Всероссийский конкурс учебно-исследовательских работ старшеклассников
по политехническим, естественным, математическим дисциплинам
для учащихся 9-11 классов

Информатика и информационные технологии

3D моделирование Солнечной системы

Ванюков Владимир Владимирович,
Вяткин Яков Витальевич,
11 класс, МБОУ «Лицей №1», г. Пермь

Ашихмин Валерий Николаевич,
к.т.н., доцент каф. ММСП ПНИПУ

Пермь. 2018.

СОДЕРЖАНИЕ

ВВЕДЕНИЕ

Зачастую очень сложно объяснить словами самые простые вещи или устройство того или иного механизма. Понимание обычно приходит достаточно легко, если увидеть всё своими глазами, а еще лучше и покрутить в руках. Но некоторые вещи невидимы для нашего зрения и, даже будучи простыми, очень сложны для понимания.

Компьютерное моделирование является одним из эффективных методов изучения сложных систем. Компьютерные модели проще и удобнее исследовать в силу их возможности проводить вычислительные эксперименты, в тех случаях, когда реальные эксперименты затруднены из-за финансовых или физических препятствий, или могут дать непредсказуемый результат. Логичность компьютерных моделей позволяет выявить основные факторы, определяющие свойства изучаемого объекта-оригинала (или целого класса объектов), в частности, исследовать отклик моделируемой физической системы на изменения ее параметров и начальных условий [1].

В данной работе с помощью компьютерного моделирования рассмотрим действие такого процесса, как сила тяготения.

Сам закон прост - сила тяготения прямо пропорциональна массам и обратно пропорциональна квадрату расстояния между ними, но сложность заключается в невообразимом количестве взаимодействующих объектов.

Объектом исследования нашей работы является компьютерное 3D моделирование.

Предмет исследования – особенности этапов работы со средой программирования.

Цель учебно-исследовательской работы - создание программы для компьютерного 3D-моделирования объектов Солнечной системы, демонстрирующей работу закона силы тяготения.

Для достижения поставленной цели в ходе исследования решался ряд задач:

1. Проанализировать учебно-методическую литературу по математическим основам трёхмерной графики и теоретическим аспектам строения Солнечной системы.

2. Определить программные средства, необходимые для создания работоспособной программы и изучить их возможности.

3. Разработать и осуществить алгоритм работы в программной среде для создания компьютерной модели.

Глава 1. ТЕОРЕТИЧЕСКАЯ ЧАСТЬ

1.1 Закон всемирного тяготения

В рамках классической механики гравитационное взаимодействие описывается законом всемирного тяготения Ньютона, который гласит, что сила гравитационного притяжения **F** между двумя материальными точками массы $\mathbf{m_1}$ и $\mathbf{m_2}$, разделёнными расстоянием **r**, пропорциональна обеим массам и обратно пропорциональна квадрату расстояния - то есть:

$$\boldsymbol{F = G\frac{m_1 m_2}{r^2}} \qquad (1.1.)$$

где **G** - гравитационная постоянная, равная примерно $6{,}67384\times10^{-11}$ Н×м^2×кг$^{-2}$

Следует заметить, что:

- сила тяготения всегда положительна, не имеет отрицательных значений, т.е. масса не может быть отрицательной;
- сила тяготения не может быть равна нулю, т.е. объект либо существует с какой-то массой, либо не существует вообще
- силу тяготения нельзя ни заэкранировать, ни отразить (как луч света зеркалом). [9]

Солнечная система состоит из планет с их спутниками, астероидов, комет, мелких метеорных тел, космической пыли. Законы движения и происхождения всех этих тел неразрывно связаны с центральным объектом системы — Солнцем. Основной силой, управляющей движением планет и связывающей воедино Солнечную систему, является электрическая сила Солнца. При этом для тел Солнечной системы характерны два признака.

Во-первых, тело за счет своей кинетической энергии не может преодолеть силы солнечного притяжения и покинуть Солнечную систему.

Во-вторых, тело, принадлежащее Солнечной системе, должно постоянно находиться в области преобладающего притяжения Солнца [9].

Заметим, что для всех планет с их спутниками, астероидов, практически всех комет, находящихся в сфере действия Солнца, оба условия выполняются. Данные об орбитах и некоторых физических свойствах планет, являющихся главными членами Солнечной системы, приведены в таблице 1.1., именно эти данные будут положены в основу будущей модели Солнечной системы нашего проекта.

Таблица 1.1. Планеты солнечной системы

Планеты	**Радиус орбиты R, а.е.**	**Скорость по орбите V, км/с**	**Масса планет M, кг**
Меркурий	0,38	47,0	$0{,}3\times10^{21}$
Венера	0,72	34,9	$3{,}3\times 10^{21}$
Земля	1,0	29,7	$4{,}9\times 10^{21}$
Марс	1,52	24,0	$2{,}2\times 10^{21}$
Юпитер	5,2	13,1	$3{,}0\times10^{24}$
Сатурн	9,5	9,6	$4{,}4\times10^{24}$
Уран	19,2	6,8	$1{,}6\times10^{24}$
Нептун	30,0	5,4	$2{,}2\times10^{24}$
Плутон	39,5	4,7	$6{,}7\times10^{21}$

1.2. Системы трёхмерного моделирования

Методы трехмерного моделирования делятся на 3 вида:

- Каркасное (проволочное) моделирование;
- Поверхностное (полигональное) моделирование;
- Твердотельное (сплошное, объемное) моделирование.

Каркасная модель полностью описывается в терминах точек и линий. Это моделирование самого низкого уровня и имеет ряд серьезных ограничений, большинство из которых возникает из-за недостатка информации о гранях, которые заключены между линиями, и невозможности выделить внутреннюю и внешнюю область изображения твердого объемного тела.

Однако каркасная модель требует меньше памяти и вполне пригодна для решения задач, относящихся к простым. Каркасное представление часто

используется не при моделировании, а при отображении моделей как один из методов визуализации.

Поверхностное моделирование определяется в терминах точек, линий и поверхностей. При построении поверхностной модели предполагается, что технические объекты ограничены поверхностями, которые отделяют их от окружающей среды. Такая оболочка изображается графическими поверхностями. Поверхность технического объекта снова становится ограниченной контурами, но эти контуры уже являются результатом двух касающихся или пересекающихся поверхностей. Точки объектов - вершины, могут быть заданы пересечением трех поверхностей.

Поверхностное моделирование имеет следующие преимущества по сравнению с каркасным:

- способность распознавания и изображения сложных криволинейных граней;

- изображение грани для получения тоновых изображений;

- особые построения на поверхности (отверстия);

- возможность получения качественного изображения;

-обеспечение более эффективных средств для имитации функционирования роботов.

В основу поверхностной модели положены два основных математических положения:

- Любую поверхность можно аппроксимировать многогранником, каждая грань которого является простейшим плоским многоугольником;

- Наряду с плоскими многоугольниками в модели допускаются поверхности второго порядка и аналитически неописываемые поверхности, форму которых можно определить с помощью различных методов аппроксимации и интерполяции.

В отличие от каркасного моделирования каждый объект имеет внутреннюю и внешнюю часть.

Твердотельная модель описывается в терминах того трехмерного объема, который занимает определяемое ею тело. Твердотельное моделирование является самым совершенным и самым достоверным методом создания копии реального объекта.

Преимущества твердотельных моделей:

- Полное определение объемной формы с возможностью разграничивать внутренний и внешние области объекта, что необходимо для взаимовлияний компонент.

- Обеспечение автоматического удаления скрытых линий.

- Автоматическое построение 3D разрезов компонентов, что особенно важно при анализе сложных сборочных изделий.

-Применение методов анализа с автоматическим получением изображения точных весовых характеристик методом конечных элементов.

- Получение тоновых эффектов, манипуляции с источниками света [3].

1.2.1. Полигональное моделирование

Полигональное моделирование - метод описания трёхмерных объектов, при котором любой объект формируется из треугольников (полигонов). Чем больше полигонов на площадь модели, тем точнее модель. Основан на утверждении, что любую плоскую фигуру, кроме полукруга и эллипса, возможно условно “разбить” на треугольники. При моделировании округлых объектов (цилиндров, шаров и т.д.), окружность изображается вписанным многоугольником. Этот метод чаще всего используется в современном компьютерном моделировании, так как он довольно прост в реализации и не очень требователен к ресурсам компьютера [3].

Полигональное моделирование происходит путем манипуляций с полигонами в пространстве. Вытягивание, вращение, перемещение и т.д. Пионером в этой отрасли является компания Autodesk.

Рисунок 1.1. Полигональная модель

1.2.2. Open Graphics Library

OpenGL – программное обеспечение, набор дополнительных функций, необходимых для работы над трёхмерными объектами. Создан в 1992 году компанией Silicon Graphics (США). Основным принципом работы OpenGL является получение наборов векторных графических примитивов в виде точек, линий и многоугольников с последующей математической обработкой полученных данных и построением растровой картинки на экране и/или в памяти. Векторные трансформации и растеризация выполняются графическим конвейером (graphics pipeline), который по сути представляет собой дискретный автомат. Абсолютное большинство команд OpenGL попадают в одну из двух групп: либо они добавляют графические примитивы на вход в конвейер, либо конфигурируют конвейер на различное исполнение трансформаций [5].

1.3. Язык программирования Lazarus

Lazarus - открытая среда разработки программного обеспечения на языке Object Pascal для компилятора Free Pascal (часто используется сокращение FPC - Free Pascal Compiler, бесплатно распространяемый компилятор языка программирования Pascal). Интегрированная среда разработки предоставляет возможность кроссплатформенной разработки приложений в Delphi-подобном окружении.

До сих пор среды разработки программ, подобные Lazarus, были исключительно платными. Lazarus же стал первой (и пока единственной) IDE,

доступной образовательным и государственным учреждениям совершенно бесплатно. Более того, Lazarus является проектом Open Source - проектом с открытым исходным кодом. Многие программисты по всему миру принимают участие в его развитии, исходный код Lazarus доступен для изучения и модификации. Lazarus имеет поддержку множества языков, в том числе и русского, что выгодно отличает его от других IDE.

Данный проект базируется на оригинальной кроссплатформенной библиотеке визуальных компонентов Lazarus Component Library (LCL).

Кроссплатформенное программное обеспечение – это программное обеспечение, работающее более чем на одной аппаратной платформе и/или операционной системе [3].

Таким образом, разработанные приложения могут функционировать практически под любой операционной системой. Все, что вы видите на экране во время работы различных приложений, все элементы (кнопки, бегунки, меню и т.п.) можно реализовать в Lazarus.

В Lazarus используется технология визуального программирования. Пользователь для создания графического интерфейса приложения использует готовые компоненты, значки которых находятся на панели компонентов. После того как он помещает компонент на форме, программный код для него генерируется автоматически. Вручную остается запрограммировать только те действия, которые будет выполнять это приложение.

Процесс создания приложения можно разделить на следующие этапы:

1. Создание проекта. В результате на экране появляется пустая форма.

2. Создание графического интерфейса проекта – расположение необходимых элементов, задание размеров, изменение свойств;

3. Написание программного кода, который определит, что будет делать ваша программа.

4. Отладка программы [3].

Глава 2. ПРАКТИЧЕСКАЯ ЧАСТЬ

2.1. План работы над компьютерной программой

1. Составить основу программного кода по рисованию объектов.
2. Схематически определить расстояния и размеры для лучшего понимания наблюдения за процессами движений объектов.

   Рисунок 2.1.1. Процедура объекта

```
procedure GObject(var x,z,Vx,Vz,Ax,Az,R:Double; Ra,cr,cg,cb:Double);
begin
   Gravity(x,z,Vx,Vz,Ax,Az,R);//реализация гравитации

   glPushMatrix;
   glEnable(GL_COLOR_MATERIAL);//задание материала
   glColor3f(cr,cg,cb);//цвет
   glTranslatef(x,0,z);//определение координат
   glRotatef(90, 1, 0, 0);//вращение объекта
   glutSolidSphere(Ra,20,20);//прорисовка сферы
   glPopMatrix;
end;
```

3. Составить основу программного кода, определяющая законы движения объектов.

   Рисунок 2.1.2. Процедура силы тяготения

```
procedure Gravity(var x,z,Vx,Vz,Ax,Az,R:Double);//параметры процедуры
begin
   R  := sqrt(x*x+z*z);//определение расстояния до солнца

   Ax := G * (Mass * (-x)) / (R*R*R);//ускорение по оси X
   Az := G * (Mass * (-z)) / (R*R*R);//ускорение по оси Z

   Vx := Vx + (Ax*T);//скорость по оси X
   Vz := Vz + (Az*T);//скоорсть по оси Z

   x := x + (Vx*T);//координата по оси X
   z := z + (Vz*T);//координата по оси Z
```

4. Дать возможность с помощью мыши управлять поворотом камеры.

Рисунок 2.1.3. Процедура поворота камеры

```
procedure Cam(x,y,b:Double);//параметры процедуры
begin
  R := sqrt(x*x+y*y);//определение радиуса вращения
  CosAlpha := (2*R*R-b*b)/(2*R*R);//определение косинуса угла поворота
  if (CosAlpha>=0) then
  begin
     iZ := R*CosAlpha;//определение положения камеры по оси Z
     if (b<0) then
     iY := -sqrt(R*R-iZ*iZ)//определение положения камеры по оси Y
     else  iY := sqrt(R*R-iZ*iZ);
  end;
end;
```

5. Организовать движение камеры в пространстве с помощью клавиатуры.

Рисунок 2.1.4. Процедура движения камеры

```
procedure TForm1.GLboxKeyDown(Sender: TObject; var Key: Word; Shift: TShiftState
  );
begin
   //перемещение по выбраным осям
   if Key = VK_Down  then iZ := iZ + 5;
   if Key = VK_Up    then iZ := iZ - 5;
   if Key = VK_Left  then iX := iX-0.1;
   if Key = VK_Right then iX := iX+0.1;
end;
```

2.2. Алгоритм работы по созданию компьютерной программы

1. Загрузка данных о планетах.
2. Вычисление по формулам координат объектов.
3. Создание массива трёхмерных точек и его заполнение результатами предыдущего пункта.
4. Реализация перемещения и поворота камеры.

Рисунок 2.2.1. Прорисовка объектов программы

```
//прорисовка всех объектов с их параметрами
With Mercury do GObject(dX,dZ,Vx,Vz,Ax,Az,R, 0.08,  0.62, 0.62, 0.64);
With Venus   do GObject(dX,dZ,Vx,Vz,Ax,Az,R, 0.1,   0.94, 0.83, 0.59);
With Earth   do GObject(dX,dZ,Vx,Vz,Ax,Az,R, 0.13,  0.52, 0.8,  0.98);
With Mars    do GObject(dX,dZ,Vx,Vz,Ax,Az,R, 0.13,  1.0,  0.72, 0.38);
With Upiter  do GObject(dX,dZ,Vx,Vz,Ax,Az,R, 0.2,  0.76, 0.66, 0.58);
With Saturn  do GObject(dX,dZ,Vx,Vz,Ax,Az,R, 0.18,  0.98, 0.98, 0.82);
With Uranus  do GObject(dX,dZ,Vx,Vz,Ax,Az,R, 0.15,  0.23, 0.51, 0.74);
With Neptune do GObject(dX,dZ,Vx,Vz,Ax,Az,R, 0.2,  0.16, 0.32, 0.745);
With Pluto   do GObject(dX,dZ,Vx,Vz,Ax,Az,R, 0.2,  0.6,  0.46, 0.33);
```

5. Создание перспективы и выбор направления взгляда.

Рисунок 2.2.2. Задание перспективы

```
gluPerspective(45, GLbox.Width / GLbox.Height, 0.1, 1000);//задание перспективы
```

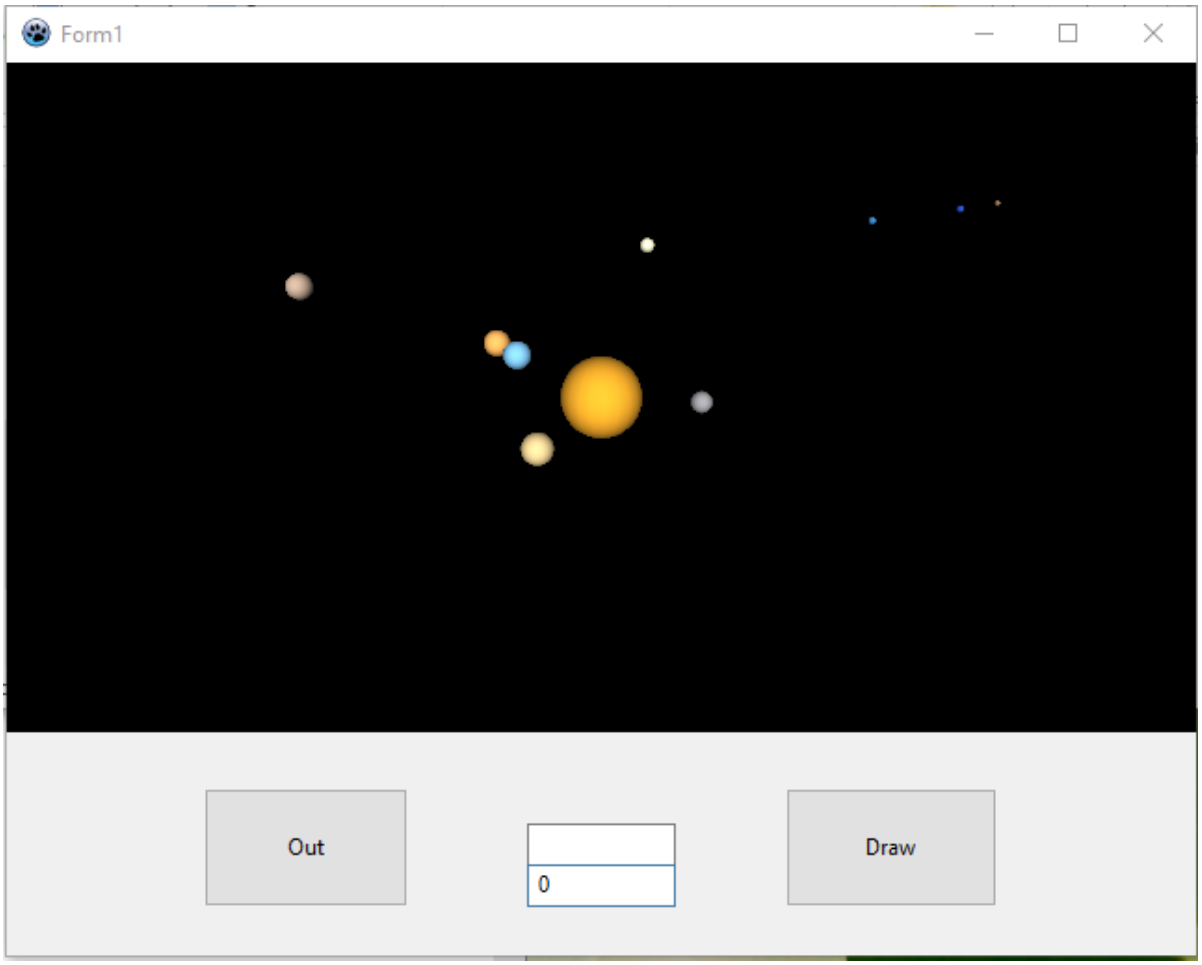
2.3. Управление камерой

Для наиболее наглядного показа необходимо, чтобы пользователь мог свободно перемещаться в пространстве, а также вращать сцену. Для этого были использованы клавиатура и манипулятор типа «мышь». В Lazarus имеются встроенные возможности для считывания данных с мыши и клавиатуры. После нажатия на кнопку мыши или клавишу клавиатуры Lazarus автоматически запускает определенный фрагмент кода. Затем осуществляется переход к нужной части кода в зависимости от нажатой кнопки или клавиши. Изменяются соответствующие переменные, после чего они обрабатываются OpenGL с помощью вышеописанных функций.

2.4. Результат работы

В результате работы была написана программа, названная Solar System. Программа позволяет наблюдать за процессом движения объектов солнечной системы в реальном времени. Пользователь получает возможность свободно перемещаться в пространстве и менять точку наблюдения. Также программа имитирует поведение планет, позволяет менять массу Солнца, что визуально даёт представление о работе закона всемирного тяготения.

Рисунок 2.4.1. Окно программы Solar System



ЗАКЛЮЧЕНИЕ

В процессе научно-исследовательской работы были проанализированы принципы и системы компьютерного программирования, моделирования, а также их основные инструменты. Помимо этого, в работе был описан закон всемирного тяготения и состав Солнечной Системы, также обнаружена познавательная информация об её современных исследованиях.

В практической части работы описан план и краткий алгоритм исследования, ход работы над проектом.

В результате продуктом проекта является компьютерная программа Solar System, предназначенная для трёхмерного моделирования Солнечной Системы.

Данная программа в дальнейшем может использоваться в учебно-образовательном процессе, а именно, на уроках физики для иллюстрации работы закона всемирного тяготения. Впоследствии данная компьютерная программа может быть усовершенствована и положена в основу будущих проектных работ.

СПИСОК ИСПОЛЬЗУЕМОЙ ЛИТЕРАТУРЫ

1. Майер Р.В. Основы компьютерного моделирования: Учебное пособие. – Глазов: ГГПИ, 2005. – 25 с.
2. http://grafika.me/node/182
3. http://intuit.valrkl.ru/course-1265/
4. http://kappasoft.narod.ru/info/3d/3d.htm
5. http://rsdn.org/article/opengl/ogltut2.xml#E3ZAE
6. http://wiki.lazarus.freepascal.org/OpenGL_Tutorial/ru
7. http://www.opengl-master.ru/catalog-functions.php
8. https://ru.wikipedia.org/wiki/Lazarus
9. https://studopedia.ru/7_95774_gravitatsionnoe-vzaimodeystvie-v-solnechnoy-sisteme.html
10. Видео-хостинг www.youtube.com

ANNOTATION

The main subject of this research work is the peculiarity of stages of working with the programming environment.

The aim of the research work is to develop a programme for computer 3D modeling of the objects of the solar system, demonstrating the work of the law of gravity.

To accomplish the task, it was necessary to analyze educational and methodological literature on the mathematical foundations of three-dimensional graphics and theoretical aspects of the structure of the solar system.

During the study of the programming environment, a work plan was developed for the program.

In conclusion, it should be noted that in the process of work, the principles of computer programming, basic tools, were studied. The composition of the Solar System was studied, and information about its investigation was also found.

The product of this project work is a computer program designed for 3D modeling of the Solar System.