**МИНОБРНАУКИ РОССИИ**

**Санкт-Петербургский государственный**

**электротехнический университет**

**«ЛЭТИ» им. В.И. Ульянова (Ленина)**

**Кафедра САПР**

отчет

**по лабораторной работе №5**

**по дисциплине «Компьютерная графика»**

Тема: **Исследование алгоритмов выявления видимости сложных сцен**

|  |  |  |
| --- | --- | --- |
| Студент гр. 7307 |  | Торопов В.А. |
| Преподаватель |  | Матвеева И.В. |

**Цель работы**

Обеспечить реализацию алгоритма выявления видимых граней и ребер для одиночного выпуклого объемного тела.

**Основные теоретические положения**

Алгоритм Робертса представляет собой первое известное решение задачи об удалении невидимых линий. Это математически элегантный метод, работающий в объектном пространстве. Алгоритм прежде всего удаляет из каждого тела те ребра или грани, которые экранируются самим телом. Затем каждое из видимых ребер каждого тела сравнивается с каждым из оставшихся тел для определения того, какая его часть или части, если таковые есть, экранируются этими телами. Поэтому вычислительная трудоемкость алгоритма Робертса растет теоретически как квадрат числа объектов.

В алгоритме Робертса требуется, чтобы все изображаемые тела или объекты были выпуклыми. Невыпуклые тела должны быть разбиты на выпуклые части. В этом алгоритме выпуклое многогранное тело с плоскими гранями должно представляться набором пересекающихся плоскостей. Уравнение произвольной плоскости в трехмерном пространстве имеет вид:

aх + by + cz + d = 0

Если любая точка S(xs, ys, zs) лежит на плоскости, то axs + bys + czs + d = 0. Если же S не лежит на плоскости, то знак этого скалярного произведения показывает, по какую сторону от плоскости расположена точка. В алгоритме Робертса предполагается, что точки, лежащие внутри тела, дают положительное скалярное произведение.

Пусть F1, F2, …, Fn — грани многогранника. Рассмотрим одну из граней. Обозначим вершины, инцидентные грани, через V1, V2, …, Vk. Найдем вектор нормали к грани, вычислив векторное произведение любых двух смежных ребер этой грани V1V2 = [x1,y1,z1] и V2V3 = [x2,y2,z2]:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| ni = [V1V2,V2V3] = |  | i     j    k x1  y1  z1 x2  y2  z2 |  | = (y1z2 — y2z1)·i + (z1x2 — z2x1)·j + (x1y2 — x2y1)·k = Ai·i + Bi·j + Ci·k |

Тогда опорная функция грани имеет вид:

Li(x,y,z) = Aiх + Biy + Ciz + D

Величина D вычисляется с помощью произвольной точки на плоскости. В частности, если компоненты этой точки на плоскости (х1,y1,z1), то:

D = -(ах1 + by1 + cz1)

Так как многогранник выпуклый, коэффициенты Ai, Bi, Ci легко выбрать так, чтобы ni(Ai, Bi, Ci) был вектором внешней нормали. Для этого найдем какую-либо внутреннюю точку, например, барицентр многогранника:

W = (V1 + V2 + … + Vk)/k

Если скалярное произведение уравнения плоскости и этой точки меньше 0, то необходимо поменять знак уравнения этой плоскости, чтобы отразить правильное направление внешней нормали. Остается только вычислить скалярное произведение уравнения плоскости на точку, в которой находится наблюдатель. Если это скалярное произведение меньше 0, то плоскость невидима и необходимо удалить весь многоугольник, лежащий в этой плоскости.

**Математическое обоснование (Python)**

Реализация алгоритма Робертса:

def Roberts(data):  
 *"""  
 The Roberts algorithm* ***:param*** *data: matrix of the coordinates of the cube* ***:return****: the matrix of transformed coordinates of the cube  
 """* verts = data[0]  
 W = np.zeros(3)  
 P = [1, -1, 1]  
  
 # search for the barycenter of a cube  
 for i in range(2):  
 max\_value = -10  
 min\_value = 10  
 for point in verts:  
 max\_value1 = max(point[j][0] for j in range(len(point)))  
 max\_value = max(max\_value, max\_value1)  
 min\_value1 = min(point[j][0] for j in range(len(point)))  
 min\_value = min(min\_value, min\_value1)  
 W[i] = (max\_value + min\_value)/2  
  
 new\_data = []  
  
 # loop across all faces  
 for vector in verts:  
 # find the coordinates of two vectors that lie in the plane of the face  
 Vec1\_x = vector[0][0] - vector[1][0]  
 Vec2\_x = vector[2][0] - vector[1][0]  
 Vec1\_y = vector[0][1] - vector[1][1]  
 Vec2\_y = vector[2][1] - vector[1][1]  
 Vec1\_z = vector[0][2] - vector[1][2]  
 Vec2\_z = vector[2][2] - vector[1][2]  
  
 # calculate the coefficients of the plane equation  
 A = Vec1\_y\*Vec2\_z - Vec2\_y\*Vec1\_z  
 B = Vec1\_z\*Vec2\_x - Vec2\_z\*Vec1\_x  
 C = Vec1\_x\*Vec2\_y - Vec2\_x\*Vec1\_y  
 D = -(A\*vector[0][0] + B\*vector[0][1] + C\*vector[0][2])  
  
 # find the coefficient that changes the sign of the plane  
 m = -(A\*W[0] + B\*W[1] + C\*W[2] + D)  
  
 # correcting the direction of the plane  
 A = A\*m  
 B = B\*m  
 C = C\*m  
 D = D\*m  
  
 # defining the visibility of faces  
 if (A\*P[0] + B\*P[1] + C\*P[2] + D) > 0:  
 new\_data.append(vector)  
  
 return new\_data

**Экспериментальные результаты.**

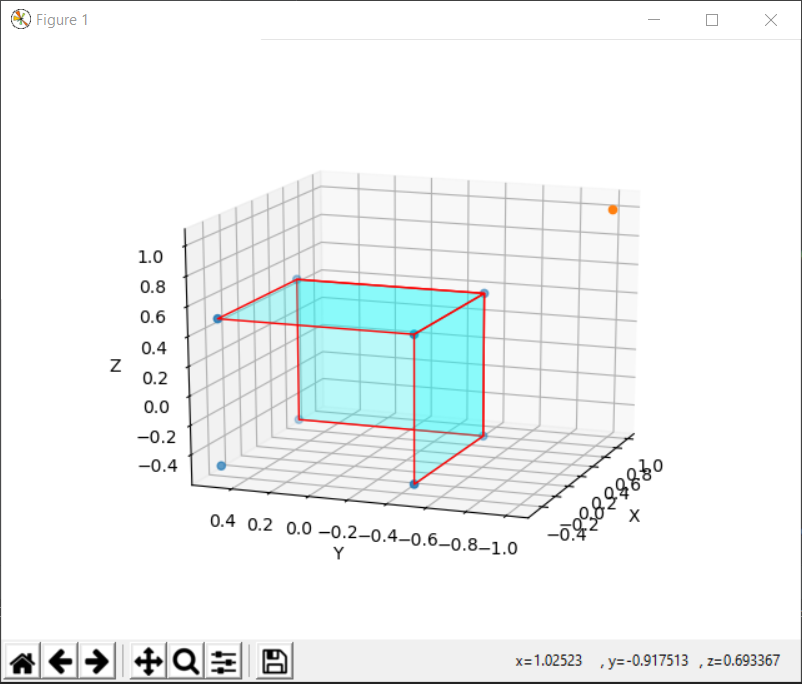
На рисунке 1 представлено выявление видимых граней куба. Оранжевая точка – зритель.

Рисунок 1 – выявление видимых граней

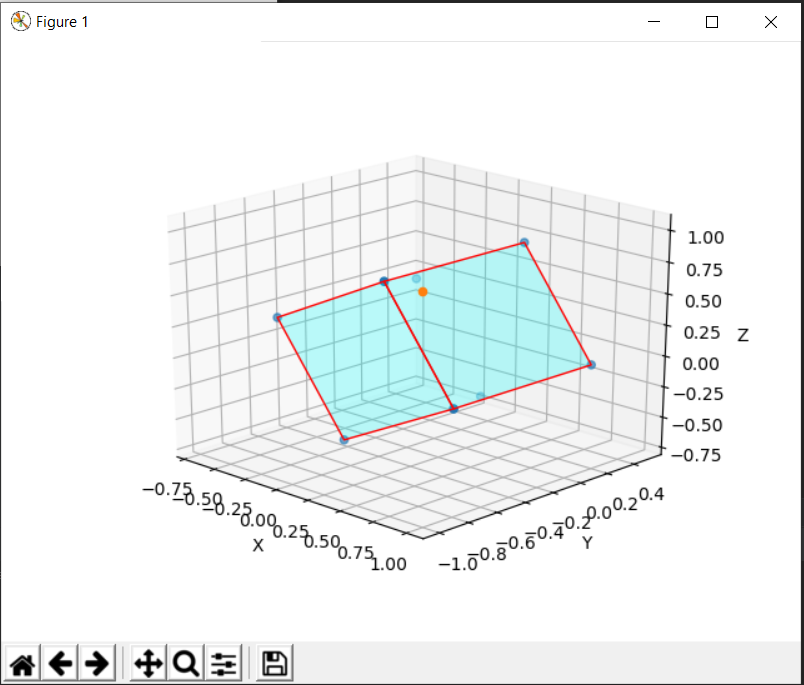
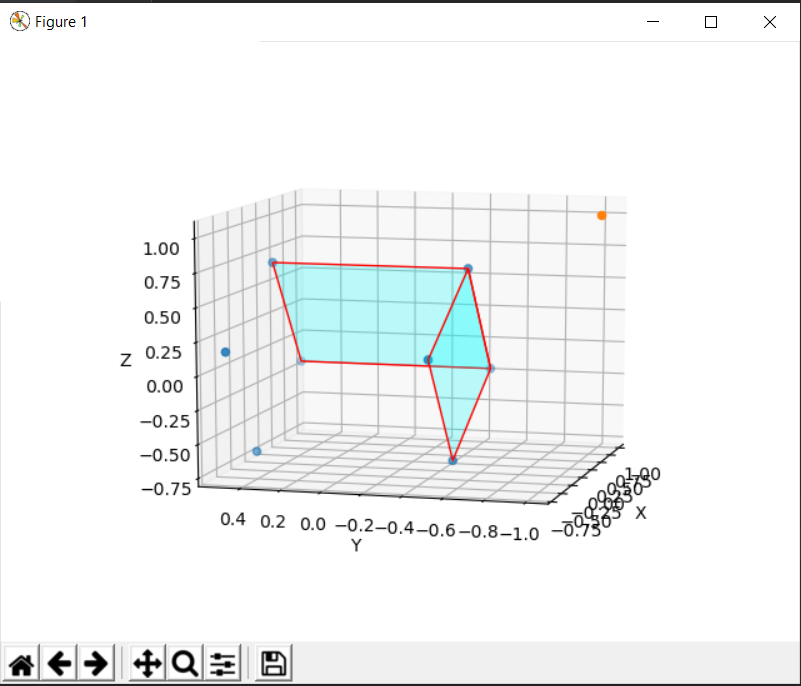
Повернём куб на 45̊ и посмотрим на результат. На рисунке 2.1 представлен вид зрителя, на рисунке 2.2 вид со стороны:

Рисунок 2.1 -–выявление видимых граней с углом 45. Вид зрителя

Рисунок 2.2 -–выявление видимых граней с углом 45. Вид со стороны

**Вывод**

В данной лабораторной работе был реализован алгоритм выявления видимых граней и ребер для одиночного выпуклого объемного тела.