Санкт-Петербургский государственный политехнический университет

Институт Прикладной математики и механики  
Кафедра Теоретической механики

Н.Д. Мущак

ДВИЖЕНИЕ СПУТНИКА В ДВОЙНОЙ СИСТЕМЕ

Курсовой проект

Направление подготовки бакалавров: 010800 Механика и математическое моделирование

Группа 23604/1

Руководитель проекта: Панченко А.Ю.

Допущен к защите:

«\_\_» 20\_\_ г.

Санкт-Петербург

2015

# 

# **Оглавление**

[**Оглавление** 2](file:///C:\Users\Се\Downloads\Programming_project1.docx#_Toc388226015)

[**Введение** 3](file:///C:\Users\Се\Downloads\Programming_project1.docx#_Toc388226016)

[**Глава 1. Решение задачи.** 4](file:///C:\Users\Се\Downloads\Programming_project1.docx#_Toc388226017)

[1.1 **С помощью уравнения Лаграджа 2-ого рода.** 4](file:///C:\Users\Се\Downloads\Programming_project1.docx#_Toc388226018)

[1.2 **С помощью 2-ого закона Ньютона** 7](file:///C:\Users\Се\Downloads\Programming_project1.docx#_Toc388226019)

[**Глава 2. . Написание программы** 10](file:///C:\Users\Се\Downloads\Programming_project1.docx#_Toc388226021)

[2.1 **Задачи, поставленные перед программой** 10](file:///C:\Users\Се\Downloads\Programming_project1.docx#_Toc388226022)

[2.2 **Написание кода** 11](file:///C:\Users\Се\Downloads\Programming_project1.docx#_Toc388226023)

[**Итоги работы**  14](file:///C:\Users\Се\Downloads\Programming_project1.docx#_Toc388226021)

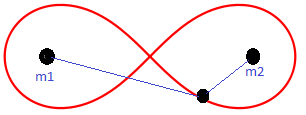
[**Литература**](file:///C:\Users\Се\Downloads\Programming_project1.docx#_Toc388226027) 15

**Введение**

Данный проект посвящен изучению движения спутника в двойной системе под действием гравитации. В процессе выполнения курсовой работы необходимо решить следующие задачи:

* Написать программу, моделирующую движение спутника в двойной системе. Двойная система состоит из 2 неподвижных планет и спутника вращающегося вокруг них как показано на рисунке снизу.
* Определить стационарные орбиты спутника, а также устойчивость движения спутника.

В первой главе приведена информация решения задачи со стороны физики .В ней будет показано два вида решения данной задачи. Во второй главе описывается то, как создается программа с примерами кодстраниц.

****

**Глава 1. Решение задачи.**

* 1. **С помощью уравнения Лаграджа 2-ого рода.**

Запишим уравнение Лагранжа 2-ого рода:

C:\Users\Никита\AppData\Local\Microsoft\Windows\INetCache\Content.Word\Lagrange.png

**-** функция Лагранжа, которая представляет собой разность кинетической и потенциальной энергий.



***q***- обобщенная координата, ***t*** — время, ***i***— число степеней свободы механической системы.

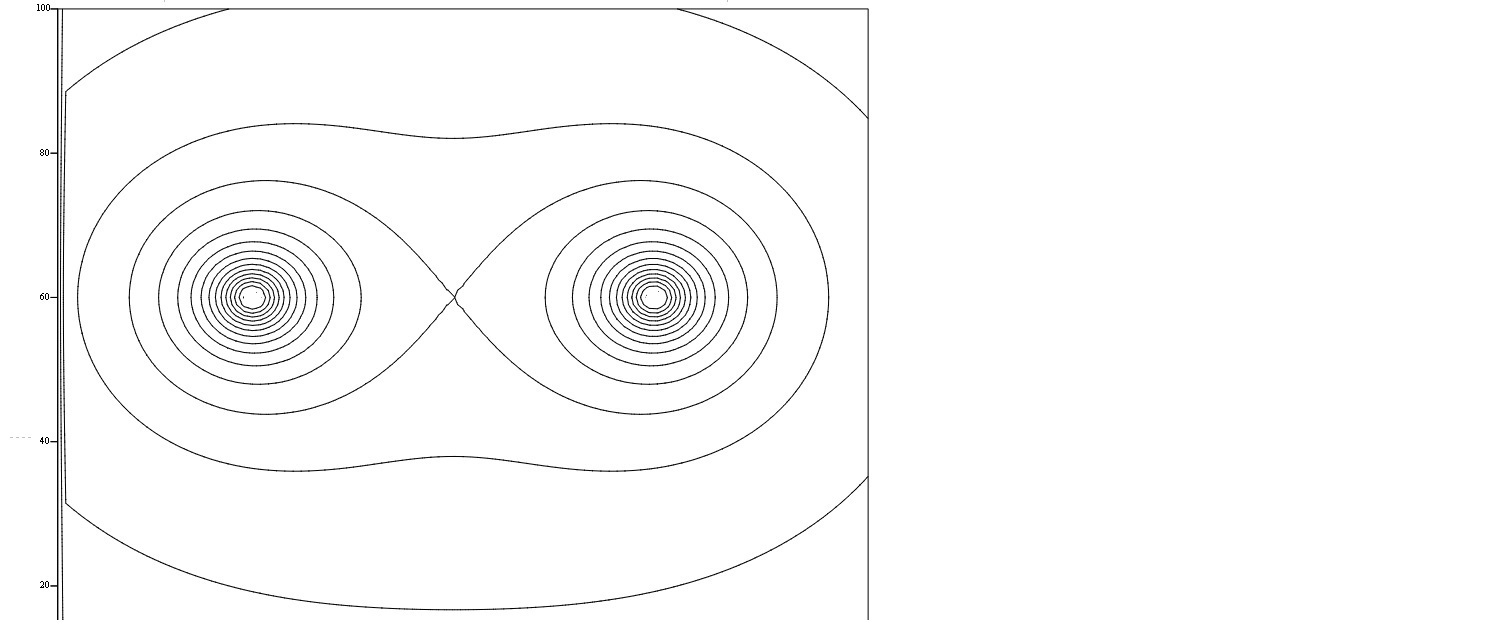
Дальнейшим дифференцированием получаем уравнение движения:



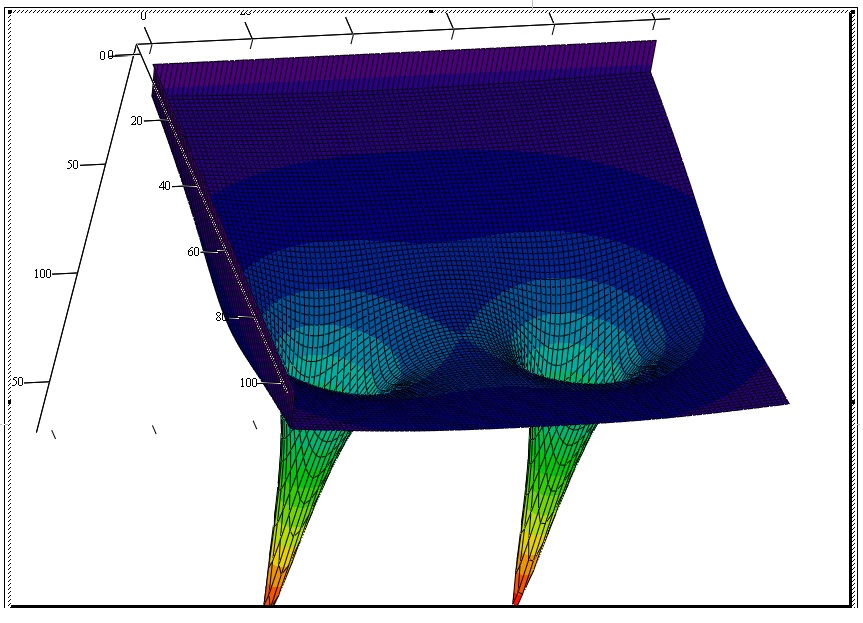
При этом потенциал гравитационного поля вокруг двух планет примет вид:



График силовых линий потенциального гравитационного поля будет выглядеть:



**Рис.1**



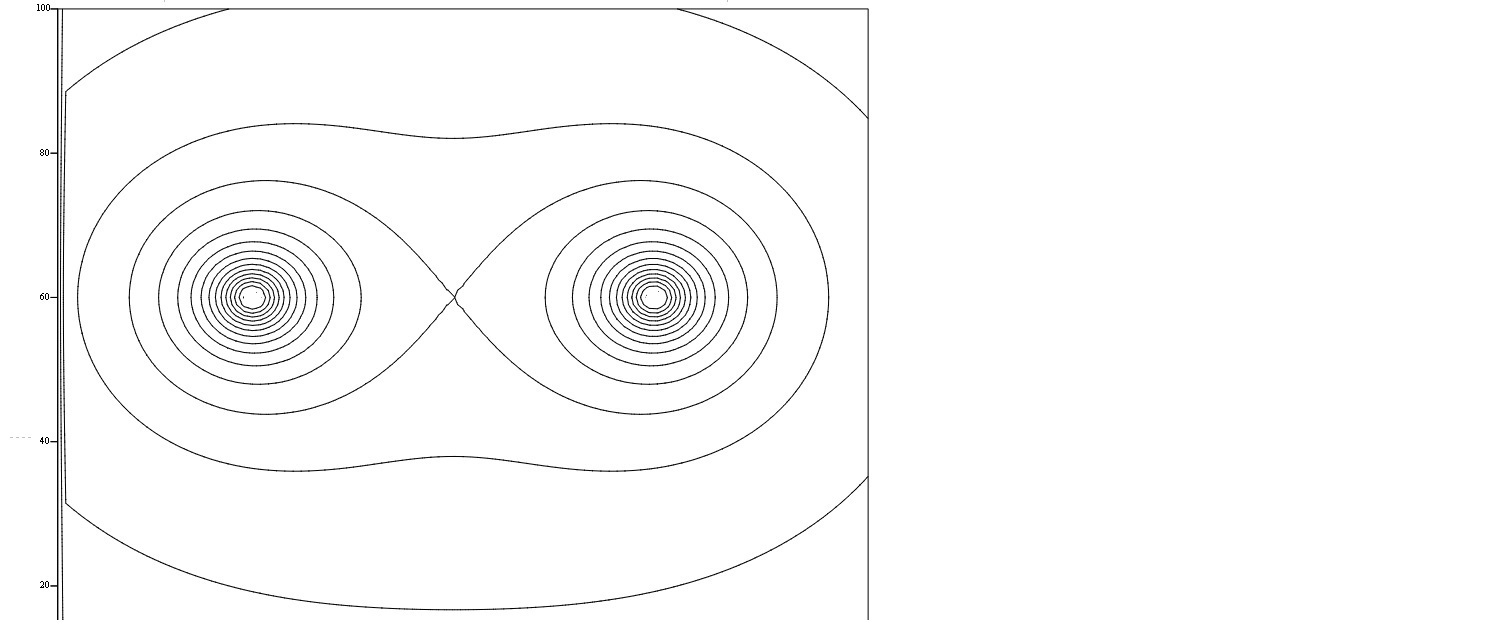
**Рис.2**

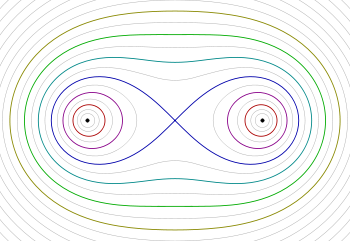
На рис.1 представлен контурный график потенцильного поля, а на рис.2 3-D график.

Если сравнить рис.1 с овалами Кассини[1], то можно обнаружить некоторое сходстство.

Ниже можно посмотреть сравнение рис.1 с овалами Кассини.

[1] Овалы Кассини и лемниската- Курс высшей математики, Т.1





Теперь немного подробней про овалы Кассини. **Овал Кассини** — [кривая](https://ru.wikipedia.org/wiki/%D0%9A%D1%80%D0%B8%D0%B2%D0%B0%D1%8F), являющаяся [геометрическим местом точек](https://ru.wikipedia.org/wiki/%D0%93%D0%B5%D0%BE%D0%BC%D0%B5%D1%82%D1%80%D0%B8%D1%87%D0%B5%D1%81%D0%BA%D0%BE%D0%B5_%D0%BC%D0%B5%D1%81%D1%82%D0%BE_%D1%82%D0%BE%D1%87%D0%B5%D0%BA), произведение расстояний от которых до двух заданных точек (фокусов) постоянно и равно квадрату некоторого числа a.

Частным случаем овала Кассини при фокусном расстоянии равном 2a является [лемниската Бернулли](https://ru.wikipedia.org/wiki/%D0%9B%D0%B5%D0%BC%D0%BD%D0%B8%D1%81%D0%BA%D0%B0%D1%82%D0%B0_%D0%91%D0%B5%D1%80%D0%BD%D1%83%D0%BB%D0%BB%D0%B8). С другой стороны, сам овал является частным случаем [лемнискаты](https://ru.wikipedia.org/wiki/%D0%9B%D0%B5%D0%BC%D0%BD%D0%B8%D1%81%D0%BA%D0%B0%D1%82%D0%B0).

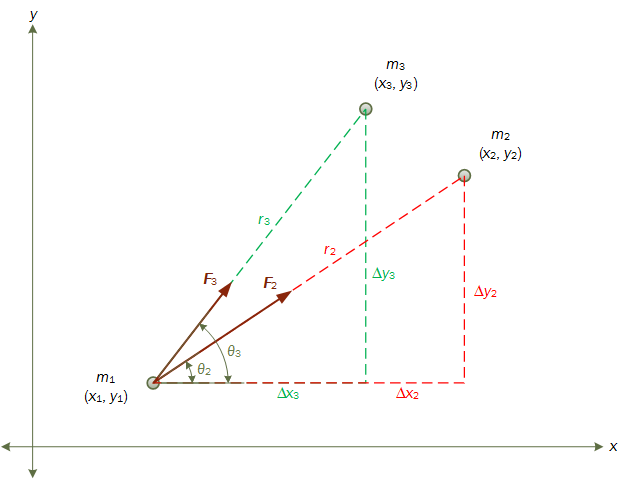
Кривая была придумана астрономом [Джованни Кассини](https://ru.wikipedia.org/wiki/%D0%9A%D0%B0%D1%81%D1%81%D0%B8%D0%BD%D0%B8,_%D0%94%D0%B6%D0%BE%D0%B2%D0%B0%D0%BD%D0%BD%D0%B8_%D0%94%D0%BE%D0%BC%D0%B5%D0%BD%D0%B8%D0%BA%D0%BE). Он ошибочно считал, что она точнее определяет [орбиту](https://ru.wikipedia.org/wiki/%D0%9E%D1%80%D0%B1%D0%B8%D1%82%D0%B0) [Земли](https://ru.wikipedia.org/wiki/%D0%97%D0%B5%D0%BC%D0%BB%D1%8F_(%D0%BF%D0%BB%D0%B0%D0%BD%D0%B5%D1%82%D0%B0)), чем [эллипс](https://ru.wikipedia.org/wiki/%D0%AD%D0%BB%D0%BB%D0%B8%D0%BF%D1%81). Хотя эту линию называют [овалом](https://ru.wikipedia.org/wiki/%D0%9E%D0%B2%D0%B0%D0%BB) Кассини, она не всегда овальна.

Неявное уравнение в прямоугольных координатах:



**1.2.С помощью 2-ого закона Ньютона**

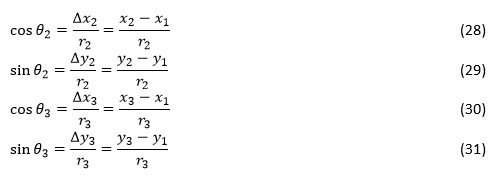
Расмотрим систему, состоящую из трех тел:



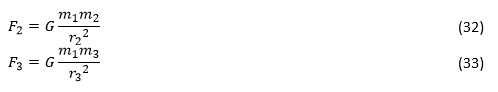
Сначала заметим, что результирующая сила F₁, действующая на тело m₁, будет суммой сил F₂ и F₃. Это значит, что F₁ = m₁a₁ = F₂ + F₃. Теперь по тригонометрическим законам, мы можем разложить модуль результирующей силы F₁, действующей на тело m₁, на компоненты x и y:

[IC694007.png](http://tm.spbstu.ru/%D0%A4%D0%B0%D0%B9%D0%BB:IC694007.png)

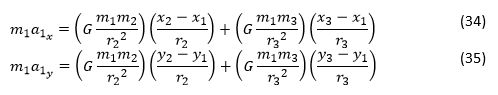
В красном и зеленом треугольниках на рис. мы видим:

[](http://tm.spbstu.ru/%D0%A4%D0%B0%D0%B9%D0%BB:IC694008.png)

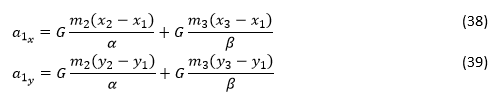
Согласно закону всемирного тяготения Ньютона, F₂ и F₃ можно выразить как:

[](http://tm.spbstu.ru/%D0%A4%D0%B0%D0%B9%D0%BB:IC694009.png)

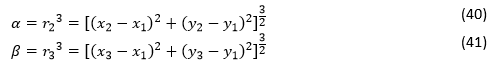
Подставляя формулы, получим:

[](http://tm.spbstu.ru/%D0%A4%D0%B0%D0%B9%D0%BB:IC694010.png)

Упрощая формулы, имеем:

[](http://tm.spbstu.ru/%D0%A4%D0%B0%D0%B9%D0%BB:IC694012.png)

Здесь α и β равны:

[](http://tm.spbstu.ru/%D0%A4%D0%B0%D0%B9%D0%BB:IC694013.png)

Эту систему уравнений (34,35,38-41) можно решить численно методом интегрирования "чехарда"[2] (формулы 22–24) по заданным начальным условиям (значения массы, положения и скорости для каждого тела) с приемлемой точностью и стабильностью. Чтобы быстро добиться высокой точности, можно использовать рабочий веб-процесс для выполнения численного интегрирования в потоке, отдельном от потока пользовательского интерфейса главной страницы.

Рассмотрим N небесных тел. Пусть i обозначает одно из тел (i = 1, …, N), а h — малый интервал времени. В позиционном алгоритме Верле следующие значения положения и скорости тела i вычисляются следующим образом:

[](http://tm.spbstu.ru/%D0%A4%D0%B0%D0%B9%D0%BB:IC693998.png)

[2] Интегрирование чехарда-[Лекция «Реализация физики на основе интегрирования Верлета»](http://www.gamedev.ru/community/gamedev_lecture/articles/?id=26)

**Глава 2. Написание программы**

**2.1 Задачи, поставленные перед программой**

Программа должна выполнять следующие функции:

1. Визуализация двух планет и их спутника.
2. Взаимодействие между спутникоми планетами.
3. У пользователя должна быть возможность изменять параметры для всех трех тел.

**2.2 Написание кода**

Ниже приведены части кода, которые отвечают за расчётную часть программы: рассчитывается взаимодействие между спутником и планетами.

// Расчетная часть программы

var N = 3; // Количество тел.

var G = 6.67384E-11; // Гравитационая постоянная .

var h = 0.000001; // Интервал между шагами интегрирования.

var iterationsPerFrame = 400; // Число итераций в потоке.

var m1;

var m1\_half; // Копия m1.

var m2;

var m2\_half;

var m3;

var m3\_half;

self.onmessage = function (evt) { // передаем данные на экран пользователя.

switch (evt.data.cmd) {

case 'init':

init(evt.data.initialConditions); // Передаем начальные условия.

break;

case 'crunch':

crunch();

break;

default:

console.error("ERROR FROM worker.js: SWITCH STATEMENT ERROR IN self.onmessage");

} // switch

};

function alpha(m1, m2) {

var delta\_x = m2.p.x - m1.p.x;

var delta\_y = m2.p.y - m1.p.y;

var delta\_x\_squared = delta\_x \* delta\_x;

var delta\_y\_squared = delta\_y \* delta\_y;

var base = delta\_x\_squared + delta\_y\_squared;

return Math.sqrt(base \* base \* base);

}

function beta(m1, m3) {

var delta\_x = m3.p.x - m1.p.x;

var delta\_y = m3.p.y - m1.p.y;

var delta\_x\_squared = delta\_x \* delta\_x;

var delta\_y\_squared = delta\_y \* delta\_y;

var base = delta\_x\_squared + delta\_y\_squared;

return Math.sqrt(base \* base \* base);

}

function gamma(m2, m3) {

var delta\_x = m3.p.x - m2.p.x;

var delta\_y = m3.p.y - m2.p.y;

var delta\_x\_squared = delta\_x \* delta\_x;

var delta\_y\_squared = delta\_y \* delta\_y;

var base = delta\_x\_squared + delta\_y\_squared;

return Math.sqrt(base \* base \* base);

}

this.init = function (initialConditions) {

function Mass(initialCondition) {

this.m = initialCondition.mass;

this.p = { x: initialCondition.position.x, y: initialCondition.position.y };

this.v = { x: initialCondition.velocity.x, y: initialCondition.velocity.y };

this.a = {};

}

if (initialConditions.length != N) {

console.error("ERROR FROM worker.js: THE initialConditions ARRAY DOES NOT CONTAIN EXACTLY " + N + " OBJECTS - init() TERMINATED");

return;

}

// Устанавливаем глобальные переменные

m1 = new Mass(initialConditions[0]);

m1\_half = new Mass(initialConditions[0]); // Создаем копию m1.

m2 = new Mass(initialConditions[1]);

m2\_half = new Mass(initialConditions[1]);

m3 = new Mass(initialConditions[2]);

m3\_half = new Mass(initialConditions[2]);

// Вычисляем ускорение

m1.a.x = G \* m2.m \* (m2.p.x - m1.p.x) / alpha(m1, m2) + G \* m3.m \* (m3.p.x - m1.p.x) / beta(m1, m3); // Equation 42.

m1.a.y = G \* m2.m \* (m2.p.y - m1.p.y) / alpha(m1, m2) + G \* m3.m \* (m3.p.y - m1.p.y) / beta(m1, m3); // Equation 43.

function equation25(x, v, a) {

return x + 0.5 \* h \* v + 0.25 \* (h \* h) \* a;

}

m1\_half.p.x = equation25(m1.p.x, m1.v.x, m1.a.x);

m1\_half.p.y = equation25(m1.p.y, m1.v.y, m1.a.y);

} // this.init

this.crunch = function () {

for (var i = 0; i < iterationsPerFrame; i++) {

m1\_half.a.x = G \* m2\_half.m \* (m2\_half.p.x - m1\_half.p.x) / alpha(m1\_half, m2\_half) + G \* m3\_half.m \* (m3\_half.p.x - m1\_half.p.x) / beta(m1\_half, m3\_half);

m1\_half.a.y = G \* m2\_half.m \* (m2\_half.p.y - m1\_half.p.y) / alpha(m1\_half, m2\_half) + G \* m3\_half.m \* (m3\_half.p.y - m1\_half.p.y) / beta(m1\_half, m3\_half);

// Вычисляем скорость

m1.v.x = equation23(m1.v.x, m1\_half.a.x);

m1.v.y = equation23(m1.v.y, m1\_half.a.y);

// вычисляем координату

m1.p.x = equation24(m1\_half.p.x, m1.v.x);

m1.p.y = equation24(m1\_half.p.y, m1.v.y);

m1\_half.p.x = equation22(m1.p.x, m1.v.x);

m1\_half.p.y = equation22(m1.p.y, m1.v.y);

} // for

self.postMessage([m1]); // передаем посчитанные данные на отрисовку пользователю

function equation23(v, a) {

return v + h \* a;

}

function equation24(x, v) {

return x + 0.5 \* h \* v;

}

function equation22(x, v) {

return x + 0.5 \* h \* v;

}

**2.3 Итог работы.**   
  
 Таким образом, в ходе работы над проектом, смоделирован процесс движения спутника в двойной системе. Обработка результатов показала, что спутник пролетает вокруг двух тел по лемнискате один раз, а потом отклоняется. Это отклонение получается за счет накопления ошибки вычислений в программе.

**Список литературы**

1. Баррет Д. **JavaScript. Web-профессионалам.** - Киев: БХВ - Киев, 2001.

2. Вайк А. **JavaScript в примерах.** - Киев: ДиаСофт, 2000.

3. Интегрирование чехарда-[Лекция **«Реализация физики на основе интегрирования Верлета**»](http://www.gamedev.ru/community/gamedev_lecture/articles/?id=26)

4. **Ландау Л.Д., Лифшиц Е.М.,** Механика

5. Овалы Кассини и лемниската- **Курс высшей математики, Т.1**