БЕЛОРУССКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ
ФАКУЛЬТЕТ ПРИКЛАДНОЙ МАТЕМАТИКИ И ИНФОРМАТИКИ

Пояснительная записка к проекту **Color Cube**

Автор:
Архангельский И.А.

30 апреля 2012 г.

Содержание

Часть 1

Соответсвие техническому заданию

Возможности менять положение наблюдателя

Вращение куба фактически, есть изменение положения наблюдателя вокруг куба. Изменение размера куба, есть изменение расстояния между наблюдателем и кубом.

Вращать куб

Реализовано

Менять размер куба

Реализовано

Задавать шаг (в примере 16x16x16, а надо бы и 8x8x8)

Реализовано изменение количества слоев в диапазоне [2 : 20], отрисовка с большего количества слоев требует более оптимальных алгоритмов отрисовки(Z-buffer, двойная буфферизация) для которых процессор не предназначен, а значит использование видеокарты напрямую, что ограничевает кроссплатформенность приложения. Однако, класс реализующий вычисления достаточно абстрактен для того, чтобы использовать его с любой библиотекой реализующей графический вывод на экран средствами видеокарты.

Включать отключать вывод рёбер.

Не реализовано

Построение сечений

Не реализовано

Часть 2

Коментарии и пояснения по реализации

Куб задается 8 точками в пространстве (вершины). Центр куба находится в точке $(0, 0, 0)$. Грани строятся по 4 точкам.

1 Вращение

Вращение куба реализовано с при помощи преобразования координат. Если рассматривать вершину куба как вектор, то поворот в некоторой плоскости есть умножение вектора на специальную матрицу.

Rolling. Вращение вокруг оси x

Матрица поворота: $M_x(\alpha) = \begin{pmatrix} 1 & 0 & 0 \\ 0 & \cos\alpha & -\sin\alpha \\ 0 & \sin\alpha & \cos\alpha \end{pmatrix}$

Pitching. Вращение вокруг оси y

Матрица поворота $M_y(\alpha) = \begin{pmatrix} \cos\alpha & 0 & \sin\alpha \\ 0 & 1 & 0 \\ -\sin\alpha & 0 & \cos\alpha \end{pmatrix}$

Yawing. Вращение вокруг оси z

Матрица поворота: $M_z(\alpha) = \begin{pmatrix} \cos\alpha & -\sin\alpha & 0 \\ \sin\alpha & \cos\alpha & 0 \\ 0 & 0 & 1 \end{pmatrix}$

2 Проецирование

Для проецирования было решено использовать диметрическую проекцию. То есть X и Y координаты искажаются. Формула искажения была выбрана такая :

$$x' = x\frac{d - z}{d + a}$$

где,

d - это некоторая константа, фактически отвечающая за расстояние от объекта до плоскости экрана.

a - длинна ребра куба.

В текущей реализации $d = 8a$ Точно такой же коэффициент используется для преобразования

$$y' = y\frac{d - z}{d + a}$$

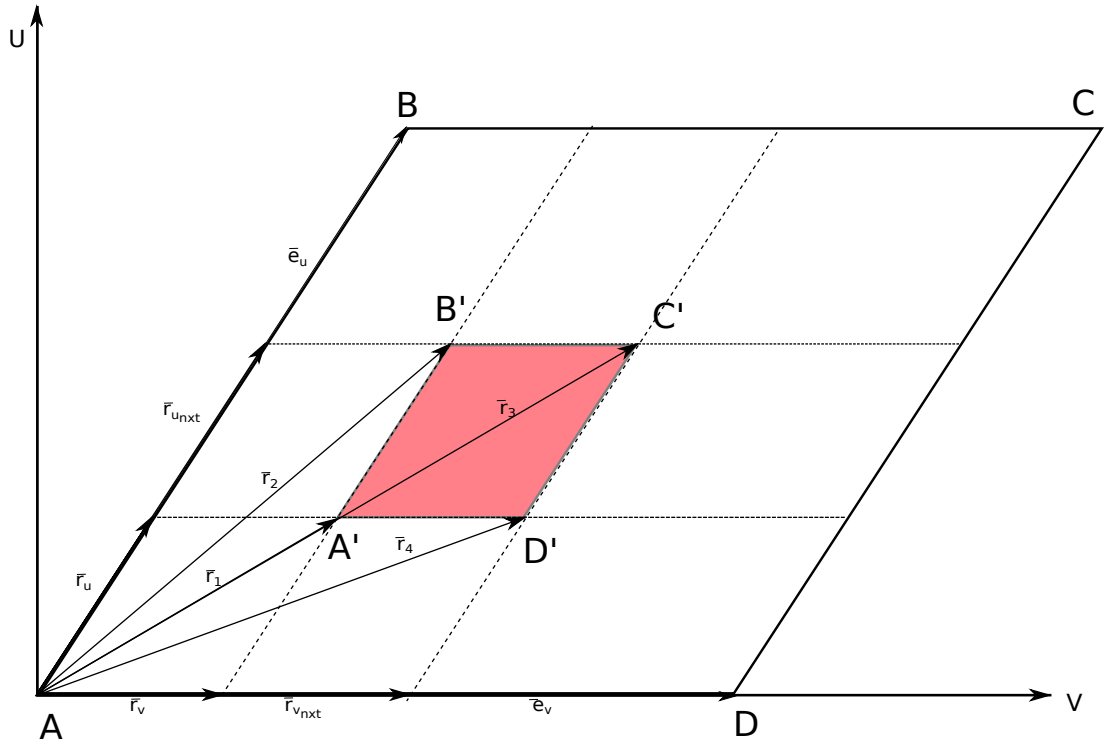
Для создания эффекта схода параллельных прямых мы "искажаем"координаты. Суть в том, что два одинаковых по длине отрезка, будут отображены различной длинны (в случае прямой перспективы, тот что ближе к точке наблюдения будет выглядеть длиннее). Для этого мы искажаем координаты по двум осям (X и Y).

3 Изменение длины ребра

Так как куб отпозиционирован так, что его центр находится в точке $(0, 0, 0)$, для изменения длины ребра куба, достаточно изменить длинну векторов, которые соответсвуют вершинам куба. Тем самым к каждой вершине применяем преобразование

$$\begin{cases} x' = x\frac{l^{\frac{1}{3}}}{\sqrt{x^2+y^2+z^2}} \\ y' = y\frac{l^{\frac{1}{3}}}{\sqrt{x^2+y^2+z^2}} \\ z' = z\frac{l^{\frac{1}{3}}}{\sqrt{x^2+y^2+z^2}} \end{cases}$$

4 Разбиение на сегменты



Пусть имеем грань куба, уже спроецированную на плоскость UV (плоскость экрана). И грань в пространстве UV описывается четырьмя точками $A(v_a, u_a), B(v_b, u_b), C(v_c, u_c), D(v_d, u_d)$. Построим векторы $\vec{e_u}$ и $\vec{e_v}$:

$$\vec{e_u} = (v_b, u_b)$$
$$\vec{e_v} = (v_d, u_d)$$

Пологаем, что стороны не обязательно параллельны осям, но точка совпадает с началом координат.[1] Пусть c - количество сегментов в разбиении по каждому вектору $(\vec{e_u}, \vec{e_v})$, а i, j - индексы текущего сегмента. Тогда, векторы $\vec{r_u}, \vec{r_{u_{nxt}}}, \vec{r_v}, \vec{r_{v_{nxt}}}$ вычисляются по формулам$(\forall i, j = \overline{1, 2, \cdots, c-1})$:

$$\vec{r_u} = \frac{i}{c}\vec{e_u}$$

$$\vec{r_v} = \frac{j}{c}\vec{e_v}$$

$$\vec{r_{u_{nxt}}} = \frac{i+1}{c}\vec{e_u}$$
$$\vec{r_{v_{nxt}}} = \frac{j+1}{c}\vec{e_v}$$

Тогда точке A' сегмента соответсвует вектор $\vec{r_u} + \vec{r_v}$, точке $B' - \vec{r_{u_{nxt}}} + \vec{r_v}$, точке $C' - \vec{r_{u_{nxt}}} + \vec{r_{v_{nxt}}}$, точке $D' - \vec{r_u} + \vec{r_{v_{nxt}}}$. Координаты сегмента цветового куба определяются в полной аналогии, но работаетм в пространстве RGB. Найденный сегмент закрашивается средним цветовым значением сегмента.

[1] В реализации разбиения это вообще говоря не так, но сделаем это допущение для простоты изложения.

5 Преобразование перемещения мыши в углы поворота

Пусть $roll$ – угол поворота вокруг оси X, $pitch$ – угол поворота вокруг оси Y, yaw – угол поворота вокруг оси Z
Δx, Δy – изменение координаты указателя мыши на экране
k - коэффициент (в текущей реализации $k = 3600$)
Тогда новые углы поворота вычисляются по формулам:

$$roll' = roll - \Delta y \frac{2\pi}{k}$$

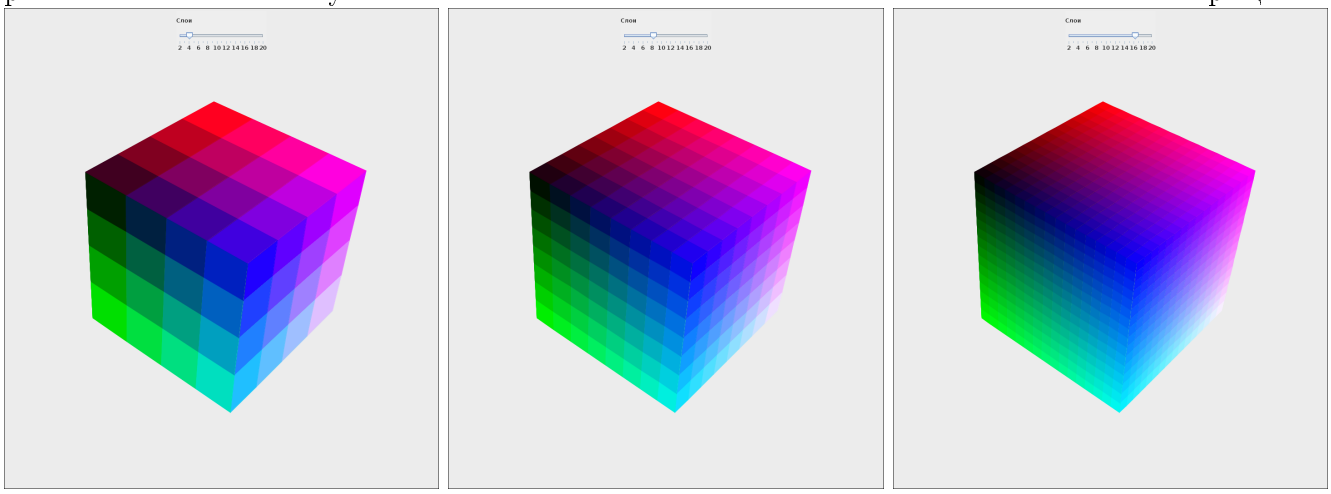
$$pitch' = pitch - \Delta y \frac{2\pi}{k}$$

$$yaw' = \begin{cases} yaw - \frac{2\pi}{k}\frac{\Delta x}{\Delta y}, y \neq 0 \\ yaw - \frac{2\pi}{k}\Delta x, y = 0 \end{cases}$$

Часть 3

Управление

Вращение колеса мыши изменяет размер куба. Перемещение мыши с зажатой ПКМ или ЛКМ поворачивает куб. Клик останавливает вращение.



Бегунок изменяет количество слоев.

Часть 4

Ссылки

- Последняя ревизия исходных текстов проекта
- Исходный текст LaTeXэтого документа
- Последняя ревизия этого документа
- Сгенерированная документация проекта