УДК 629.78(075)

Куренков В.И., Алиппа В.С.

ВИЗУАЛИЗАЦИЯ ПРОЦЕССА КОСМИЧЕСКОЙ СЪЁМКИ НАЗЕМНЫХ ОБЪЕКТОВ НАБЛЮДЕНИЯ СРЕДСТВАМИ OPENGL

В процессе моделирования целевого функционирования космических аппаратов (КА) наблюдения часто возникает задача проверки адекватности разрабатываемых или используемых математических моделей. Одним из инструментов проверки адекватности моделей является визуализация результатов моделирования. С помощью визуализации можно контролировать процесс орбитального полёта КА, нахождение объектов наблюдения (ОН) в полосах обзора, повороты корпуса КА при съёмке ОН, вхождение КА в тень Земли и выход на Солнце, повороты панелей солнечных батарей, текущее изменение энергобаланса на борту КА и т. п.

Кроме того визуализация процесса орбитального полёта и съёмки делает программное обеспечение более эффектным и помогает пользователям осмыслить поведение КА в процессе съёмки.

Математические модели и программное обеспечение для моделирования целевого функционирования КА наблюдения (без визуализации) были разработаны ранее с использованием системы программирования ***Delphi*** и представлены в работах [1, 2]. В настоящей статье приводятся результаты разработки модуля визуализации для упомянутого программного обеспечения. Для этого используется графическая библиотека ***OpenGL***, которая является неотъемлемым инструментом современной операционной системы ***Windows***.

Все необходимые исходные данные для визуализации берутся из основной программы с учётом их изменения в процессе моделирования функционирования КА. В частности, основная программа выдаёт массив ОН, попавших в полосы обзора КА за сутки полёта, массив ОН включённых в маршруты съёмки на каждом витке полёта, текущие значения координат КА, изменения направлений оптической оси аппаратуры наблюдения и т. п.

Рассмотрим вопросы визуализации модели Земли и процесса съёмки в трёхмерном пространстве с использованием средств графической библиотеки ***OpenGL***.

Визуализация трёхмерной модели Земли. Для этого в графической библиотеке ***OpenGL*** имеется несколько стандартных функций и наборов команд.

С помощью функции ***glFrustum*** задаются параметры вида, определяющие область воспроизведения в пространстве (перспективу с учётом координат наблюдателя).

Всё, что выходит за пределы заданной области, будет отсекаться при воспроизведении.

С помощью функции ***gluSphere*** задаются параметры сферы. Эта функция имеет четыре аргумента: первый – объект типа ***GLUquadricObj*** (***Quadric***-объект), второй – радиус сферы, третий и четвёртый задают параметры сглаживания при изображении трёхмерных фигур на экране.

Quadric-объекты содержатся в библиотеке ***glu***, которая имеется в составе графической библиотеки ***OpenGL***. Библиотека ***glu*** предоставляет набор команд, с использованием которых решение многих задач визуализации существенно упрощается. С помощью команд библиотеки ***glu*** можно изображать простейшие фигуры: цилиндр, куб, сферу и т.д. Для работы с этими командами вводится переменная специального типа: ***quadObj: GLUquadricObj;***. При создании программного окна визуализации вызывается команда, создающая ***quadric***-объект: ***quadObj = gluNewQuadric.*** Без этого действия обращение к объекту приведёт к исключениям (ошибкам). По окончании работы память, используемую quadric-объектами, необходимо освобождать. Сделать это нужно до освобождения контекста воспроизведения с помощью следующей последовательности функций:

gluDeleteQuadric (quadObj);

wglMakeCurrent (0, 0);

wglDeleteContext(hrc);.

Отметим, что здесь (и далее) в конце предложения стоит два знака препинания: точка с запятой ставится согласно правилам языка программирования, а точка – согласно правилам русского языка.

С помощью функции ***ReadBitmap ('..\earth.bmp', sWidth, tHeight)*** выбирается необходимое изображение. В рассматриваемом случае это файл ***earth.bmp***, в котором имеется плоская картинка с физической картой Земли в формате ***bmp***. С помощью параметров ***sWidth*** и ***tHeight*** задаются размеры картинки.

Ниже (без подробного пояснения) приведена последовательность необходимых функций для нанесения текстуры (карты) на поверхность сферы:

wrkPointer := ReadBitmap('..\earth.bmp', sWidth, tHeight); //создаём указатель;

glTexImage2D(GL_TEXTURE_2D, 0, 3, sWidth, tHeight, 0, GLRGB, GLJJNSIGNEDBYTE, wrkPointer);

Freemem(wrkPointer); //удаляем указатель;

glTexParameteri(GL_TEXTURE_2D, GL_TEXTURE_MIN_FILTER, GL_NEAREST);

glTexParameteri(GL_TEXTURE_2D, GL_TEXTURE_MAG_FILTER, GL_NEAREST);

glEnable(GL_TEXTURE_2D); //включаем режим нанесения текстуры;

glEnable(GL_DEPTH_TEST); //включаем режим глубины.

Более подробные сведения по этим функциям приведены в [3].

Представление процесса съёмки в трёхмерном пространстве. Рассмотрим вопрос о создании анимационной модели движения космического аппарата и Земли с нанесёнными на неё объектами наблюдения.

Для поворота используется функция *glRotatef* с четырьмя аргументами: угол поворота и три вещественных числа, характеризующих направляющие косинусы вектора поворота.

Для смены координат и направления взгляда наблюдателя библиотека *glu* располагает командой – *gluLookAt*. У неё девять аргументов: координаты позиции глаза наблюдателя в пространстве; координаты точки, располагающейся в центре экрана, и направляющие косинусы вектора, задающего поворот «сцены».

Приведём укрупнённый алгоритм и основные элементы кода программы, связанные только с технологией использования *OpenGL*.

1. Задаются масштабные коэффициенты для отображения (пропорции масштабирования):

glscale (clientheight/clientwidth, 1, 1);.

2. Задаётся начальное направление угла взгляда наблюдателя в системе координат *OpenGL*.

3. Определяется текущее значение угла истиной аномалии КА (для круговых орбит соответствует углу поворота надирной линии КА) в зависимости от текущего значения времени и периода обращения КА.

4. Рассчитываются координаты *x, y, z* подспутниковой точки КА наблюдения в системе координат *OpenGL*, где располагается центр экрана (точка, в которую смотрит наблюдатель).

5. Рассчитываются координаты *x1, y1, z1* подвижной точки взгляда наблюдателя с учётом масштабного коэффициент (k>1) и угла взгляда наблюдателя.

6. Записывается функция, задающая связи между координатами точки «взгляда» наблюдателя, координатами подспутниковой точки и направлением «взгляда»:

glulookat (y1, z1, x1, y, z, x, 0, 1, 0);.

7. Осуществляется прорисовки объектов наблюдения (точек на поверхности сферы) с помощью процедуры *RisON;* (см. ниже).

8. Осуществляется вращение сферы (модели Земли) на угол *beta*:

Glrotate (beta, 0, 1, 0);

где ***beta*** – угол вращения Земли (из основной программы).

9. Пункты 3…8 циклически повторяются на каждом шаге по времени имитационного моделирования до окончания «отработки» маршрута съёмки или до остановки данной части программы.

В процедуре ***RisON*** осуществляется перевод координат ОН в систему координат, связанную с ***OpenGL***, с использованием следующего фрагмента кода программы:

glbegin (gl_points); // Задание размера отображающих точек

for i:= 1 to num do // Цикл прорисовки всех ОН на сфере.

glvertex3f (1.01*cos(fi[i])*sin ((lm[i])), 1.01*sin (fi[i]), 1.01*cos (fi[i])*cos (lm[i]));.

Здесь ***fi[i]*** и ***lm[i]*** – одномерные массивы широты и долготы объектов наблюдения. Коэффициент 1.01 вводится для того, чтобы точки, соответствующие объектам наблюдения, выводились поверх текстуры, которая наложена на сферу с координатами центра (0,0,0) и радиусом 1.

С использованием приведённого алгоритма и кода разработан модуль визуализации для программного обеспечения, имитирующего орбитальный полёт и функционирование КА наблюдения по целевому назначению.

На рис. 1 представлено окно программы для визуализации орбитального движения и программных поворотов КА при отработке маршрута съёмки. Кроме того на экран выводятся данные о текущем энергобалансе на борту КА (дополнительное окно «Заряд аккумулятора» в левой части рисунка).

Рис. 1. Окно программы для иллюстрации орбитального движения и программных поворотов КА при отработке маршрута съёмки

Библиографический список

1. Куренков, В И. Основы устройства и моделирования целевого функционирования КА наблюдения [Текст]/В.И. Куренков, В.В. Салмин, Б А. Абрамов – Самара: Изд-во Самар. гос. аэрокосм. ун-та, 2006. – 296 с.
2. Куренков, В И. Моделирование целевого функционирования космических аппаратов наблюдения с учётом энергобаланса: учеб. пособие [Текст]/ В.И. Куренков, В.В. Салмин, Б.А. Абрамов. – Самара, Изд-во Самар. гос. аэрокосм. ун-та, 2007. – 160 с.
3. Краснов, М.В. OpenGL. Графика в проектах Delphi [Текст]/М.В. Краснов. – СП.: БХВ-Петербург, 2004. – 352 с.