**SECTION 2. Applied mathematics. Mathematical modeling.**

**Shevtsov Alexandr Nikolayevich**
candidate of technical Sciences,
President of International Academy of Theoretical & Applied Sciences, Kazakhstan
Shev_AlexXXXX@mail.ru

## SOME QUESTIONS SIMULATION OF INTERACTIVE DYNAMIC SYSTEMS

***Abstract***: *The process of developing an interactive model is a rather complicated process. In the structure of the model is necessary not only to draw up the conditions that influence its work, and also to connect directly with the graphic libraries and the matrices of transformation of the coordinates. In this article the approach and the implementation of the three-dimensional model calculation of the orbital trajectories of objects interactively in the Delphi-XE4.*

***Key words***: *interactive model, orbits, collisions of objects.*

## НЕКОТОРЫЕ ВОПРОСЫ МОДЕЛИРОВАНИЯ ИНТЕРАКТИВНЫХ ДИНАМИЧЕСКИХ СИСТЕМ

***Аннотация***: *Процесс реализации любой интерактивной модели представляет собой довольно сложный процесс. В самой структуре модели необходимо не только правильно составить условия, влияющие на ее работу, но и связать непосредственно с графическими библиотеками и матрицами преобразования координат. В данной статье рассмотрен подход, и реализация трехмерной модели расчета орбитальных траекторий объектов, в интерактивном режиме в среде Delphi-XE4.*

***Ключевые слова***: *интерактивная модель, орбиты, столкновения объектов.*

Отслеживание траекторий движения объектов на орбите Земли представляет собой очень важную проблему. По данным ежеквартального отчета НАСА, посвященного проблеме космического мусора, на данный момент на орбите находятся 16,094 тысячи "мусорных объектов", в том числе вышедших из строя 3,396 тысячи спутников и 12,968 тысячи верхних ступеней ракет-носителей и других обломков. К примеру, ВВС США готовят к запуску первый космический аппарат, предназначенный для наблюдения за космическим мусором на орбите Земли. Спутник космического наблюдения будет следить за 1000 действующими спутниками и 22 000 фрагментами космического мусора. Три компании, в том числе, Lockheed-Martin, получили от ВВС США по 30 миллионов долларов на разработку электронной изгороди, которая позволит отслеживать перемещения космических объектов размером более пяти сантиметров [1].

Причем большинство объектов не являются геостационарными, а значит движутся по определенным эллиптическим орбитам. Зная орбитальные координаты объектов, можно проанализировать данные и отследить движение всех объектов. Разработаем математическую и компьютерную модель, движения объектов на орбите Земли. Из курса физики известно, что тело, движущееся по орбите вокруг другого тела, подчинено трём законам Кеплера. Нас будут интересовать только два из них - первый и третий.

Согласно первому закону Кеплера, тело, обращающееся вокруг Земли движется по эллипсу, в одном из фокусов которого находится центр Земли [2].

**Орбитальные элементы**

Для того, чтобы задать параметры и ориентацию орбиты околоземных объектов в пространстве, нужно указать 6 так называемых кеплеровских элементов (орбитальных элементов) (рис. 1). Орбита ИСЗ полностью задаётся шестью орбитальными элементами.

| | Наименование | Обозначение в программе |
|---|---|---|
| 1. | Большая полуось "a". Равна среднему расстоянию ИСЗ от центра Земли. | **D[ ].a1** |
| 2. | Эксцентриситет "e" (см. формулу 1) - мера сплюснутости эллипса. | **D[ ].a2** |
| 3. | Наклонение орбиты "i" к экваториальной плоскости Земли - угол пересечения плоскости орбиты ИСЗ с плоскостью экватора Земли. | **D[ ].a3** |
| 4. | Аргумент перигея (АП) ω - угол, отсчитываемый в плоскости орбиты ИСЗ от восходящего узла орбиты до точки перигея (точка, где расстояние между ИСЗ и центром Земли наименьшее). | **D[ ].a5** |
| 5. | Долгота восходящего узла (ДВУ) Ω - угол, отсчитываемый в плоскости земного экватора от восходящего узла до точки весеннего равноденствия. Угол отсчитывается против часовой стрелки, если смотреть с северного полюса мира. | **D[ ].a4** |
| 6. | Средняя аномалия (СА) M0 - угол, отсчитываемый в плоскости орбиты ИСЗ от перигея до ИСЗ на орбите. Угол отсчитывается против часовой стрелки, если смотреть с северного полюса мира. | **D[ ].a6** |


**Рисунок 1 - Орбитальные элементы. [2]**

Двигаясь по эллиптической орбите, объект ближе всего подлетает к центру Земли в перигее, а дальше всего находится в апогее. Также происходит - эволюция

орбиты, прецессия орбиты, смещение восходящего узла за один виток [3], вращение эллиптической орбиты [1], атмосферное торможение, влияние давления света. В заключение нужно отметить, что перечисленные факторы влияния на эволюцию орбиты спутника не составляют полный список. Например, на ИСЗ воздействуют своим притяжением Солнце и Луна, но это воздействие в 10000 раз слабее действия экваториального "горба" Земли (Рис.2.), но его нужно учитывать для орбит с большим эксцентриситетом. Экваториальный "горб" также вызывает незначительные колебания плоскости орбиты ИСЗ при пересечении экваториальной плоскости. Наконец, неравномерность распределения масс под поверхностью Земли также сказывается на движении спутника.

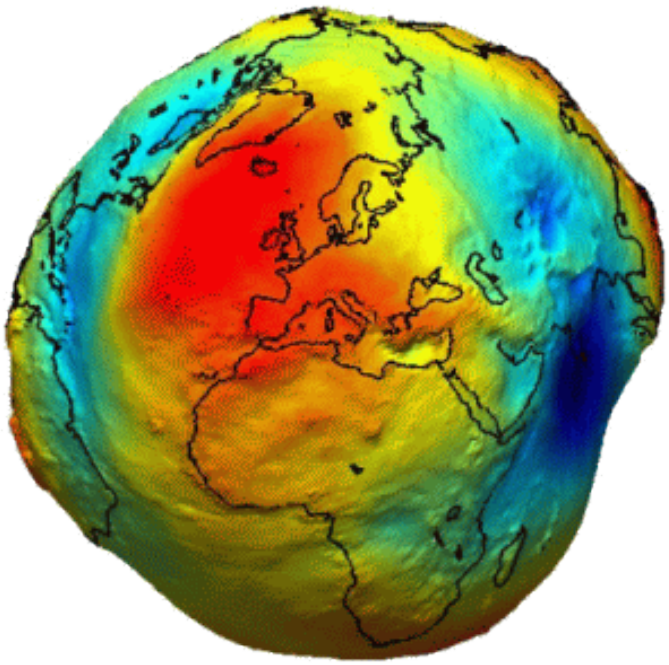

**Рисунок 2 - Форма земного геоида по данным ИСЗ "GOCE" [4-5].**

Используя все выше описанные алгоритмы разработаем интерактивную компьютерную математическую модель.

**База данных орбит объектов.**

Разработку будем вести посредством Directx11 и OpenGL в среде XE4. В качестве нормы примем среднюю длину радиуса Земли. Загрузим Quad объект шара и наложим текстуру. Траектории орбит будем загружать из базы данных. Все данные в базе - согласно таблице, указанных выше, орбитальных элементов:

| | |
|---|---|
| Название орбитального объекта | D[ ].name |
| Большая полуось | D[ ].a1 |
| Эксцентриситет | D[ ].a2 |
| Наклонение орбиты | D[ ].a3 |
| Долгота восходящего узла | D[ ].a4 |
| Аргумент перигея | D[ ].a5 |
| Средняя аномалия | D[ ].a6 |

Вращение модели в пространстве будем осуществлять в автоматическом режиме и вручную с помощью правой кнопки мыши в любую произвольную сторону.

Увеличение и уменьшение размера с помощью колесика. Модель поддерживает более 100 млн. отдельных объектов, причем базу можно редактировать в реальном времени, а также при необходимости дополнить более подробными характеристиками и параметрами отдельных объектов, или связать с другими базами данных.

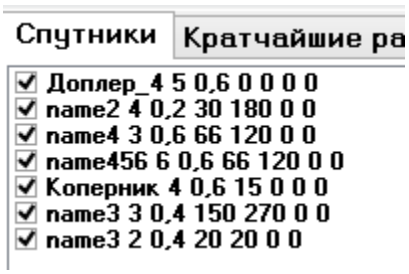


**Рисунок 3 – Выбор спутника из базы.**

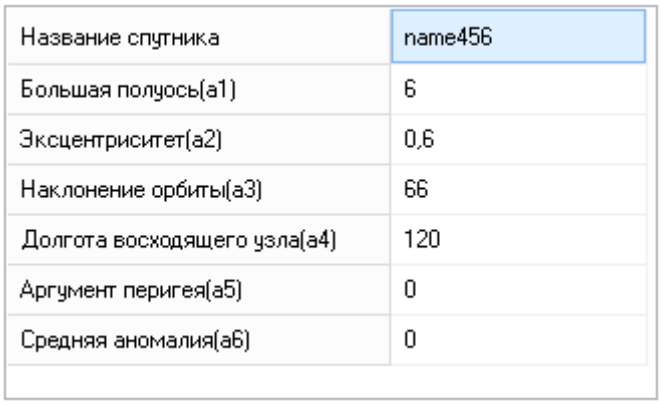


**Рисунок 4 – Редактирование базы спутников.**

Расчет кратчайших возможных расстояний между отдельными траекториями будем расчитывать как функцию минимизации длинны множества векторов соответствующих точек орбит в глобальной системе координат.

После нахождения, данные сортируются по возрастанию и выводятся на экран в указанием названий спутников. Имеется возможность не только найти минимальное расстояние но и определить глобальные координаты в каждом из случаев.

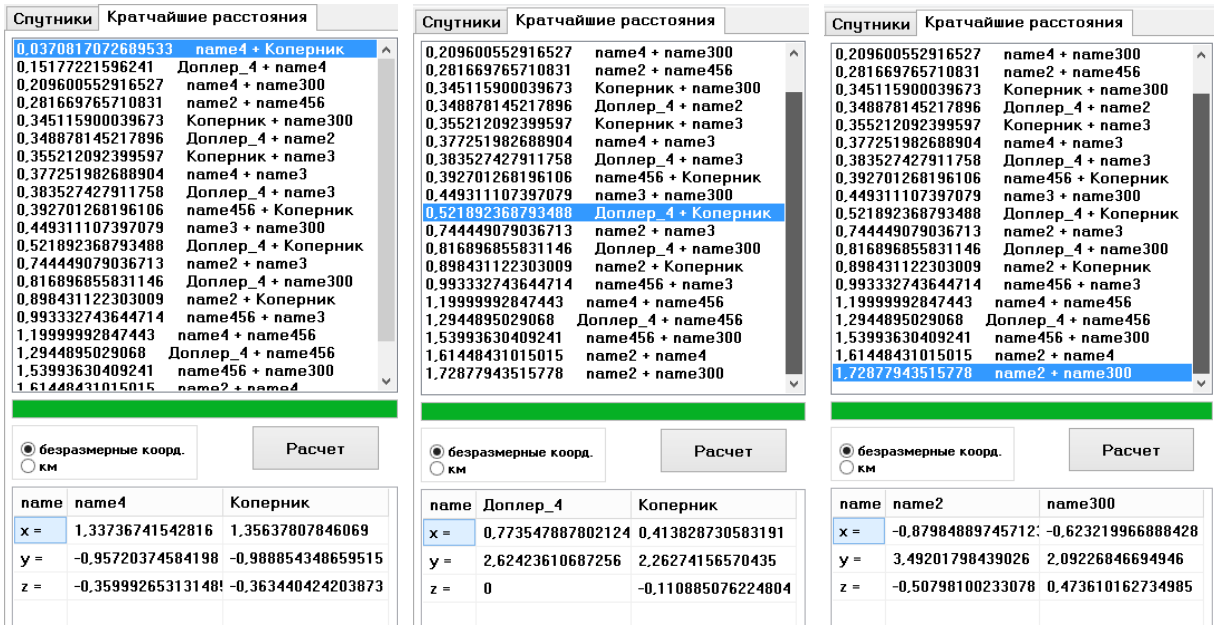


**Рисунок 5 – Расчет кратчайшего, произвольного, и максимального расстояния между орбитами спутников в безразмерных координатах.**

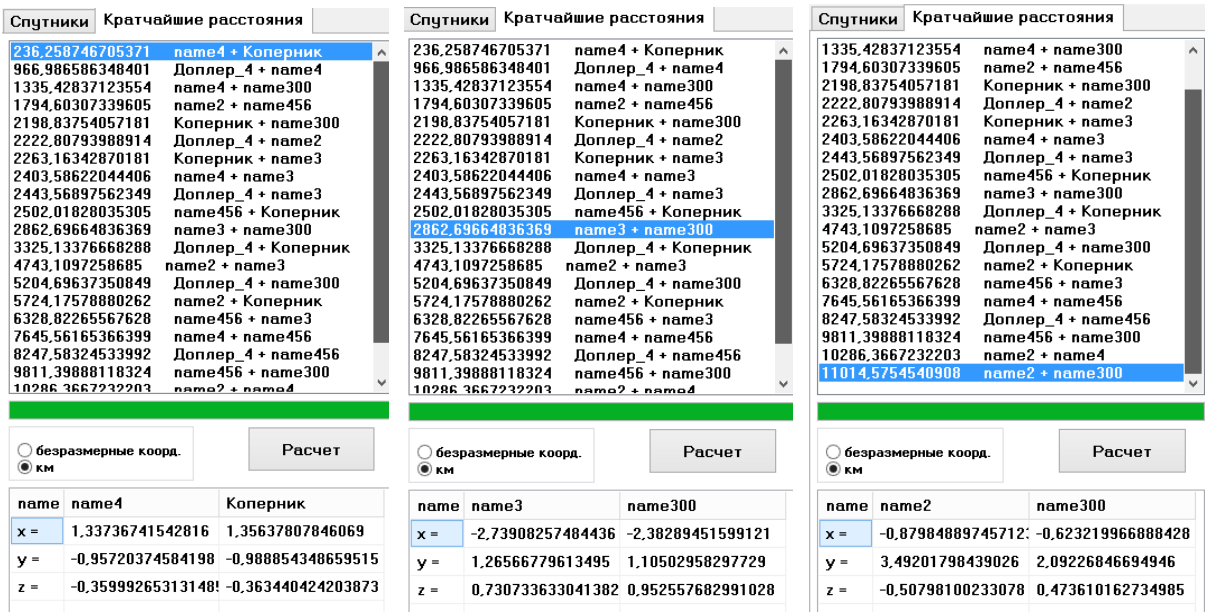


**Рисунок 6 – Расчет кратчайшего, произвольного, и максимального расстояния между орбитами спутников в системе СИ.**

Полученная интерактивная модель отображена на рисунках 7-10.

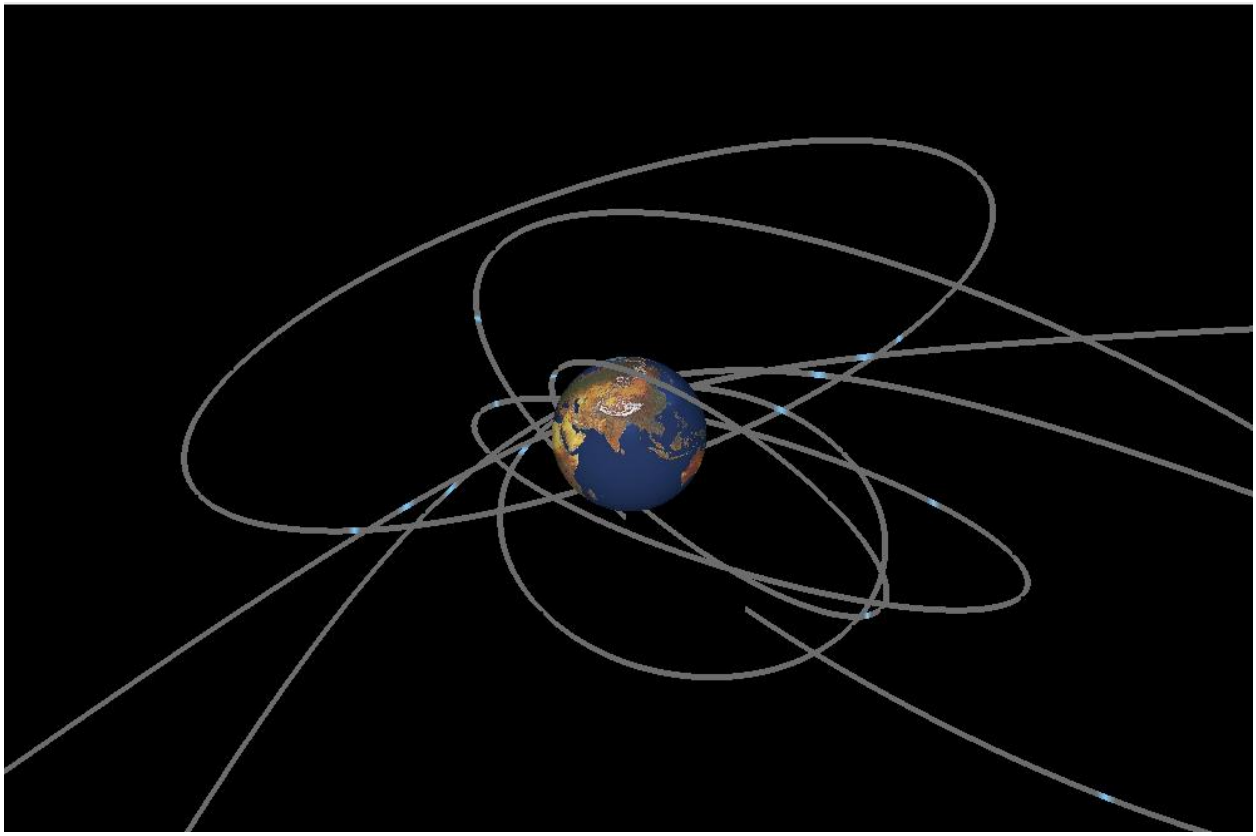

**Рисунок 7 – Разработанная модель.**

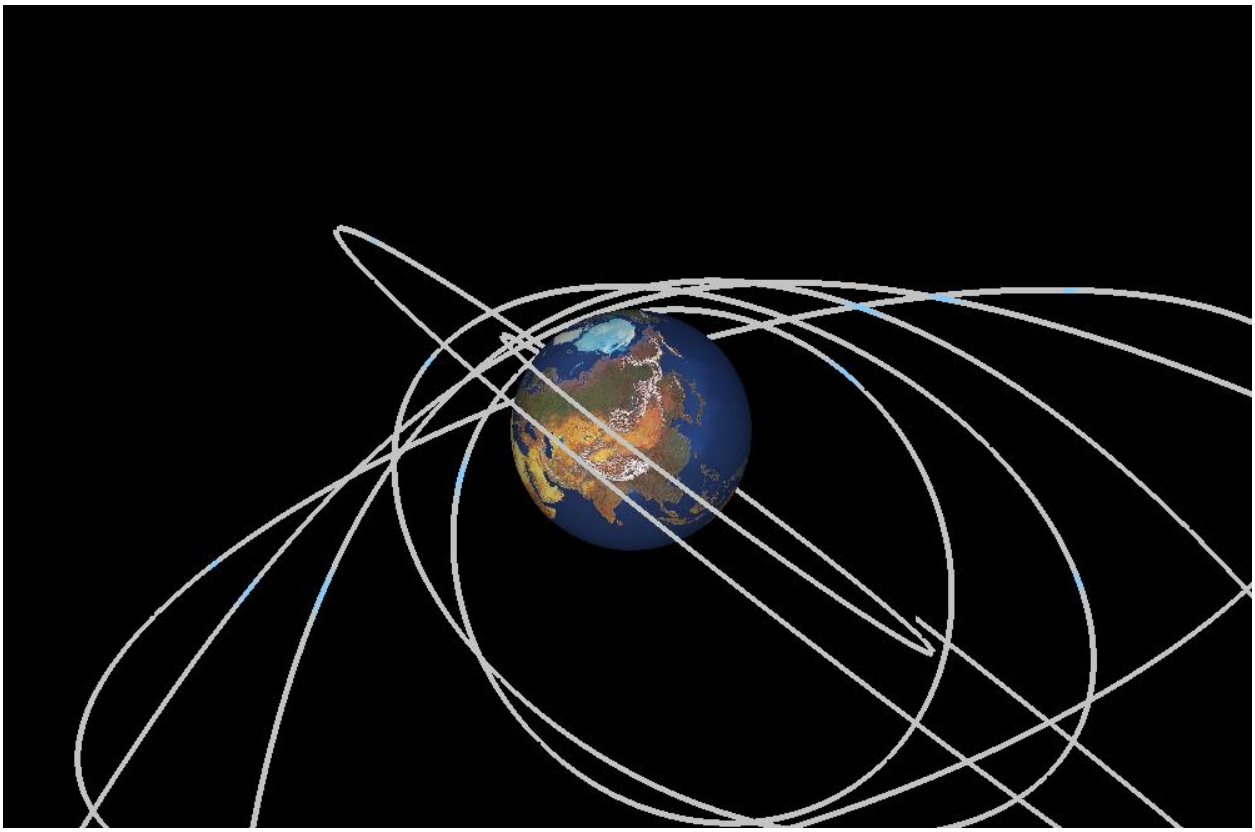

**Рисунок 8 – Разработанная модель.**

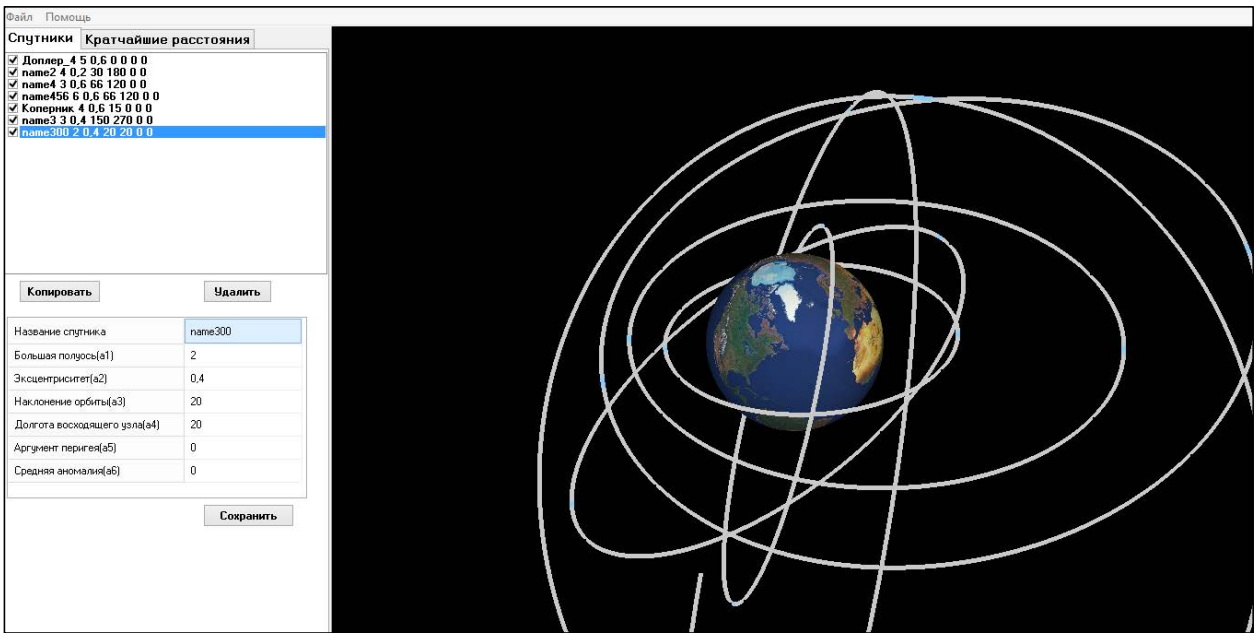


**Рисунок 9 – Разработанная модель.**

**Рисунок 10 – Разработанная модель.**

Полученная модель позволяет просчитывать минимальные расстояния между орбитами спутников и вероятности их столкновений. Модель основана на аппробированных методиках и алгоритмах расчета. Траектории спутников можно отображать и исследовать в реальном времени на данной интерактивной модели.

## ЗАКЛЮЧЕНИЕ

В результате проведенного исследования и компьютерного моделирования получены следующие результаты:

- Разработана интерактивная компьютерная модель отображающая траектории околоземных объектов,
- Алгоритмы расчета траекторий движения спутников в глобальной системе координат связанной с Землей,
- Компьютерные алгоритмы трехмерных преобразований проекций,
- Алгоритмы определяющие все возможные минимальные расстояния между траекториями отдельных объектов.

Разработанные модели и программное обеспечение могут быть использованны как при научных исследованиях и построении более сложных систем слежения, так и в процессе преподавания.

## ЛИТЕРАТУРА

1. Д. Кинг-Хили, "Искусственные спутники и научные исследования", М., 1963 г.
2. Кеплеровы элементы орбиты. [Электронный ресурс]. URL: http://ru.wikipedia.org/wiki/%D0%9A%D0%B5%D0%BF%D0%BB%D0%B5%D1%80%D0%BE%D0%B2%D1%8B_%D1%8D%D0%BB%D0%B5%D0%BC%D0%B5%D

0%BD%D1%82%D1%8B_%D0%BE%D1%80%D0%B1%D0%B8%D1%82%D1%8B
3. Левантовский В.И., "Механика космического полёта в элементарном изложении", М., 1980 г.
4. Предложен новый метод определения формы Земли. [Электронный ресурс]. URL: http://mestechko.info/science/11010-predlozhen-novyj-metod-opredeleniya-formy-zemli.html (дата обращения: 20.01.2014).
5. ESA показало самую точную "математическую форму Земли". [Электронный ресурс]. URL: HTTP://LENTA.RU/NEWS/2011/04/01/GOCE (дата обращения: 12.01.2014).
6. Наблюдение искусственных спутников Земли. [Электронный ресурс]. URL: http://www.sat.belastro.net/glava1/glava1.php BelAstro.Net, Lupus, 01.09.2013 (дата обращения: 29.01.2014).
7. Наблюдение искусственных спутников Земли. [Электронный ресурс]. URL: http://www.sat.belastro.net/glava2/glava2.php BelAstro.Net, Lupus, 01.09.2013 (дата обращения: 29.01.2014).

## ПРИЛОЖЕНИЕ
## (алгоритмы и программа)

```
unit MainUnit;
interface
uses
  Windows, Messages, SysUtils, Variants, Classes, Graphics, Controls, Forms,
  Dialogs, OpenGL, glTextures, ExtCtrls, DGlut, OpenGLEx, Vcl.StdCtrls,
  Vcl.ComCtrls, Vcl.AppEvnts, Vcl.CheckLst, Vcl.Grids, Vcl.Buttons, Vcl.Menus;
const
  // Идентификатор текстуры
  EARTH_TEXT_ID = 1;
//  Радиус Земли
  EARTH_RAD = 1;
  // Файл текстуры
  EARTH_TEXT_FILE_NAME = 'earth.jpg';
  // Кол-во звезд на экране
  STARS_COUNT = 1000;
  // Максимальная скорость вращения земли, по горизонтали и по вертикали
  MAX_X_SPEED = 10;
  MAX_Y_SPEED = 10;
  // Позиция источника света (Солнца)
  LIGHT_POS: array[1..3] of GLFloat = ( 1, 1, 1);
//  LIGHT_POS1: array[0..3] of GLFloat = (-1.0, -1.0, -1.0, 0.0);
// LIGHT_POS2: array[0..3] of GLFloat = (1.0, 1.0, 1.0, 0.0);
//  LIGHT_POS3: array[0..3] of GLFloat = (1.0, 1.0, 1.0, 0.0);
 ambient:array[1..4] of GLFloat = ( 1, 1, 1, 1);
  // Цвет тумана (Атмосферы Земли)
  FOG_COLOR: array[0..3] of GLFloat = (0.5, 0.8, 1.0, 0.0);
type
  // Структура, описывающая орбиту
  TStar = record
    name: string;    a1:GLFloat;    a2:GLFloat;    a3:GLFloat;    a4:GLFloat;    a5:GLFloat;
    a6:GLFloat;
  end;
```

```
 Dlina = record
   name1: string;   name2: string;   d:GLFloat;   x1:GLFloat;   y1:GLFloat;   z1:GLFloat;
   x2:GLFloat;   y2:GLFloat;   z2:GLFloat;
   end;

 TMainForm = class(TForm)
   Timer: TTimer;
   Panel1: TPanel;
   Timer1: TTimer;
   ApplicationEvents1: TApplicationEvents;
   TrackBar1: TTrackBar;
   Memo1: TMemo;
   PageControl1: TPageControl;
   TabSheet1: TTabSheet;
   CheckListBox1: TCheckListBox;
   StringGrid1: TStringGrid;
   BitBtn1: TBitBtn;   BitBtn2: TBitBtn;   BitBtn3: TBitBtn;
   TabSheet2: TTabSheet;
   BitBtn4: TBitBtn;
   Memo2: TMemo;
   RadioGroup1: TRadioGroup;
   ProgressBar1: TProgressBar;
   StringGrid2: TStringGrid;
   ListBox1: TListBox;
   MainMenu1: TMainMenu;
   N1: TMenuItem;   N2: TMenuItem;   N3: TMenuItem;   N4: TMenuItem;
   N5: TMenuItem;
   procedure FormPaint(Sender: TObject);
   procedure FormCreate(Sender: TObject);
   procedure FormResize(Sender: TObject);
   procedure TimerTimer(Sender: TObject);
   procedure FormDestroy(Sender: TObject);
   procedure FormClose(Sender: TObject; var Action: TCloseAction);
   procedure FormMouseDown(Sender: TObject; Button: TMouseButton; Shift: TShiftState;
X, Y: Integer);
   procedure FormMouseMove(Sender: TObject; Shift: TShiftState; X, Y: Integer);
   procedure Timer1Timer(Sender: TObject);
   procedure ApplicationEvents1Message(var Msg: tagMSG; var Handled: Boolean);
   procedure TrackBar1Change(Sender: TObject);
   procedure CheckListBox1Click(Sender: TObject);
   procedure BitBtn1Click(Sender: TObject);
   procedure BitBtn2Click(Sender: TObject);
   procedure BitBtn3Click(Sender: TObject);
   procedure BitBtn4Click(Sender: TObject);
   procedure ListBox1Click(Sender: TObject);
   procedure N2Click(Sender: TObject);
   procedure N4Click(Sender: TObject);
   procedure N5Click(Sender: TObject);
  private
    // Переменные для инициализации OpenGL
```

```
    HRC:          HGLRC;
    pfd:          TPixelFormatDescriptor;
    nPixelFormat: Integer;
    // Координаты центра экрана
    FOX: Integer;
    FOY: Integer;
    // Старые координаты мыши
    FOldMouseX: Double;
    FOldMouseY: Double;
    // Направление вращения земли, по горизонтали и по вертикали
    FVRotateDirection: Integer;
    FHRotateDirection: Integer;
    // Текстура земли
    FEarthText: TGLTexture;
    // Скорость вращения земли, по горизонтали и по вертикали
    FSpeedX: Double;
    FSpeedY: Double;
    // Сфера
    FGLUObj: GLUquadricObj;
    // Массив звезд
    FStars: array[1..STARS_COUNT] of TStar;
   D: array[0..1000000] of Dlina;
      // Процедура рисования сцены
    procedure DrawScene;
    // Сброс параметров вращения земли (остановка)
    procedure ResetMoveParams;
    procedure Traekt;
  end;

var
  MainForm: TMainForm;
     stars_N:integer;
implementation

{$R *.dfm}
function GetToken(aString, SepChar: string; TokenNum: Byte): string;    var Token: string;StrLen: Byte;TNum: Byte;TEnd: Byte;
begin StrLen := Length(aString);TNum := 1;TEnd := StrLen; while ((TNum <= TokenNum) and (TEnd <> 0)) do begin     TEnd := Pos(SepChar, aString);    if TEnd <> 0 then    begin Token := Copy(aString, 1, TEnd - 1);    Delete(aString, 1, TEnd);    Inc(TNum); end    else begin       Token := aString;    end; end; if TNum >= TokenNum then begin     GetToken := Token; end else begin     GetToken := '';  end;end;
function NumToken(aString, SepChar: string): Byte; var RChar: Char; StrLen,TNum,TEnd: Byte;
begin if SepChar = '#' then begin    RChar := '*' end else begin     RChar := '#' end; StrLen := Length(aString); TNum := 0;TEnd := StrLen;while TEnd <> 0 do begin    Inc(TNum);    TEnd := Pos(SepChar, aString);    if TEnd <> 0 then    begin      aString[TEnd] := RChar;    end; end;Result := TNum;end;

procedure TMainForm.FormCreate(Sender: TObject);
var
```

```
  I: Integer;
  s:string;
begin
PageControl1.TabIndex:=0;
memo1.Lines.LoadFromFile('sput.txt');
 STARS_N:=0 ;
  for I := 0 to memo1.Lines.Count-1 do
  begin
  s:=memo1.Lines.Strings[i];
  if NumToken(s,' ')>=7 then
   begin
     inc(STARS_N);
     FStars[STARS_N].name := gettoken(s,' ',1);
    FStars[STARS_N].a1 := strtofloat(gettoken(s,' ',2));
    FStars[STARS_N].a2 := strtofloat(gettoken(s,' ',3));
    FStars[STARS_N].a3 := strtofloat(gettoken(s,' ',4));
    FStars[STARS_N].a4 := strtofloat(gettoken(s,' ',5));
    FStars[STARS_N].a5 := strtofloat(gettoken(s,' ',6));
    FStars[STARS_N].a6 := strtofloat(gettoken(s,' ',7));
CheckListBox1.Items.Add(s);
   end;
  end;

 for i := 0 to CheckListBox1.Items.Count-1 do
  CheckListBox1.Checked[i]:=true;
 stringgrid1.Cells[0,0]:='Название спутника'   ;
stringgrid1.Cells[0,1]:='Большая полуось(a1)'   ;
stringgrid1.Cells[0,2]:='Эксцентриситет(a2)'   ;
stringgrid1.Cells[0,3]:='Наклонение орбиты(a3)'   ;
stringgrid1.Cells[0,4]:='Долгота восходящего узла(a4)'   ;
stringgrid1.Cells[0,5]:='Аргумент перигея(a5)'   ;
stringgrid1.Cells[0,6]:='Средняя аномалия(a6)'   ;
  ResetMoveParams;
  // Вычисляем центр экрана
  FoX := (ClientWidth-panel1.Width)  div 2;
  FoY := ClientHeight div 2;
  // Инициализируем OpenGL
  FillChar(pfd, SizeOf(pfd), 0);
  pfd.dwFlags    :=  PFD_DOUBLEBUFFER       or
                     PFD_SUPPORT_OPENGL     or
                     PFD_DRAW_TO_WINDOW     or
                     PFD_GENERIC_ACCELERATED;
  nPixelFormat := ChoosePixelFormat(Canvas.Handle, @pfd);
  SetPixelFormat(Canvas.Handle, nPixelFormat, @pfd);
  hrc := wglCreateContext(Canvas.Handle);
  wglMakeCurrent(Canvas.Handle, hrc);
  // Создаем и загружаем текстуру земли
  FEarthText := TGLTexture.Create(EARTH_TEXT_ID);
  FEarthText.LoadFromJpg(EARTH_TEXT_FILE_NAME);
  ReSize;
  Timer.Enabled := True;
```

```
end;
procedure TMainForm.FormClose(Sender: TObject; var Action: TCloseAction);
begin
  wglMakeCurrent(0, 0);
  wglDeleteContext(hrc);
end;

procedure TMainForm.FormPaint(Sender: TObject);
begin
  DrawScene;
end;

procedure TMainForm.FormResize(Sender: TObject);
begin
  FoX := ClientWidth  div 2;
  FoY := ClientHeight div 2;
  glViewport(panel1.Width, 0, ClientWidth-panel1.Width, ClientHeight);
  glMatrixMode(GL_PROJECTION);
  glLoadIdentity;
  gluPerspective(trackbar1.Position, (ClientWidth-panel1.Width) / ClientHeight, 0.2, 10);
  glMatrixMode(GL_MODELVIEW);
  InvalidateRect(Handle, nil, False);
end;

procedure TMainForm.ListBox1Click(Sender: TObject);
var s:string;
i:integer;
pp:double;
begin
s:= ListBox1.Items.Strings[ListBox1.ItemIndex];
if RadioGroup1.ItemIndex=0 then  pp:=1 else pp:=6371.302;

for I := 1 to ListBox1.Items.Count do

if pos(d[i].name1+' + '+d[i].name2,s)>1 then
begin

StringGrid2.Cells[1,0]:=d[i].name1;
StringGrid2.Cells[2,0]:=d[i].name2;
StringGrid2.Cells[1,1]:=floattostr(d[i].x1);
StringGrid2.Cells[1,2]:=floattostr(d[i].y1);
StringGrid2.Cells[1,3]:=floattostr(d[i].z1);
StringGrid2.Cells[2,1]:=floattostr(d[i].x2);
StringGrid2.Cells[2,2]:=floattostr(d[i].y2);
StringGrid2.Cells[2,3]:=floattostr(d[i].z2);
end;
StringGrid2.Cells[0,0]:='name';
StringGrid2.Cells[0,1]:='x =';
StringGrid2.Cells[0,2]:='y =';
StringGrid2.Cells[0,3]:='z =';
end;
```

```
procedure TMainForm.N2Click(Sender: TObject);
begin
Application.Terminate;
end;

procedure TMainForm.N4Click(Sender: TObject);
begin
Showmessage('Математическое и компьютерное моделирование траекторий орбитальных объектов. Данная программа предназначенна для исследования траекторий движения околоземных объектов.');
end;

procedure TMainForm.Timer1Timer(Sender: TObject);
begin
application.ProcessMessages;
end;

procedure TMainForm.TimerTimer(Sender: TObject);
begin
  DrawScene;
  if FSpeedX > 0.2 then
  begin
    FSpeedX := FSpeedX - 0.05;
  end else
  begin
    FSpeedX := 0.2;
  end;

  if FSpeedY > 0.05 then
  begin
    FSpeedY := FSpeedY - 0.05;
  end else
  begin
    FSpeedY := 0;
  end;
end;

procedure TMainForm.TrackBar1Change(Sender: TObject);        begin    MainForm.Resize;
end;

procedure TMainForm.FormMouseDown(Sender: TObject; Button: TMouseButton; Shift: TShiftState; X, Y: Integer);
begin
  FOldMouseX := X;        FOldMouseY := Y;
  if ssLeft in Shift then ResetMoveParams;
end;

procedure TMainForm.FormMouseMove(Sender: TObject; Shift: TShiftState; X, Y: Integer);
begin
  if ssRight in Shift then
```

```
  begin
    FSpeedX := Abs(FOldMouseX - X);
    if FSpeedX > MAX_X_SPEED then
    begin
      FSpeedX := MAX_X_SPEED;
    end;

    FSpeedY := Abs(FOldMouseY - Y);
    if FSpeedY > MAX_Y_SPEED then
    begin
      FSpeedY := MAX_Y_SPEED;
    end;

    if FOldMouseX > X then FHRotateDirection :=  - 1;
    if FOldMouseX < X then FHRotateDirection :=    1;
    if FOldMouseY > Y then FVRotateDirection :=  - 1;
    if FOldMouseY < Y then FVRotateDirection :=    1;
    FOldMouseY := Y;
    FOldMouseX := X;
  end;
end;

procedure TMainForm.FormDestroy(Sender: TObject);
var i:integer;
begin
  FEarthText.Free;
memo1.Clear;
for I := 0 to CheckListBox1.Items.Count-1 do
memo1.Lines.Add(CheckListBox1.Items.Strings[i]);
memo1.Lines.SaveToFile('sput.txt');
end;

procedure TMainForm.ApplicationEvents1Message(var Msg: tagMSG;
  var Handled: Boolean);
begin
  with Msg do                                   // собственно само сообщение
  if message = WM_MOUSEWHEEL Then                       // если крутим колесо
  begin
if wParam/10000000>1 then     trackbar1.Position:=trackbar1.Position+5;
if wParam/10000000<1 then     trackbar1.Position:=trackbar1.Position-5;
  end;
end;

procedure TMainForm.Traekt;
var
  I,j: Integer;
 x,y,z,a,b,c,x1,y1,z1:GLfloat;
begin
   glPolygonMode(GL_FRONT_AND_BACK,GL_Line);
glLineWidth(5);
 for I := 1 to STARS_N do
```

```
begin
 a:= FStars[I].a1;
 c:=a*FStars[I].a2;
 b:= sqrt(sqr(a)-sqr(c));
 glBegin(GL_Polygon);
 glcolor4f(255, 255,255,255);
 if checklistbox1.Checked[i-1] then
 for j := 0 to 360 do
 begin
 x:=-c+a*cos(j*pi/180);
 y:=b*sin(j*pi/180);
 z:=0;
x1:=  x*cos(FStars[I].a3*pi/180)+z*sin(FStars[I].a3*pi/180);
y1:= y;
z1:=-x*sin(FStars[I].a3*pi/180)+z*cos(FStars[I].a3*pi/180);
x:=x1;y:=y1;z:=z1;
x1:=  x*cos(FStars[I].a4*pi/180)+y*sin(FStars[I].a4*pi/180);
y1:= -x*sin(FStars[I].a4*pi/180)+y*cos(FStars[I].a4*pi/180);
z1:=z;
 if i-1=checklistbox1.ItemIndex then glcolor3f(255, 0,0);
 glVertex(x1, y1,z1);
 end;
   glEnd;
 end;
 glPolygonMode(GL_FRONT_AND_BACK,GL_fill);
end;

procedure TMainForm.BitBtn1Click(Sender: TObject);
var s:string;  i,k:integer;
begin
s:='';
for i := 0 to 6 do
s:=s+stringgrid1.Cells[1,i]+' ';
k:=checklistbox1.ItemIndex;
checklistbox1.Items.Strings[k]:=s;
k:=k+1;
  FStars[k].name := gettoken(s,' ',1);    FStars[k].a1 := strtofloat(gettoken(s,' ',2));
    FStars[k].a2 := strtofloat(gettoken(s,' ',3));    FStars[k].a3 := strtofloat(gettoken(s,' ',4));
    FStars[k].a4 := strtofloat(gettoken(s,' ',5));    FStars[k].a5 := strtofloat(gettoken(s,' ',6));
    FStars[k].a6 := strtofloat(gettoken(s,' ',7));
end;

procedure TMainForm.BitBtn2Click(Sender: TObject);
var s:string;
begin
  s:=checklistbox1.Items.Strings[checklistbox1.ItemIndex];
checklistbox1.Items.Add(s);
inc(STARS_N);
  FStars[STARS_N].name := gettoken(s,' ',1);
    FStars[STARS_N].a1 := strtofloat(gettoken(s,' ',2));
    FStars[STARS_N].a2 := strtofloat(gettoken(s,' ',3));
```

```
    FStars[STARS_N].a3 := strtofloat(gettoken(s,' ',4));
    FStars[STARS_N].a4 := strtofloat(gettoken(s,' ',5));
    FStars[STARS_N].a5 := strtofloat(gettoken(s,' ',6));
    FStars[STARS_N].a6 := strtofloat(gettoken(s,' ',7));
checklistbox1.Checked[STARS_N-1]:=true;
end;

procedure TMainForm.BitBtn3Click(Sender: TObject);
var s:string;  i,k:integer;
begin
k:=checklistbox1.ItemIndex;
for I := k+1 to STARS_N-1 do
begin
    FStars[i].name := FStars[i+1].name;
    FStars[i].a1 := FStars[i+1].a1;    FStars[i].a2 := FStars[i+1].a2;
    FStars[i].a3 := FStars[i+1].a3;    FStars[i].a4 := FStars[i+1].a4;
    FStars[i].a5 := FStars[i+1].a5;    FStars[i].a6 := FStars[i+1].a6;
end;
checklistbox1.Items.Delete(k);
STARS_N:=STARS_N-1 ;
checklistbox1.ItemIndex:=0;
end;

function Sort(List: TStringList; Index1, Index2: Integer): integer;
begin
  Result:=trunc(strtofloat(gettoken(List[Index1],' ',1))-strtofloat(gettoken(List[Index2],' ',1)));
end;
//****************** расстояние
procedure TMainForm.BitBtn4Click(Sender: TObject);
var
  I,j,k,kk,w: Integer;
a0,b0,c0, x,y,z,a,b,c,x1,y1,z1,x2,y2,z2,d1,d0,pp:GLfloat;
var
  tmp:TStringList;
begin
w:=0;
  ProgressBar1.Max:=   STARS_N-1;
 for I := 1 to STARS_N-1 do
 begin
 ProgressBar1.Position:=i;
 a0:= FStars[I].a1;
 c0:=a0*FStars[I].a2;
 b0:= sqrt(sqr(a0)-sqr(c0));
 for j := i+1 to STARS_N do
 begin
 a:= FStars[j].a1;
 c:=a*FStars[j].a2;
 b:= sqrt(sqr(a)-sqr(c));
d0:=1000000;
inc(w);
 for k := 0 to 360 do
```

```
 begin
 x:=-c0+a0*cos(k*pi/180);
 y:=b0*sin(k*pi/180);
 z:=0;
x1:=  x*cos(FStars[I].a3*pi/180)+z*sin(FStars[I].a3*pi/180);
y1:= y;
z1:=-x*sin(FStars[I].a3*pi/180)+z*cos(FStars[I].a3*pi/180);
x:=x1;y:=y1;z:=z1;
x1:=  x*cos(FStars[I].a4*pi/180)+y*sin(FStars[I].a4*pi/180);
y1:= -x*sin(FStars[I].a4*pi/180)+y*cos(FStars[I].a4*pi/180);
z1:=z;
 x2:=x1;y2:=y1;z2:=z1;
for kk := 0 to 360 do
 begin
 x:=-c+a*cos(kk*pi/180);
 y:=b*sin(kk*pi/180);
 z:=0;
x1:=  x*cos(FStars[j].a3*pi/180)+z*sin(FStars[j].a3*pi/180);
y1:= y;
z1:=-x*sin(FStars[j].a3*pi/180)+z*cos(FStars[j].a3*pi/180);
x:=x1;y:=y1;z:=z1;
x1:=  x*cos(FStars[j].a4*pi/180)+y*sin(FStars[j].a4*pi/180);
y1:= -x*sin(FStars[j].a4*pi/180)+y*cos(FStars[j].a4*pi/180);
z1:=z;
d1:=sqrt(sqr(x2-x1)+sqr(y2-y1)+sqr(z2-z1));
 if d1<d0 then
 begin
  d0:=d1;
d[w].name1:=FStars[i].name;
d[w].name2:=FStars[j].name;
d[w].d:=d0;
d[w].x1:=x2;    d[w].y1:=y2;     d[w].z1:=z2;
d[w].x2:=x1;    d[w].y2:=y1;     d[w].z2:=z1;
 end; end; end; end; end;
memo2.Clear;
if RadioGroup1.ItemIndex=0 then  pp:=1 else pp:=6371.302;
for I := 1 to w do
memo2.Lines.Add(floattostr(d[i].d*pp) +'     '+d[i].name1+' + '+d[i].name2);
  tmp:=TStringList.Create;
  tmp.Assign(Memo2.Lines);
if RadioGroup1.ItemIndex=1 then  tmp.CustomSort(Sort) else
tmp.Sort;
  Memo2.Lines.Assign(tmp);
  listbox1.Items.Assign(tmp);
    FreeAndNil(tmp);
end;
procedure TMainForm.CheckListBox1Click(Sender: TObject);
var s:string;  i:integer;
begin
s:=checklistbox1.Items.Strings[checklistbox1.ItemIndex];
for i := 0 to 6 do
```

```
stringgrid1.Cells[1,i]:=gettoken(s,' ',i+1);
end;
procedure TMainForm.DrawScene;
var  mvMatrix: array[0..15] of glFloat;
begin    // Очищаем буфер цвета и буфер глубины
  glClearColor(0, 0, 0, 0);
  glClear(GL_COLOR_BUFFER_BIT or GL_DEPTH_BUFFER_BIT);
  // Рисуем землю   // Включаем буфер глубины
  glEnable(GL_DEPTH_TEST);
  // Включаем освещение
  glEnable(GL_LIGHTING);      glEnable(GL_LIGHT0);
  // Включаем туман (Атмосферу)
  glEnable(GL_FOG);
  glFogi(GL_FOG_MODE,  GL_LINEAR);
  glFogfv(GL_FOG_COLOR, @FOG_COLOR);
  glFogf(GL_FOG_DENSITY, 1.0);
  glFogf(GL_FOG_START, 0.9);
  glFogf(GL_FOG_END, 0.3);
  glMatrixMode(GL_PROJECTION);
  glPushMatrix;     glTranslate(0, 0, - 4);      glPushMatrix;
  Traekt;
   glPopMatrix;
  // Рисуем землю
  glEnable(GL_NORMALIZE);       glEnable(GL_TEXTURE_2D);
  glBindTexture(GL_TEXTURE_2D, FEarthText.TextureID);
  FGLUObj := gluNewQuadric;
  gluQuadricTexture(FGLUObj, GL_TRUE);
  gluQuadricDrawStyle(FGLUObj, GLU_FILL);
  gluSphere(FGLUObj, EARTH_RAD, 64, 64);
  gluDeleteQuadric(FGLUObj);
  glDisable(GL_TEXTURE_2D);
  glPopMatrix;
  glMatrixMode(GL_MODELVIEW);
  glDisable(GL_LIGHTING);
  glDisable(GL_DEPTH_TEST);
  glMatrixMode(GL_MODELVIEW);
  glGetFloatv(GL_MODELVIEW_MATRIX, @mvMatrix);
  glLoadIdentity();
  glRotatef(FHRotateDirection * FSpeedX / 2, 0, 1, 0);
  glRotatef(FVRotateDirection * FSpeedY / 2, 1, 0, 0);
  glLightModelfv(GL_LIGHT_MODEL_AMBIENT, @ambient);
  glMultMatrixf(@mvMatrix);
  SwapBuffers(Canvas.Handle);
end;
procedure TMainForm.ResetMoveParams;
begin
  FSpeedX := 0.2;  FSpeedY := 0;
  FHRotateDirection := 1;
end;   end.
```