

Министерство науки и высшего образования Российской Федерации

Пензенский государственный университет

Кафедра «Системы автоматизированного проектирования»

ПОЯСНИТЕЛЬНАЯ ЗАПИСКА

к курсовой работе

по курсу «Компьютерная графика и 3D моделирование»

на тему: «Разработка программы построения и визуализации 3D объекта»

Выполнили:

Студенты группы 21BBS1

Шутихин А.Э.

Савкин В.В.

Принял:

Финогеев А. А.

Пенза 2023

ПЕНЗЕНСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ

Кафедра

Системы автоматизации проектирования

УТВЕРЖДАЮ

Зав. кафедрой

“ ”

г.

ЗАДАНИЕ

на курсовое проектирование по курсу

Компьютерная графика и 3D моделирование

Студенту

Савинну Василию Владимировичу

Группа

21ВВС

Тема

проекта

«Разработка программы построения и визуализации 3D объекта»

Исходные данные (технические требования) на проектирование

Основная цель проекта: Разработка графической программы для построения визуализации 3D модели на языке программирования C++, для реализации процесса динамики объекта, фона и камеры в трехмерной сцене.

Объект моделирования: 3D модель реального объекта -

Задачи проекта:

1. Спроектировать 3D твердотельную модель заданного объекта средствами Autocad (3D Max и т.д.)
2. Синтезировать 2D чертежи проекционных видов 3D модели на плоскости XY, XZ, YZ.
3. Импортировать полученные 3D модели объекта в программную среду C++.
4. Подобрать соответствующие текстуры для объектов сцены.
5. С помощью собственных функций синтезировать 3D объект и визуализировать 3D сцену.
6. Реализовать движения частей объекта относительно друг друга, движение камеры вокруг объекта.
7. Реализовать пользовательское управление движением камеры вокруг 3D объекта посредством клавиатуры (мыши)

Учебно-методический материал: Сервер - CAD-Filer, Диск - Т, Раздел - Компьютерная графика

Программное обеспечение: C++ Builder

CAD Import VCL - DXF DWG PLT HPGL SVG CGM библиотека для Delphi и C++.

Объем работы по курсу

1. Расчетная часть (содержание пояснительной записки)
 1. Анализ и описание процесса синтеза 3D объекта методом твердотельного моделирования
 2. Декомпозиция на графические примитивы, аппроксимация гранями, синтез видов
 3. Подбор текстур, источников освещения и камеры
 4. Описание расположения объекта/сцены в системе координат
 5. Импортирование созданных моделей и ландшафта в программную среду
 6. Разработка программы визуализации объекта/сцены с текстурами, освещением, камерой
 7. Программная реализация динамики объекта и камеры посредством задания их траекторий
 8. Реализация интерфейса в отношении управления объектом или камерой в сцене
 9. Отладка и тестирование программы
 10. Визуализация управляемой анимации объекта, камеры и сцены в целом
11. Графическая часть
 1. Чертежи трех проекций объекта
 2. Структурная схема алгоритма,
 3. Листинг программ.
12. Экспериментальная часть
 1. Тестирование и верификация программного обеспечения

Срок выполнения проекта по разделам

1	Моделирование 3D объекта и чертежей	к	22.02.23
2	Импорт моделей в среду	к	8.03.23
3	Разработка алгоритма и программы	к	22.03.23
4	Реализация движения объекта и камеры	к	5.04.23
5	Реализация интерфейса и элементов управления	к	12.04.23
6	Верификация и отладка ПО	к	19.04.05
7	Подготовка пояснительной записки и защита проекта	к	17.05.23

Дата выдачи задания "8" февраля 2023г.

Дата защиты проекта "17" мая 2023г.

Руководитель к.т.н., доцент

/А.А. Финогеев/

Задание получил "8" февраля 2023г.

Студент Савин В.В.

Оглавление

Введение	7
1 Постановка задачи.....	8
2 Расчётная часть.....	9
2.1 Анализ и описание процесса синтеза 3D объекта методом твердотельного моделирования	9
2.2 Описание расположения объекта в системе координат	11
2.3 Импортрование созданных моделей в программную среду	13
2.4 Программная реализация.....	16
2.5 Отладка и тестирование.....	22
Заключение	23
Список Литературы.....	24
Приложение 1. Визуализация объекта	25
Приложение 2. Листинг	29

Введение

Компьютерные технологии предоставляют широкий спектр возможностей для визуализации в трёхмерном пространстве. Трёхмерное моделирование как пространственное изображение объекта в трёхмерной системе координат даёт возможность максимально информативно, детально и реалистично представить его форму, текстуру, размер и цвет, позволяет рассмотреть объект с любого интересующего ракурса.

Трёхмерная графика считается незаменимым средством для демонстрации разного рода сложных технических узлов, многоступенчатых производств, архитектурных сооружений. Трёхмерные модели наглядно показывают все особенности строения объекта, его мельчайшие элементы, которые скрыты от глаз наблюдателя. На сегодняшний момент 3D изображения являются пиком совершенства в рекламной и дизайнерской индустрии.

Компьютерное моделирование — это процесс создания компьютерных моделей. Компьютерная модель — компьютерная программа, работающая на отдельном компьютере или множестве взаимодействующих компьютеров, реализующая абстрактную модель некоторой системы. Компьютерные модели стали обычным инструментом математического моделирования и применяются в физике, астрофизике, механике, химии, биологии, экономике, социологии и других науках.

Компьютерные модели используются для получения новых знаний о моделируемом объекте или для приближенной оценки поведения систем, слишком сложных для аналитического исследования.

В качестве среды разработки 3D объекта была выбрана программа Blender 3.4., для синтезирования 2D чертежа – Microsoft Visual Studio 2019, язык программирования C++.

Целью данной курсовой работы является создание программы построения каркасной модели 3D объекта.

1 Постановка задачи

Требуется разработать программу, которая визуализирует 3D объект на экране на основе заданных координат точек и связей между ними. Сам 3D объект должен быть спроектирован в среде моделирования и импортирован в программу. Пользователь должен иметь возможность управлять движением камеры посредством клавиатуры или мыши.

2 Расчётная часть

2.1 Анализ и описание процесса синтеза 3D объекта методом твердотельного моделирования

Шаг 1 Добавили цилиндр

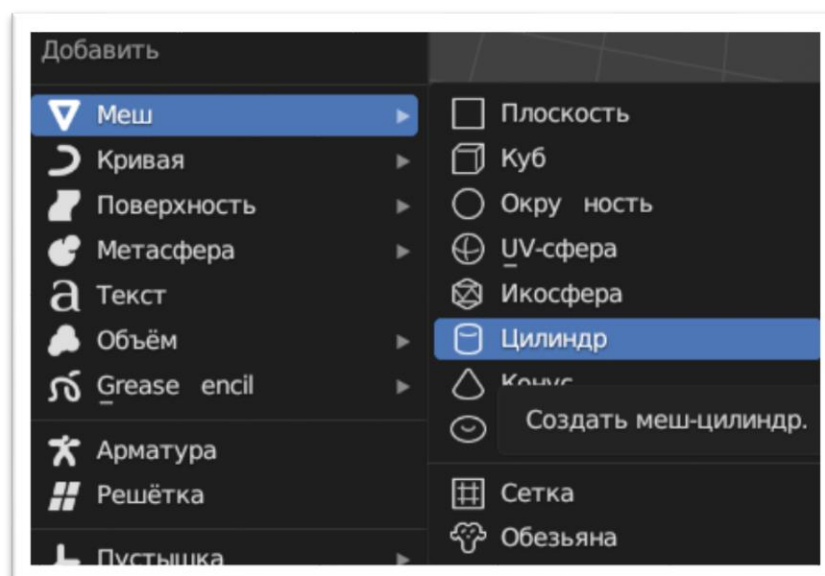


Рисунок 1 – Создание цилиндра

Шаг 2 Используя режим Edit mode изменили форму цилиндра для моделирования фюзеляжа, а также использовали модификатор “Mirror” для отражения объекта по оси у и удалили ту половину объекта, которая отразилась.

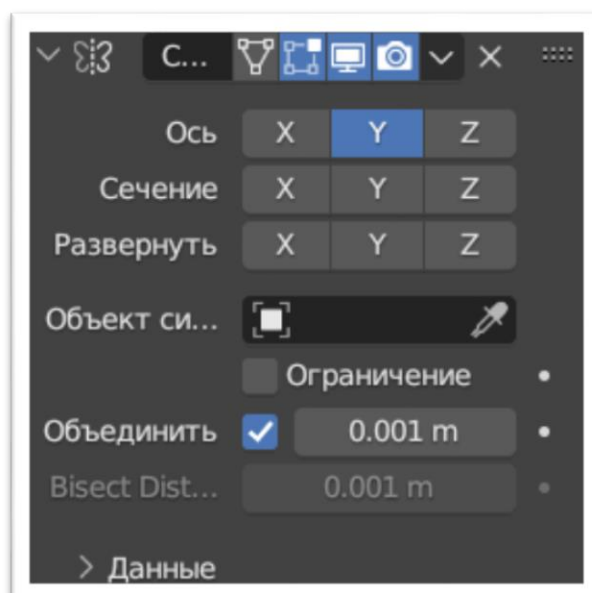


Рисунок 2 – Отражение по оси у

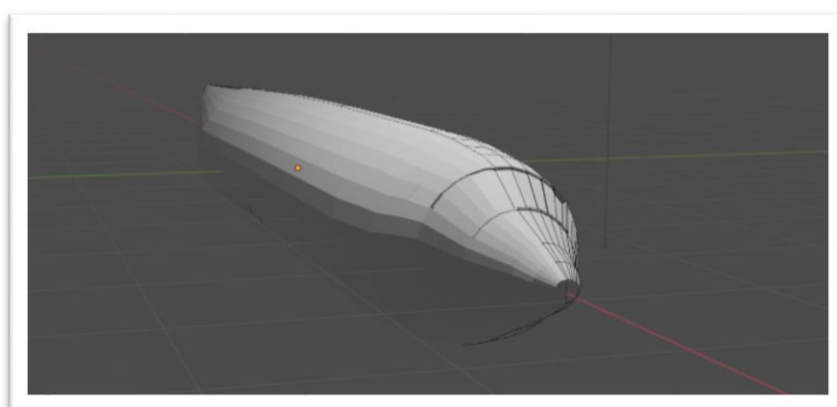


Рисунок 3 – Моделирование фюзеляжа

Шаг 3 С помощью функции «Extrude» смоделировали хвостовое оперение и крылья самолёта

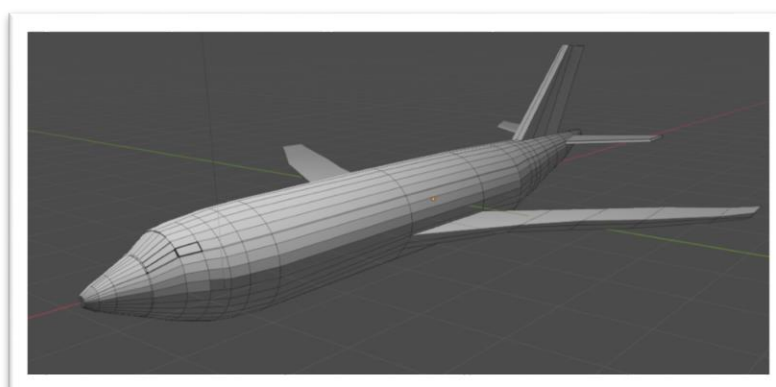


Рисунок 4 – Моделирование хвоста и крыльев

Шаг 4 Добавили новый объект цилиндр, чтобы смоделировать двигателя

Шаг 5 Смоделировали цилиндр двигателя, после чего с помощью функции «Extrude» закончили его детализацию



Рисунок 5 – Моделирование двигателя

Шаг 6 Совместили ранее созданные объекты

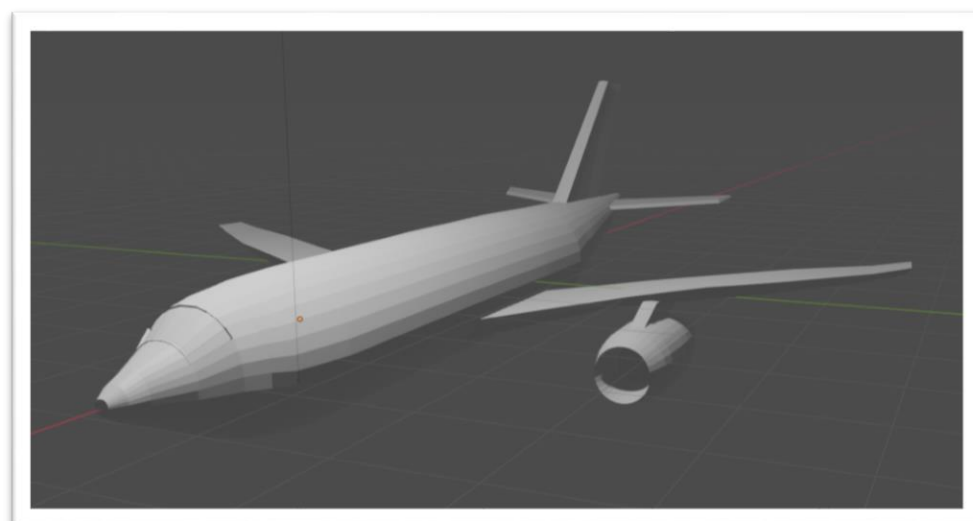


Рисунок 6 – Результат моделирования

2.2 Описание расположения объекта в системе координат

Проекция объекта на плоскости (x, y) , (y, z) , (x, z)

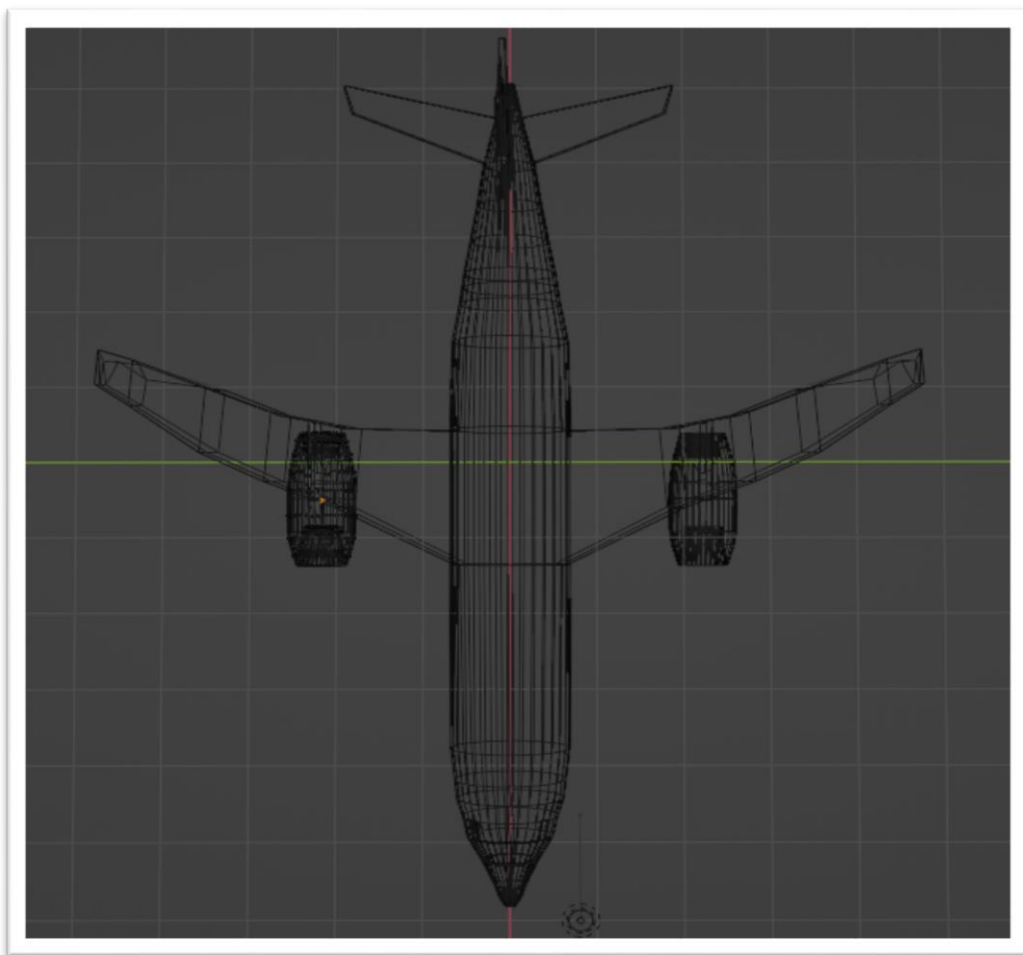


Рисунок 7 – Проекция на (x, y)

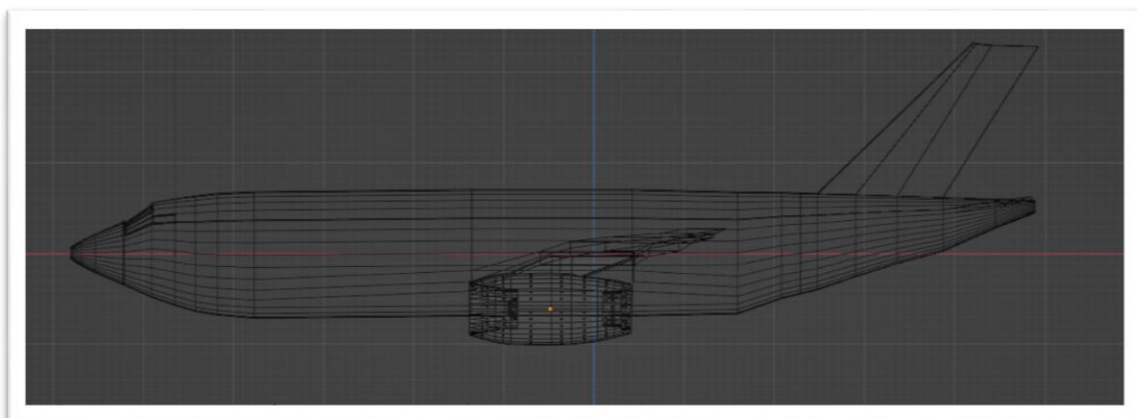


Рисунок 8 – Проекция на (y, z)

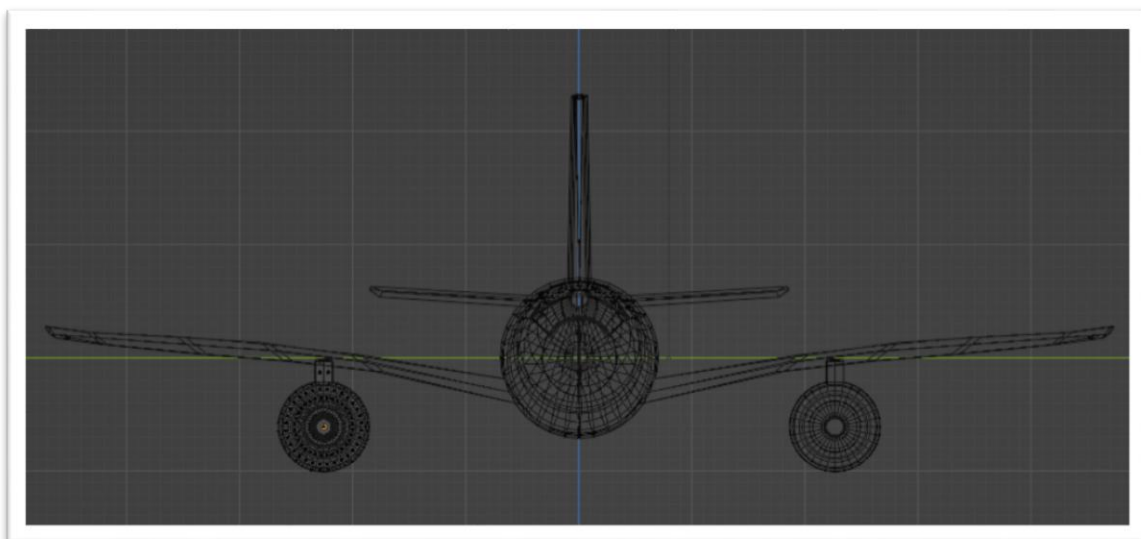


Рисунок 9 – Проекция на (x, z)

2.3 Импортирование созданных моделей в программную среду

Для импортирования созданной модели был использован obj файл. Obj файл – это файл, содержащий информацию о геометрии объекта. Он хранит координаты вершин, координаты текстуры объекта, координаты нормалей, параметры одной поверхности объекта. В данной курсовой работе, в программной среде реализована отрисовка проволочной модели, а значит достаточно, чтобы в obj файле хранилась лишь информация о координатах вершин(v) и рёбрах (l).

```
l 853 865
l 867 868
l 868 838
l 837 867
l 717 718
l 716 715
l 716 717
o Цилиндр.001_Цилиндр.005
v 0.986428 -0.608018 2.030288
v -0.410964 -0.608019 1.980003
v 0.986428 -0.651967 2.034616
v -0.410964 -0.661777 1.985298
v 0.986428 -0.694227 2.047435
v -0.410964 -0.713470 2.000978
```

Рисунок 10 – содержание файла

Для импортирования модели в среду программирования был написан алгоритм для чтения из obj файла с сохранением результата в вектор.

Алгоритм работает следующим образом: функция void read_file() построчно проходят файл и записывают значения в вектор строк, из которого в зависимости от первого символа (v или l) записываются значения в соответствующие массивы.

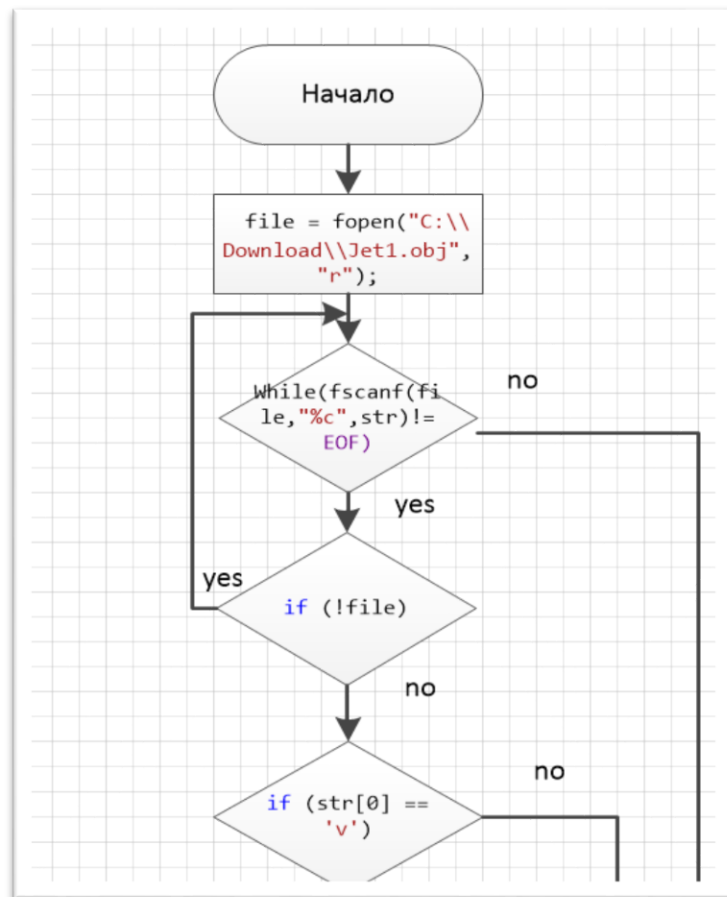


Рисунок 11 – граф схема функции чтения файла

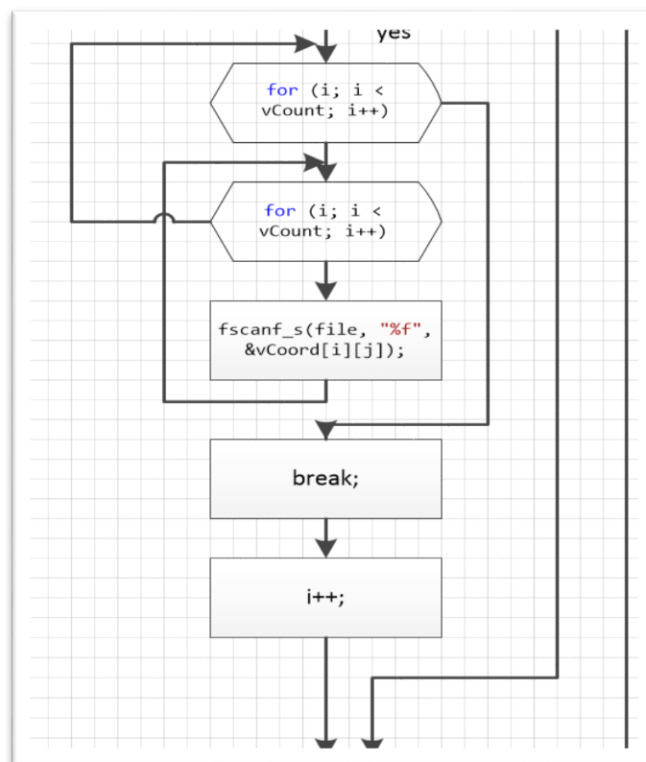


Рисунок 12 – продолжение Рисунок 11

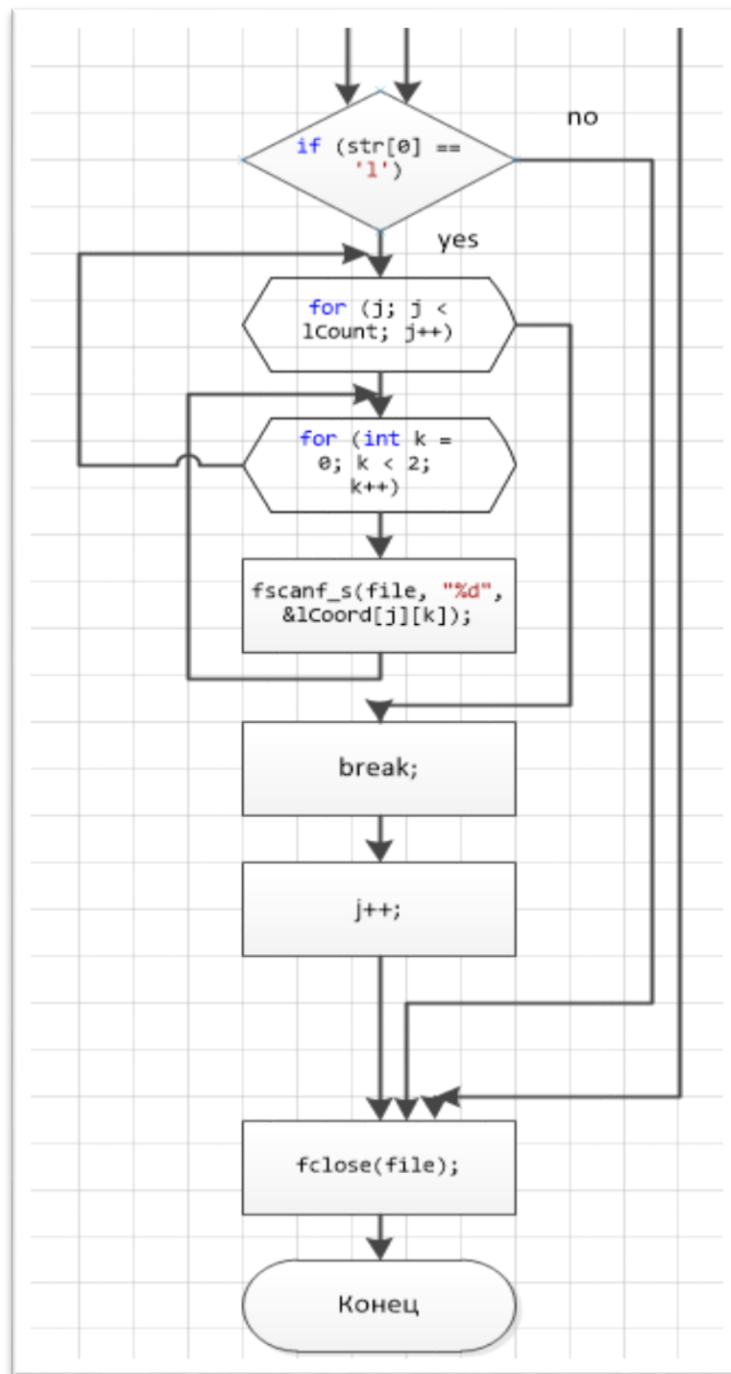


Рисунок 13 – продолжение Рисунок 12

Отрисовка модели реализована с помощью цикла и работает следующим образом: Из массива, хранящего рёбра считываются два числа – индекс исходной вершины и конечной вершины. После чего с помощью функций `MoveToEx` и `LineTo` рисуется линия. Координаты вершин получаю из массива с вершинами, индекс которых узнал ранее.

2.4 Программная реализация

Чтобы реализовать динамику объекта необходимо преобразовать мировые координаты вершин. Преобразование будет проходить в три этапа:
а) видовое б) перспективное в) экранное.

Для видового преобразования необходима величина R - расстояние от точки наблюдения до начала мировой системы координат, полярные углы Phi и Theta (в горизонтальной и вертикальной плоскости соответственно).

Для реализации видового преобразования была написана функция void transformation(). Функция выполняет перемножение матрицы видовых преобразований на координаты вершин объекта и записывает результат в двумерный массив plane.

$$\begin{bmatrix} X_e & Y_e & Z_e & 1 \end{bmatrix} = \begin{bmatrix} X_w & Y_w & Z_w & 1 \end{bmatrix} * \begin{bmatrix} -\sin \theta & -\cos \varphi \cos \theta & -\sin \varphi \cos \theta & 0 \\ \cos \theta & -\cos \varphi \sin \theta & -\sin \varphi \sin \theta & 0 \\ 0 & \sin \varphi & -\cos \varphi & 0 \\ 0 & 0 & R & 1 \end{bmatrix}$$

Рисунок 14 – Матрица видовых преобразований

Для перспективных преобразований используется параметр d, который задает расстояние между экраном и точкой наблюдения. Чтобы реализовать данное преобразование была написана функция void proec().

$$\frac{X}{d} = \frac{x}{z} \quad Y = d \cdot \frac{y}{z}$$

Рисунок 15 – Перспективные преобразования

Для изменения размера изображения была написана функция zoom(), которая умножает координаты x, y, z на одно число.

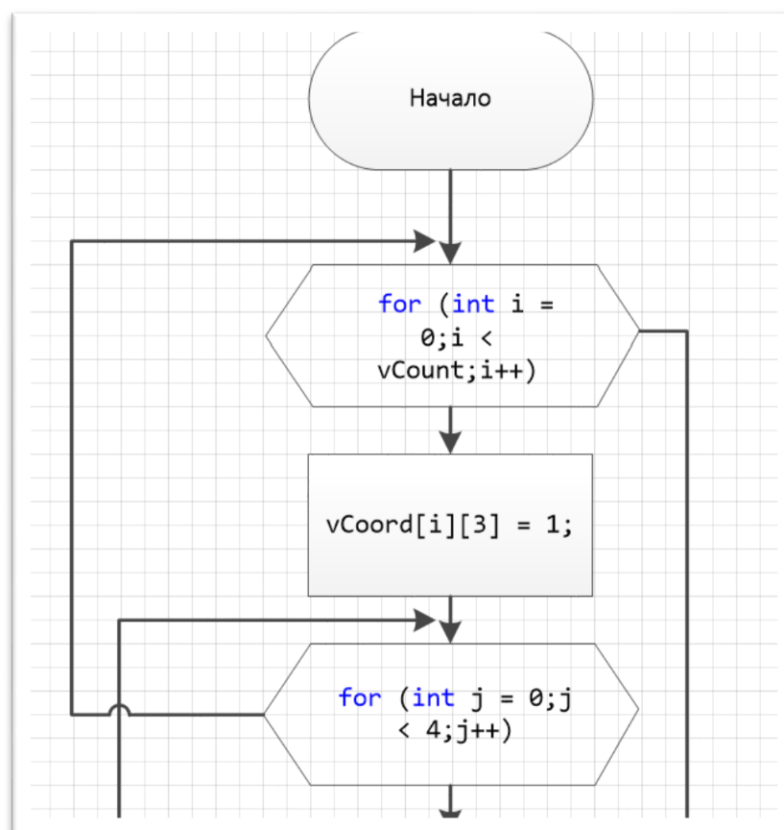


Рисунок 16 – граф схема функции transformation()

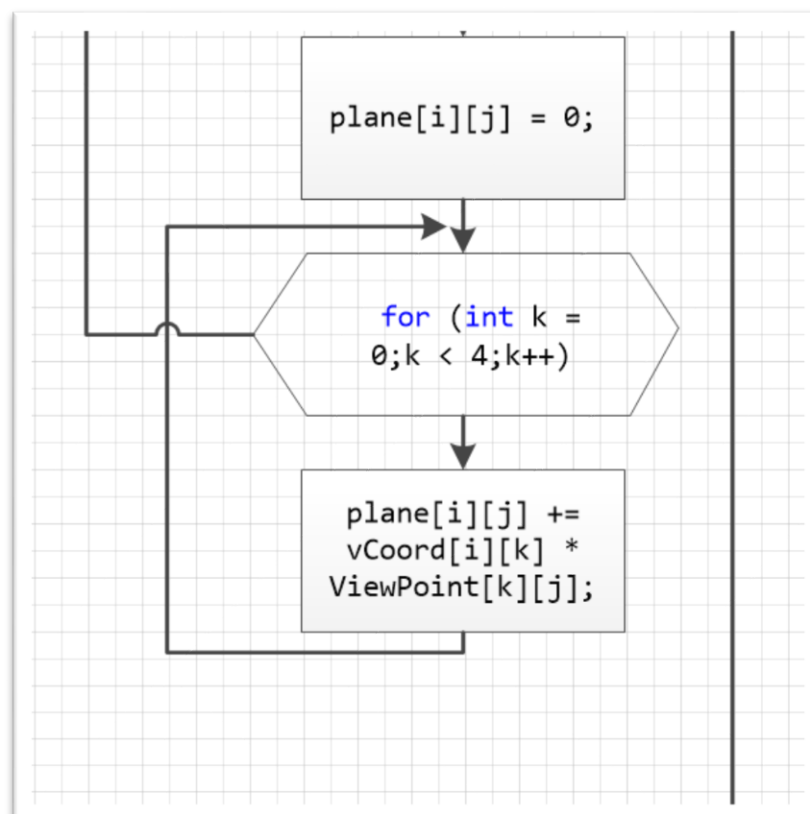


Рисунок 17 – Продолжение рисунок 16

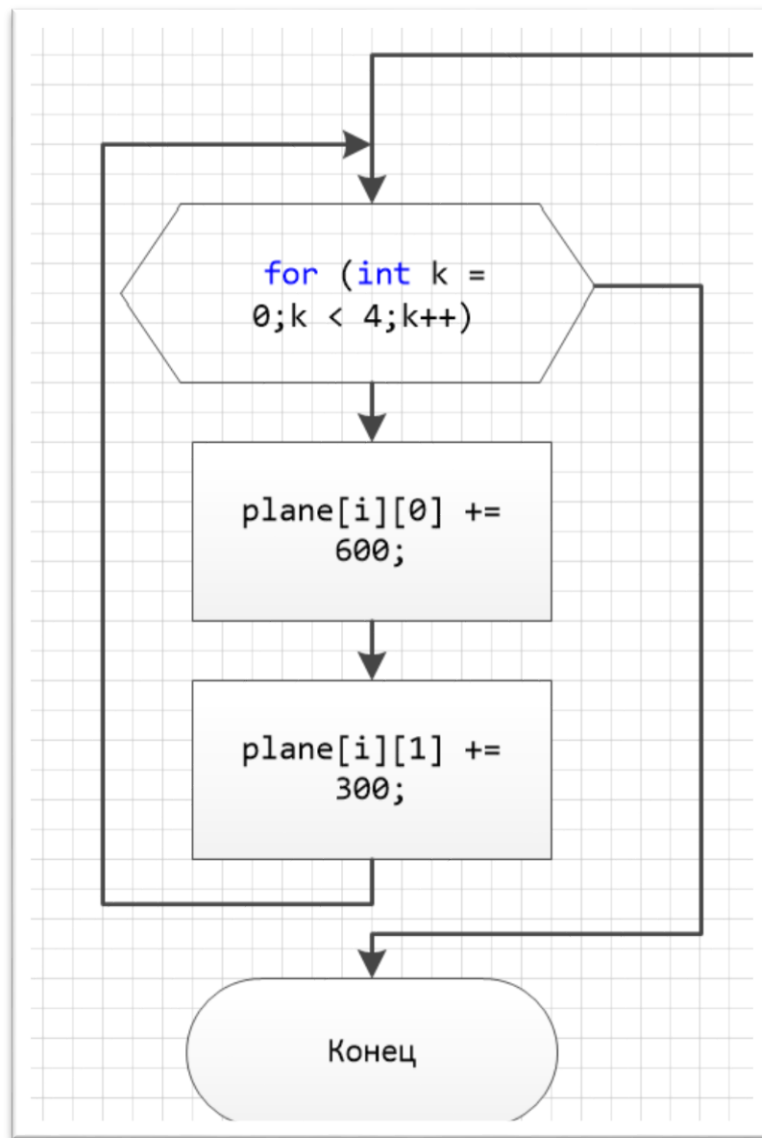


Рисунок 18 – Продолжение рисунок 17

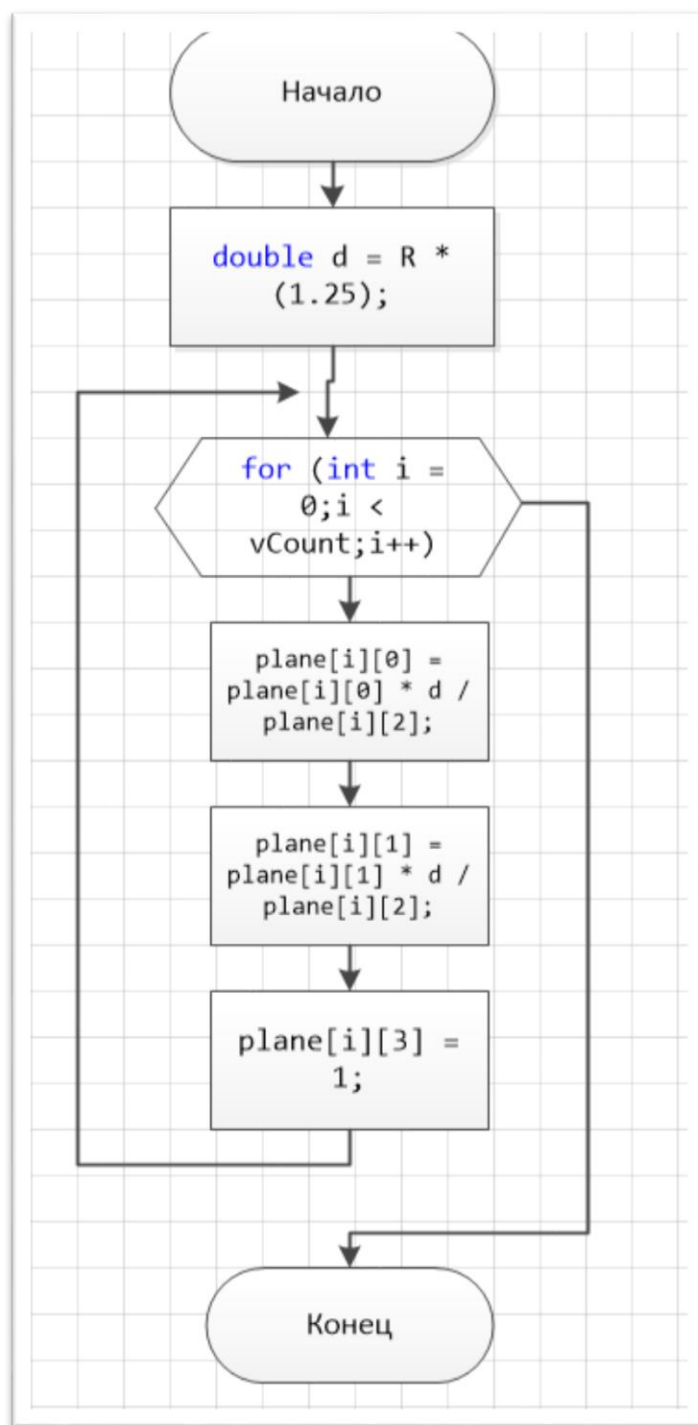


Рисунок 19 – граф схема функции proes()

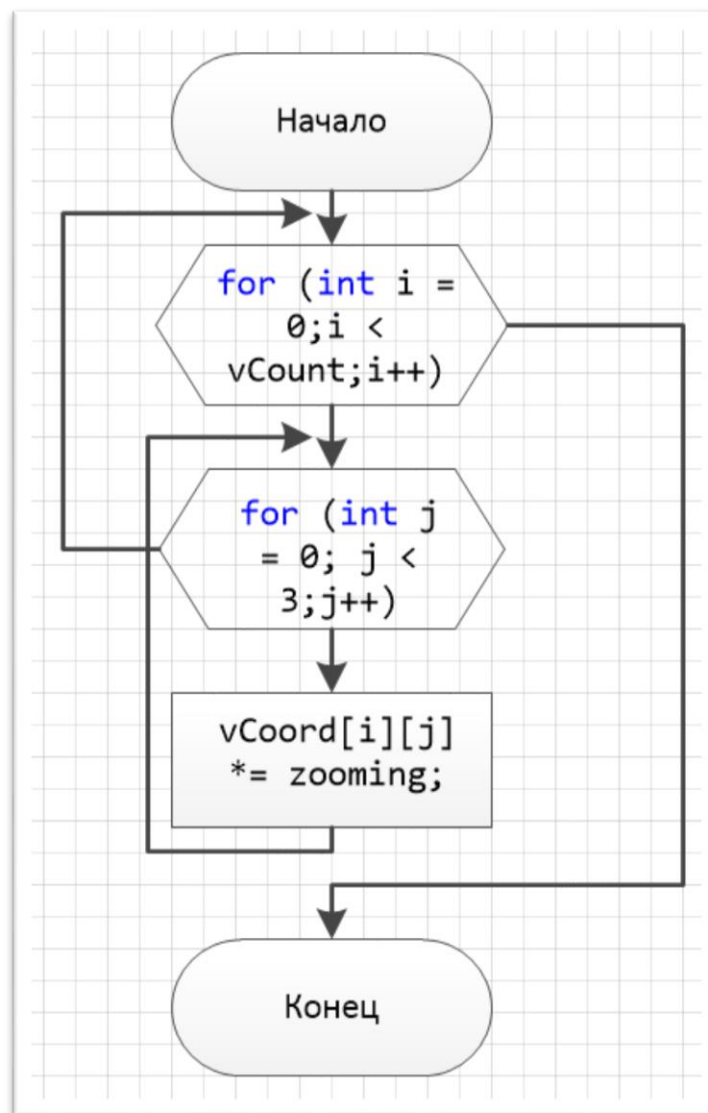


Рисунок 20 – граф схема функции zoom()

2.5 Отладка и тестирование

Управление параметрами динамики объекта производится с помощью клавиатуры

стрелочки - уменьшение/увеличение углов поворота фигуры.

NUMPAD1, NUMPAD2 – уменьшение/увеличение коэффициента масштабирования.

NUMPAD4, NUMPAD5 – включение/выключение функции проекции.

+ - – уменьшение/увеличение расстояния до экранной плоскости.

Выполнение действия	Ожидаемый результат	Полученный результат
Загрузка существующего файла	Импорт файла, рисование модели	Успех
Нажатие клавиш стрелочек	Поворот проекции.	Успех
Нажатие клавиш NUMPAD1 и NUMPAD2	Увеличение, уменьшение объекта	Успех
Нажатие клавиш NUMPAD4 и NUMPAD5	Включение и выключение проекции	Успех
Нажатие клавиш плюс и минус	Приближение и отдаление от объекта	Успех

Таблица 1 - Тестирование

Заключение

В ходе выполнения курсовой работы я закрепил навыки получения отображения трехмерного объекта на экране, получил опыт проектирования 3D моделей твердотельных объектов.

Была разработана программа, осуществляющая построение визуализации 3D модели на языке программирования C++.

Список Литературы

1. Компьютерная графика: Методические указания к лабораторным работам/[Л.С. Курилов]; под ред. Е.Г. Бершадская – Пенза:Издательство Пензенский государственный университет, 2004. – 52с.
2. Blender.org: Справочное руководство Blender 3.1– URL: <https://docs.blender.org/manual/ru/latest/> (дата обращения: 14.02.2022).
3. Геометрическое моделирование/[Голованов Н.В.]; под ред. Л.А. Панюшкина – Москва: Издательство Физико-математической литературы, 2002. – 472с.
4. Конспект лекций: Компьютерная графика/ [Постнов К.В.]; - Москва: Издательство НИУ МГСУ, 2009. – 247с.
5. Конспект лекций: Геометрические преобразования в компьютерной графике/[Косников Ю.Н.] – Пенза: Издательство Пензенский государственный университет, 2011. – 50с.

Приложение 1. Визуализация объекта

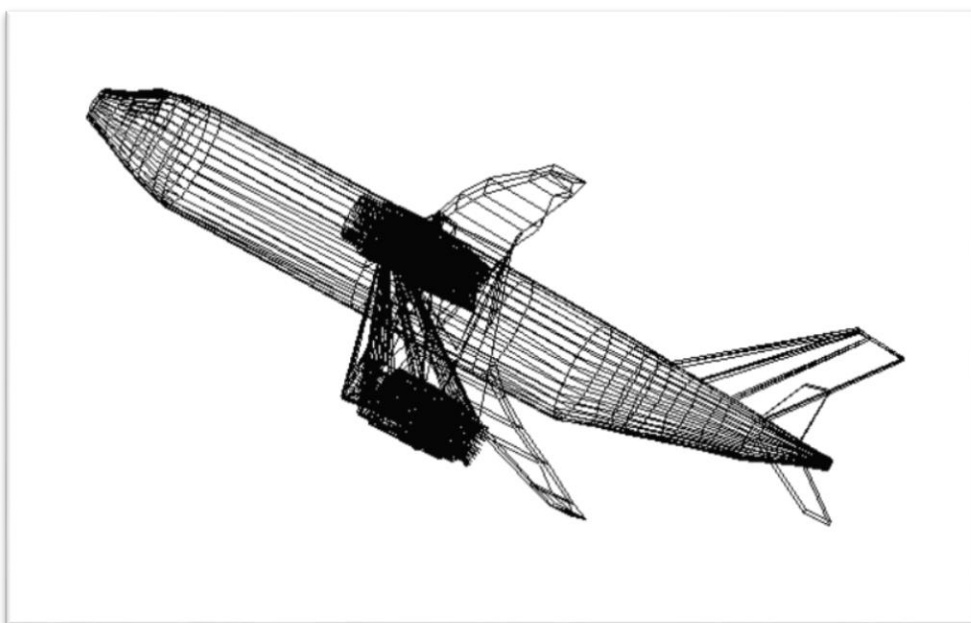


Рисунок 21 – Отрисовка изображения

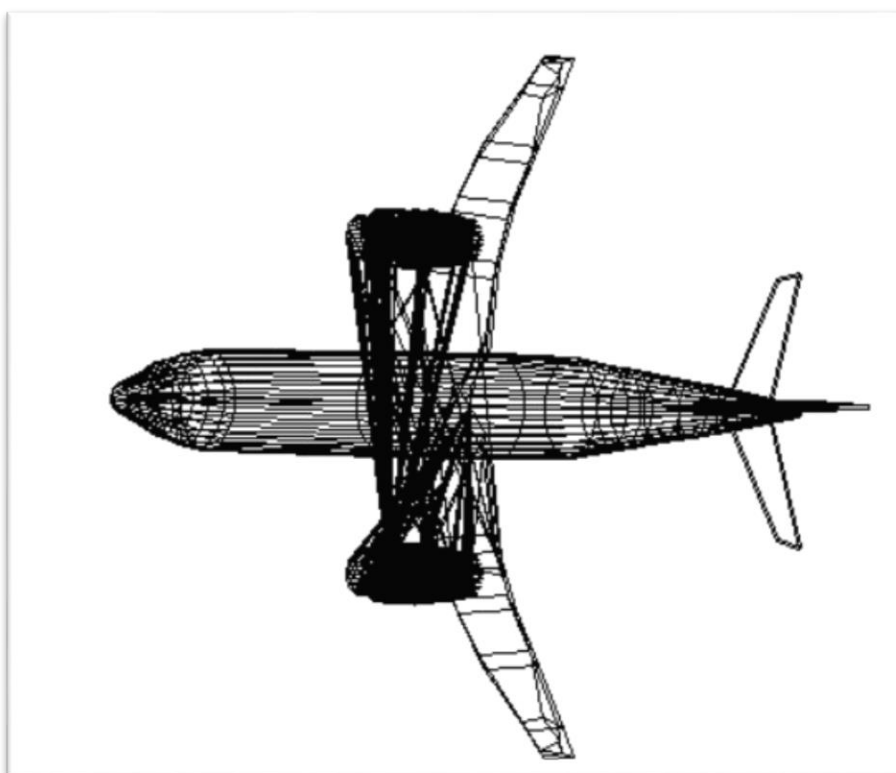


Рисунок 22 – Поворот по оси y

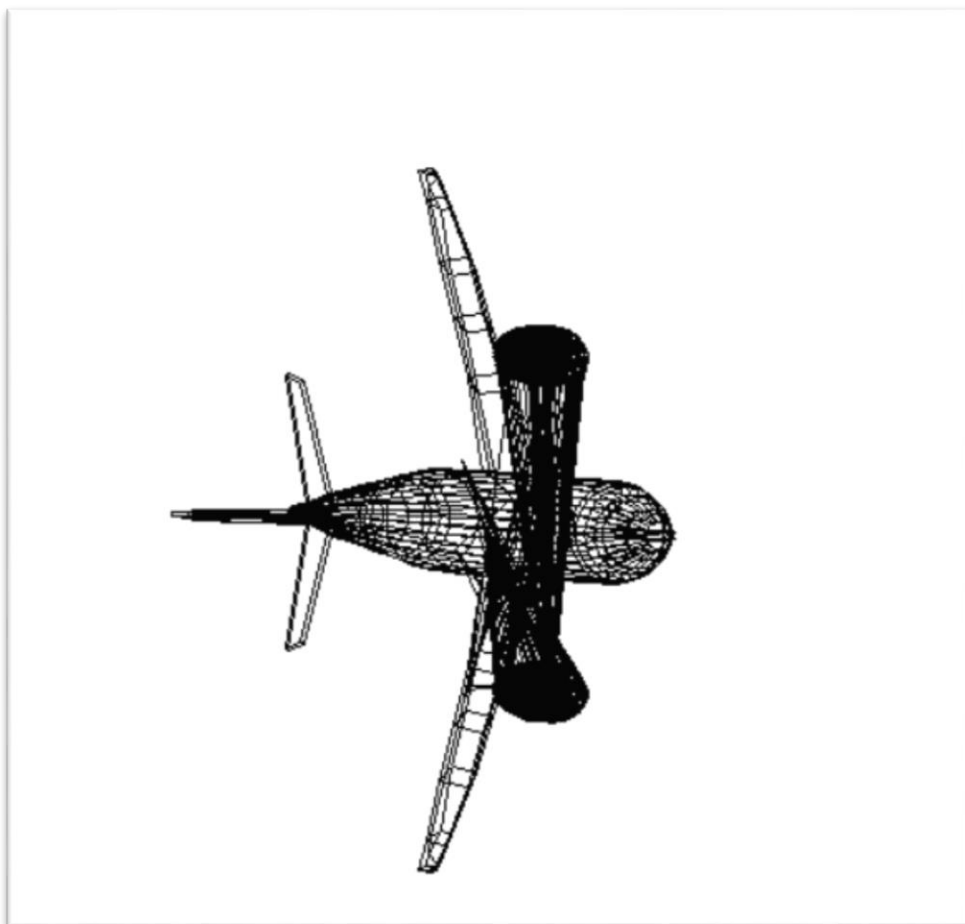


Рисунок 23 – Поворот по оси x



Рисунок 24 – Уменьшение размера изображения

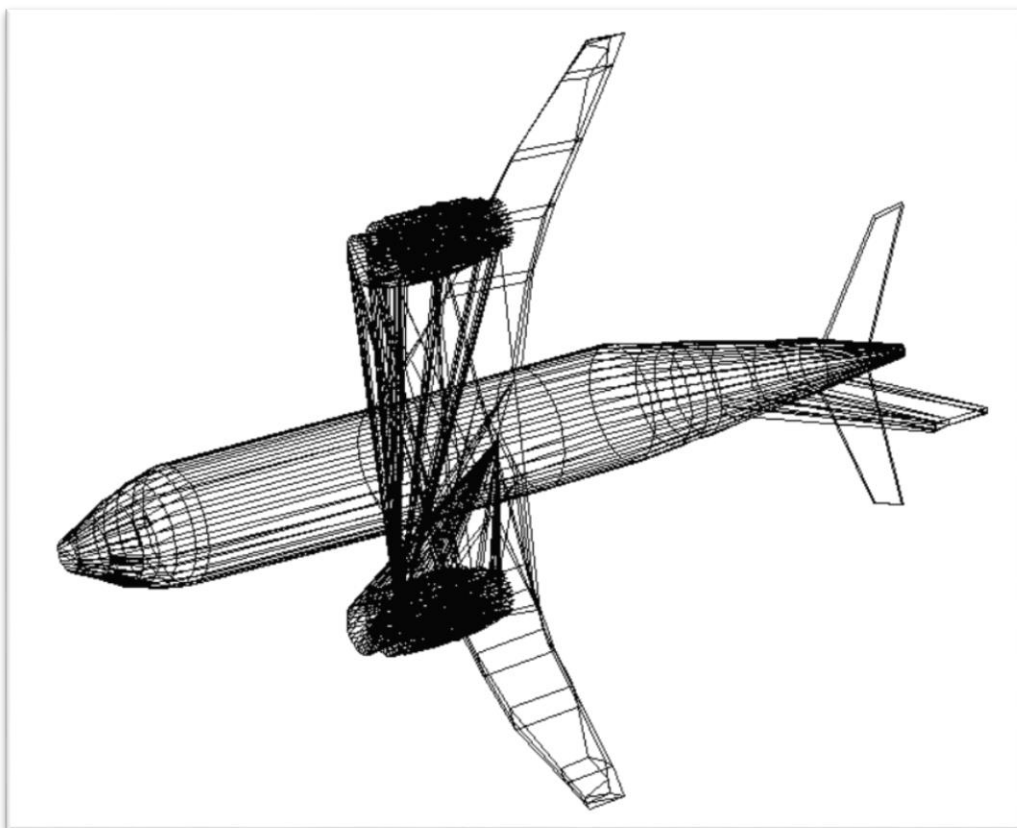


Рисунок 25 – Увеличение размера изображения

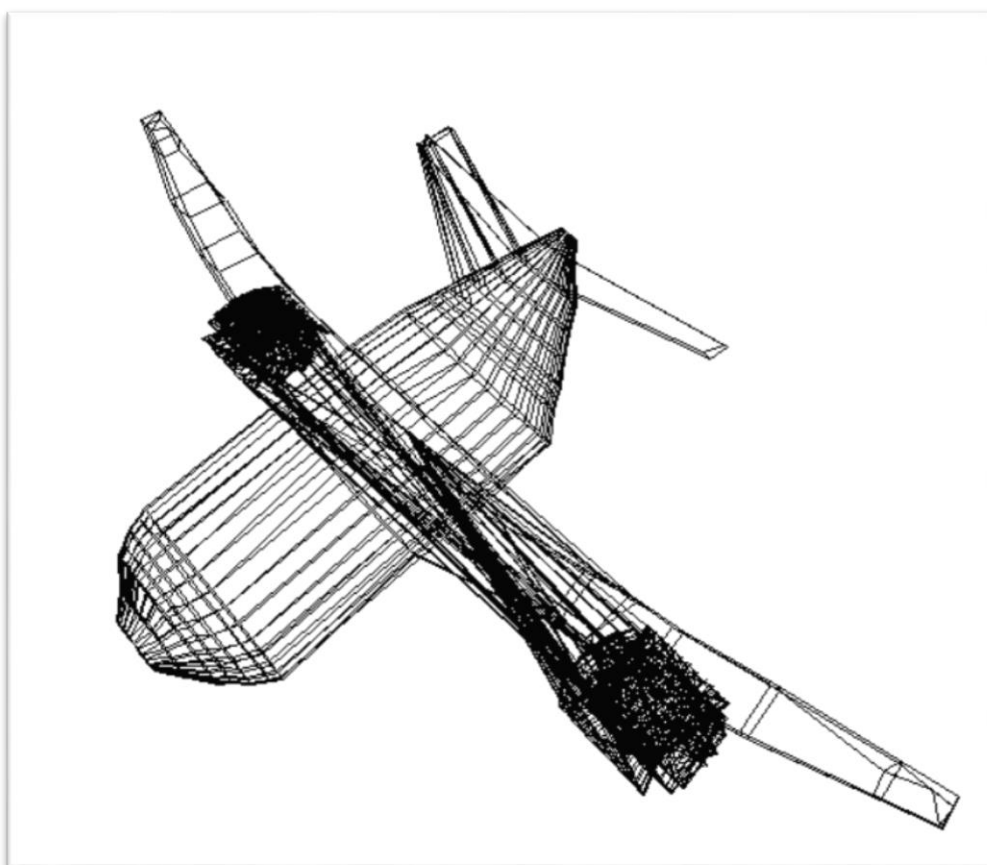


Рисунок 26 – Проекция

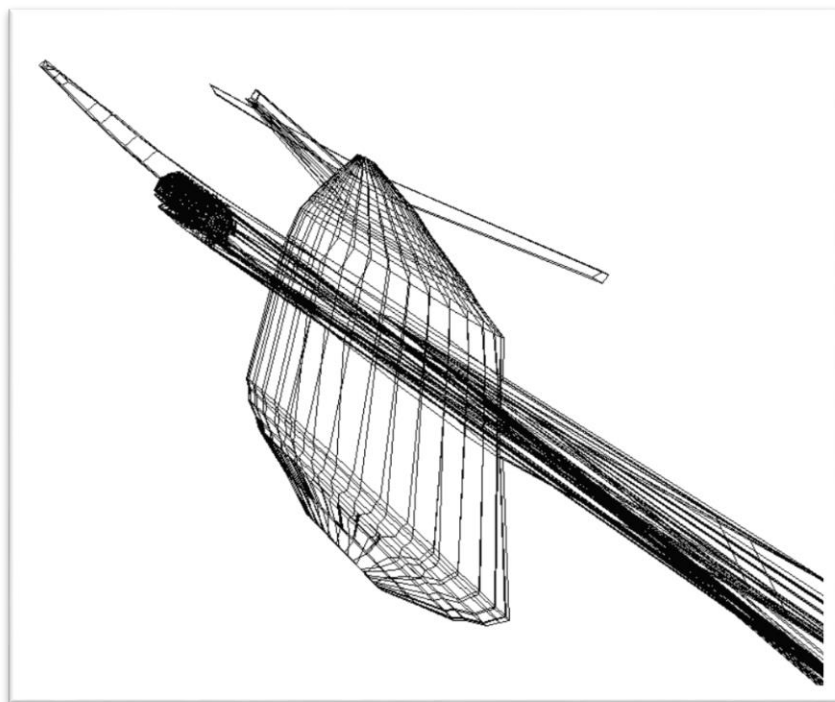


Рисунок 27 – Приближение

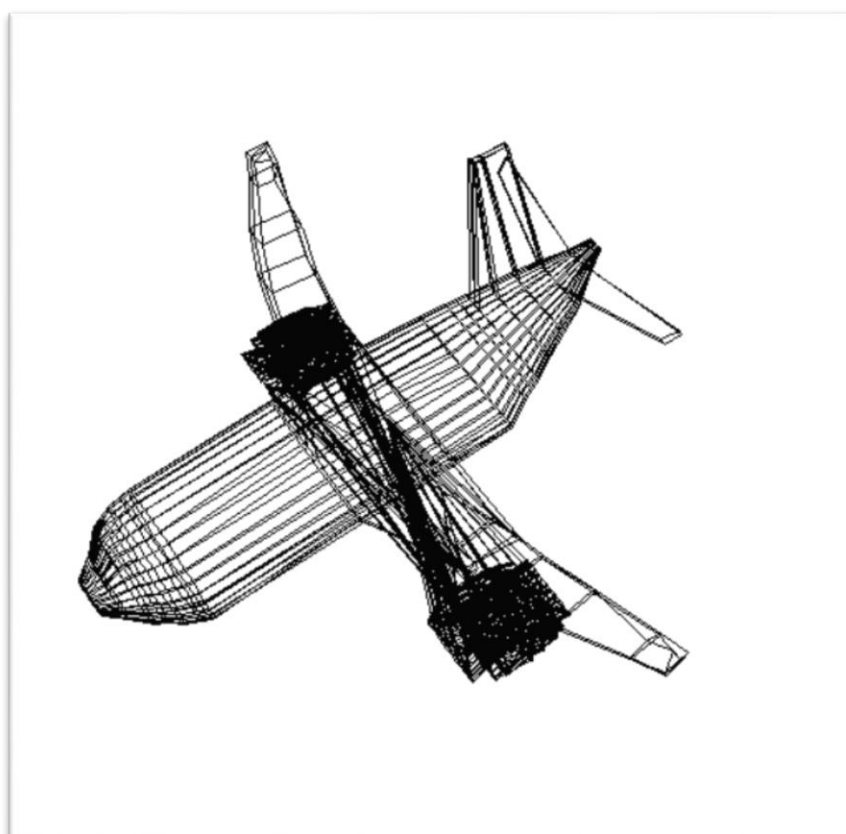


Рисунок 28 – Отдаление

Приложение 2. Листинг

```
// kurs3d.cpp : Определяет точку входа для приложения.  
//
```

```
#include "framework.h"  
#include "kurs3d.h"  
#include <string>  
#include <stdio.h>  
#include <math.h>
```

```
using namespace std;
```

```
#define MAX_LOADSTRING 100  
#pragma warning(disable : 4996)
```

```
const int vCount = 1976, lCount = 3726;  
bool proec_status = false;  
FILE* file;  
int R = 100;  
int zooming = 5;  
float Phi = 90, Theta = 45;  
float vCoord[vCount][4];  
int lCoord[lCount][2];  
float plane[vCount][4];  
int from, to;  
char str[100];
```

```
void read_file() {  
    int i = 0;  
    int j = 0;  
    file = fopen("C:\\Download\\Jet1.obj", "r");  
    while ((fscanf(file, "%c", str) != EOF)) {  
        if (!file)  
            break;  
        if (str[0] == 'v') {  
            for (i; i < vCount; i++) {  
                for (int p = 0; p < 3; p++) {  
                    fscanf_s(file, "%f", &vCoord[i][p]);  
                }  
                break;  
            }  
            i++;  
        }  
        if (str[0] == 'l')  
        {  
            for (j; j < lCount; j++) {
```

```

        for (int k = 0; k < 2; k++) {
            fscanf_s(file, "%d", &lCoord[j][k]);
        }
        break;
    }
    j++;
}
}
fclose(file);
}

void zoom() {
    for (int i = 0; i < vCount; i++)
        for (int j = 0; j < 3; j++)
            vCoord[i][j] *= zooming;
}

void transform() {
    double ViewPoint[4][4] =
    {
        {-sin(Theta), -cos(Phi) * cos(Theta), -sin(Phi) * cos(Theta), 0},
        {cos(Theta), -cos(Phi) * sin(Theta), -sin(Phi) * sin(Theta), 0},
        {0, sin(Phi), cos(Phi), 0},
        {0, 0, R, 0}
    };

    for (int i = 0; i < vCount; i++) {
        vCoord[i][3] = 1;
        for (int j = 0; j < 4; j++) {
            plane[i][j] = 0;
            for (int k = 0; k < 4; k++) {
                plane[i][j] += vCoord[i][k] * ViewPoint[k][j];
            }
        }
    }
    for (int i = 0; i < vCount; i++)
    {
        plane[i][0] += 600;
        plane[i][1] += 300;
    }
};

void proec() {
    double d = R * (1.25);
    for (int i = 0; i < vCount; i++)
    {
        plane[i][0] = plane[i][0] * d / plane[i][2];
        plane[i][1] = plane[i][1] * d / plane[i][2];
        plane[i][3] = 1;
    }
}

```

```

    }
};

// Глобальные переменные:
HINSTANCE hInst;                // текущий экземпляр
WCHAR szTitle[MAX_LOADSTRING]; // Текст строки заголовка
WCHAR szWindowClass[MAX_LOADSTRING]; // имя класса главного окна

// Отправить объявления функций, включенных в этот модуль кода:
ATOM MyRegisterClass(HINSTANCE hInstance);
BOOL InitInstance(HINSTANCE, int);
LRESULT CALLBACK WndProc(HWND, UINT, WPARAM, LPARAM);
INT_PTR CALLBACK About(HWND, UINT, WPARAM, LPARAM);

int APIENTRY wWinMain(_In_ HINSTANCE hInstance,
                     _In_opt_ HINSTANCE hPrevInstance,
                     _In_ LPWSTR lpCmdLine,
                     _In_int_ nCmdShow)
{
    UNREFERENCED_PARAMETER(hPrevInstance);
    UNREFERENCED_PARAMETER(lpCmdLine);

    // TODO: Разместите код здесь.

    // Инициализация глобальных строк
    LoadStringW(hInstance, IDS_APP_TITLE, szTitle, MAX_LOADSTRING);
    LoadStringW(hInstance, IDC_KURS3D, szWindowClass, MAX_LOADSTRING);
    MyRegisterClass(hInstance);

    // Выполнить инициализацию приложения:
    if (!InitInstance (hInstance, nCmdShow))
    {
        return FALSE;
    }

    HACCEL hAccelTable = LoadAccelerators(hInstance,
    MAKEINTRESOURCE(IDC_KURS3D));

    MSG msg;

    // Цикл основного сообщения:
    while (GetMessage(&msg, nullptr, 0, 0))
    {
        if (!TranslateAccelerator(msg.hwnd, hAccelTable, &msg))
        {
            TranslateMessage(&msg);
            DispatchMessage(&msg);
        }
    }
}

```

```

    return (int) msg.wParam;
}

//
// ФУНКЦИЯ: MyRegisterClass()
//
// ЦЕЛЬ: Регистрирует класс окна.
//
ATOM MyRegisterClass(HINSTANCE hInstance)
{
    WNDCLASSEXW wcex;

    wcex.cbSize = sizeof(WNDCLASSEX);

    wcex.style      = CS_HREDRAW | CS_VREDRAW;
    wcex.lpfnWndProc = WndProc;
    wcex.cbClsExtra  = 0;
    wcex.cbWndExtra  = 0;
    wcex.hInstance   = hInstance;
    wcex.hIcon        = LoadIcon(hInstance, MAKEINTRESOURCE(IDI_KURS3D));
    wcex.hCursor      = LoadCursor(nullptr, IDC_ARROW);
    wcex.hbrBackground = (HBRUSH)(COLOR_WINDOW+1);
    wcex.lpszMenuName = MAKEINTRESOURCEW(IDC_KURS3D);
    wcex.lpszClassName = szWindowClass;
    wcex.hIconSm      = LoadIcon(wcex.hInstance, MAKEINTRESOURCE(IDI_SMALL));

    return RegisterClassExW(&wcex);
}

//
// ФУНКЦИЯ: InitInstance(HINSTANCE, int)
//
// ЦЕЛЬ: Сохраняет маркер экземпляра и создает главное окно
//
// КОММЕНТАРИИ:
//
// В этой функции маркер экземпляра сохраняется в глобальной переменной, а также
// создается и выводится главное окно программы.
//
BOOL InitInstance(HINSTANCE hInstance, int nCmdShow)
{
    hInst = hInstance; // Сохранить маркер экземпляра в глобальной переменной

    HWND hWnd = CreateWindowW(szWindowClass, szTitle, WS_OVERLAPPEDWINDOW,
        CW_USEDEFAULT, 0, CW_USEDEFAULT, 0, nullptr, nullptr, hInstance, nullptr);

    if (!hWnd)
    {
        return FALSE;
    }
}

```

```

    }

    ShowWindow(hWnd, nCmdShow);
    UpdateWindow(hWnd);

    return TRUE;
}

//
// ФУНКЦИЯ: WndProc(HWND, UINT, WPARAM, LPARAM)
//
// ЦЕЛЬ: Обрабатывает сообщения в главном окне.
//
// WM_COMMAND - обработать меню приложения
// WM_PAINT - Отрисовка главного окна
// WM_DESTROY - отправить сообщение о выходе и вернуться
//
//
LRESULT CALLBACK WndProc(HWND hWnd, UINT message, WPARAM wParam,
LPARAM lParam)
{
    switch (message)
    {
    case WM_COMMAND:
        {
            int wmId = LOWORD(wParam);
            // Разобрать выбор в меню:
            switch (wmId)
            {
            case IDM_ABOUT:
                DialogBox(hInst, MAKEINTRESOURCE(IDD_ABOUTBOX), hWnd, About);
                break;
            case IDM_EXIT:
                DestroyWindow(hWnd);
                break;
            default:
                return DefWindowProc(hWnd, message, wParam, lParam);
            }
        }
        break;
    case WM_KEYDOWN:
        {
            int wmId = LOWORD(wParam);
            // Разобрать выбор в меню:
            switch (wmId)
            {
            case VK_NUMPAD1:
                zooming *= 1.5;
                zoom();
                transform();
                if (proec_status)
                    proec();
            }
        }
    }
}

```

```

        InvalidateRect(hWnd, NULL, TRUE);
        UpdateWindow(hWnd);
        break;
case VK_NUMPAD2:
    zooming /= 2;
    zoom();
    transform();
    if (proec_status)
        proec();
    InvalidateRect(hWnd, NULL, TRUE);
    UpdateWindow(hWnd);
    break;
case VK_UP: // Поворот вверх
    Phi -= 0.5;
    zoom();
    transform();
    if (proec_status)
        proec();
    InvalidateRect(hWnd, NULL, TRUE);
    UpdateWindow(hWnd);
    break;
case VK_DOWN:// Поворот вниз
    Phi += 0.5;
    zoom();
    transform();
    if (proec_status)
        proec();
    InvalidateRect(hWnd, NULL, TRUE);
    UpdateWindow(hWnd);
    break;
case VK_LEFT: // Поворот влево
    Theta -= 0.5;
    zoom();
    transform();
    if (proec_status)
        proec();
    InvalidateRect(hWnd, NULL, TRUE);
    UpdateWindow(hWnd);
    break;
case VK_RIGHT: // Поворот вправо
    Theta += 0.5;
    zoom();
    transform();
    if (proec_status)
        proec();
    InvalidateRect(hWnd, NULL, TRUE);
    UpdateWindow(hWnd);
    break;

case VK_NUMPAD4: // Включение проекции
    proec_status = true;

```

```

        zoom();
        transform();
        proec();
        InvalidateRect(hWnd, NULL, TRUE);
        UpdateWindow(hWnd);
        break;
case VK_NUMPAD5: // Выключение проекции
    proec_status = false;
    zoom();
    transform();
    InvalidateRect(hWnd, NULL, TRUE);
    UpdateWindow(hWnd);
    break;

case VK_ADD: // Отдаление от проекции
    zoom();
    transform();
    if (proec_status)
        proec();
    R += 10;
    InvalidateRect(hWnd, NULL, TRUE);
    UpdateWindow(hWnd);
    break;
case VK_SUBTRACT: // Приближение к проекции;
    zoom();
    transform();
    if (proec_status)
        proec();
    R -= 10;
    InvalidateRect(hWnd, NULL, TRUE);
    UpdateWindow(hWnd);
    break;
    }
}
case WM_PAINT:
{
    PAINTSTRUCT ps;
    HDC hdc = BeginPaint(hWnd, &ps);
    read_file();

    for (int i = 0; i < lCount; i++)
    {
        from = lCoord[i][0];
        to = lCoord[i][1];
        MoveToEx(hdc, plane[from][0], plane[from][1], 0);
        LineTo(hdc, plane[to][0], plane[to][1]);
    }

    // TODO: Добавьте сюда любой код прорисовки, использующий HDC...
    EndPaint(hWnd, &ps);
}

```

```

        break;
    case WM_DESTROY:
        PostQuitMessage(0);
        break;
    default:
        return DefWindowProc(hWnd, message, wParam, lParam);
    }
    return 0;
}

// Обработчик сообщений для окна "О программе".
INT_PTR CALLBACK About(HWND hDlg, UINT message, WPARAM wParam, LPARAM
lParam)
{
    UNREFERENCED_PARAMETER(lParam);
    switch (message)
    {
    case WM_INITDIALOG:
        return (INT_PTR)TRUE;

    case WM_COMMAND:
        if (LOWORD(wParam) == IDOK || LOWORD(wParam) == IDCANCEL)
        {
            EndDialog(hDlg, LOWORD(wParam));
            return (INT_PTR)TRUE;
        }
        break;
    }
    return (INT_PTR)FALSE;
}

```