Отчет по проекту

Ray tracing

Ларов Даниил

5103, Мфти

Общая задача:

Построение изображения сцены с трехмерными сферами методом обратной трассировки лучей.

Используемый язык:

Стандартная библиотека C++ и Qt.

Входные данные

Текстовый файл (cond.txt) со следующим содержанием:

1 << количество источников света

0 0 0 5 << X Y Z света и его интенсивность (0 - 10)

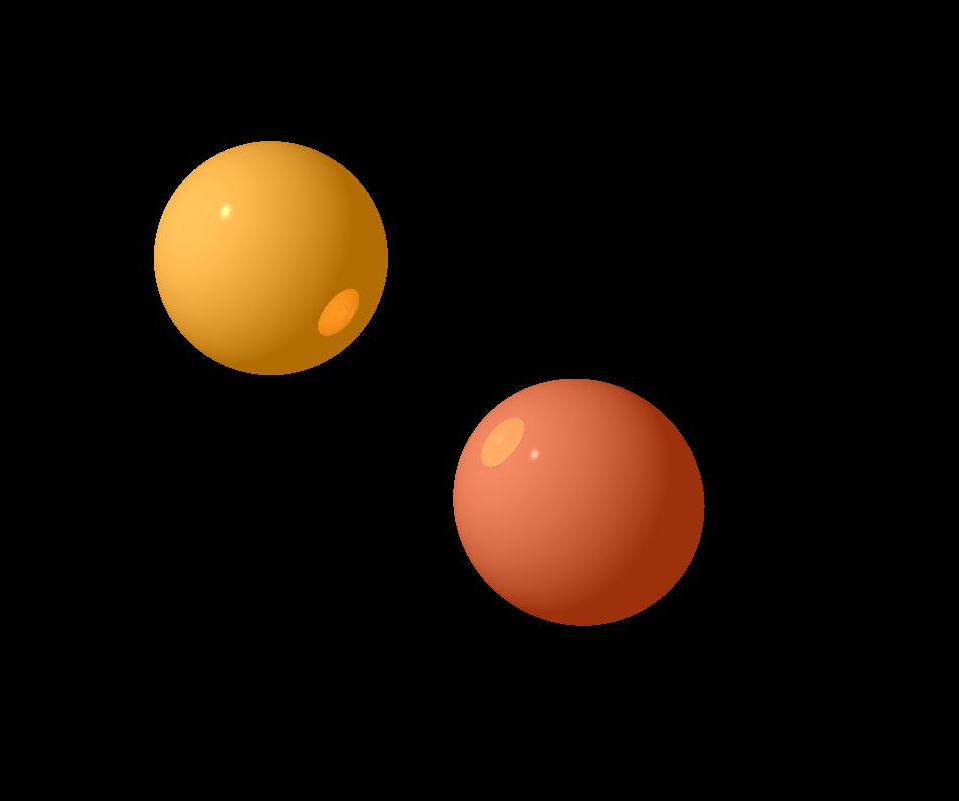
2 << к-во сфер

200 1000 1000 500 255 156 7 7 << R X Y Z R G B и коэф. отражения (0 - 10)

200 1500 900 900 225 71 17 7

Выходные данные:

Файл-изображение формата .jpg



Основные объекты, параметры и структуры, используемые в программы

**Константные переменные:**

*Разрешение:*

int width - ширина получаемого изображения

int height - высота получаемого изображения

*Константы для бликов:*

Float Ks – яркость блика (0 - 1)

Float p – влияет на размытость блика ( 0-200 (чем меньше, тем размытее))

*Прочее:*

int MAXrecurs

**Основные структуры данных:**

struct lightning{ //источник света

int x; //

int y; // расположение в комнате

int z; //

float Il; //интенсивность света

};

struct object { // Сфера

int x; //

int y; // Расположение в комнате

int z; //

int R; // Радиус сферы

int Red;

int Green; // компоненты цвета сферы

int Blue;

float Kd; // коэф. Отражения сферы (0 - 1)

};

struct viewer{

int x;

int y; // расстояние до изображения

int z;

};

**Общая идея алгоритма:**

Имеем сцену с примитивами(сферами), сетку (матрицу) пикселей а также камеру, расположенную за сеткой пикселей на заданном расстоянии перед заданным пикселем.

Через каждый пиксель проводится прямая с заданным направлением. Отслеживая движение этой прямой, мы определяем интенсивность красного зеленого и синего для данного пикселя и заносим его цвет в файл через цикл for.

Направление луча, проходящего через один из пикселей определяем следующим образом:

a = ox – camera.x <<<ox – это расположение пикселя по горизонтали

b = oy – camera.y <<< oy=0

c = oz – camera.z <<< oz – это расположение пикселя по вертикали

Общее уравнение для луча идущего из сетки пикселей в сцену выглядит следующим образом:

X = ox + a \* t

Y = oy + b \* t

Z = oz + c \* t

После этого запускается функция **Intensity**.

Intensity

1. Определяется пересечение луча со всеми сферами. Если пересечение не найдено то функция завершается, а интенсивность всех цветов остается равной 0.
2. Если пересечение найдено и шаг рекурсии не больше MAXrecurs, то определяется точка пересечения со сферой. После этого находится луч отражения и снова запускается функция Intensity. Так будет продолжатся до тех пор, пока шаг рекурсии будет меньше MAXrecurs или луч не отправится в пустоту.
3. Теперь, имея интенсивность по отраженному лучу определяется интенсивность цвета, где мы сейчас находимся. Интенсивность каждого цвета определяется следующим образом:

**Ir**= sphere[numberspher].Red / 255 \* sphere[f].Kd + sphere[f].Kd \* **Shadow** + Ks \* **Bliks** + (1-sphere[f].Kd) \* **IotrRi**;

**Ig**=sphere[numberspher].Green / 255 \* sphere[f].Kd + sphere[f].Kd \* **Shadow** + Ks \* **Bliks** + (1-sphere[f].Kd) \* **IotrGi**;

**Ib**=sphere[numberspher].Blue / 255 \* sphere[f].Kd + sphere[f].Kd \* **Shadow** + Ks \* **Bliks** + (1-sphere[f].Kd) \* **IotrBi**;

Где переменные **Shadow** и **Bliks** определяются следующим образом:

**Shadow** = Cosa \* light[j].Il; << Cosa – это косинус угла между источником света и нормалью в точке пересечения

**Bliks** = pow(Cosb,p) \* light[j].Il; <<Cosb – это косинус угла между отраженным лучом от источника света и лучом, направленным в камеру

**IotrRi, IotrGi, IotrBi** – это **Ir,Ig,Ib** для отраженного луча, которые определяются через рекурсию.

Математические предпосылки

**Пересечение со сферой**

Коэффициент t, который необходим для поиска Xinters, Yinters и Zinters находится из решения следующего квадратного уравнения:

At^2 + Bt +C = 0

A= a\*a+b\*b+c\*c

B= 2\*(X\*a+Y\*b+Z\*c)

C= (X\*X+Y\*Y+Z\*Z-(sphere.R)\*(sphere.R))

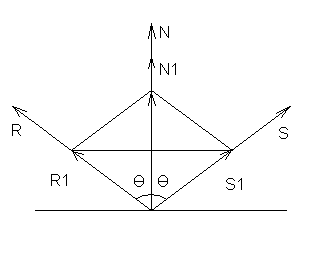
X=Xo-sphere.x; // sphere.x sphere.y sphere.z sphere.R центр и радиус сферы

Y=Yo-sphere.y; // Xo,Yo,Zo – точка, откуда идет луч

Z=Zo-sphere.Z; // a,b,c – направление луча

### 

### Вычисление отраженного луча



xR = 2xN(xNxS+yNyS+zNzS)/(xN2+yN2+zN2) – xS,

yR = 2yN(xNxS+yNyS+zNzS)/(xN2+yN2+zN2) – yS,

zR = 2zN(xNxS+yNyS+zNzS)/(xN2+yN2+zN2) – zS,