Пользовательская документация к проекту:

RayTracing

Автор: Евтеева А.В.

4103 группа

ФНБИК 2016

Содержание:

Цель проекта:

Программа "RayTracing" предназначена для построения изображений из трёхмерной сцены методом трассировки лучей. Пользователь заносит в файл формата TXT данные об объектах сцены и на выходе получает соответствующее изображение.

Описание алгоритма трассировки лучей

Суть алгоритма

Есть наблюдатель, смотрящий на сцену. Поставим между ним и сценой пиксельную сетку с некоторым разрешением и в каждый пиксель этой сетки будем записывать информацию о цвете луча, проходящего через этот пиксель в точку наблюдения (в глаз). Таким образом мы получим информацию о сцене, проецированную на пиксельную сетку изображения. Но обсчитывать все лучи, генерируемые источниками лучей (источниками света) в сцене очень трудоемко (их, вообще говоря, бесконечное количество). поэтому поступают иначе. Из точки наблюдения генерируются лучи, определяется, могли ли они придти из источника света, если да, то с какими характеристиками.

Таким образом, алгоритм следующий

1) Для каждого пикселя на изображении, мы вычисляем его положение в трехмерном пространстве и пускаем луч из точки наблюдения в заданном направлении
2) Для каждого объекта на сцене вычисляем, пересекается ли он с лучом, находим ближайший объект и точку его пересечения
3) Для каждого источника света вычисляем направление на этот источник
4) Для каждого объекта в сцене определяем, пересекается ли луч на источник света с объектом (в тени ли точка)
5) Если точка не в тени, то добавим к цвету интенсивность света, которую мы вычисляем с помощью моделей освещения Ламберта и Фонга

Реализация алгоритма

Использованные библиотеки и модули

Базовый язык программирования - python 3.5.2

Для реализации программы дополнительно были подключены:

1) Модули Image и ImageTk из Python Imaging Library (сокращенно PIL)
2) Модуль os.path из стандартной библиотеки Python
3) Библиотека numPy
4) Библиотека matplotlib.pyplot
5) Модуль readfile2 – написан мною для считывания данных из файла

Описание функций

def make_3Dimage(filename,w,h)
Основная функция, принимает на вход:

- filename – имя текстового файла, который содержит информацию об объектах на сцене
- w – ширина изображения
- h – длина изображения

Содержит внутри себя следующие функции:

1) *def normalize(x)*
   Функция нормализации вектора *x*
   Возвращает нормализованный вектор
2) *def ray_intersects(objectType,parameter1, parameter2, startPoint,directionPoint):*
   Функция определения пересечения луча с объектом сцены
   - *objectType* – тип объекта, в нашей программе это либо сфера, либо плоскость
   - *parameter1* – в зависимости от объекта это либо радиус-вектор(плоскость), либо позиция в трехмерном пространстве(сфера)
   - *parameter2* - в зависимости от объекта это либо нормаль(плоскость), либо радиус(сфера)
   - *startPoint* – точка, из которой выпускается луч
   - *directionPoint* – направление луча

   Возвращает точку пересечения с лучом
3) *def normal(obj, M)*
   Функция получения нормали
   - *obj* – объект сцены, с которым пересекся луч
   - *M* – точка пересечения

   Возвращает нормаль объекта

4) *def tracing(startPoint, directionPoint,scene)*
   Функция, которая находит ближайший объект, пересекающийся с данным лучем, проверяет, не в тени ли этого объекта остальные объекты и добавляет к цвету интенсивность света в соответствии с моделями Ламберта и Фонга
   - *startPoint* – точка, из которой выпускается луч
   - *directionPoint* – направление луча
   - *scene* – трехмерная сцена

Возвращает ближайший объект, пересекающийся с данным лучем, точку пересечения, нормаль объекта и цвет

5) В основной части функции строится рисунок по соответствующему алгоритму и сохраняется изображение в формате png

def read_fpath_w_h(event)

Функция принимает на вход некое событие – нажатие пользователем кнопки "Enter”, вызывает функцию make_3Dimage, тем самым создает изображение и выводит его в отдельном окне. (Подробнее описано в разделе «Инструкция для пользователя. Графический интерфейс»)

Модульное тестирование

Для модульного тестирования использовались:

- Библиотека numpy
- Модуль unittest

Были импортированы модули module.py и readfile2.py – непосредственная программа без интерфейса и программа для считывания данных из текстового файла

В программе имеется класс FTestCase(unittest.TestCase), в котором содержатся следующие функции:

- *def test_readData(self)*

Функция проверки правильности считывания данных из текстового файла

- *def test_planeInteresect(self):*

Функция проверки правильности нахождения точки пересечения плоскости с лучом

- *def test_sphereInteresect(self):*

Функция проверки правильности нахождения точки пересечения сферы с лучом

Инструкция для пользователя

Заполнение текстового файла

В текстовом файле должны быть следующие строки

Light: *,*,* - точка расположения источника света в трехмерном пространстве

ambient: * - фоновая компонента освещения

diffuse_c: * - рассеянная компонента освещения

specular_c: * - зеркальная компонента освещения

specular_k: * – коэффициент блеска

depth_max: # - максимальное количество отражений

Camera: *,*,* - расположение камеры в трехмерном пространстве

Camera pointing to: *,*,* - вектор, куда указывает камера

Plane: position; color; normal; diffuse; reflection; specular_c

,,*;*,*,*;*,*,*;*;*;*

Sphere: position; radius; color; reflection

,,*;*;*,*,*;*

…

,,*;*;*,*,*;*

Пояснение:

* - числа типа float, # - числа типа int

Работа с графическим интерфейсом

В окно “File” пользователь вводит имя файла.

В окна “Width” и “Height” – ширину и высоту нужного изображения.

Далее нажимает кнопку “Enter”

Если все значения введены корректно, то откроется новое окно с изображением, иначе появится надпись “Enter correct value”