ностью. Определение температуры в центральной части реактора с помощью математического моделирования является актуальной задачей, позволяющей оптимизировать технологический процесс.

СПИСОК ЛИТЕРАТУРЫ

1. Копченова Н.В. Вычислительная математика в примерах и задачах: Учебное пособие / Копченова Н. В. Марон И. А. 2 – е изд., стер. – СПб.: Издательство «Лань», 2008. – 368с.

УДК 004.02

Сенотова Светлана Анатольевна,
к.т.н., доцент, доцент кафедры «Вычислительные машины и комплексы», ФГБОУ ВО «Ангарский государственный технический университет», тел.: 89021723488
Маслюченко Андрей Павлович,
магистрант кафедры «Вычислительные машины и комплексы», ФГБОУ ВО «Ангарский государственный технический университет», тел.: 89025420060

КОМПЬЮТЕРНОЕ МОДЕЛИРОВАНИЕ ОРБИТАЛЬНОЙ ТРОСОВОЙ СИСТЕМЫ

Senotova S.A., Masluchenko A. P.

COMPUTER MODELING OF ORBITAL CABLE SYSTEM

Аннотация. *В статье рассматриваются дифференциальные уравнения движения орбитальной тросовой системы. Системы уравнений решаются методом Эйлера и методом Рунге-Кутта. Программы составлены на языке Python в графическом редакторе Blender с визуальным представлением математических моделей.*

Ключевые слова: *орбитальные тросовые системы, математические модели, дифференциальные уравнения, язык Python, графический редактор Blender.*

Abstract. *The article discusses the differential equations of motion of the orbital cable system. Systems of equations are solved by the Euler method and the Runge-Kutt method. Programs are written in Python in the graphic editor Blender with a visual representation of mathematical models.*

Keyword: *orbital cable systems, mathematical models, differential equations, Python language, graphic editor Blender.*

Исследованию движения сложных космических систем посвящено большое количество работ, как в нашей стране [1-3], так и за рубежом. В последнее время приобрели большое значение задачи вращательного движения орбитальных космических систем, содержащих панели солнечных батарей, антенны, гибкие упругие тросы. Космические аппараты с тросами получили название орбитальных тросовых систем. При изучении движения орбитальных тросовых систем используются различные математические модели.

Рассмотрим уравнения относительного движения привязанного субспутника в сферических координатах [1]:

$$\frac{d^2\theta}{dt^2} = 2\left(\frac{d\varphi}{dt}\right) tg\,\varphi \left(\frac{d\theta}{dt} + 1\right) - 3\sin\theta\cos\theta,$$

$$\frac{d^2\varphi}{dt^2} = -\sin\varphi\cos\varphi\left[\left(\frac{d\theta}{dt} + 1\right)^2 + 3\,cos^2\,\theta\right] \tag{1}$$

при начальных условиях

$$\varphi(0) = 0{,}3,$$

$$\frac{d\varphi}{dt}(0) = 0{,}1,$$

$$\theta(0) = 0{,}3, \tag{2}$$

$$\frac{d\theta}{dt}(0) = 0{,}1.$$

Запишем дифференциальные уравнения второго порядка (1) в виде системы четырёх дифференциальных уравнений первого порядка

$$\frac{dy}{dt} = y1,$$

$$\frac{dx}{dt} = x1, \qquad (3)$$

$$\frac{dy1}{dt} = 2\,x1\,tg\,x\,(y1+1) - 3\sin y\,\cos y,$$

$$\frac{dx1}{dt} = -\sin x\,\cos x\,\left((y1+1)^2 + 3\cos^2 y\right)$$

с начальными условиями

$$x(0) = 0{,}3,$$

$$x1(0) = 0{,}1, \qquad (4)$$

$$y(0) = 0{,}3,$$

$$y1(0) = 0{,}1,$$

где x, y – углы, $x1, y1$ – угловые скорости (рисунок 1).

Решим систему (3) - (4) методом Эйлера (рисунок 2) и методом Рунге-Кутта (рисунок 3). Программы составлены на языке Python в графическом редакторе Blender [4] с визуальным представлением математической модели [5].

Метод Эйлера

```
import bge #Импорт модуля Bleder Game Engine
import math #Импорт модуля математических команд
o = bge.logic.getCurrentController().owner
h=o['h']
def f1 (x,x1,y,y1):
   return y1
def f2 (x,x1,y,y1):
   return x1
def f3 (x,x1,y,y1):
return 2*x1*math.tan(x)*(y1+1)-3*math.sin(y)*math.cos(y)
def f4 (x,x1,y,y1):
return math.sin(x)*math.cos(x)*((y1+1)**2+3*math.cos(y)**2)
o['x']=o['x']+h*f2(o['x'],o['x1'],o['y'],o['y1'])
o['x1']=o['x1']+h*f4(o['x'],o['x1'],o['y'],o['y1'])
o['y']=o['y']+h*f1(o['x'],o['x1'],o['y'],o['y1'])
o['y1']=o['y1']+h*f3(o['x'],o['x1'],o['y'],o['y1'])
o.applyRotation ([0,o['x'],o['y']])
```

Рисунок 1 – Начальные условия

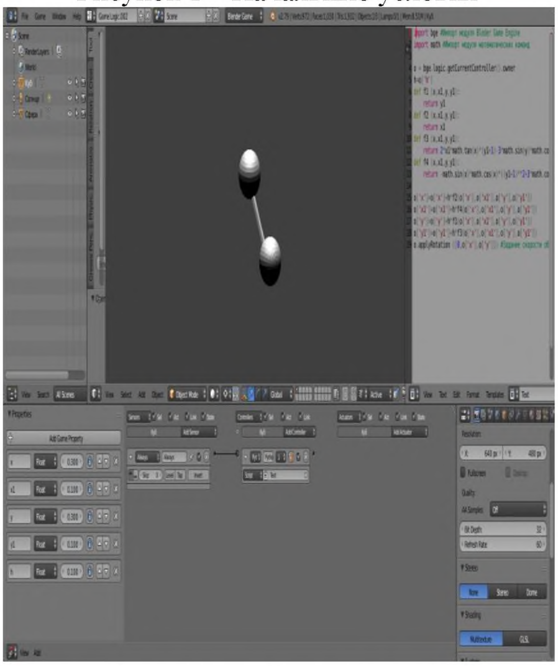

Рисунок 2 – Метод Эйлера для субспутника

Метод Рунге-Кутта

```
import bge #Импорт модуля Bleder Game Engine
import math #Импорт модуля математических команд
o = bge.logic.getCurrentController().owner
h=o['h']
def f1 (x,x1,y,y1):
   return y1
def f2 (x,x1,y,y1):
   return x1
def f3 (x,x1,y,y1):
return 2*x1*math.tan(x)*(y1+1)-3*math.sin(y)*math.cos(y)
def f4 (x,x1,y,y1):
return math.sin(x)*math.cos(x)*((y1+1)**2+3*math.cos(y)**2)
s1=h*f1(o['x'],o['x1'],o['y'],o['y1'])
t1=h*f2(o['x'],o['x1'],o['y'],o['y1'])
```

```
u1=h*f3(o['x'],o['x1'],o['y'],o['y1'])
v1=h*f4(o['x'],o['x1'],o['y'],o['y1'])
s2=h*f1(o['x']+t1/2,o['x1']+v1/2,o['y']+s1/2,o['y1']+u1/2)
t2=h*f2(o['x']+t1/2,o['x1']+v1/2,o['y']+s1/2,o['y1']+u1/2)
u2=h*f3(o['x']+t1/2,o['x1']+v1/2,o['y']+s1/2,o['y1']+u1/2)
v2=h*f4(o['x']+t1/2,o['x1']+v1/2,o['y']+s1/2,o['y1']+u1/2)
s3=h*f1(o['x']+t2/2,o['x1']+v2/2,o['y']+s2/2,o['y1']+u2/2)
t3=h*f2(o['x']+t2/2,o['x1']+v2/2,o['y']+s2/2,o['y1']+u2/2)
u3=h*f3(o['x']+t2/2,o['x1']+v2/2,o['y']+s2/2,o['y1']+u2/2)
v3=h*f4(o['x']+t2/2,o['x1']+v2/2,o['y']+s2/2,o['y1']+u2/2)

s4=h*f1(o['x']+t3,o['x1']+v3,o['y']+s3,o['y1']+u3)
t4=h*f2(o['x']+t3,o['x1']+v3,o['y']+s3,o['y1']+u3)
u4=h*f3(o['x']+t3,o['x1']+v3,o['y']+s3,o['y1']+u3)
v4=h*f4(o['x']+t3,o['x1']+v3,o['y']+s3,o['y1']+u3)
o['x']=o['x']+(t1+2*t2+2*t3+t4)/6
o['x1']=o['x1']+(v1+2*v2+2*v3+v4)/6
o['y']=o['y']+(s1+2*s2+2*s3+s4)/6
o['y1']=o['y1']+(u1+2*u2+2*u3+u4)/6
o.applyRotation ([0,o['x'],o['y']])
```

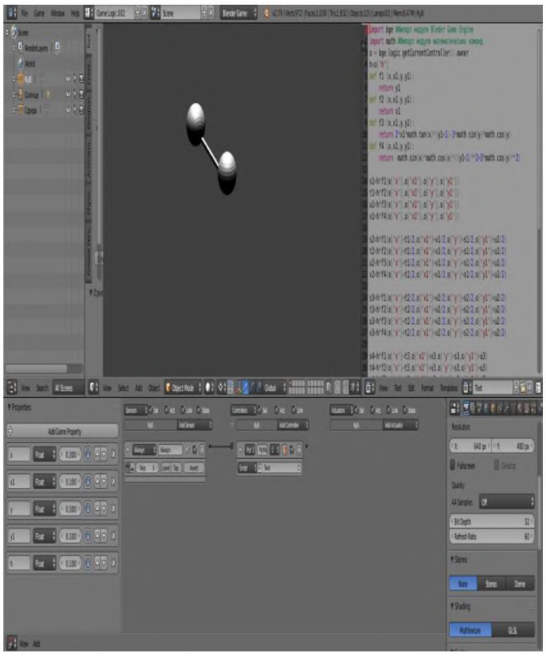

Рисунок 3 – Метод Рунге-Кутта для субспутника

Рассмотрим уравнения движения центра масс тросовой системы в декартовых координатах [1]

$$\frac{d^2x}{dt^2} = 2\omega\frac{dy}{dt} + 3\omega^2 x, \qquad (5)$$
$$\frac{d^2y}{dt^2} = -2\omega\frac{dx}{dt},$$
$$\frac{d^2z}{dt^2} = -\omega^2 z$$

с начальными условиями

$$x(0) = 0,$$
$$\frac{dx}{dt} = 0{,}1,$$
$$y(0) = 0, \qquad (6)$$
$$\frac{dy}{dt} = 0{,}1,$$
$$z(0) = 0,$$
$$\frac{dz}{dt} = 0{,}1,$$

где ω – угловая скорость движения по круговой орбите. Решим систему (5)-(6) методом Эйлера (рисунок 4). Программа составлена на языке Python в графическом редакторе Blender с визуальным представлением математической модели.

Метод Эйлера

```
import bge
import math
o = bge.logic.getCurrentController().owner
h=o['h']
w=o['w']
o['x2']=2*w*o['y1']+3*w**2*o['x']
o['y2']=-2*w*o['x1']
o['z2']=-w**2*o['z']
o['x1']=o['x1']+o['x2']*h
o['y1']=o['y1']+o['y2']*h
o['z1']=o['z1']+o['z2']*h
o['x']=o['x']+o['x1']*h
o['y']=o['y']+o['y1']*h
o['z']=o['z']+o['z1']*h
o.worldPosition.x+=o['x1']*h
o.worldPosition.y+=o['y1']*h
o.worldPosition.z+=o['z1']*h
x=o.worldPosition.x
y=o.worldPosition.y
z=o.worldPosition.z
o.applyMovement([x,y,z])
```

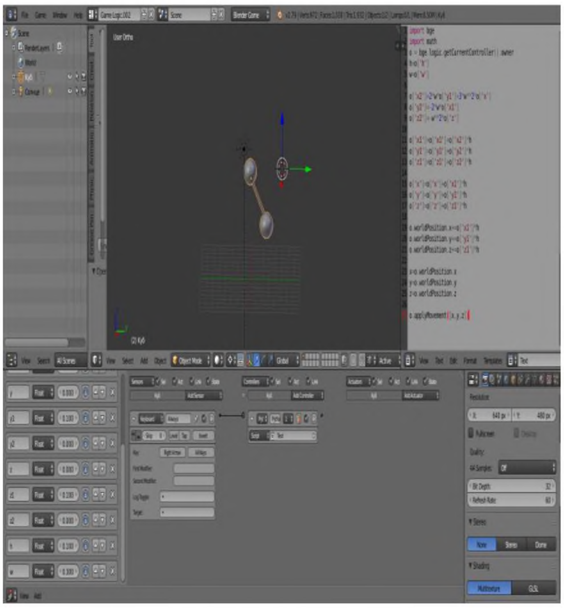

Рисунок 4 – Метод Эйлера для тросовой системы

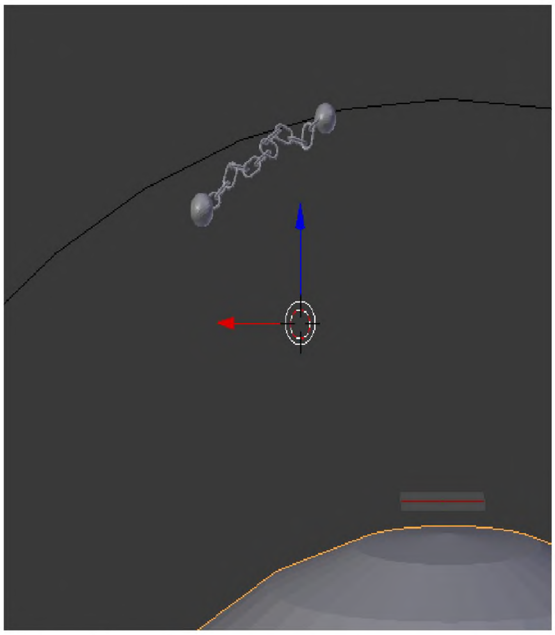

Рисунок 5 – Физическая модель орбитальной тросовой системы

Физическое моделирование

В случае если математическая модель не известна, Blender даёт возможность создать физическую модель без каких-либо формул. В качестве примера физического моделирования в Blender создана орбитальная тросовая система с использованием встроенной физики твёрдых тел (рисунок 5).

В результате созданы компьютерные модели орбитальных тросовых систем. Составлены программы на языке Python в графическом редакторе Blender с визуальным представлением математических моделей. Разработана физическая модель тросовой системы.

Сделан вывод, что при визуальном представлении математической модели метод Эйлера и метод Рунге-Кутта работают одинаково. На самом деле, метод Рунге-Кутта намного точнее метода Эйлера.

СПИСОК ЛИТЕРАТУРЫ

1. Белецкий В.В. Динамика космических тросовых систем. / В.В. Белецкий, Е.М. Левин. – М.: Наука. – 1990.

2. Набиуллин М.К. Стационарные движения и устойчивость упругих спутников. / М.К. Набиуллин. – Новосибирск: Наука. – 1990.

3. Асламов В.С. Влияние упругости орбитальной тросовой на колебания спутника. / В.С. Асламов. // Прикладная математика и механика. – 2010. –т. 74. – № 4. – стр. 582-593.

4. Прахов А.А. Blender: 3D-моделирование и анимация. Руководство для начинающих. / А.А. Прахов. – СПб.: «БХВ-Петербург», 2012.

5. Матвеев М.А. Исследование хаотических процессов с помощью Blender // Молодой ученый. 2016. № 11. С. 196-202.