения, чтобы сформировать solid пересечения. Пересечение пересеченных solid-объектов для всех пар проекций создаст итоговый solid.

Как конструктивная сплошная геометрия, так и метод отображения границ используются для создания итоговых 3D объектов. Как правило, восстановление псевдокаркасной модели является этапом работы алгоритмов обоих подходов. Однако задачей работы является восстановления каркасных моделей из 2D проекций. Основной вклад в разработку алгоритма восстановления каркасной модели внесли Markowsky и Wesley. При разработке последующих алгоритмов именно их подход всегда используется при восстановлении каркасных моделей объектов.

* 1. **Технологические этапы формирования 3D объекта по набору их 2d проекций**
     1. **Извлечение 2D ортографической информации**

Каркасная модель W(Ο) объекта Ο строится по фронтальной проекции (FV – front view), боковой проекции – виду справа (SV – side view) и верхней проекции – вид сверху (TV – top view), которые подаются на вход программе в формате систем автоматизированного проектирования (САПР) DXF. Файл каждой проекции необходимо проанализировать и загрузить информацию о хранящихся в нем геометрических компонентах – вершинах и ребрах – в матрицы FI(Ο),TI(Ο) и SI(Ο):

,

где - координаты вершины - начала ребра,

- координаты вершины - конца ребра,

где ,

n – количество ребер в проекции.

Каждое ребро задается координатами двух вершин, координаты одной вершины могут встречаться в описании несколько раз по количеству сходящих в ней ребер.

* + 1. **Разметка 2D граней и вершин**

Чтобы уменьшить количество параметров, определяющих ребра, каждая вершина отмечается порядковым номером. Номера вершин будут описывать ребра вместо координат этих вершин. Кроме того, сами ребра также отмечаются порядковыми номерами. Таким образом, для каждой проекции создаются матрицы и для хранения информации о вершинах и ребрах соответственно:

,

где – порядковый номер i-ой вершины,

- координаты i-ой вершины, i =1,2,…,k

k – количество вершин проекции.

,

где - объект типа «ребро», i = 1,2,…,n

n – количество ребер проекции.

Каждый объект типа «ребро» содержит4 следующую информацию:

,

где - порядковый номер ребра,

– порядковый номер вершины, определяющей начало ребра,

- порядковый номер вершины, определяющей конец ребра,

- тип ребра,

i = 1,2,…,n,

n – количество ребер проекции.

Таким образом, матрица вершин содержит порядковые номера вершин проекции и их координаты, а матрица ребер – порядковые номера ребер и порядковые номера вершин, определяющих эти ребра. Тип ребер предлагается запоминать для обработки пунктирных линий на проекции, которые обычно используются для обозначения ребер, находящихся не на лицевых гранях проекции.

Координаты вершин проекций могут значительно отличаться, поэтому для упрощения работы с ними предлагается сдвинуть каждую проекцию таким образом, чтобы она находилась в первом квадранте и хотя бы одна точка лежала на оси X и хотя бы одна точка лежала на оси Y. Для этого необходимо найти минимальную координату и минимальную координату вершин в каждой проекции. После этого необходимо выполнить следующую операцию для матриц каждой проекции:

,

,

где fv[i,2] – элемент i-ой строки и второго столбца матрицы ,

fv[i,3] – элемент i-ой строки и третьего столбца матрицы .

Впоследствии ребра проекции будут сравниваться с ребрами каркасной модели. Так как одно ребро проекции может описывать большое количество ребер в каркасной модели, ребра проекции должны быть максимальной длины, а не множеством коротких отрезков, лежащих на одной прямой.

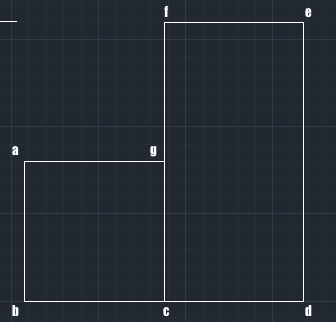


Рисунок 3 Вид сбоку

Например, на рисунке 3 представлена проекция некоторого объекта. В конечной матрице проекции не должно существовать несколько ребер, лежащих на одной прямой и пересекающихся в вершинах, таких как bc, cd и fg, gc. Если они были обнаружены, то оба ребра, лежащие на одной прямой, удаляются из , и добавляются ребра bd и fc соответственно.

* + 1. **Объединение граней и вершин**

Для последующей работы с данными о проекциях необходимо создать две новых матрицы и для верхней и боковой проекции соответственно. После этого нужно выполнить следующие операции над новыми матрицами:

,

,

где rtv[i,3] – элемент i-ой строки и третьего столбца матрицы (координата y),

rtv[i,4] – элемент i-ой строки и четвертого столбца матрицы (координата z),

rsv[i,2] – элемент i-ой строки и второго столбца матрицы (координата x),

rsv[i,4] – элемент i-ой 4строки и четвертого столбца матрицы (координата z).

Предполагается, что проекции, полученные в начале работы программы из файлов, созданы в едином масштабе. Для упрощения работы с каркасной моделью необходимо перенести данные о проекциях из матриц вершин в общую матрицу вершин и из матриц ребер в матрицу ребер .

Данные о проекциях расположены в и последовательно, поэтому теперь легко можно провести перпендикуляры из всех вершин: по z для фронтальной проекции, по y вверх для верхней проекции и по x вправо для боковой проекции, - и добавить новые вершины в , а новые ребра в .

На этом этапе работы алгоритма часть ребер никак не связана между собой, а также существуют повторяющиеся ребра, поэтому нужно удалить повторяющиеся вершины из и заменить порядковые номера этих вершин в на соответствующие не удаленные. Кроме того, нужно удалить повторяющиеся ребра из .

* + 1. **Нахождение точек пересечения ребер**

Перпендикуляры, проведенные на предыдущем шаге все еще никак между собой не связны, поэтому необходимо найти точки их пересечения. Заранее не известно, какие именно ребра пересекаются; для каждого ребра проверяются все остальные, еще не проверенные ребра, на предмет их пересечения. В этой проверке можно выделить четыре случая: ребра параллельны, находятся на одной прямой, скрещиваются (лежат в параллельных плоскостях), пересекаются. Проверку нужно провести только на нахождение ребер на одной прямой и на пересечение.

* + - 1. **Определение нахождения ребер на одной прямой**

Уравнение прямой, проходящей через две точки, в трехмерном пространстве выглядит следующим образом:

где - координаты первой точки,

- координаты второй точки.

Так как проверяемые ребра (восстановленные перпендикуляры) параллельны хотя бы одной оси координат, в этой формуле будет происходить деление на ноль, во избежание этого нужно воспользоваться правилом пропорции:

(2).

Допустим, сравнивается ребро AB и ребро CD. Если подставить во (2) уравнение вместо координаты вершины A, вместо координаты вершины B, а вместо поочередно координаты вершин C и D, и оба выражения будут выполняться, то ребра лежат на одной прямой.

В этом случае ребра могут, не пересекаются. Если же ребра ab и cd пересекаются, могут возникнуть пять ситуаций:

* ребра пересекаются в одной из существующих вершин (рис. 4);

Macintosh HD:Users:niksan93:Desktop:ВКР:пересечение в вершинах.jpg

Рисунок 4

* ребра совпадают, в этом случае второе ребро удаляется из ;
* одна из вершин одного ребра находится на другом ребре, к примеру, вершины ребер ab и cd расположены в следующем порядке: a c b d, в этом случае ребра ab и cd удаляются из , а новые ребра ac, cb и bd добавляются в матрицу (рис. 5);

Macintosh HD:Users:niksan93:Desktop:ВКР:acbd.jpg

Рисунок 5

* одно ребро находится внутри другого, но вершины не совпадают, в этом случае меньшее ребро удаляется из . Например, на рис. 6 нужно удалить ребро ab;

Macintosh HD:Users:niksan93:Desktop:ВКР:cabd.jpg

Рисунок 6

* одно ребро находится внутри другого, и одна из вершин ребра ab совпадает с одной из вершин ребра cd, к примеру, если a = c и вершины расположены в порядке: a=c, d, b (рис. 7), ребро ab удаляется из , а новое ребро db добавляется в матрицу.

Macintosh HD:Users:niksan93:Desktop:ВКР:a_cdb.jpg

Рисунок 7

* + - 1. **Определение пересечения ребер**

Две прямые AB и CD:

проходящие через точки , и , соответственно, пересекаются, если выполняется условие:

и если ранг этой матрицы равен двум. Иными словами для прямых AB и CD нужно найти такие коэффициенты u и v, что система из трех уравнений (3) имеет решение:

(3)

Определитель матрицы 3х3 считается по формуле (4):

(4).

Ранг матрицы равен количеству ненулевых строк в матрице ступенчатого вида. Для того чтобы привести матрицу к ступенчатому виду можно воспользоваться методом Гаусса, для этого введем элементарные преобразования матрицы:

1. Перестановка двух столбцов или строк матрицы;
2. Умножение элементов одной строки или столбца на одно и то же число, отличное от нуля;
3. Прибавление к элементам одной строки или столбца соответствующих элементов другой строки или столбца, умноженных на одно и то же число.

Для того чтобы привести матрицу к ступенчатому виду методом Гаусса необходимо выполнить следующие шаги:

1. Выбрать в первом столбце элемент, отличный от нуля (ведущий элемент). Если строка с ведущим элементов (ведущая строка) не первая, переставить ее на место первой строки (преобразование I типа). Если в первом столбце нет ведущего элемента, то продолжить поиск в оставшейся части матрицы. Нужно закончить преобразования, если исключены все столбцы.
2. Разделить все элементы ведущей строки на ведущий элемент. Если строка последняя, то преобразования заканчиваются.
3. К каждой строке ниже ведущей строки прибавить ведущую строку, умноженное на такое число, что все элементы ниже ведущего элемента становятся равны нулю.
4. Исключив ведущую строку и ведущий столбец, перейти к пункту 1 для оставшейся части матрицы.

Если определитель равен нулю и ранг матрицы равен двум, систему уравнений (2) можно решить при помощи формул Крамера [12]. Для решения системы из трех уравнений с двумя неизвестными достаточно двух уравнений, поэтому решение, использующее первое и второе уравнение будет иметь вид:

.

Если плоскость, в которой находятся пересекающиеся ребра, перпендикулярна плоскости XY, то D будет равен нулю. Аналогично плоскость пересечения может быть перпендикулярна плоскостям XZ и YZ. Поэтому для нахождения точки пересечения выбираем пару уравнений, где . Таким образом, решение для первого и третьего уравнений:

.

Решение для второго и третьего уравнений:

.

В случае пересечения ребер возможны четыре случая:

* Прямые, определенные вершинами ребер пересекаются, но сами ребра не пересекаются;
* Ребра пересекаются в одной из их вершин. Если ребро ab пересекает cd в точке a (рис. 8), то cd удаляется из , а новые ребра ca и ad добавляются в список;

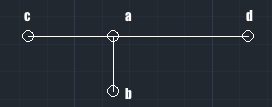


Рисунок 8

* Ребра пересекаются в новой вершине. Если ребро ab пересекает cd в некоторой точке m (рис. 9), ab и cd удаляются из , а новые ребра am, mb, cm, md добавляются в список;

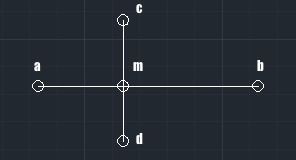


Рисунок 9

* Ребра пересекаются двумя вершинами (рис. 10).



Рисунок 10

Таким образом, в каркасную модель добавлены все ребра, ортогональные ко всем трем проекциям. Следующим шагом является добавление ребер, не ортогональных ни к одной проекции.

* + 1. **Создание не ортогональных ребер**

Для того чтобы создать все возможные ребра не ортогональные ни к одной плоскости проекции, необходимо спроецировать вершины на каждую из плоскостей XY, XZ и YZ. Под проецированием понимается приравнивание к нулю одной из координат трехмерной вершины, Таким образом, при проецировании на плоскость XY координата z приравнивается к нулю, при проецировании на XZ координата y приравнивается к нулю, при проецировании на YZ координата x приравнивается к нулю.

После этого для каждой вершины каркасной модели перебираются вершины соответствующей проекции ( для XY, для XZ, для YZ). Для каждой i-ой вершины проекции выделяются координаты вершин, соответствующие плоскости, на которую модель спроецирована, которые соединяются ребрами с i-ой вершиной (соседних вершин). В каркасной модели создаются ребра, соединяющие i-ю вершину и все вершины с выделенными двухмерными координатами.

На рисунке 11 приведен пример проекций некоторого объекта (a,b,c) и результат, который был, достигнут на момент выполнения шага 2.4.2.

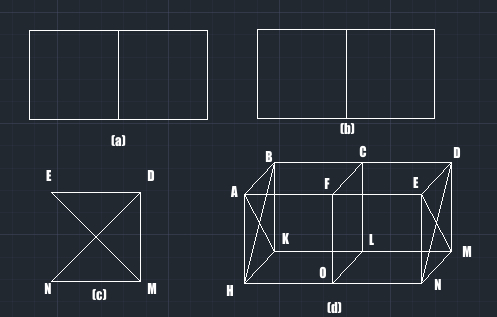


Рисунок 11 (a) – фронтальная проекция, (b) – вид сверху, (c) – вид сбоку, (d) – текущая каркасная модель

На текущем этапе алгоритма необходимо восстановить ребра FL и CO. Из модели видно, что для этого необходимо спроецировать модель на плоскость YZ (боковая проекция). В этом случае координаты y и z каждой вершины сравниваются с соответствующими координатами матрицы . К примеру, вершина C по этим координатам равна вершине D проекции. Для вершины D находятся все вершины, которые соединяются ребрами с D. На проекции это вершины E, N, M. После этого в каркасной модели находятся все вершины, чьи координаты y и z (в данном случае) равны координатам y и z вершин E, N или M. Из проверяемой вершины C проводятся все возможные ребра к найденным вершинам, если таких ребер еще не существует. Таким образом, в каркасную модель добавятся ребра CA, CE, CK, CM, CH, CO, CN (рис. 12).

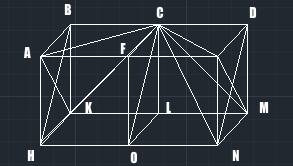


Рисунок 12

Таким образом, необходимое ребро CO, которое не ортогональной ни одной проекции, было добавлено в список ребер. Все остальные ребра, скорее всего, не принадлежат всем проекциям и будут удалены на следующем шаге алгоритма. Такие операции производятся для каждой вершины модели, а так же для каждой проекции.

* + 1. **Удаление лишних ребер**

На данном этапе выполнения алгоритма в псевдокаркасной модели существует большое количество ребер, не принадлежащих проекциям. Для того чтобы их удалить, необходимо спроецировать вершины на каждую из плоскостей XY, XZ и YZ и сравнить ребра ) с , и соответственно. Если спроецированное ребро модели совпадает с одним из ребер соответствующей проекции, но не выходит за его пределы, оно остается в списке, в противном случае – удаляется.

В дополнение к алгоритму построения псевдокаркасной модели объекта по трем 2D проекциям, предложенному Markowsky и Wesley [11], в работе предлагается алгоритм обработки пунктирных линий, который можно использовать для несимметричных относительно центра проекций.

Изначально предполагается что объект, чья псевдокаркасная модель создается в ходе алгоритма, симметричен относительно центра, поскольку известны только три его проекции. На предыдущем шаге могли быть созданы ребра, которые принадлежат проекциям, но в итоговую каркасную модель не входят.

Рассмотрим возможность восстановления каркасных моделей несимметричных объектов. Предположим, что фигура симметрична только относительно двух плоскостей XY и YZ. Для каждого пунктирного ребра проекции создается список ребер из , которые совпадают с пунктирным ребром. Если, удалять ребра на ближайшей и дальней гранях каждой плоскости, то может возникнуть ситуация, при которой на дальней грани удалятся нужные ребра. Поэтому предлагается для фронтальной проекции из получившихся списков удалять ребро с наименьшей координатой z. Для боковой проекции удалять ребро с наибольшей координатой x. Для верхней проекции удалять ребро с наибольшей координатой y.

Таким образом, из трех двумерных проекций в результате работы алгоритма получается псевдокаркасная модель с допущением, что на дальних гранях объекта могут сформироваться ребра, принадлежащие проекциям, но, возможно, не принадлежащие объекту. Пользователю предоставляется возможность интерактивного удаления лишних ребер.

1. **Практическая часть** 
   1. **Описание разработанных и используемых алгоритмов**
2. **Информационное обеспечение системы**
   1. **Формат CDR**

Формат CDR (CorelDRAW file format) предназначен для хранения векторных изображений или рисунков, созданных с помощью программы CorelDRAW. Данный формат файла разработан компанией Corel для использования в собственных программных продуктах. CDR-файлы не поддерживаются многими программами, предназначенными для редактирования изображений. Однако, файл можно экспортировать с помощью CorelDRAW в другие, более распространенные и популярные форматы изображений. Формат известен в прошлом низкой устойчивостью и плохой совместимостью файлов, тем не менее, пользоваться CorelDRAW чрезвычайно удобно. В файлах этих версий применяется раздельная компрессия для векторных и растровых изображений, могут внедряться шрифты, файлы CDR имеют огромное рабочее поле 45х45 метров, поддерживается многостраничность.

* 1. **Формат AI**

Формат AI (Adobe Illustrator Document) предназначен для хранения векторных файлов изображения, созданных в программе Adobe Illustrator. Формат AI поддерживают практически все программы, так или иначе связанные с векторной графикой. Является лучшим посредник при передаче векторных изображений из одной программы в другую, с РС на Macintosh и наоборот (или др вариант «в другие программы на разные платформы»). Отличается наибольшей стабильностью и совместимостью с языком PostScript, на который ориентируются практически все издательско-полиграфические приложения. PostScript в полной реализации представляет собой мощный и сложный язык и способен определять почти все, что может быть отображено на двумерном устройстве вывода, формат AI адаптирован для хранения традиционных графических данных: рисунков, чертежей и декоративных надписей.

* 1. **Формат DXF**

Формат DXF (Drawing eXchange Format) это открытый формат файлов для обмена данными между приложениями систем автоматизированного проектирования (САПР). Был создан для Autocad фирмой Autodesk и поддерживается практически всеми системами САПР на платформе PC. Является одним из самых распространенных и бесплатных форматов файлов для обмена графической информацией [**Ошибка! Источник ссылки не найден.**]. Формат DXF содержит все информацию об изображении в виде маркированных данных. Это означает, что различные данные обозначаются целыми числами.

1. **Условие разработки и функционирования ПО**

Среда разработки - Microsoft Visual Studio 2017.

Язык разработки - C++.

Среда визуализации - графическая библиотека OpenGL.

Требования к квалификации пользователя - оператор ЭВМ.

Минимальные технические требования для работы с программным продуктом:

1. ПК типа Sony VAIO SVE1511X1RW;
2. ОС - не ниже Windows 7 SP1;
3. Видеоадаптер AMD Radeon HD 7650M;
4. Разрешение экрана 1366x768;
5. Процессор - не ниже 2.50GHz;
6. Оперативная память - не ниже 4 Гб;
7. Место на жестком диске (HDD) - не ниже 40 Гб;
8. Периферийные устройства ввода информации: клавиатура, мышь или тачпад.

Дополнительные требования: платформа.NET Framework 4.5 или более новые версии.

Объем программного кода - строк

Разработано классов -

Примерное время разработки - 5 месяцев.

1. **Детальное описание разрабатываемого ПО**
   1. **Назначение**
   2. **Входные/Выходные данные**

Исходные параметры:

На вход программе подаются размеры трех проекции детали: длина, ширина, высота. Проекции окружности задаются параметрическим уравнением с радиусом r и углом angle = 2 \* PI \* (double)(p - 1) / amountSegments.

Параметрические уравнения окружности:

Варьируемые параметры:

* dlina - длина;
* shirina - ширина;
* visota - высота;
* r - радиус;
* lats - количество подразделов по широте;

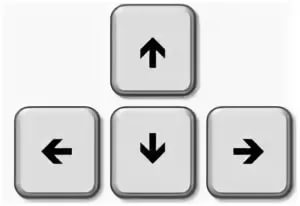
Ограничения:

* lats <= (50);
  1. **Коды завершения**
  2. **Руководство оператора**
* Для запуска программы нажмите на значок запуска в верхней панели Visual Studio.



Программа должна вывести на экран 2 окна. В первом окне отображается фигура, во втором - координаты точек пересечения линий.

* Окно с фигурой можно двигать с помощью специальных клавиш на клавиатуре.



* Окно с координатами можно просматривать с помощью «полосы прокрутки».



* Для завершения работы программы нужно нажать на  в правом верхнем углу окна или нажать на  расположенного в верхней панели Visual Studio.
  1. **Действия оператора при сбое**

Если при запуске программа не запустилась в течение 30 секунд или выдала ошибку, нужно нажать на  или  чтобы завершить работу. Повторите попытку запуска программы заново.

* 1. **Результаты**

**Заключение**

В ходе проделанной работы были

1. Решены следующие задачи:

- изучены существующие форматы представления инженерных чертежей;

- изучены принципы и технологии моделирования двухмерных и трехмерных графических объектов;

- изучены принципы работы библиотеки OpenGL;

- рассмотрены основные подходы в восстановлении 3D объектов по 2D чертежам проекции;

- изучены существующие алгоритмы восстановления трехмерной модели по трем двухмерным ортогональным проекциям;

1. Разработано:

- алгоритмы

- информационное обеспечение системы

- архитектура программного комплекса синтеза 3D объекта по набору их 2D проекций…

- программное обеспечение на языке С++, состоящее из таких то классов..

1. Произведена комплексная отладка и тестирование разработанного ПО.
2. Практическая применение данного комплекса обнаружило его работоспособность и эффективность при использование.

**Список литературы**

1. Markowsky, G. & Wesley, M. (1980). Fleshing out wireframes. IBM Journal of Research and Development, 24 (5), 582–597.
2. Wesley, M.A. & Markowsky, G. (1981). Fleshing out projections, IBM Journal of Research and Development, 25 (6), 934–954.

**Приложение**