Порхающие бабочки – групповая анимация объектов.

Автор: Васильчикова Елена, студентка 3-го курса
Руководитель:. Кульман Никита Юрьевич , кандидат физико-математических наук, доцент кафедры «Информационных технологии»
Образовательное учреждение: филиал «Протвино» Международного университета природы, общества и человека «Дубна», г. Протвино

В работе изучаются методы анимации группы объектов на примере порхающих бабочек. Разработано приложение, выводящее на экран группу движущихся фигур. Используется загрузка текстур из файла.

Butterflies – group objects animation.

Vasilchikova E.

This article describes the questions to ask when studying methods for a group objects animation. Butterflies are used to as example such objects. an application An application was developed as a result a group of moving figures displays. It was used texture loading from a file.

Работа выполнялась с помощью библиотеки **OpenGL** (Open Graphics Library — открытая графическая библиотека, графическое API), определяющей независимый от языка программирования программный интерфейс для написания приложений, использующих двухмерную и трёхмерную компьютерную графику для различных ОС.

OpenGL включает в себя около 150 различных функций, при помощи которых программист может задавать свойства различных трехмерных и двумерных объектов и выполнять их визуализацию. Таким образом, в программе надо задавать местоположение объектов в трехмерном пространстве, определять другие параметры (поворот, растяжение), задавать свойства объектов (цвет, текстуру, материал), положение наблюдателя, а затем библиотека OpenGL выполнит генерацию двумерной этой трехмерной сцены.

В работе использовались следующие **возможности OpenGL:**

- Создание геометрических и растровых примитивов

Примитив - это фигура, такая как точка, линия, многоугольники, прямоугольник пикселов или битовый массив, которая рисуется, хранится и которой манипулируют как единой дискретной сущностью. Другими словами, примитивы — это те элементы, из которых строятся графические объекты любой степени сложности. Они определяются группами из одной или нескольких вершин, каждая из которых имеет ассоциированные с ней данные: текущий цвет, текущая позиция растра, текущая нормаль, текущие координаты текстуры.

- Применение видовых и координатных преобразований

Для: перехода к новым координатам в OpenGL есть две функции: glTranslatcd (Dx,Dy,Dz) - сдвигает начало системы координат на (Dx,Dy,Dz), glRotated(p.xO,yO,zO) - поворачивает на угол φ против часовой стрелки вокруг вектора (xO.yO.zO).

Удаление невидимых линий и поверхностей

Использование сплайнов для построения линий и поверхностей

Наложение текстуры и применение освещения

Для создания реалистичных изображений необходимо определить как свойства самого объекта, так и свойства среды, в которой он находится. Первая группа свойств включает в себя параметры материала, из которого сделан объект, способы нанесения текстуры на его поверхность, степень прозрачности объекта. Ко второй группе можно отнести количество и свойства источников света, уровень прозрачности среды, а также

модель освещения. Все эти свойства можно задавать, вызывая соответствующие команды OpenGL.

- Добавление специальных эффектов: тумана, изменение прозрачности, сопряжение цветов, устранение ступенчатости.

Иногда, компьютерные изображения могут казаться нереально четкими я резкими. Антиалиасинг делает объекты более реалистичными за счет сглаживания их ребер.

Эффект ступенчатости возникает в результате погрешностей растеризации примитивов в буфере кадра из-за конечного разрешения буфера. Есть несколько подходов к решению данной проблемы. Например, можно применять фильтрацию полученного изображения. Также этот эффект можно устранять на этапе растеризации, сглаживая образ каждого примитива.

Можно выделить следующие **этапы создания приложения.**

1. Сперва было сформировано изображение в файле ресурсов.

2. Была разработана структура для информации о 50 различных объектах, которые должны двигаться по экрану.

tex - содержит информацию о текстуре объекта.

x – это положение объекта по оси X,

y- это Y положение объекта,

z – положение объекта по оси Z,

yi будет случайным числом, описывающим скорость падения объекта

spinz используется для вращения объекта по оси z во время падения

spinzi это еще одно случайное число для описания скорости вращения объекта

flap используется для описания крыльев объекта

fi – это случайное число описывающее направление хлопка крыльев.

3. Изображение из файла ресурсов было преобразовано в текстуру. **hBMP** – это указатель на файл изображения. Он говорит о том, откуда взять данные. **BMP** - это структура изображения, которую можно заполнить данными из файла ресурсов.

4. Была использована функция **LoadImage** для создания текстур из изображений.

5. Использована функция **glPixelStorei**, которая говорит OpenGL о том, что данные сохранены с выравниванием по размеру двойного слова (4 байта на каждый пиксель).

6. Произведено связывание текстуры, для установления фильтра на плавное сглаживание и генерацию текстуры.

7. Добавлена функция **LoadGLTextures** для загрузки текстур. Цвет пустого экрана – черный. Проверка на глубину выключена.

8. Было инициировано 50 обьектов таким образом, чтобы они не появлялись все сразу в середине экрана или в одном и том же месте, а были распределены естественным образом.

9. Изображение можно поместить на любую поверхность. Изображение может совпасть с формой поверхности, но может также принять совершенно другую форму.

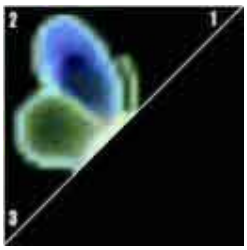
10. Вызвана функция **glLoadIdentity()** для сброса матрицы просмотра модели. Затем выбрана текстура, которая была присвоена объекту (**obj[loop].tex**). Бабочка помещается на экран с помощью **glTranslatef()** и поворачивается на 45 градусов по оси X. В этом случае бабочка будет наклонена слегка вперед к зрителю, чтобы не было эффекта плоского 2-мерного объекта.

11. Заключительное вращение закрутит бабочку по оси **Z**, пока она будет падать вниз на экране.

12. Было сделано текстурирование треугольника, которое почти не отличается от стандартного текстурирования квадрата. Единственное отличие заключается в том, что

надо немного больше позаботиться о правильности задания координат текстуры.

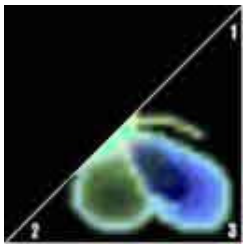
13. Создание первого треугольника начинается с верхнего правого угла невидимого квадрата. Затем двигаемся влево, пока не достигнем верхнего левого угла. Отсюда идем в нижний левый угол.



Половина бабочки рисуется на первом треугольнике. Другая половина рисуется на втором треугольнике. Координаты текстуры совпадают с координатами вершин и, хотя задается всего 3 координаты для текстуры, этого оказалось достаточно, чтобы OpenGL определил какая порция изображения должна быть нарисована в треугольнике.

14. Тот же подход был применен для второй половины треугольника, но теперь рисование начинается с верхнего правого угла и далее в нижний левый, а затем в правый нижний угол.

Вторая точка первого треугольника и третья точка второго треугольника движутся вперед-назад по оси Z, что создает иллюзию хлопающих крыльев. Если посмотреть на оба изображения можно заметить, что точки 2 и 3 это кончики крыльев. Получается очень красивый эффект машущих крыльев с минимальным набором команд.



15. Вычитание obj[loop].yi из obj[loop].y приведет к тому, что бабочка будет двигаться вниз по экрану. Значение переменной spinz увеличивается на величину spinzi (которое может быть как положительным, так и отрицательным), а крылья увеличиваются на величину fi. fi также может иметь отрицательное или положительное значения в зависимости от того, в каком направлении должны хлопать крылья

16. После перемещения бабочки вниз по экрану надо проверить, не вышла ли она за пределы экрана (она больше невидима). Если это произошло, вызываем **SetObject(loop)** для задания этой бабочке новой текстуры, новой скорости и т.д.

17. Чтобы крылья махали, проверяем значение «хлопанья» на диапазон **+1.0f -1.0f**. Если значение вне диапазона, то мы меняем направление хлопанья: **fi=-fi**. Таким образом, если крыло идет вверх и достигает значения **1.0f**, то **fi** принимает отрицательное значение и крыло идет вниз.

В результате выполненной работы было создано приложение, которое выводит на экран группу движущихся фигур. Бабочки двигаются сверху вниз, машут крыльями и вращаются вокруг собственной оси.

Для выполнения данной работы были использованы различные элементы управления, функции для работы с объектами. Отрегулирована скорость, наклон и поворот изображения, загруженного из файла. Если посмотреть на оба изображения можно заметить, что точки 2 и 3 это кончики крыльев. Получается очень красивый эффект машущих крыльев с минимальным набором команд.

1) Список использованной литературы

1. Богуславский А.В. Си++ и компьютерная графика. Лекции и практикум по

программированию на Си++. – М.: КомпьютерПресс, 2003.
2. Баяковский Ю.М., Игнатенко А.В., Фролов А.И.. Графическая библиотека OpenGL. Учебно-методическое пособие. – Москва, 2003.