Министерство образования Российской Федерации

ПЕНЗЕНСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ

Кафедра «Системы автоматизированного проектирования»

**ПОЯСНИТЕЛЬНАЯ ЗАПИСКА**

К курсовой работе

по курсу «Инженерная и компьютерная графика»

на тему «Разработка программы построения каркасной модели трехмерного объекта»

Выполнили:

студенты группы 14ВВ2

Матюнин А.Р.

Пехтерева Д.А.

Приняли:

Кандидат технических наук, доцент

Финогеев Антон Алексеевич

Пенза 2015

**Аннотация**

Данный курсовой проект включает в себя интерактивную программу на языке Cи, реализующую процесс считывания информации об объекте из файла .obj, а также построения и визуализации трехмерного объекта, его закраску и экранные преобразования.

Оглавление

[1. Основная часть 3](#_Toc438460422)

[2. Общие сведения 3](#_Toc438460423)

[3. Функциональное назначение 3](#_Toc438460424)

[4. Описание логической структуры 5](#_Toc438460425)

[4.1. Алгоритм рисования линии в функции rastr 6](#_Toc438460426)

[3.2 Алгоритм преобразования координат в функции viewport\_transformation 7](#_Toc438460427)

[3.3 Алгоритм преобразования координат в функции perspective\_transformation 8](#_Toc438460428)

[3.4 Алгоритм преобразования координат в функции Screen\_conversion или построения каркасной модели объекта 9](#_Toc438460429)

[3.5 Алгоритм закраски каркасной модели 11](#_Toc438460430)

[1. Разработка 3D объекта 12](#_Toc438460431)

[2. Руководство пользователя 13](#_Toc438460432)

[3. Заключение 23](#_Toc438460433)

[4. Список использованной литературы 24](#_Toc438460434)

[5. Приложение 25](#_Toc438460435)

[5.1 Приложение А 25](#_Toc438460436)

[5.2 Приложение Б(Файл Unit1.h) 41](#_Toc438460437)

[5.3 Приложение В(Файл Unit1.cpp) 43](#_Toc438460438)

# Основная часть

## Общие сведения

Разработанный проект программы имеет название Project1, а сама программа Project1.exe. Для полноценного функционирования программы необходимо иметь файл формата .obj, который хранит информацию о вершинах и полигонах объекта (информация о нормалях и текстурных картах объекта необязательна). Программа написана на языке Cи в интегрированной среде разработчика С++ Builder 6, библиотеки которой также потребуются для запуска данной программы. Текст программы приведён в приложении.

## Функциональное назначение

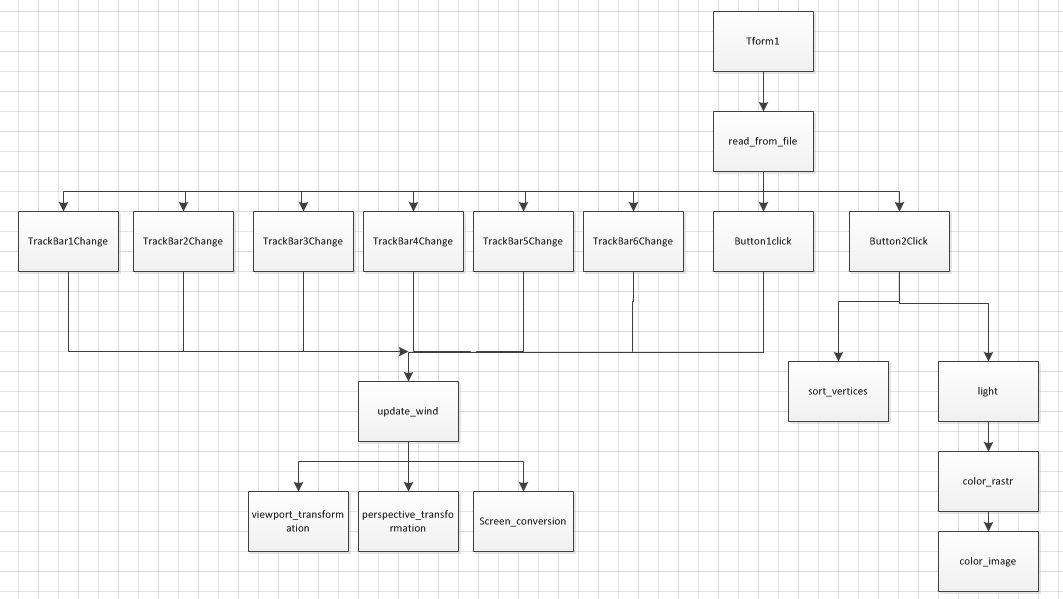
Программа реализует процесс построения и визуализации трехмерного объекта (модель кружки) с элементами динамики (вращение объекта по различным осям).

|  |  |
| --- | --- |
| **Название функции** | **Назначение** |
| void \_\_fastcall TForm1::Button1Click(TObject \*Sender) | Функция кнопки, вызывающее диалоговое окно выбора файла с последующим вызовом функции считывания информации с файла. |
| void \_\_fastcall TForm1::TrackBar1Change(TObject \*Sender) | Изменение смещения фигуры по оси ОХ |
| void \_\_fastcall TForm1::TrackBar2Change(TObject \*Sender) | Изменение смещения фигуры по оси ОУ |
| void \_\_fastcall TForm1::TrackBar3Change(TObject \*Sender) | Изменение расстояния до объекта |
| void \_\_fastcall TForm1::TrackBar4Change(TObject \*Sender) | Изменение угла θ |
| void \_\_fastcall TForm1::TrackBar5Change(TObject \*Sender) | Изменение угла φ |
| void \_\_fastcall TForm1::TrackBar6Change(TObject \*Sender) | Изменение параметра object size или размер объекта |
| void read\_from\_file(char\* MyFile) | Функция считывания с файла информации об объекте |
| void update\_wind() | Функция, вызываемая при каждом изменении одного из параметров каждого TrackBar и вызывающая функции преобразований координат объекта |
| void viewport\_transformation(vert v[1000],vert res[1000],int R,int teta,int phi) | Выполнение видовых преобразований |
| float perspective\_transformation(int R,int Width,int Height,int object\_size,vert res[1000],vert v[1000]) | Выполнение перспективных преобразований |
| void Screen\_conversion(vert res[1000],indexs index[1000],float d,int smech\_x,int smech\_y) | Выполнение экранных преобразований |
| void rastr(float fx1,float fy1,float fx2,float fy2) | Соединение двух точек между собой посредством алгоритма разложения в растр |
| void sort\_vertices(vert res[1000],indexs index[1000]) | Сортировка вершин и их индексов по среднему значению Z |
| void \_\_fastcall TForm1::Button2Click(TObject \*Sender) | Функция кнопки, выполняющая вызов функций заливки объекта. |
| void light(vert res[1000],indexs index[1000]) | Нахождение нормалей (расчет освещения каждого полигона) |
| void color\_image(float cos\_phi,float cos\_teta,struct vert res[1000],int i,int j,int s) | Заливка полигонов объекта согласно найденным нормалям |
| int color\_rastr(float fx1,float fy1,float fx2,float fy2,int k) | Функция, выполняющая разложение в растр с поиском изменившихся координат Y и запись пар X,Y в структуру. Возвращает k- кол-во записанных пар. |

## Описание логической структуры

При разработке программы использовалось принципы модульного и структурного программирования, суть которых заключается в том, что программа разбивается на отдельные функциональные части – подпрограммы, которые распределяются по отдельным уровням иерархии. Каждая из подпрограмм решает только свою небольшую задачу по преобразованию данных, что позволяет упростить процесс написания и отладки программы в целом. Основная идея структурного программирования состоит в том, что программа любой сложности состоит из структур следования, разветвления и повторения, т.е. линейных, условных и циклических.

Структурная схема программы



## Алгоритм рисования линии в функции rastr

Используется алгоритм простого ЦДА. В простом ЦДА нам должны быть известны координаты двух наших точек.

Обозначим:

∆x – шаг при построении по оси ОХ.

∆y - шаг при построении по оси OY.

LX – длина отрезка. LY – высота отрезка.

N – кол-во итераций.

Алгоритм простого ЦДА имеет вид:

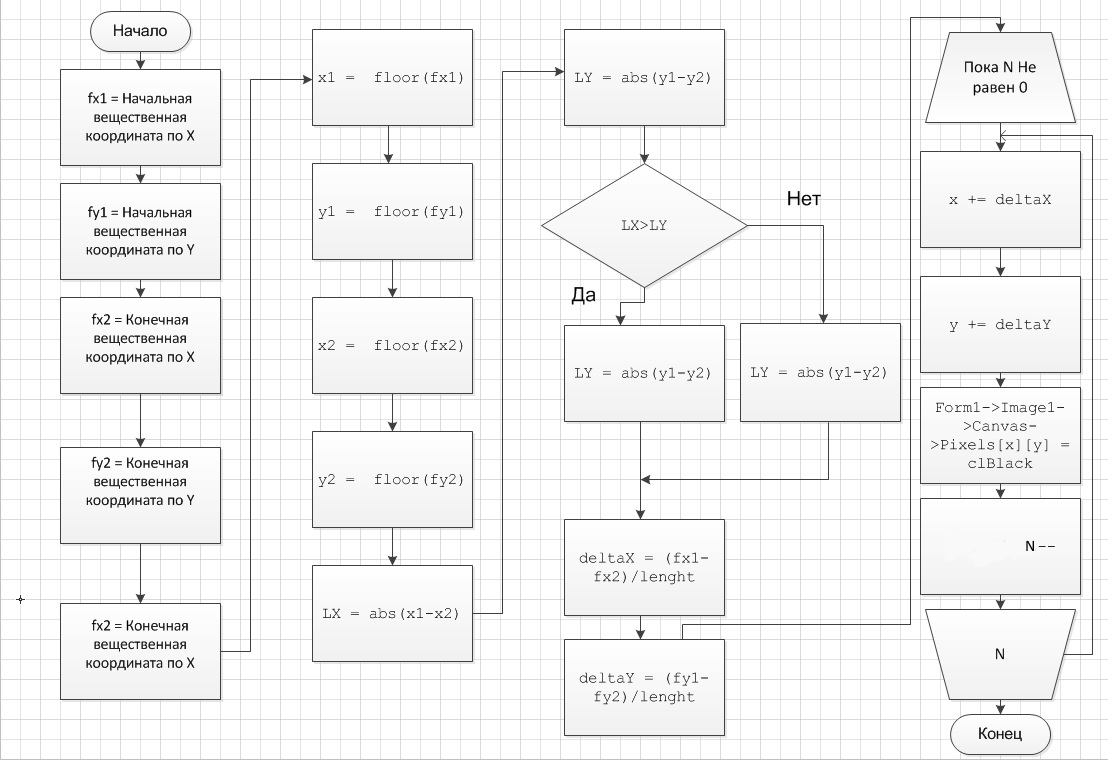
1. Находим длину и высоту будущего отрезка:

1. Если длина отрезка больше высоты, то количество итераций берем равное длине, и наоборот:
2. Находим шаг для координаты Х и Y:
3. Повторяем действия N раз:

4.1)

4.2) Закрашиваем на экране пиксел с координатами ().

4.3) Кол-во итераций уменьшаем на 1.

****

## Алгоритм преобразования координат в функции viewport\_transformation

В данной функции реализовано видовое преобразование. А именно реализовано следующее перемножение матриц:

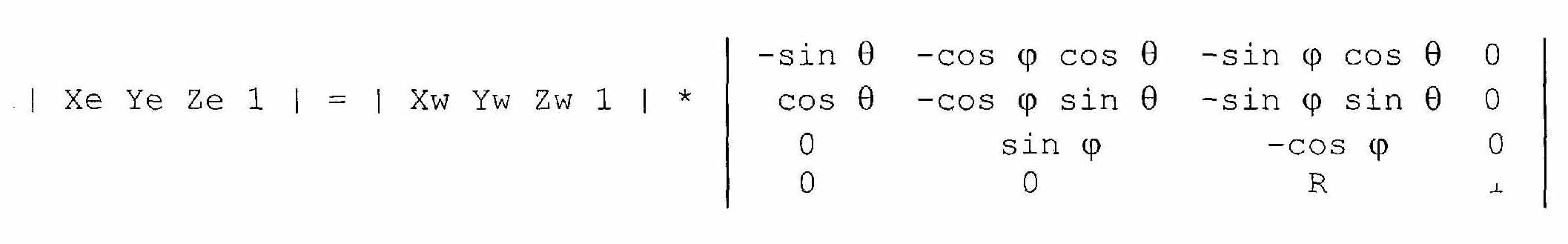
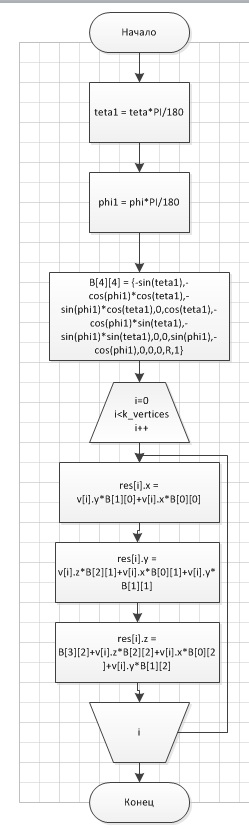


Схема выполнения видового преобразования:

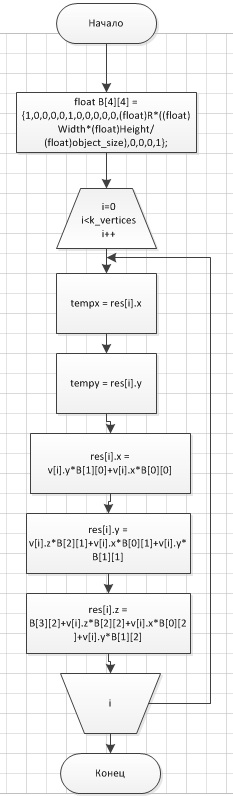
****

## Алгоритм преобразования координат в функции perspective\_transformation

В данной функции реализовано перспективное преобразование. А именно реализовано следующее перемножение матриц:



Схема выполнения перспективного преобразования:

****

## Алгоритм преобразования координат в функции Screen\_conversion или построения каркасной модели объекта

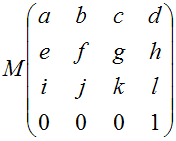
В данной функции реализовано экранное преобразование и также аффинное преобразование. Экранное преобразование осуществляет коррекцию координат (X,Y) в (Xs,Ys) с учетом таких параметров, как текущее разрешение графического дисплея, сдвига по осям, направления оси ординат.

Сначала мы получаем координаты нашего объекта:

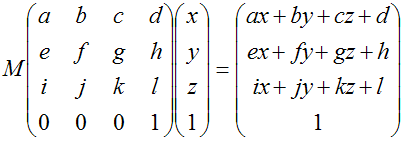
В нашем случае r заменяется на d = r \* ( <размер изображения> / <размер объекта>).

Также были выполнено аффинное преобразования перемещения:

Преобразование M точки T(*x,y,z*), где

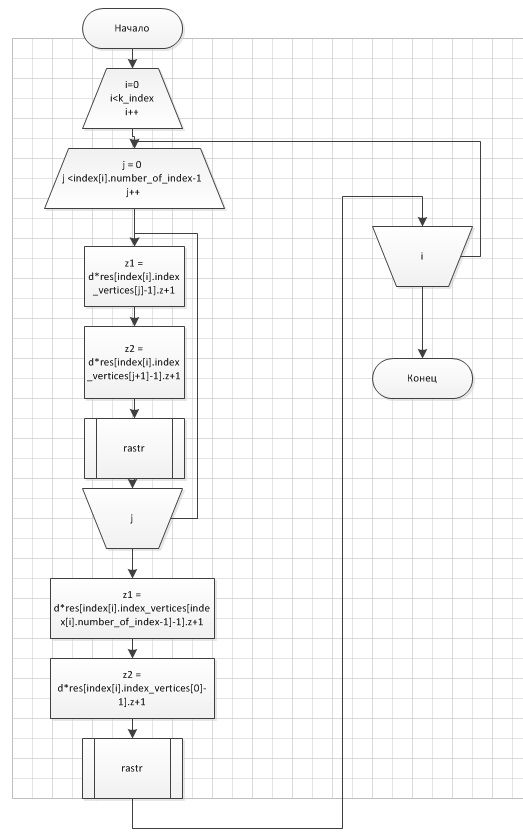


является произведение



Значит вид матрицы:

Схема выполнения экранного преобразования:

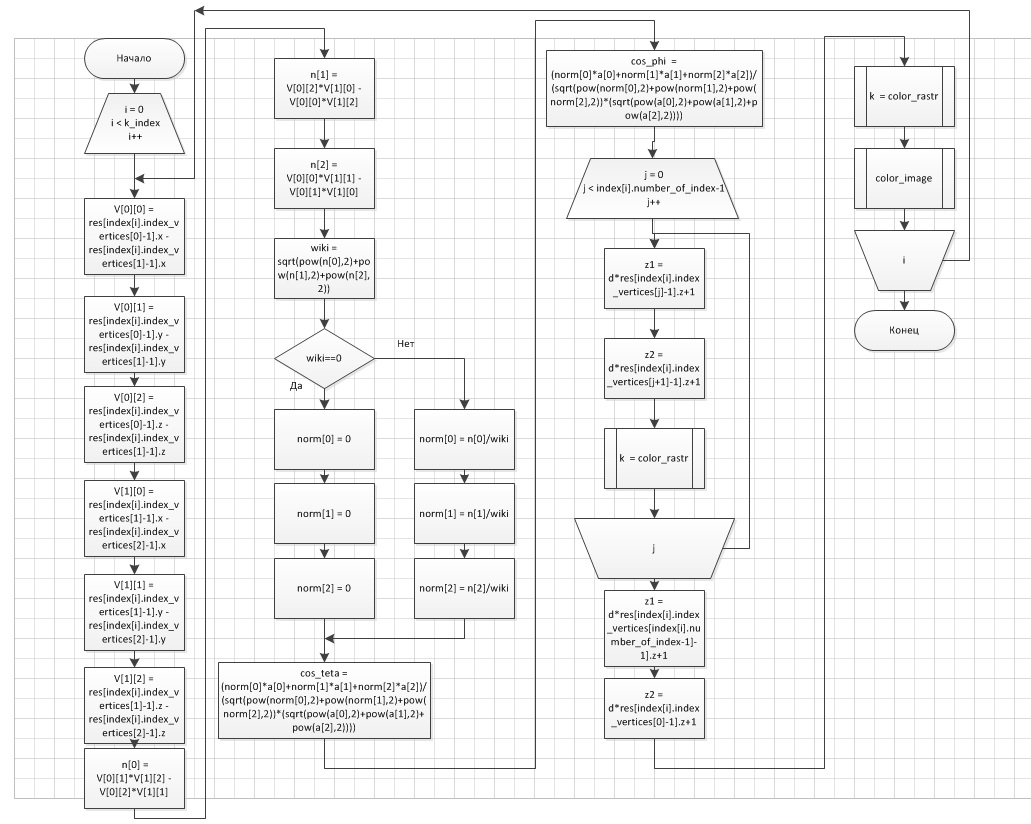


## Алгоритм закраски каркасной модели

Принцип работы алгоритма:

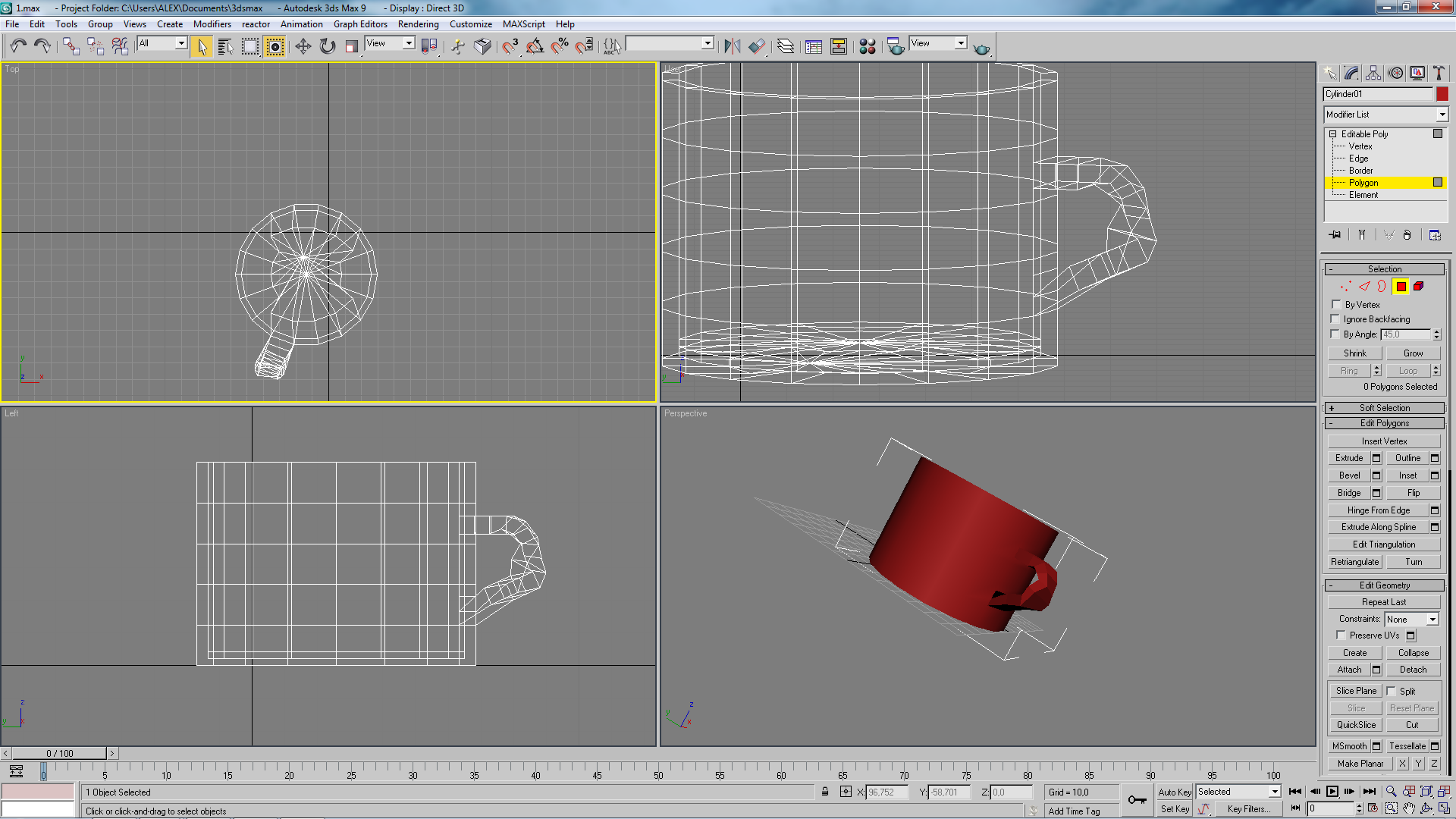
Задаем цикл от первого полигона до последнего. Во время работы этого цикла мы вычисляем нормали для этого полигона согласно координатам вершин в этом полигоне. После просчета нормалей вычисляем углы, которые будут характеризовать положение фигуры относительно источника освещения. Создаем внутренний цикл, который позволит нам пройти по всем сторонам данной фигуры. Вызываем функцию color\_rastr(), она выполняет разложение в растр нашей фигуры и ищет при выполнении алгоритма изменившиеся значения Y. Если Y был изменен, то пара Х и Y заносится в структуру coord. Далее вызывается функция color\_image(). Эта функция ищет такие пары, где Y совпадают, а затем проводит линию от одного Х до другого, при этом цвет выбирается палитрой RGB с учетом косинуса угла, который характеризует степень освещенности объекта относительно источника света.

Схема выполнения закраски каркасной модели:



# Разработка 3D объекта

Для данного курсового проекта был создан 3D объект "Кружка". Для создания использовались следующие программные средства: 3DS Max 9. Сначала была создана основа из цилиндра. Задали кол-во делений верхней части равным 2-м. Расширили верхний внутренний контур так, чтобы получилось приблизительное изображение дна, как у настоящей кружки. После этого с помощью редактора вершин вырезали небольшие участи в цилиндре под ручки кружки и постепенно из одной части стали вести и поворачивать будущую ручку кружки. В результате получилась 3D модель:

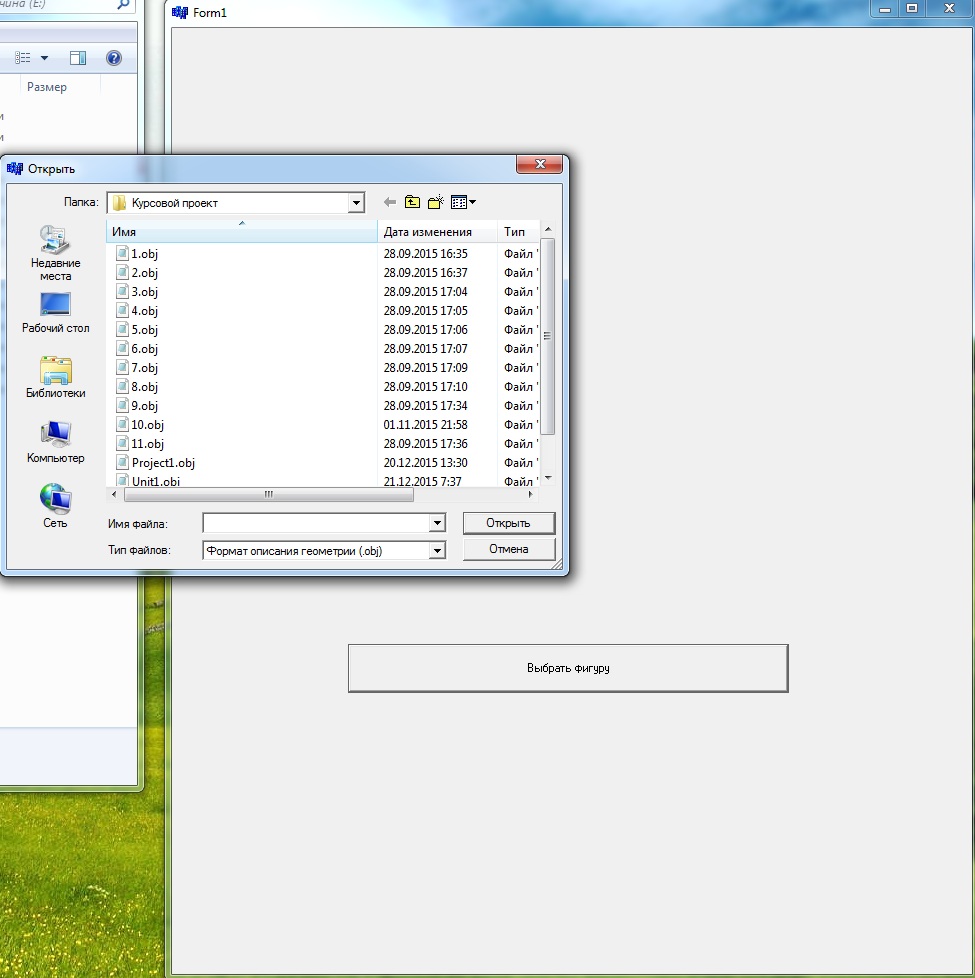


# Руководство пользователя

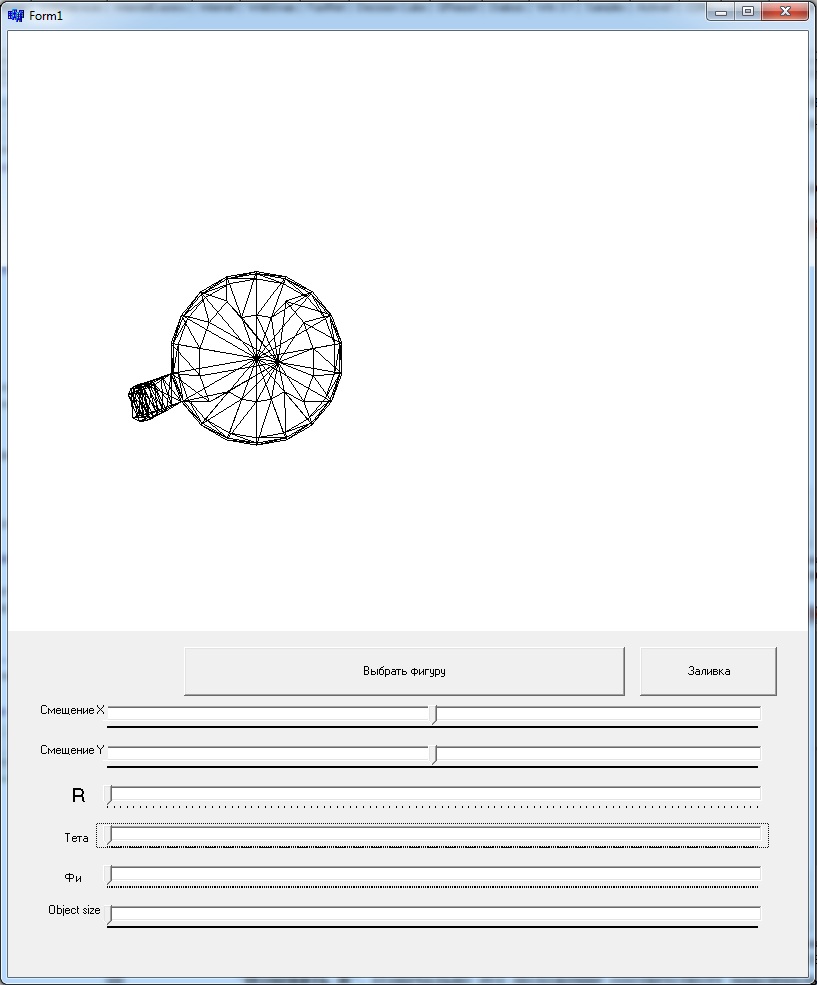
Данная программа предназначена для просмотра 3D - объектов. При запуске вы будете наблюдатель следующий интерфейс:



Для того , что бы начать пользоваться программой нажмите кнопку **"Выбрать фигуру"**. После этого вы увидите диалоговое окно для открытия файлов формата .obj .

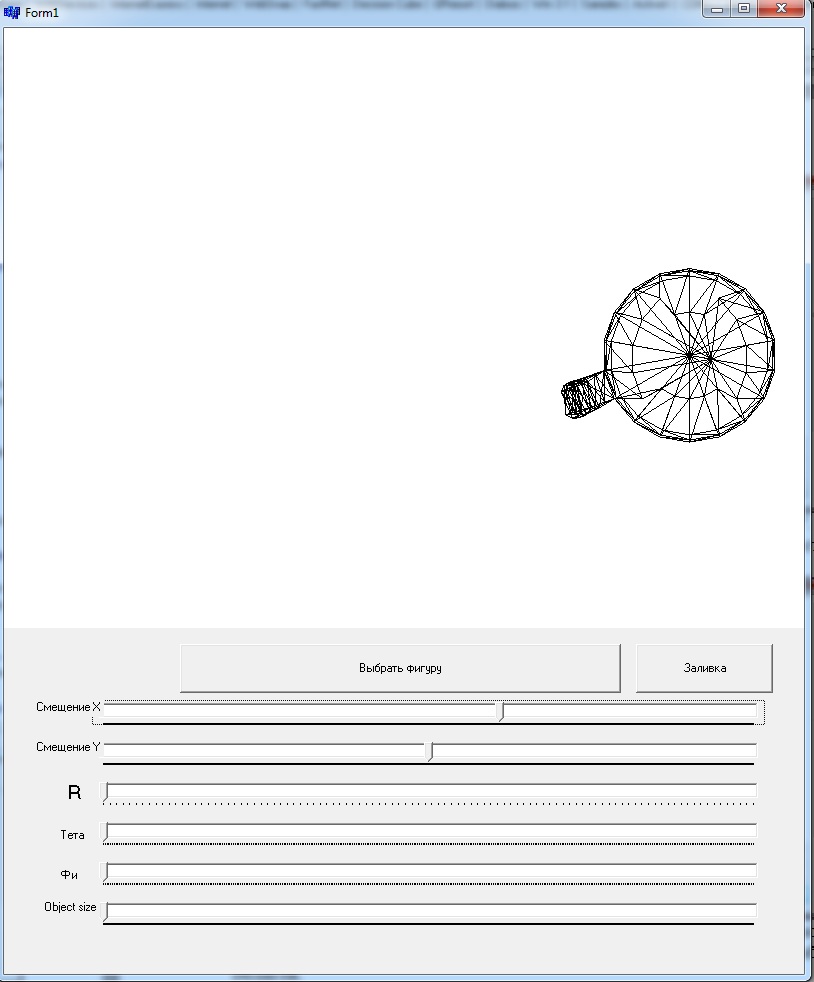


После открытия файла данные будут считаны, и вы увидите на экране каркасную модель вашего объекта, при этом станут доступны все функции программы:

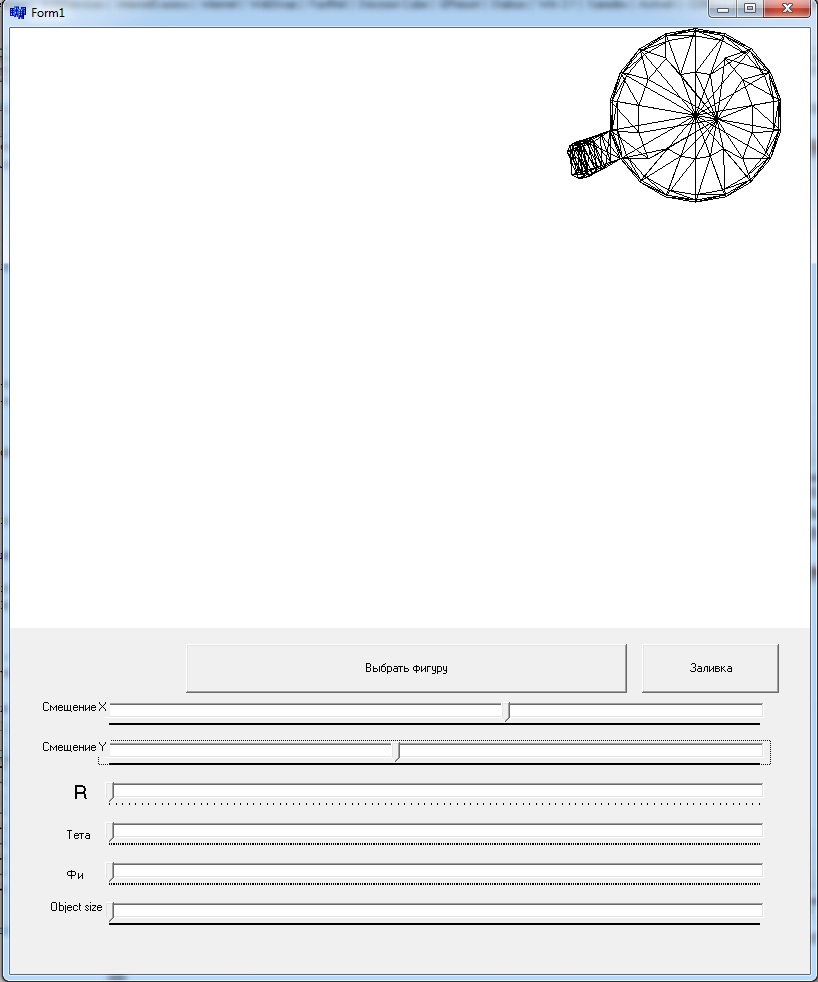


В программе есть возможность использования специальных ползунков, которые позволяют пользователю взаимодействовать с фигурой.

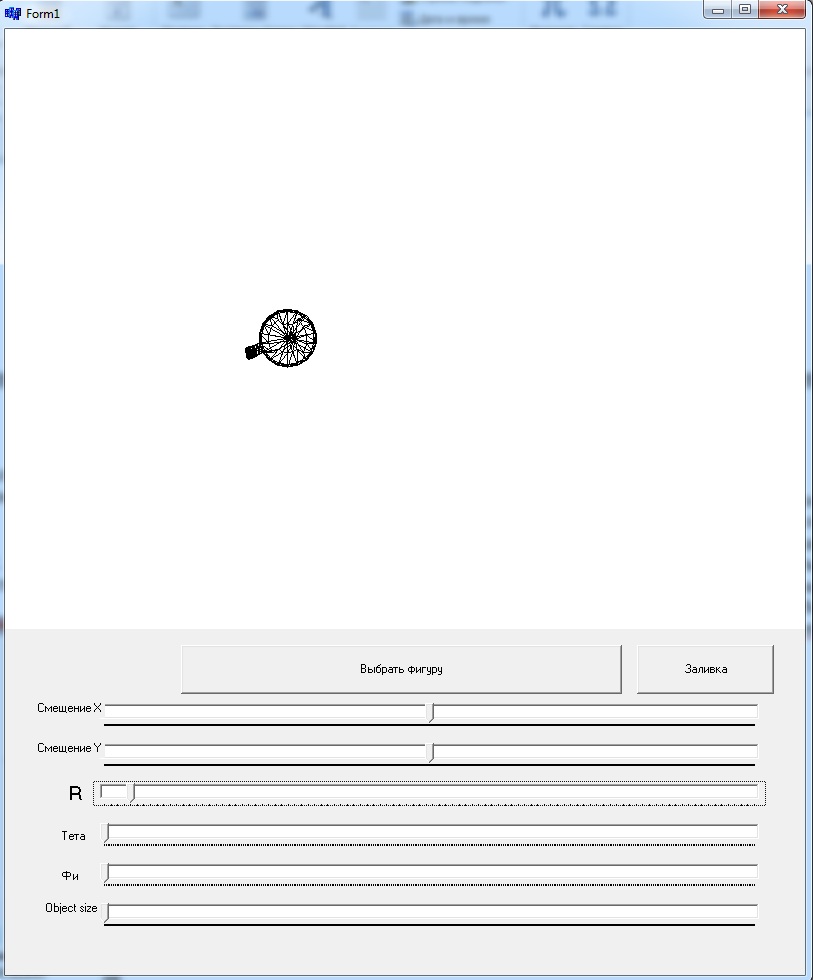
При использовании ползунка "**Смещение Х**" можно изменить положение объекта по горизонтальной оси:



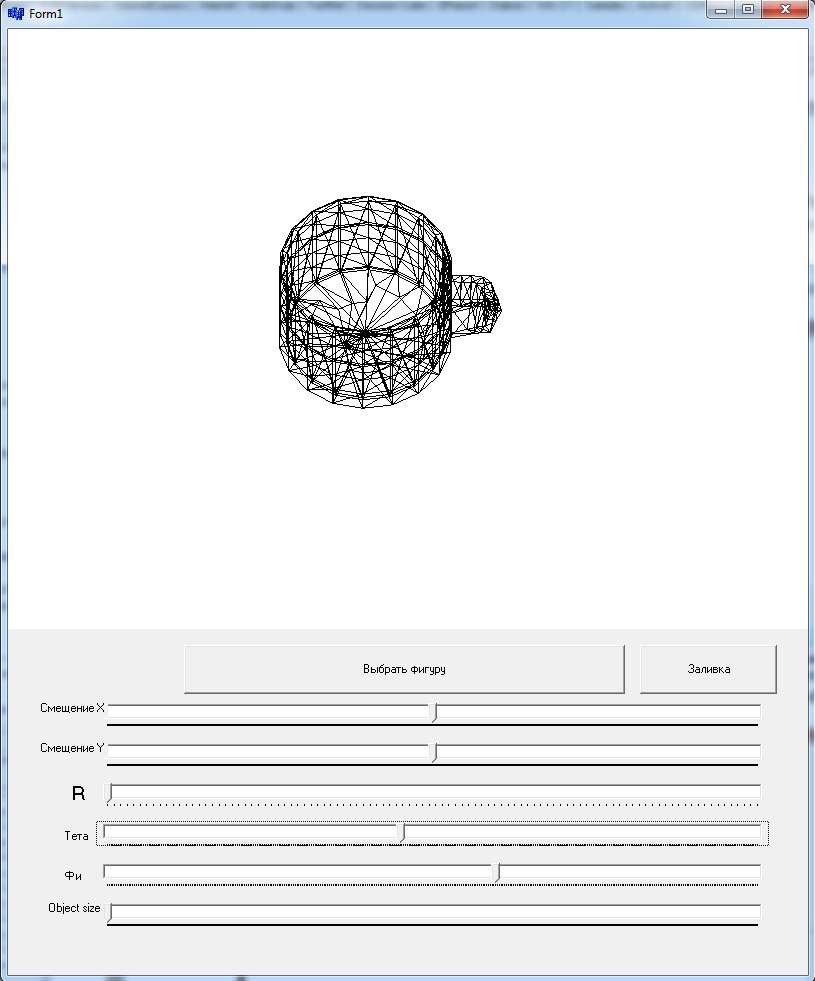
При использовании ползунка "**Смещение Y**" можно изменить положение объекта по вертикальной оси:



Ползунок "**R**" позволит вам изменить расстояние до объекта. Изначальное выбрано минимальное значение, поэтому увеличение этого параметра приведет к уменьшению изображения объекта.

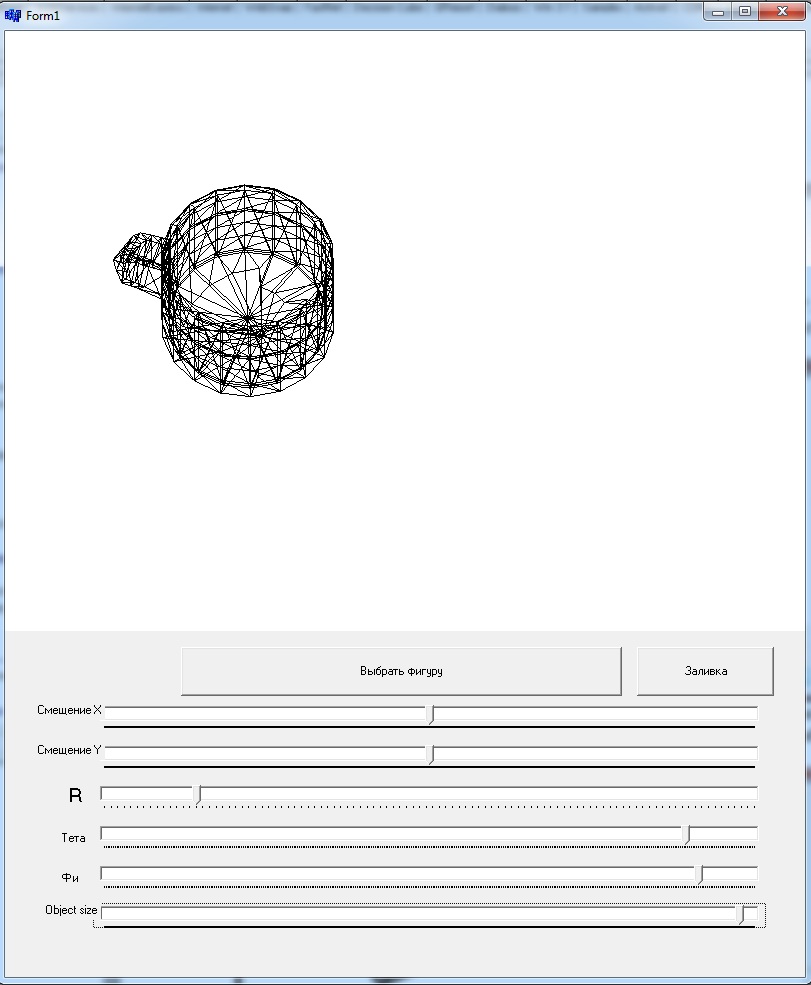


Ползунки "**Тета**" и "**Фи**" позволят вам изменить угол обзора объекта. Каждый угол меняет положение объекта на экране, осуществляя ее поворот.



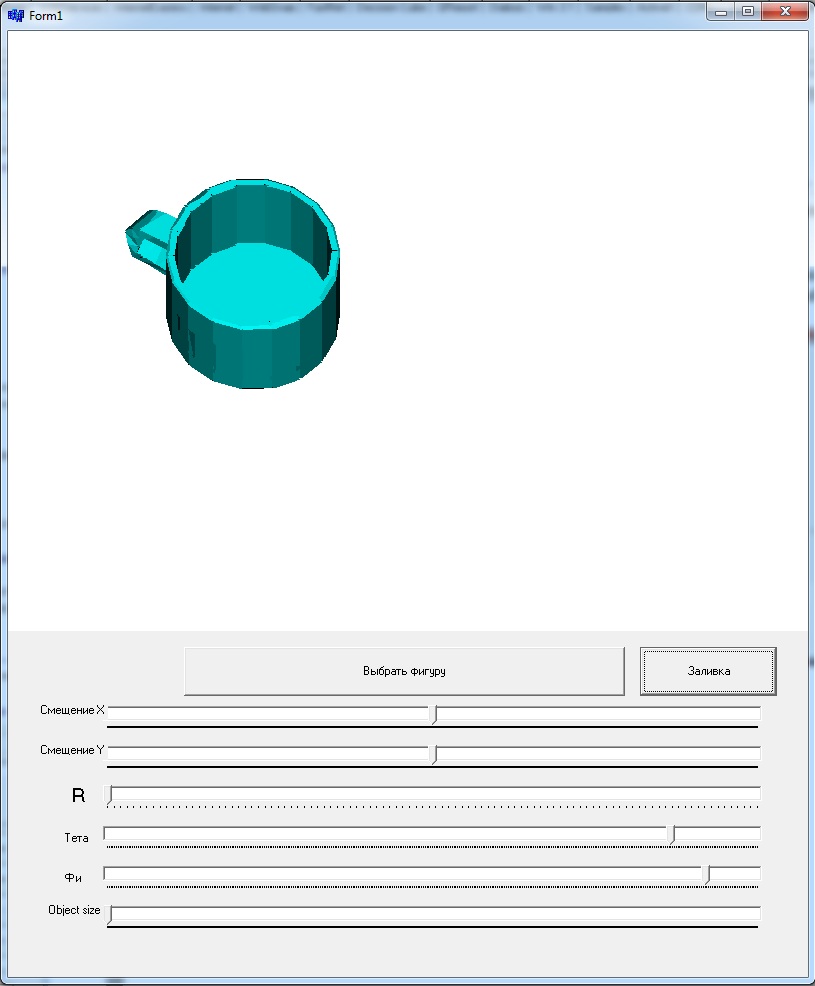
Наглядное практическое применение.

Ползунок "**Object size**" позволит вам изменить размер объекта. В основном используется вместе с изменением параметра ползунка "**R**", что позволяет увидеть, как будет выглядеть объект на каком-либо расстоянии от вас и при разных его габаритах.

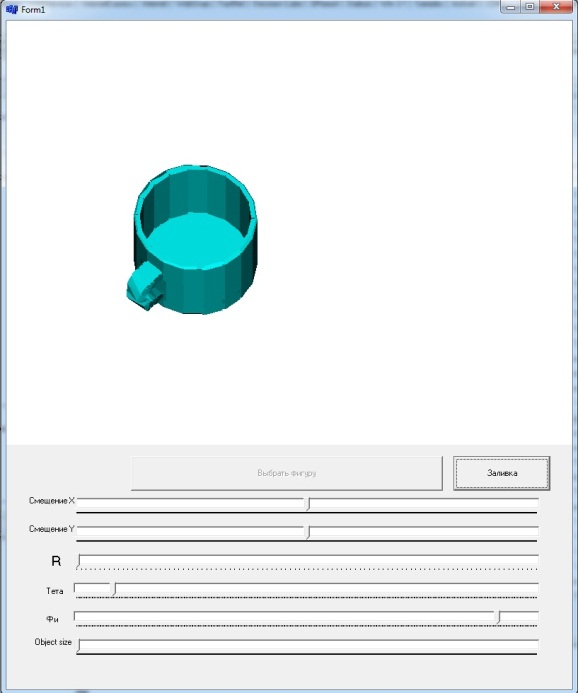
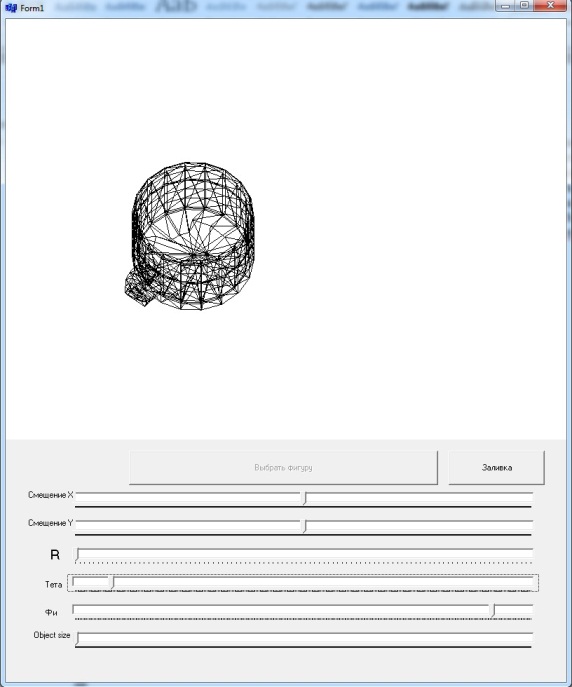


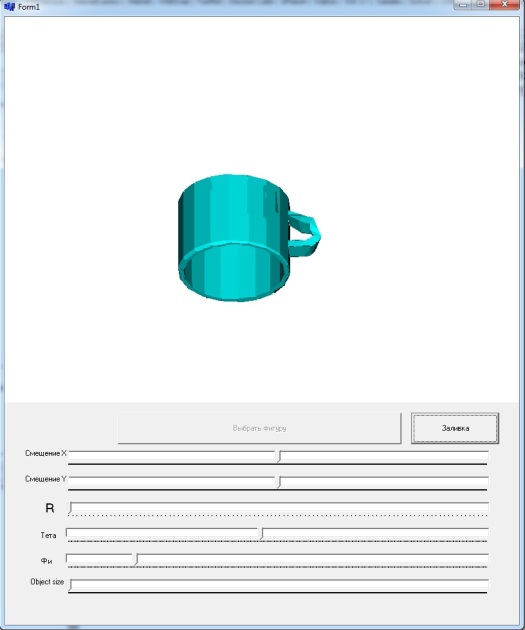
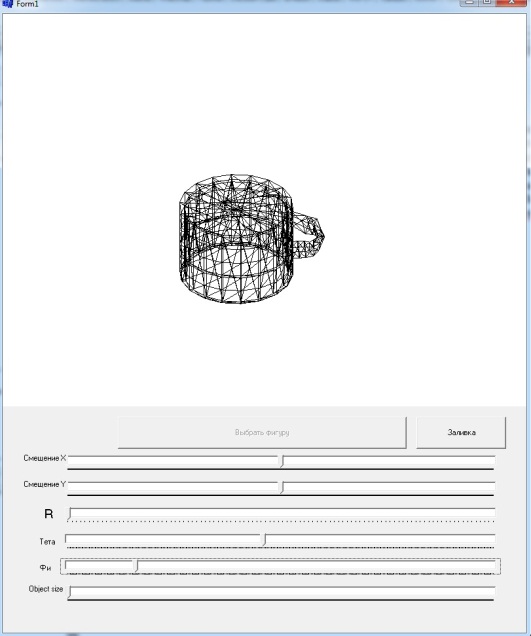
При большом отдалении и больших размерах объекта его изображение практически не изменилось.

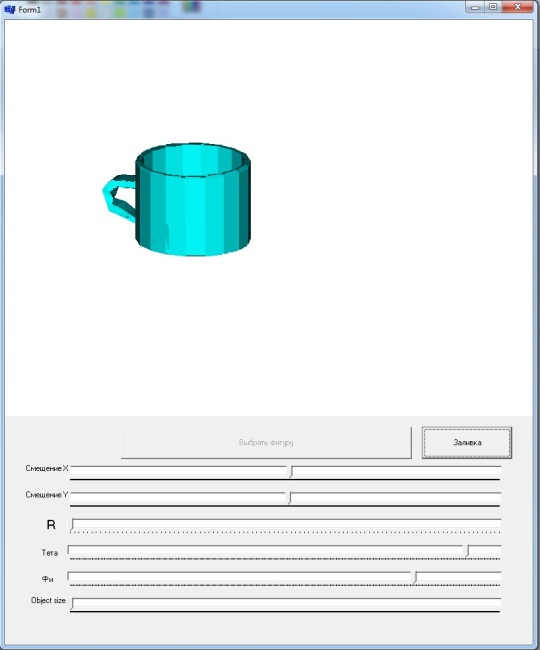
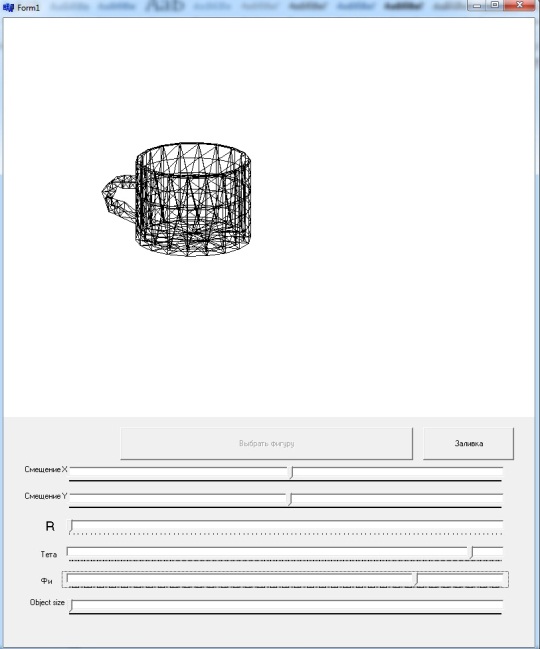
Для того, что бы закрасить объект нажмите кнопку "**Заливка**". Она позволяет закрасить объект оттенками смеси зеленого цвета и синего в зависимости от уровня освещенности соответствующей грани.



При изменении одного из ползунков на экране снова появится каркасная модель объекта. Для повторной закраски объекта еще раз нажмите кнопку "**Заливка**".







Некоторые примеры закраски фигуры со сменой ее положения**.**

# Заключение

В результате работы над данным курсовым проектом была реализована программа для просмотра 3D-объектов в формате .obj. Для данной работы был создан трехмерный объект "Кружка", который может быть открыт в данном приложении. Программа позволяет перемещать объект относительно осей, изменять расстояние наблюдателя до объекта, менять углы обзора, изменять размеры объекта, а также производить закраску в соответствии с уровнем освещенности. Были получены навыки работы с такими программными продуктами как: С++ Builder 6 и 3DS Max 9.

# Список использованной литературы

1. Карпов Е.В. Геометрическое моделирование и машинная графика в САПР: Методические указания к курсовому проектированию. - Пенза: ПГТУ, 1994. - 41 с.
2. Шикин Е.В., Боресков А.В. Компьютерная графика. Динамика, реалистические изображения - М.: "Диалог-МИФИ", 1995. - 288 с.
3. Фролов А.В., Фролов Г.В. Графический интерфейс GDI в MS Windows. - М.: ДИАЛОГ-МИФИ, 1994.
4. http://bibliofond.ru/view.aspx?id=600422
5. http://vbbook.ru/c/
6. http://www.cyberforum.ru/
7. https://msdn.microsoft.com/ru-ru/library
8. Аммерал Л. Машинная графика на языке Си. В 4-х кн. Кн.1. Принципы программирования в машинной графике. Кн.2. Машинная графика на персональных компьютерах. Кн.З. Интерактивная трехмерная машинная графика. Кн.4. Программирование графики на Турбо Си: Пер. с англ.- М: Сол Систем, 1992.

# Приложение

## Приложение А

(Коррдинаты vn и vt опущены, так как не используются программой)

# Max2Obj Version 4.0 Mar 10th, 2001

#

# object (null) to come ...

#

v 29.687828 -25.754061 0.000000

v 27.071880 -10.918279 0.000000

v 19.539555 2.128090 0.000000

v 7.999364 11.811457 0.000000

v -6.156775 16.963869 0.000000

v -21.221424 16.963869 0.000000

v -35.377560 11.811457 0.000000

v -46.917744 2.128088 0.000000

v -54.450066 -10.918276 0.000000

v -57.066017 -25.754055 0.000000

v -54.450073 -40.589832 0.000000

v -46.917751 -53.636200 0.000000

v -35.377571 -63.319572 0.000000

v -21.221437 -68.471985 0.000000

v -6.156793 -68.471992 0.000000

v 7.999345 -63.319588 0.000000

v 19.539539 -53.636230 0.000000

v 27.071869 -40.589870 0.000000

v 29.687828 -25.754061 12.436194

v 27.071880 -10.918279 12.436194

v 19.539555 2.128090 12.436194

v 7.999364 11.811457 12.436194

v -6.156775 16.963869 12.436194

v -21.221424 16.963869 12.436194

v -35.377560 11.811457 12.436194

v -46.917744 2.128088 12.436194

v -54.450066 -10.918276 12.436194

v -57.066017 -25.754055 12.436194

v -54.450073 -40.589832 12.436194

v -46.917751 -53.636200 12.436194

v -35.377571 -63.319572 12.436194

v -21.221437 -68.471985 12.436194

v -6.156793 -68.471992 12.436194

v 7.999345 -63.319588 12.436194

v 19.539539 -53.636230 12.436194

v 27.071869 -40.589870 12.436194

v 29.687828 -25.754061 24.872389

v 27.071880 -10.918279 24.872389

v 19.539555 2.128090 24.872389

v 7.999364 11.811457 24.872389

v -6.156775 16.963869 24.872389

v -21.221424 16.963869 24.872389

v -35.377560 11.811457 24.872389

v -46.917744 2.128088 24.872389

v -54.450066 -10.918276 24.872389

v -57.066017 -25.754055 24.872389

v -54.450073 -40.589832 24.872389

v -46.917751 -53.636200 24.872389

v -35.377571 -63.319572 24.872389

v -21.221437 -68.471985 24.872389

v -6.156793 -68.471992 24.872389

v 7.999345 -63.319588 24.872389

v 19.539539 -53.636230 24.872389

v 27.071869 -40.589870 24.872389

v 29.687828 -25.754061 36.308586

v 27.071880 -10.918279 36.308586

v 19.539555 2.128090 36.308586

v 7.999364 11.811457 36.308586

v -6.156775 16.963869 36.308586

v -21.221424 16.963869 36.308586

v -35.377560 11.811457 36.308586

v -46.917744 2.128088 36.308586

v -54.450066 -10.918276 36.308586

v -57.066017 -25.754055 36.308586

v -54.450073 -40.589832 36.308586

v -46.917751 -53.636200 36.308586

v -35.377571 -63.319572 36.308586

v -21.221437 -68.471985 36.308586

v -6.156793 -68.471992 36.308586

v 7.999345 -63.319588 36.308586

v 19.539539 -53.636230 36.308586

v 27.071869 -40.589870 36.308586

v 29.687828 -25.754061 49.744778

v 27.071880 -10.918279 49.744778

v 19.539555 2.128090 49.744778

v 7.999364 11.811457 49.744778

v -6.156775 16.963869 49.744778

v -21.221424 16.963869 49.744778

v -35.377560 11.811457 49.744778

v -46.917744 2.128088 49.744778

v -54.450066 -10.918276 49.744778

v -57.066017 -25.754055 49.744778

v -54.450073 -40.589832 49.744778

v -46.917751 -53.636200 49.744778

v -35.377571 -63.319572 49.744778

v -21.221437 -68.471985 49.744778

v -6.156793 -68.471992 49.744778

v 7.999345 -63.319588 49.744778

v 19.539539 -53.636230 49.744778

v 27.071869 -40.589870 49.744778

v 29.687828 -25.754061 61.180973

v 27.071880 -10.918279 61.180973

v 19.539555 2.128090 61.180973

v 7.999364 11.811457 61.180973

v -6.156775 16.963869 61.180973

v -21.221424 16.963869 61.180973

v -35.377560 11.811457 61.180973

v -46.917744 2.128088 61.180973

v -54.450066 -10.918276 61.180973

v -57.066017 -25.754055 61.180973

v -54.450073 -40.589832 61.180973

v -46.917751 -53.636200 61.180973

v -35.377571 -63.319572 61.180973

v -21.221437 -68.471985 61.180973

v -6.156793 -68.471992 61.180973

v 7.999345 -63.319588 61.180973

v 19.539539 -53.636230 61.180973

v 27.071869 -40.589870 61.180973

v 7.999366 -25.754061 0.000000

v 6.691392 -18.336170 0.000000

v 14.845271 -11.010910 0.000000

v 4.274169 -2.140703 0.000000

v -8.693209 2.579035 0.000000

v -22.492804 2.579035 0.000000

v -35.460175 -2.140703 0.000000

v -30.303421 -11.812986 0.000000

v -34.069580 -18.336168 0.000000

v -35.377556 -25.754057 0.000000

v -34.069584 -33.171947 0.000000

v -30.303425 -39.695129 0.000000

v -35.460182 -49.367409 0.000000

v -22.492811 -54.087151 0.000000

v -8.693221 -54.087158 0.000000

v 4.274154 -49.367424 0.000000

v 14.845260 -40.497227 0.000000

v 6.691386 -33.171967 0.000000

v -15.592999 -14.956463 0.000000

v 26.045250 -25.754061 62.180984

v 23.648979 -12.164114 62.180984

v 16.749180 -0.213314 62.180984

v 6.178077 8.656893 62.180984

v -6.789301 13.376631 62.180984

v -20.588896 13.376631 62.180984

v -33.556267 8.656893 62.180984

v -44.127365 -0.213316 62.180984

v -51.027161 -12.164111 62.180984

v -53.423439 -25.754055 62.180984

v -51.027168 -39.343998 62.180984

v -44.127373 -51.294796 62.180984

v -33.556278 -60.165009 62.180984

v -20.588907 -64.884750 62.180984

v -6.789317 -64.884758 62.180984

v 6.178060 -60.165024 62.180984

v 16.749165 -51.294823 62.180984

v 23.648972 -39.344032 62.180984

v -13.689094 -25.754061 4.180984

v 26.045250 -25.754061 2.180984

v 23.648979 -12.164114 2.180984

v 16.749180 -0.213314 2.180984

v 6.178077 8.656893 2.180984

v -6.789301 13.376631 2.180984

v -20.588896 13.376631 2.180984

v -33.556267 8.656893 2.180984

v -44.127365 -0.213316 2.180984

v -51.027161 -12.164111 2.180984

v -53.423439 -25.754055 2.180984

v -51.027168 -39.343998 2.180984

v -44.127373 -51.294796 2.180984

v -33.556278 -60.165009 2.180984

v -20.588907 -64.884750 2.180984

v -6.789317 -64.884758 2.180984

v 6.178060 -60.165024 2.180984

v 16.749165 -51.294823 2.180984

v 23.648972 -39.344032 2.180984

v 26.045250 -25.754061 62.180984

v 23.648979 -12.164114 62.180984

v 16.749180 -0.213314 62.180984

v 6.178077 8.656893 62.180984

v -6.789301 13.376631 62.180984

v -20.588896 13.376631 62.180984

v -33.556267 8.656893 62.180984

v -44.127365 -0.213316 62.180984

v -51.027161 -12.164111 62.180984

v -53.423439 -25.754055 62.180984

v -51.027168 -39.343998 62.180984

v -44.127373 -51.294796 62.180984

v -33.556278 -60.165009 62.180984

v -20.588907 -64.884750 62.180984

v -6.789317 -64.884758 62.180984

v 6.178060 -60.165024 62.180984

v 16.749165 -51.294823 62.180984

v 23.648972 -39.344032 62.180984

v 26.045250 -25.754061 62.180984

v 23.648979 -12.164114 62.180984

v 16.749180 -0.213314 62.180984

v 6.178077 8.656893 62.180984

v -6.789301 13.376631 62.180984

v -20.588896 13.376631 62.180984

v -33.556267 8.656893 62.180984

v -44.127365 -0.213316 62.180984

v -51.027161 -12.164111 62.180984

v -53.423439 -25.754055 62.180984

v -51.027168 -39.343998 62.180984

v -44.127373 -51.294796 62.180984

v -33.556278 -60.165009 62.180984

v -20.588907 -64.884750 62.180984

v -6.789317 -64.884758 62.180984

v 6.178060 -60.165024 62.180984

v 16.749165 -51.294823 62.180984

v 23.648972 -39.344032 62.180984

v 26.045250 -25.754061 4.180984

v 23.648979 -12.164114 4.180984

v 16.749180 -0.213314 4.180984

v 6.178077 8.656893 4.180984

v -6.789301 13.376631 4.180984

v -20.588896 13.376631 4.180984

v -33.556267 8.656893 4.180984

v -44.127365 -0.213316 4.180984

v -51.027161 -12.164111 4.180984

v -53.423439 -25.754055 4.180984

v -51.027168 -39.343998 4.180984

v -44.127373 -51.294796 4.180984

v -33.556278 -60.165009 4.180984

v -20.588907 -64.884750 4.180984

v -6.789317 -64.884758 4.180984

v 6.178060 -60.165024 4.180984

v 16.749165 -51.294823 4.180984

v 23.648972 -39.344032 4.180984

v -35.377571 -63.319572 45.862915

v -21.221437 -68.471985 45.692207

v -21.630796 -68.322990 40.332016

v -35.377571 -63.319572 40.335846

v -21.221437 -68.471985 21.295856

v -35.377571 -63.319572 21.329195

v -21.221437 -68.471985 16.471891

v -35.377571 -63.319572 16.608782

v -22.931540 -73.170448 45.692207

v -37.087673 -68.018036 45.862915

v -37.087673 -68.018036 40.335846

v -23.340899 -73.021454 40.332016

v -24.641642 -77.868912 45.692207

v -38.797775 -72.716499 45.862915

v -38.797775 -72.716499 40.335846

v -25.051001 -77.719917 40.332016

v -26.452988 -83.689156 44.183949

v -40.960941 -79.984932 45.848480

v -40.393227 -76.217049 41.844833

v -26.321442 -79.924545 40.345734

v -27.720184 -87.280083 39.413101

v -42.228138 -84.380486 42.256390

v -41.660419 -79.460152 39.803558

v -27.588636 -82.420052 37.114349

v -28.987381 -88.510551 32.872765

v -43.495335 -87.718262 36.855751

v -42.927616 -82.271492 37.603302

v -28.855833 -83.200447 33.713585

v -31.575100 -89.929276 28.444233

v -45.163662 -84.323952 31.746340

v -42.862183 -79.355263 32.497147

v -29.733994 -84.944061 29.268522

v -31.576717 -88.048279 23.881706

v -44.921227 -81.207611 22.431889

v -44.354588 -79.158844 27.533836

v -31.412563 -85.861717 28.792255

v -29.302540 -84.009193 22.007166

v -42.647049 -77.168526 20.557348

v -42.080410 -75.119759 25.659296

v -29.138386 -81.822632 26.917715

v -27.028362 -80.277725 21.352510

v -40.372871 -74.045387 18.181597

v -39.806232 -70.745941 22.579439

v -26.864208 -76.894722 25.529814

v 28.672401 -25.754061 62.180977

v 26.117691 -11.265575 62.180977

v 18.761692 1.475386 62.180977

v 7.491651 10.932070 62.180977

v -6.333102 15.963869 62.180977

v -21.045097 15.963869 62.180977

v -34.869846 10.932074 62.180977

v -46.139881 1.475384 62.180977

v -53.495876 -11.265573 62.180977

v -56.050591 -25.754055 62.180977

v -53.495884 -40.242538 62.180977

v -46.139889 -52.983498 62.180977

v -34.869854 -62.440186 62.180977

v -21.045109 -67.471985 62.180977

v -6.333119 -67.471992 62.180977

v 7.491633 -62.440201 62.180977

v 18.761677 -52.983528 62.180977

v 26.117680 -40.242573 62.180977

v 29.687828 -25.754061 38.308586

v 27.071880 -10.918279 38.308586

v 19.539555 2.128090 38.308586

v 7.999364 11.811457 38.308586

v -6.156775 16.963869 38.308586

v -21.221424 16.963869 38.308586

v -35.377560 11.811457 38.308586

v -46.917744 2.128088 38.308586

v -54.450066 -10.918276 38.308586

v -57.066017 -25.754055 38.308586

v -54.450073 -40.589832 38.308586

v -46.917751 -53.636200 38.308586

v -35.377571 -63.319572 38.308586

v -21.221437 -68.471985 38.308586

v -6.156793 -68.471992 38.308586

v 7.999345 -63.319588 38.308586

v 19.539539 -53.636230 38.308586

v 27.071869 -40.589870 38.308586

# 298 vertices

vt\*\*………

# 359 texture vertices

vn\*\*………

# 298 vertex normals

g (null)

s 8

f 1/1/1 2/2/2 20/20/20

f 20/20/20 19/19/19 1/1/1

f 2/2/2 3/3/3 21/21/21

f 21/21/21 20/20/20 2/2/2

f 3/3/3 4/4/4 22/22/22

f 22/22/22 21/21/21 3/3/3

f 4/4/4 5/5/5 23/23/23

f 23/23/23 22/22/22 4/4/4

f 5/5/5 6/147/6 24/148/24

f 24/148/24 23/23/23 5/5/5

f 6/6/6 7/7/7 25/25/25

f 25/25/25 24/24/24 6/6/6

f 7/7/7 8/8/8 26/26/26

f 26/26/26 25/25/25 7/7/7

f 8/8/8 9/9/9 27/27/27

f 27/27/27 26/26/26 8/8/8

f 9/9/9 10/10/10 28/28/28

f 28/28/28 27/27/27 9/9/9

f 10/10/10 11/11/11 29/29/29

f 29/29/29 28/28/28 10/10/10

f 11/11/11 12/12/12 30/30/30

f 30/30/30 29/29/29 11/11/11

f 12/12/12 13/13/13 31/31/31

f 31/31/31 30/30/30 12/12/12

f 13/13/13 14/14/14 32/32/32

f 32/32/32 31/31/31 13/13/13

f 14/14/14 15/15/15 33/33/33

f 33/33/33 32/32/32 14/14/14

f 15/15/15 16/16/16 34/34/34

f 34/34/34 33/33/33 15/15/15

f 16/16/16 17/17/17 35/35/35

f 35/35/35 34/34/34 16/16/16

f 17/17/17 18/18/18 36/36/36

f 36/36/36 35/35/35 17/17/17

f 18/18/18 1/1/1 19/19/19

f 19/19/19 36/36/36 18/18/18

f 19/19/19 20/20/20 38/38/38

f 38/38/38 37/37/37 19/19/19

f 20/20/20 21/21/21 39/39/39

f 39/39/39 38/38/38 20/20/20

f 21/21/21 22/22/22 40/40/40

f 40/40/40 39/39/39 21/21/21

f 22/22/22 23/23/23 41/41/41

f 41/41/41 40/40/40 22/22/22

f 23/23/23 24/148/24 42/149/42

f 42/149/42 41/41/41 23/23/23

f 24/24/24 25/25/25 43/43/43

f 43/43/43 42/42/42 24/24/24

f 25/25/25 26/26/26 44/44/44

f 44/44/44 43/43/43 25/25/25

f 26/26/26 27/27/27 45/45/45

f 45/45/45 44/44/44 26/26/26

f 27/27/27 28/28/28 46/46/46

f 46/46/46 45/45/45 27/27/27

f 28/28/28 29/29/29 47/47/47

f 47/47/47 46/46/46 28/28/28

f 29/29/29 30/30/30 48/48/48

f 48/48/48 47/47/47 29/29/29

f 30/30/30 31/31/31 226/304/226

f 30/30/30 226/304/226 224/302/224

f 30/30/30 224/302/224 49/49/49

f 30/30/30 49/49/49 48/48/48

f 32/32/32 33/33/33 51/51/51

f 51/51/51 50/50/50 223/301/223

f 51/51/51 223/301/223 225/303/225

f 32/32/32 51/51/51 225/303/225

f 33/33/33 34/34/34 52/52/52

f 52/52/52 51/51/51 33/33/33

f 34/34/34 35/35/35 53/53/53

f 53/53/53 52/52/52 34/34/34

f 35/35/35 36/36/36 54/54/54

f 54/54/54 53/53/53 35/35/35

f 36/36/36 19/19/19 37/37/37

f 37/37/37 54/54/54 36/36/36

f 37/37/37 38/38/38 56/56/56

f 56/56/56 55/55/55 37/37/37

f 38/38/38 39/39/39 57/57/57

f 57/57/57 56/56/56 38/38/38

f 39/39/39 40/40/40 58/58/58

f 58/58/58 57/57/57 39/39/39

f 40/40/40 41/41/41 59/59/59

f 59/59/59 58/58/58 40/40/40

f 41/41/41 42/149/42 60/150/60

f 60/150/60 59/59/59 41/41/41

f 42/42/42 43/43/43 61/61/61

f 61/61/61 60/60/60 42/42/42

f 43/43/43 44/44/44 62/62/62

f 62/62/62 61/61/61 43/43/43

f 44/44/44 45/45/45 63/63/63

f 63/63/63 62/62/62 44/44/44

f 45/45/45 46/46/46 64/64/64

f 64/64/64 63/63/63 45/45/45

f 46/46/46 47/47/47 65/65/65

f 65/65/65 64/64/64 46/46/46

f 47/47/47 48/48/48 66/66/66

f 66/66/66 65/65/65 47/47/47

f 48/48/48 49/49/49 67/67/67

f 67/67/67 66/66/66 48/48/48

f 49/49/49 50/50/50 68/68/68

f 68/68/68 67/67/67 49/49/49

f 50/50/50 51/51/51 69/69/69

f 69/69/69 68/68/68 50/50/50

f 51/51/51 52/52/52 70/70/70

f 70/70/70 69/69/69 51/51/51

f 52/52/52 53/53/53 71/71/71

f 71/71/71 70/70/70 52/52/52

f 53/53/53 54/54/54 72/72/72

f 72/72/72 71/71/71 53/53/53

f 54/54/54 37/37/37 55/55/55

f 55/55/55 72/72/72 54/54/54

f 281/341/281 282/342/282 74/74/74

f 74/74/74 73/73/73 281/341/281

f 282/342/282 283/343/283 75/75/75

f 75/75/75 74/74/74 282/342/282

f 283/343/283 284/344/284 76/76/76

f 76/76/76 75/75/75 283/343/283

f 284/344/284 285/345/285 77/77/77

f 77/77/77 76/76/76 284/344/284

f 285/345/285 286/346/286 78/151/78

f 78/151/78 77/77/77 285/345/285

f 286/347/286 287/348/287 79/79/79

f 79/79/79 78/78/78 286/347/286

f 287/348/287 288/349/288 80/80/80

f 80/80/80 79/79/79 287/348/287

f 288/349/288 289/350/289 81/81/81

f 81/81/81 80/80/80 288/349/288

f 289/350/289 290/351/290 82/82/82

f 82/82/82 81/81/81 289/350/289

f 290/351/290 291/352/291 83/83/83

f 83/83/83 82/82/82 290/351/290

f 291/352/291 292/353/292 84/84/84

f 84/84/84 83/83/83 291/352/291

f 292/353/292 293/354/293 222/300/222

f 292/353/292 222/300/222 219/297/219

f 292/353/292 219/297/219 85/85/85

f 292/353/292 85/85/85 84/84/84

f 86/86/86 85/85/85 219/297/219

f 219/297/219 220/298/220 86/86/86

f 294/355/294 295/356/295 87/87/87

f 87/87/87 86/86/86 220/298/220

f 294/355/294 87/87/87 220/298/220

f 295/356/295 296/357/296 88/88/88

f 88/88/88 87/87/87 295/356/295

f 296/357/296 297/358/297 89/89/89

f 89/89/89 88/88/88 296/357/296

f 297/358/297 298/359/298 90/90/90

f 90/90/90 89/89/89 297/358/297

f 298/359/298 281/341/281 73/73/73

f 73/73/73 90/90/90 298/359/298

f 73/73/73 74/74/74 92/92/92

f 92/92/92 91/91/91 73/73/73

f 74/74/74 75/75/75 93/93/93

f 93/93/93 92/92/92 74/74/74

f 75/75/75 76/76/76 94/94/94

f 94/94/94 93/93/93 75/75/75

f 76/76/76 77/77/77 95/95/95

f 95/95/95 94/94/94 76/76/76

f 77/77/77 78/151/78 96/152/96

f 96/152/96 95/95/95 77/77/77

f 78/78/78 79/79/79 97/97/97

f 97/97/97 96/96/96 78/78/78

f 79/79/79 80/80/80 98/98/98

f 98/98/98 97/97/97 79/79/79

f 80/80/80 81/81/81 99/99/99

f 99/99/99 98/98/98 80/80/80

f 81/81/81 82/82/82 100/100/100

f 100/100/100 99/99/99 81/81/81

f 82/82/82 83/83/83 101/101/101

f 101/101/101 100/100/100 82/82/82

f 83/83/83 84/84/84 102/102/102

f 102/102/102 101/101/101 83/83/83

f 84/84/84 85/85/85 103/103/103

f 103/103/103 102/102/102 84/84/84

f 85/85/85 86/86/86 104/104/104

f 104/104/104 103/103/103 85/85/85

f 86/86/86 87/87/87 105/105/105

f 105/105/105 104/104/104 86/86/86

f 87/87/87 88/88/88 106/106/106

f 106/106/106 105/105/105 87/87/87

f 88/88/88 89/89/89 107/107/107

f 107/107/107 106/106/106 88/88/88

f 89/89/89 90/90/90 108/108/108

f 108/108/108 107/107/107 89/89/89

f 90/90/90 73/73/73 91/91/91

f 91/91/91 108/108/108 90/90/90

s 1

f 2/153/2 1/154/1 109/109/109

f 109/109/109 110/110/110 2/153/2

f 3/155/3 2/156/2 110/110/110

f 110/110/110 111/111/111 3/155/3

f 4/157/4 3/158/3 111/111/111

f 111/111/111 112/112/112 4/157/4

f 5/159/5 4/160/4 112/112/112

f 112/112/112 113/113/113 5/159/5

f 6/161/6 5/162/5 113/113/113

f 113/113/113 114/114/114 6/161/6

f 7/163/7 6/164/6 114/114/114

f 114/114/114 115/115/115 7/163/7

f 8/165/8 7/166/7 115/115/115

f 115/115/115 116/116/116 8/165/8

f 9/167/9 8/168/8 116/116/116

f 116/116/116 117/117/117 9/167/9

f 10/169/10 9/170/9 117/117/117

f 117/117/117 118/118/118 10/169/10

f 11/171/11 10/172/10 118/118/118

f 118/118/118 119/119/119 11/171/11

f 12/173/12 11/174/11 119/119/119

f 119/119/119 120/120/120 12/173/12

f 13/175/13 12/176/12 120/120/120

f 120/120/120 121/121/121 13/175/13

f 14/177/14 13/178/13 121/121/121

f 121/121/121 122/122/122 14/177/14

f 15/179/15 14/180/14 122/122/122

f 122/122/122 123/123/123 15/179/15

f 16/181/16 15/182/15 123/123/123

f 123/123/123 124/124/124 16/181/16

f 17/183/17 16/184/16 124/124/124

f 124/124/124 125/125/125 17/183/17

f 18/185/18 17/186/17 125/125/125

f 125/125/125 126/126/126 18/185/18

f 1/187/1 18/188/18 126/126/126

f 126/126/126 109/109/109 1/187/1

f 110/110/110 109/109/109 127/127/127

f 111/111/111 110/110/110 127/127/127

f 112/112/112 111/111/111 127/127/127

f 113/113/113 112/112/112 127/127/127

f 114/114/114 113/113/113 127/127/127

f 115/115/115 114/114/114 127/127/127

f 116/116/116 115/115/115 127/127/127

f 117/117/117 116/116/116 127/127/127

f 118/118/118 117/117/117 127/127/127

f 119/119/119 118/118/118 127/127/127

f 120/120/120 119/119/119 127/127/127

f 121/121/121 120/120/120 127/127/127

f 122/122/122 121/121/121 127/127/127

f 123/123/123 122/122/122 127/127/127

f 124/124/124 123/123/123 127/127/127

f 125/125/125 124/124/124 127/127/127

f 126/126/126 125/125/125 127/127/127

f 109/109/109 126/126/126 127/127/127

f 263/189/263 264/190/264 129/129/129

f 129/129/129 128/128/128 263/189/263

f 264/191/264 265/192/265 130/130/130

f 130/130/130 129/129/129 264/191/264

f 265/193/265 266/194/266 131/131/131

f 131/131/131 130/130/130 265/193/265

f 266/195/266 267/196/267 132/132/132

f 132/132/132 131/131/131 266/195/266

f 267/197/267 268/198/268 133/133/133

f 133/133/133 132/132/132 267/197/267

f 268/199/268 269/200/269 134/134/134

f 134/134/134 133/133/133 268/199/268

f 269/201/269 270/202/270 135/135/135

f 135/135/135 134/134/134 269/201/269

f 270/203/270 271/204/271 136/136/136

f 136/136/136 135/135/135 270/203/270

f 271/205/271 272/206/272 137/137/137

f 137/137/137 136/136/136 271/205/271

f 272/207/272 273/208/273 138/138/138

f 138/138/138 137/137/137 272/207/272

f 273/209/273 274/210/274 139/139/139

f 139/139/139 138/138/138 273/209/273

f 274/211/274 275/212/275 140/140/140

f 140/140/140 139/139/139 274/211/274

f 275/213/275 276/214/276 141/141/141

f 141/141/141 140/140/140 275/213/275

f 276/215/276 277/216/277 142/142/142

f 142/142/142 141/141/141 276/215/276

f 277/217/277 278/218/278 143/143/143

f 143/143/143 142/142/142 277/217/277

f 278/219/278 279/220/279 144/144/144

f 144/144/144 143/143/143 278/219/278

f 279/221/279 280/222/280 145/145/145

f 145/145/145 144/144/144 279/221/279

f 280/223/280 263/224/263 128/128/128

f 128/128/128 145/145/145 280/223/280

f 201/279/201 202/280/202 146/146/146

f 202/280/202 203/281/203 146/146/146

f 203/281/203 204/282/204 146/146/146

f 204/282/204 205/283/205 146/146/146

f 205/283/205 206/284/206 146/146/146

f 206/284/206 207/285/207 146/146/146

f 207/285/207 208/286/208 146/146/146

f 208/286/208 209/287/209 146/146/146

f 209/287/209 210/288/210 146/146/146

f 210/288/210 211/289/211 146/146/146

f 211/289/211 212/290/212 146/146/146

f 212/290/212 213/291/213 146/146/146

f 213/291/213 214/292/214 146/146/146

f 214/292/214 215/293/215 146/146/146

f 215/293/215 216/294/216 146/146/146

f 216/294/216 217/295/217 146/146/146

f 217/295/217 218/296/218 146/146/146

f 218/296/218 201/279/201 146/146/146

s 0

f 128/128/128 129/129/129 148/226/148

f 148/226/148 147/225/147 128/128/128

f 129/129/129 130/130/130 149/227/149

f 149/227/149 148/226/148 129/129/129

f 130/130/130 131/131/131 150/228/150

f 150/228/150 149/227/149 130/130/130

f 131/131/131 132/132/132 151/229/151

f 151/229/151 150/228/150 131/131/131

f 132/132/132 133/133/133 152/230/152

f 152/230/152 151/229/151 132/132/132

f 133/133/133 134/134/134 153/231/153

f 153/231/153 152/230/152 133/133/133

f 134/134/134 135/135/135 154/232/154

f 154/232/154 153/231/153 134/134/134

f 135/135/135 136/136/136 155/233/155

f 155/233/155 154/232/154 135/135/135

f 136/136/136 137/137/137 156/234/156

f 156/234/156 155/233/155 136/136/136

f 137/137/137 138/138/138 157/235/157

f 157/235/157 156/234/156 137/137/137

f 138/138/138 139/139/139 158/236/158

f 158/236/158 157/235/157 138/138/138

f 139/139/139 140/140/140 159/237/159

f 159/237/159 158/236/158 139/139/139

f 140/140/140 141/141/141 160/238/160

f 160/238/160 159/237/159 140/140/140

f 141/141/141 142/142/142 161/239/161

f 161/239/161 160/238/160 141/141/141

f 142/142/142 143/143/143 162/240/162

f 162/240/162 161/239/161 142/142/142

f 143/143/143 144/144/144 163/241/163

f 163/241/163 162/240/162 143/143/143

f 144/144/144 145/145/145 164/242/164

f 164/242/164 163/241/163 144/144/144

f 145/145/145 128/128/128 147/225/147

f 147/225/147 164/242/164 145/145/145

f 147/225/147 148/226/148 166/244/166

f 166/244/166 165/243/165 147/225/147

f 148/226/148 149/227/149 167/245/167

f 167/245/167 166/244/166 148/226/148

f 149/227/149 150/228/150 168/246/168

f 168/246/168 167/245/167 149/227/149

f 150/228/150 151/229/151 169/247/169

f 169/247/169 168/246/168 150/228/150

f 151/229/151 152/230/152 170/248/170

f 170/248/170 169/247/169 151/229/151

f 152/230/152 153/231/153 171/249/171

f 171/249/171 170/248/170 152/230/152

f 153/231/153 154/232/154 172/250/172

f 172/250/172 171/249/171 153/231/153

f 154/232/154 155/233/155 173/251/173

f 173/251/173 172/250/172 154/232/154

f 155/233/155 156/234/156 174/252/174

f 174/252/174 173/251/173 155/233/155

f 156/234/156 157/235/157 175/253/175

f 175/253/175 174/252/174 156/234/156

f 157/235/157 158/236/158 176/254/176

f 176/254/176 175/253/175 157/235/157

f 158/236/158 159/237/159 177/255/177

f 177/255/177 176/254/176 158/236/158

f 159/237/159 160/238/160 178/256/178

f 178/256/178 177/255/177 159/237/159

f 160/238/160 161/239/161 179/257/179

f 179/257/179 178/256/178 160/238/160

f 161/239/161 162/240/162 180/258/180

f 180/258/180 179/257/179 161/239/161

f 162/240/162 163/241/163 181/259/181

f 181/259/181 180/258/180 162/240/162

f 163/241/163 164/242/164 182/260/182

f 182/260/182 181/259/181 163/241/163

f 164/242/164 147/225/147 165/243/165

f 165/243/165 182/260/182 164/242/164

f 165/243/165 166/244/166 184/262/184

f 184/262/184 183/261/183 165/243/165

f 166/244/166 167/245/167 185/263/185

f 185/263/185 184/262/184 166/244/166

f 167/245/167 168/246/168 186/264/186

f 186/264/186 185/263/185 167/245/167

f 168/246/168 169/247/169 187/265/187

f 187/265/187 186/264/186 168/246/168

f 169/247/169 170/248/170 188/266/188

f 188/266/188 187/265/187 169/247/169

f 170/248/170 171/249/171 189/267/189

f 189/267/189 188/266/188 170/248/170

f 171/249/171 172/250/172 190/268/190

f 190/268/190 189/267/189 171/249/171

f 172/250/172 173/251/173 191/269/191

f 191/269/191 190/268/190 172/250/172

f 173/251/173 174/252/174 192/270/192

f 192/270/192 191/269/191 173/251/173

f 174/252/174 175/253/175 193/271/193

f 193/271/193 192/270/192 174/252/174

f 175/253/175 176/254/176 194/272/194

f 194/272/194 193/271/193 175/253/175

f 176/254/176 177/255/177 195/273/195

f 195/273/195 194/272/194 176/254/176

f 177/255/177 178/256/178 196/274/196

f 196/274/196 195/273/195 177/255/177

f 178/256/178 179/257/179 197/275/197

f 197/275/197 196/274/196 178/256/178

f 179/257/179 180/258/180 198/276/198

f 198/276/198 197/275/197 179/257/179

f 180/258/180 181/259/181 199/277/199

f 199/277/199 198/276/198 180/258/180

f 181/259/181 182/260/182 200/278/200

f 200/278/200 199/277/199 181/259/181

f 182/260/182 165/243/165 183/261/183

f 183/261/183 200/278/200 182/260/182

f 183/261/183 184/262/184 202/280/202

f 202/280/202 201/279/201 183/261/183

f 184/262/184 185/263/185 203/281/203

f 203/281/203 202/280/202 184/262/184

f 185/263/185 186/264/186 204/282/204

f 204/282/204 203/281/203 185/263/185

f 186/264/186 187/265/187 205/283/205

f 205/283/205 204/282/204 186/264/186

f 187/265/187 188/266/188 206/284/206

f 206/284/206 205/283/205 187/265/187

f 188/266/188 189/267/189 207/285/207

f 207/285/207 206/284/206 188/266/188

f 189/267/189 190/268/190 208/286/208

f 208/286/208 207/285/207 189/267/189

f 190/268/190 191/269/191 209/287/209

f 209/287/209 208/286/208 190/268/190

f 191/269/191 192/270/192 210/288/210

f 210/288/210 209/287/209 191/269/191

f 192/270/192 193/271/193 211/289/211

f 211/289/211 210/288/210 192/270/192

f 193/271/193 194/272/194 212/290/212

f 212/290/212 211/289/211 193/271/193

f 194/272/194 195/273/195 213/291/213

f 213/291/213 212/290/212 194/272/194

f 195/273/195 196/274/196 214/292/214

f 214/292/214 213/291/213 195/273/195

f 196/274/196 197/275/197 215/293/215

f 215/293/215 214/292/214 196/274/196

f 197/275/197 198/276/198 216/294/216

f 216/294/216 215/293/215 197/275/197

f 198/276/198 199/277/199 217/295/217

f 217/295/217 216/294/216 198/276/198

f 199/277/199 200/278/200 218/296/218

f 218/296/218 217/295/217 199/277/199

f 200/278/200 183/261/183 201/279/201

f 201/279/201 218/296/218 200/278/200

s 8

f 221/299/221 222/300/222 293/354/293

f 221/299/221 293/354/293 294/355/294

f 221/299/221 294/355/294 220/298/220

f 50/50/50 49/49/49 224/302/224

f 224/302/224 223/301/223 50/50/50

s 0

f 220/298/220 219/297/219 228/306/228

f 228/306/228 227/305/227 220/298/220

f 219/297/219 222/300/222 229/307/229

f 229/307/229 228/306/228 219/297/219

f 222/300/222 221/299/221 230/308/230

f 230/308/230 229/307/229 222/300/222

f 221/299/221 220/298/220 227/305/227

f 227/305/227 230/308/230 221/299/221

f 227/305/227 228/306/228 232/310/232

f 232/310/232 231/309/231 227/305/227

f 228/306/228 229/307/229 233/311/233

f 233/311/233 232/310/232 228/306/228

f 229/307/229 230/308/230 234/312/234

f 234/312/234 233/311/233 229/307/229

f 230/308/230 227/305/227 231/309/231

f 231/309/231 234/312/234 230/308/230

f 231/309/231 232/310/232 236/314/236

f 236/314/236 235/313/235 231/309/231

f 232/310/232 233/311/233 237/315/237

f 237/315/237 236/314/236 232/310/232

f 233/311/233 234/312/234 238/316/238

f 238/316/238 237/315/237 233/311/233

f 234/312/234 231/309/231 235/313/235

f 235/313/235 238/316/238 234/312/234

f 235/313/235 236/314/236 240/318/240

f 240/318/240 239/317/239 235/313/235

f 236/314/236 237/315/237 241/319/241

f 241/319/241 240/318/240 236/314/236

f 237/315/237 238/316/238 242/320/242

f 242/320/242 241/319/241 237/315/237

f 238/316/238 235/313/235 239/317/239

f 239/317/239 242/320/242 238/316/238

f 239/317/239 240/318/240 244/322/244

f 244/322/244 243/321/243 239/317/239

f 240/318/240 241/319/241 245/323/245

f 245/323/245 244/322/244 240/318/240

f 241/319/241 242/320/242 246/324/246

f 246/324/246 245/323/245 241/319/241

f 242/320/242 239/317/239 243/321/243

f 243/321/243 246/324/246 242/320/242

f 243/321/243 244/322/244 248/326/248

f 248/326/248 247/325/247 243/321/243

f 244/322/244 245/323/245 249/327/249

f 249/327/249 248/326/248 244/322/244

f 245/323/245 246/324/246 250/328/250

f 250/328/250 249/327/249 245/323/245

f 246/324/246 243/321/243 247/325/247

f 247/325/247 250/328/250 246/324/246

f 247/325/247 248/326/248 252/330/252

f 252/330/252 251/329/251 247/325/247

f 248/326/248 249/327/249 253/331/253

f 253/331/253 252/330/252 248/326/248

f 249/327/249 250/328/250 254/332/254

f 254/332/254 253/331/253 249/327/249

f 254/332/254 250/328/250 247/325/247

f 247/325/247 251/329/251 254/332/254

f 251/329/251 252/330/252 256/334/256

f 256/334/256 255/333/255 251/329/251

f 252/330/252 253/331/253 257/335/257

f 257/335/257 256/334/256 252/330/252

f 253/331/253 254/332/254 258/336/258

f 258/336/258 257/335/257 253/331/253

f 254/332/254 251/329/251 255/333/255

f 255/333/255 258/336/258 254/332/254

f 255/333/255 256/334/256 260/338/260

f 260/338/260 259/337/259 255/333/255

f 256/334/256 257/335/257 261/339/261

f 261/339/261 260/338/260 256/334/256

f 257/335/257 258/336/258 262/340/262

f 262/340/262 261/339/261 257/335/257

f 258/336/258 255/333/255 259/337/259

f 259/337/259 262/340/262 258/336/258

s 1

f 32/32/32 225/303/225 262/340/262

f 262/340/262 259/337/259 32/32/32

s 2

f 225/1/225 226/1/226 261/339/261

f 261/339/261 262/340/262 225/1/225

s 1

f 226/304/226 31/31/31 260/338/260

f 260/338/260 261/339/261 226/304/226

s 2

f 31/31/31 32/32/32 259/337/259

f 259/337/259 260/338/260 31/31/31

s 0

f 92/92/92 264/190/264 263/189/263

f 263/189/263 91/91/91 92/92/92

f 93/93/93 265/192/265 264/190/264

f 264/190/264 92/92/92 93/93/93

f 94/94/94 266/194/266 265/192/265

f 265/192/265 93/93/93 94/94/94

f 95/95/95 267/196/267 266/194/266

f 266/194/266 94/94/94 95/95/95

f 96/152/96 268/198/268 267/196/267

f 267/196/267 95/95/95 96/152/96

f 97/97/97 269/200/269 268/198/268

f 268/198/268 96/152/96 97/97/97

f 98/98/98 270/202/270 269/200/269

f 269/200/269 97/97/97 98/98/98

f 99/99/99 271/204/271 270/202/270

f 270/202/270 98/98/98 99/99/99

f 100/100/100 272/206/272 271/204/271

f 271/204/271 99/99/99 100/100/100

f 101/101/101 273/208/273 272/206/272

f 272/206/272 100/100/100 101/101/101

f 102/102/102 274/210/274 273/208/273

f 273/208/273 101/101/101 102/102/102

f 103/103/103 275/212/275 274/210/274

f 274/210/274 102/102/102 103/103/103

f 104/104/104 276/214/276 275/212/275

f 275/212/275 103/103/103 104/104/104

f 105/105/105 277/216/277 276/214/276

f 276/214/276 104/104/104 105/105/105

f 106/106/106 278/218/278 277/216/277

f 277/216/277 105/105/105 106/106/106

f 107/107/107 279/220/279 278/218/278

f 278/218/278 106/106/106 107/107/107

f 108/108/108 280/222/280 279/220/279

f 279/220/279 107/107/107 108/108/108

f 91/91/91 263/189/263 280/222/280

f 280/222/280 108/108/108 91/91/91

s 8

f 56/56/56 282/342/282 281/341/281

f 281/341/281 55/55/55 56/56/56

f 57/57/57 283/343/283 282/342/282

f 282/342/282 56/56/56 57/57/57

f 58/58/58 284/344/284 283/343/283

f 283/343/283 57/57/57 58/58/58

f 59/59/59 285/345/285 284/344/284

f 284/344/284 58/58/58 59/59/59

f 60/150/60 286/346/286 285/345/285

f 285/345/285 59/59/59 60/150/60

f 61/61/61 287/348/287 286/346/286

f 286/346/286 60/150/60 61/61/61

f 62/62/62 288/349/288 287/348/287

f 287/348/287 61/61/61 62/62/62

f 63/63/63 289/350/289 288/349/288

f 288/349/288 62/62/62 63/63/63

f 64/64/64 290/351/290 289/350/289

f 289/350/289 63/63/63 64/64/64

f 65/65/65 291/352/291 290/351/290

f 290/351/290 64/64/64 65/65/65

f 66/66/66 292/353/292 291/352/291

f 291/352/291 65/65/65 66/66/66

f 67/67/67 293/354/293 292/353/292

f 292/353/292 66/66/66 67/67/67

f 68/68/68 294/355/294 293/354/293

f 293/354/293 67/67/67 68/68/68

f 69/69/69 295/356/295 294/355/294

f 294/355/294 68/68/68 69/69/69

f 70/70/70 296/357/296 295/356/295

f 295/356/295 69/69/69 70/70/70

f 71/71/71 297/358/297 296/357/296

f 296/357/296 70/70/70 71/71/71

f 72/72/72 298/359/298 297/358/297

f 297/358/297 71/71/71 72/72/72

f 55/55/55 281/341/281 298/359/298

f 298/359/298 72/72/72 55/55/55

# 594 faces

g

## Приложение Б(Файл Unit1.h)

//---------------------------------------------------------------------------

struct vert // Структура координат вершин

{

float x;

float y;

float z;

}vertices[1000];

struct vert transf\_vertices[1000];// Структура преобразованных координат

struct indexs // Структура индексов

{

int index\_vertices[4];

int index\_vertex\_normals[4];

int index\_texture\_coords[4];

int number\_of\_index; // Кол-во значений индексов(3-triangles,4-polygons);

}index[1000];

struct coord

{

int x; // Структура для занесения в нее координат Y при определенных Х

int y;

}coord[60000];

int number\_of\_coords[1000]; // Кол-во эл-тов в структуре координат полигона X,Y при их изменении

int R,teta,phi,object\_size,smech\_x,smech\_y; // Параметры прокрутки TrackBar

int k\_vertices; // Общее кол-во строк координат вершин

int k\_index; // Общее кол-во строк индексов

float d;

float mas[1000]; // Массив расстояний до экрана

int mas\_x[1000]; // Массив значений X с данной Y(j)

void read\_from\_file(char\*);

void update\_wind();

void viewport\_transformation(struct vert[1000],struct vert[1000],int,int,int);

float perspective\_transformation(int,int,int,int,struct vert[1000],struct vert[1000]);

void Screen\_conversion(struct vert[1000],struct indexs[1000],float,int,int);

void sort\_vertices(struct vert[1000],struct indexs[1000]);

void light(struct vert[1000],struct indexs[1000]);

void rastr(float,float,float,float);

int color\_rastr(float,float,float,float,int);

void color\_image(float,float,struct vert[1000],int,int,int);

#include <limits.h>

#include <stdio.h>

#include <string.h>

#include <math.h>

#include <Classes.hpp>

#include <Controls.hpp>

#include <ExtCtrls.hpp>

#include <StdCtrls.hpp>

#include <ComCtrls.hpp>

#include <DBCtrls.hpp>

#include <Mask.hpp>

#include <Dialogs.hpp>

#ifndef Unit1H

#define Unit1H

//---------------------------------------------------------------------------

#include <Classes.hpp>

#include <Controls.hpp>

#include <StdCtrls.hpp>

#include <Forms.hpp>

#include <ExtCtrls.hpp>

//---------------------------------------------------------------------------

class TForm1 : public TForm

{

\_\_published: // IDE-managed Components

TImage \*Image1;

TButton \*Button1;

TOpenDialog \*OpenFileDialog1;

TTrackBar \*TrackBar1;

TTrackBar \*TrackBar2;

TTrackBar \*TrackBar3;

TTrackBar \*TrackBar4;

TTrackBar \*TrackBar5;

TTrackBar \*TrackBar6;

TStaticText \*StaticText1;

TStaticText \*StaticText7;

TStaticText \*StaticText6;

TStaticText \*StaticText5;

TStaticText \*StaticText4;

TStaticText \*StaticText3;

TButton \*Button2;

void \_\_fastcall Button1Click(TObject \*Sender);

void \_\_fastcall TrackBar1Change(TObject \*Sender);

void \_\_fastcall TrackBar2Change(TObject \*Sender);

void \_\_fastcall TrackBar3Change(TObject \*Sender);

void \_\_fastcall TrackBar4Change(TObject \*Sender);

void \_\_fastcall TrackBar5Change(TObject \*Sender);

void \_\_fastcall TrackBar6Change(TObject \*Sender);

void \_\_fastcall Button2Click(TObject \*Sender);

private: // User declarations

public: // User declarations

\_\_fastcall TForm1(TComponent\* Owner);

};

//---------------------------------------------------------------------------

extern PACKAGE TForm1 \*Form1;

//---------------------------------------------------------------------------

#endif

## Приложение В(Файл Unit1.cpp)

//---------------------------------------------------------------------------

#include <vcl.h>

#pragma hdrstop

#include "Unit1.h"

//---------------------------------------------------------------------------

#pragma package(smart\_init)

#pragma resource "\*.dfm"

TForm1 \*Form1;

AnsiString FileName ="";

char\* MyFile;

//---------------------------------------------------------------------------

\_\_fastcall TForm1::TForm1(TComponent\* Owner)

: TForm(Owner)

{

}

//---------------------------------------------------------------------------

void \_\_fastcall TForm1::Button1Click(TObject \*Sender)

{

if(OpenFileDialog1->Execute())// Если в результате вызова окна файл был выбран

{

FileName = OpenFileDialog1->FileName; // Получаем имя файла

MyFile = FileName.c\_str(); // Перевод Ansi в char\* через функцию c\_str()

}

else return; // Если файл не был выбран, то уходим из функции

Image1->Picture->Bitmap = NULL;

read\_from\_file(MyFile);

Form1->Button2->Visible = true;

Form1->StaticText1->Visible = true;

Form1->StaticText4->Visible = true;

Form1->StaticText3->Visible = true;

Form1->StaticText5->Visible = true;

Form1->StaticText6->Visible = true;

Form1->StaticText7->Visible = true;

Form1->TrackBar1->Visible = true;

Form1->TrackBar2->Visible = true;

Form1->TrackBar3->Visible = true;

Form1->TrackBar4->Visible = true;

Form1->TrackBar5->Visible = true;

Form1->TrackBar6->Visible = true;

Form1->Button1->Enabled = false;

update\_wind();

}

//---------------------------------------------------------------------------

void \_\_fastcall TForm1::TrackBar1Change(TObject \*Sender)

{

smech\_x = TrackBar1->Position;

Image1->Picture->Bitmap = NULL;

update\_wind();

}

//---------------------------------------------------------------------------

void \_\_fastcall TForm1::TrackBar2Change(TObject \*Sender)

{

smech\_y = TrackBar2->Position;

Image1->Picture->Bitmap = NULL;

update\_wind();

}

//---------------------------------------------------------------------------

void \_\_fastcall TForm1::TrackBar3Change(TObject \*Sender)

{

R = TrackBar3->Position;

Image1->Picture->Bitmap = NULL;

update\_wind();

}

//---------------------------------------------------------------------------

void \_\_fastcall TForm1::TrackBar4Change(TObject \*Sender)

{

teta = TrackBar4->Position;

Image1->Picture->Bitmap = NULL;

update\_wind();

}

//---------------------------------------------------------------------------

void \_\_fastcall TForm1::TrackBar5Change(TObject \*Sender)

{

phi = TrackBar5->Position;

Image1->Picture->Bitmap = NULL;

update\_wind();

}

//---------------------------------------------------------------------------

void \_\_fastcall TForm1::TrackBar6Change(TObject \*Sender)

{

object\_size = TrackBar6->Position;

Image1->Picture->Bitmap = NULL;

update\_wind();

}

//---------------------------------------------------------------------------

void read\_from\_file(char\* MyFile)

{

FILE\* f0;

char c; // Для считывания символа

char next;

char\* buffer;

char\* v = "v ";

f0 = fopen(MyFile,"r");

buffer = (char\*)malloc(4);

// Доходим до места считывания вершин

do

{

fgets(buffer,4,f0);

}while(strspn(buffer,v)!=strlen(v));

// На выходе в buffer содержит "v "

// Считываем вершины

int i = 0;

while(fscanf(f0,"%f%c%f%c%f",&(vertices[i].x),&c,&(vertices[i].y),&c,&(vertices[i].z)) == 5)

{

vertices[i].x \*= 2;

vertices[i].y \*= 2;

vertices[i].z \*= 2;

fscanf(f0,"%c",&c); // Считываем \n

fgets(buffer,4,f0); // Читаем "v "

k\_vertices++;

i++;

}

// Доходим до места считывания индексов вершин "f "

do

{

fscanf(f0,"%c",&c);

}while(c!='f');

fscanf(f0,"%c",&c);

if(c==' ')

{

// Считываем индексы

i = 0;

bool indexs = 1;

while(indexs)

{

for(int j = 0;j<3;j++)

{

if(fscanf(f0,"%d%c%d%c%d",&(index[i].index\_vertices[j]),&c,&(index[i].index\_vertex\_normals[j]),&c,&(index[i].index\_texture\_coords[j])) != 5)

{

indexs = 0; // Если список индексов закончился

break;

}

fscanf(f0,"%c",&c); // Считываем пробел

}

do

{

fscanf(f0,"%c",&c);

}while(c!='f' && c!='#');

if(c=='#')

{

indexs = 0;

break;

}

fscanf(f0,"%c",&c);

i++;

k\_index++;

}

}

free(buffer);

fclose(f0);

smech\_x = Form1->TrackBar1->Position;

smech\_y = Form1->TrackBar2->Position;

R = Form1->TrackBar3->Min;

teta = Form1->TrackBar4->Min;

phi = Form1->TrackBar5->Min;

object\_size = Form1->TrackBar6->Min;

Form1->TrackBar1->Position = 0;

Form1->TrackBar2->Position = 0;

Form1->TrackBar3->Position = Form1->TrackBar3->Min;

Form1->TrackBar4->Position = Form1->TrackBar4->Min;

Form1->TrackBar5->Position = Form1->TrackBar5->Min;

Form1->TrackBar6->Position = Form1->TrackBar6->Min;

}

//---------------------------------------------------------------------------

void update\_wind()

{

viewport\_transformation(vertices,transf\_vertices,R,teta,phi); // Видовое преобразование

float d = perspective\_transformation(R,Form1->Image1->Width/10,Form1->Image1->Height/10,object\_size,transf\_vertices,vertices); // Перспективное преобразование

Screen\_conversion(transf\_vertices,index,d,smech\_x,smech\_y); // Экранное преобразование

}

//---------------------------------------------------------------------------

void viewport\_transformation(vert v[1000],vert res[1000],int R,int teta,int phi)

{

const float PI = 3.141592653589793238462643383279;

float teta1 = teta\*PI/180; // Радианы углов

float phi1 = phi\*PI/180;

float B[4][4] = {-sin(teta1),-cos(phi1)\*cos(teta1),-sin(phi1)\*cos(teta1),0,cos(teta1),-cos(phi1)\*sin(teta1),-sin(phi1)\*sin(teta1),0,0,sin(phi1),-cos(phi1),0,0,0,R,1};

for(int i = 0;i<k\_vertices;i++)

{

res[i].x = v[i].y\*B[1][0]+v[i].x\*B[0][0];

res[i].y = v[i].z\*B[2][1]+v[i].x\*B[0][1]+v[i].y\*B[1][1];

res[i].z = B[3][2]+v[i].z\*B[2][2]+v[i].x\*B[0][2]+v[i].y\*B[1][2];

}

}

//---------------------------------------------------------------------------

float perspective\_transformation(int R,int Width,int Height,int object\_size,vert res[1000],vert v[1000])

{

float B[4][4] = {1,0,0,0,0,1,0,0,0,0,0,(float)R\*((float)Width\*(float)Height/(float)object\_size),0,0,0,1};

for(int i = 0;i<k\_vertices;i++)

{

float tempx = res[i].x;

float tempy = res[i].y;

res[i].x = tempy\*B[1][0]+tempx\*B[0][0];

res[i].y = res[i].z\*B[2][1]+tempx\*B[0][1]+tempy\*B[1][1];

res[i].z = B[2][3]+res[i].z\*B[2][2]+tempx\*B[0][2]+tempy\*B[1][2];

}

}

//---------------------------------------------------------------------------

void Screen\_conversion(vert res[1000],indexs index[1000],float d,int smech\_x,int smech\_y)

{

float z1,z2; // Частное для получения преобразованных координат

// Находим необходимые нам вершины для соединения и соединяем их

for (int i = 0; i < k\_index; i++)

{

for (int j = 0; j < index[i].number\_of\_index-1; j++)

{

z1 = d\*res[index[i].index\_vertices[j]-1].z+1;

z2 = d\*res[index[i].index\_vertices[j+1]-1].z+1; rastr((res[index[i].index\_vertices[j]-1].x/z1)+smech\_x+300,(res[index[i].index\_vertices[j]-1].y/z1)+smech\_y+300,(res[index[i].index\_vertices[j+1]-1].x/z2)+smech\_x+300,(res[index[i].index\_vertices[j+1]-1].y/z2)+smech\_y+300);

}

z1 = d\*res[index[i].index\_vertices[index[i].number\_of\_index-1]-1].z+1; // Соединяем последнюю вершину с первой

z2 = d\*res[index[i].index\_vertices[0]-1].z+1; rastr((res[index[i].index\_vertices[index[i].number\_of\_index-1]-1].x/z1)+smech\_x+300,(res[index[i].index\_vertices[index[i].number\_of\_index-1]-1].y/z1)+smech\_y+300,(res[index[i].index\_vertices[0]-1].x/z2)+smech\_x+300,(res[index[i].index\_vertices[0]-1].y/z2)+smech\_y+300);

}

}

//---------------------------------------------------------------------------

void rastr(float fx1,float fy1,float fx2,float fy2)

{

float deltaX,deltaY,x,y;

int iterations;

int LX = abs(floor(fx1)-floor(fx2));

int LY = abs(floor(fy1)-floor(fy2));

if(LX>LY) iterations = LX;

else iterations = LY;

x = fx1;

y = fy1;

if(iterations == 0)

{

Form1->Image1->Canvas->Pixels[x][y] = clBlack;

return;

}

deltaX = (fx2-fx1)/iterations;

deltaY = (fy2-fy1)/iterations;

while(iterations)

{

x += deltaX;

y += deltaY;

Form1->Image1->Canvas->Pixels[x][y] = clBlack;

iterations--;

}

}

//---------------------------------------------------------------------------

void sort\_vertices(vert res[1000],indexs index[1000]) // Сортируем преобразованные координаты

{

for(int i = 0;i < k\_index; i++)// Обнуляем наш вектор расстояний

{

for(int j = 0; j < index[i].number\_of\_index; j++) // Пока есть индексы, определяющие вершины

{

mas[i] = 0;

}

}

for(int i = 0;i < k\_index; i++) // Считаем расстояние до экрана для каждого полигона

{

for(int j = 0; j < index[i].number\_of\_index; j++) // Пока есть индексы, определяющие вершины

{

mas[i] += res[index[i].index\_vertices[j]-1].z; // Берем координату z нужной нам вершины по индексу

}

mas[i] /= index[i].number\_of\_index; // Находим среднее арифметическое

}

float temp; // Временная переменная для хранения данных

// Производим сортировку списка

// 1) Должны получить отсортированный список расстояний до экрана по возврастанию

// 2) Поставить индексы вершин согласно отсортированному списку

for(int i = 0; i < k\_index; i++)

{

for(int j = 0; j < k\_index; j++)

{

if(mas[i] >= mas[j]) // Если это значение больше другого

{

for(int k = 0; k < 4; k++)// Производим замену индексов вершин для каждого из значений

{

temp = index[j].index\_vertices[k]; // Заносим k-ый индекс вершины во временную переменную

index[j].index\_vertices[k] = index[i].index\_vertices[k];

index[i].index\_vertices[k] = temp; // Меняем значения местами

}

temp = mas[j]; // Записываем меньшее расстояние во временную переменную

mas[j] = mas[i];

mas[i] = temp; // Меняем значения местами

}

}

}

// Совершаем пересчет кол-ва вершин, указанных индексами

for(int i = 0; i < k\_index; i++)

{

int k = 0; // Кол-во ненулевых вершин

for(int j = 0; j < 4; j++)

{

if(index[i].index\_vertices[j] != 0) k++;

}

index[i].number\_of\_index = k;

}

}

//---------------------------------------------------------------------------

void \_\_fastcall TForm1::Button2Click(TObject \*Sender)

{

sort\_vertices(transf\_vertices,index); // Сортировка полигонов по Z

light(transf\_vertices,index); // Расчет освещения каждого полигона

}

//---------------------------------------------------------------------------

void light(vert res[1000],indexs index[1000]) // Расчет освещения каждого полигона

{

float wiki;

float a[3] = {0,0,1};

float norm[3]; // Координаты нормализированного вектора

float n[3]; // Нормали по x,y,z

float V[2][3]; // Расчетные векторы сторон

float cos\_teta,cos\_phi;

int i,j;

// Рассчитываем нормаль для каждого полигона

for (i = 0; i < k\_index; i++) // Находим векторы сторон

{

// Из координат нынешней вершины вычитаем кординаты следующей

V[0][0] = res[index[i].index\_vertices[0]-1].x - res[index[i].index\_vertices[1]-1].x;

V[0][1] = res[index[i].index\_vertices[0]-1].y - res[index[i].index\_vertices[1]-1].y;

V[0][2] = res[index[i].index\_vertices[0]-1].z - res[index[i].index\_vertices[1]-1].z;

V[1][0] = res[index[i].index\_vertices[1]-1].x - res[index[i].index\_vertices[2]-1].x;

V[1][1] = res[index[i].index\_vertices[1]-1].y - res[index[i].index\_vertices[2]-1].y;

V[1][2] = res[index[i].index\_vertices[1]-1].z - res[index[i].index\_vertices[2]-1].z;

n[0] = V[0][1]\*V[1][2] - V[0][2]\*V[1][1]; // nx

n[1] = V[0][2]\*V[1][0] - V[0][0]\*V[1][2]; // ny

n[2] = V[0][0]\*V[1][1] - V[0][1]\*V[1][0]; // nz

wiki = sqrt(pow(n[0],2)+pow(n[1],2)+pow(n[2],2)); // Нормализация

if(wiki == 0) // Если результат нормализации 0, то вектор - нулевой

{

norm[0] = 0;

norm[1] = 0;

norm[2] = 0;

}

else // Если результат нормализации не 0, то нормализируем координаты вектора

{

norm[0] = n[0]/wiki;

norm[1] = n[1]/wiki;

norm[2] = n[2]/wiki;

}

// Считаем углы для закраски фигуры

cos\_teta = (norm[0]\*a[0]+norm[1]\*a[1]+norm[2]\*a[2])/(sqrt(pow(norm[0],2)+pow(norm[1],2)+pow(norm[2],2))\*(sqrt(pow(a[0],2)+pow(a[1],2)+pow(a[2],2))));

cos\_phi = (norm[0]\*a[0]+norm[1]\*a[1]+norm[2]\*a[2])/(sqrt(pow(norm[0],2)+pow(norm[1],2)+pow(norm[2],2))\*(sqrt(pow(a[0],2)+pow(a[1],2)+pow(a[2],2))));

int k = 0;

float z1,z2;

for (j = 0; j < index[i].number\_of\_index-1; j++)

{

z1 = d\*res[index[i].index\_vertices[j]-1].z+1;

z2 = d\*res[index[i].index\_vertices[j+1]-1].z+1; k = color\_rastr(res[index[i].index\_vertices[j]-1].x/z1+smech\_x+300,res[index[i].index\_vertices[j]-1].y/z1+smech\_y+300,res[index[i].index\_vertices[j+1]-1].x/z2+smech\_x+300,res[index[i].index\_vertices[j+1]-1].y/z2+smech\_y+300,k);

}

z1 = d\*res[index[i].index\_vertices[index[i].number\_of\_index-1]-1].z+1; // Соединяем последнюю вершину с первой

z2 = d\*res[index[i].index\_vertices[0]-1].z+1; k = color\_rastr(res[index[i].index\_vertices[index[i].number\_of\_index-1]-1].x/z1+smech\_x+300,res[index[i].index\_vertices[index[i].number\_of\_index-1]-1].y/z1+smech\_y+300,res[index[i].index\_vertices[0]-1].x/z2+smech\_x+300,res[index[i].index\_vertices[0]-1].y/z2+smech\_y+300,k);

color\_image(cos\_teta,cos\_phi,res,i,j,k);

}

}

//---------------------------------------------------------------------------

void color\_image(float cos\_phi,float cos\_teta,struct vert res[1000],int i,int j,int s)

{

// Нам известно кол-во координат, которые изменили координату Y

for(int k = 0;k<s;k++)

{

for(int l = 0;l<s;l++)

{

if(coord[k].y == coord[l].y && k!=l)

{

if(cos\_phi>0) // Если свет падает на полигон

{

Form1->Image1->Canvas->Pen->Color=RGB(0,cos\_phi\*255,cos\_teta\*255);

Form1->Image1->Canvas->MoveTo(coord[k].x,coord[k].y);

Form1->Image1->Canvas->LineTo(coord[l].x,coord[l].y);

Form1->Image1->Canvas->Pen->Color=clBlack;

}

}

}

}

for(int j = 0; j < s; j++)

{

coord[j].x = 0;

coord[j].y = 0;

}

Form1->Image1->Refresh();

s = 0;

}

int color\_rastr(float fx1,float fy1,float fx2,float fy2,int k)

{

float deltaX,deltaY,x,y;

int iterations;

int LX = abs(floor(fx1)-floor(fx2));

int LY = abs(floor(fy1)-floor(fy2));

if(LX>LY) iterations = LX;

else iterations = LY;

x = fx1;

y = fy1;

if(iterations == 0)

{

Form1->Image1->Canvas->Pixels[x][y] = clBlack;

return 0;

}

deltaX = (fx2-fx1)/iterations;

deltaY = (fy2-fy1)/iterations;

while(iterations)

{

x += deltaX;

float temp = y; // Записываем старый y

y += deltaY;

if(floor(temp) != floor(y)) // Если y изменился

{

coord[k].x = x - deltaX;

coord[k].y = y - deltaX;

coord[k+1].x = x;

coord[k+1].y = y;

k += 2;

}

iterations--;

}

return k;

}