Санкт-Петербургский политехнический университет Петра Великого

Институт прикладной математики и механики

**Кафедра «Теоретическая механика»**

**КУРСОВОЙ ПРОЕКТ**

**Решение и визуалтзация задачи двух тел.**

по дисциплине «Математическое моделирование»

Выполнил

студент гр.13632/1 Шелых Е.И.

«\_\_\_» \_\_\_\_\_\_\_\_\_\_ 2019 г.

Санкт-Петербург

2019

**СОДЕРЖАНИЕ**

|  |  |
| --- | --- |
| Введение. . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . Постановка задачи. . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .  | 33 |
| 1. Реализация. . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .  | 4 |
| 1.1. Описание кода HTML и JavaScript. . . . . . . . . . . . . . . . . . . . . . . . . . . . Результаты. . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .  | 713 |
| Вывод и заключение. . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . | 14 |

**Введение**

Данная работа посвящена визуализации решения одной из задач небесной механики, состоящей в определении относительного движения двух тел, взаимодействующих по закону Ньютона. Актуальность работы связана с необходимостью современного общества осваивать новейшие технологии, позволяющие исследовать космос. Данный проект может познакомить студентов, только начинающих свое обучение с наглядным примером решения данной задачи. Цель курсовой работы заключается в изучении основных принципов исследования взаимодействия двух небесных тел и визуализации данного взаимодействия. Объект исследования – процесс использования языков программирования, а также математическая теория в области небесной механики. Предмет исследования – программные инструменты языков JavaScript и HTML.

**Постановка задачи**

Реализовать и визуализировать задачу двух тел:

1. Решить задачу в общем случае
2. Визуализировать решение с заданными входными параметрами

**Реализация**

Перечень функций, которые были использованы для написания программы:

* На языке HTML размечаем головную страницу с контейнерами, содержащими изображение, краткое описание
* В основе визуализации каждого метода используется элемент <canvas>
* Код программы на языке JavaScript разбит на несколько основных поочередно вызываемых функций: ввод, подсчёты, рисование

**Описание кода HTML и JavaScript**

Рассмотрим реализацию метода трапеций.

Создадим необходимые для работы глобальные переменные:

*var gravitationalConstant = 6.67\*Math.pow(10,-11);//гравитационная постоянная*

*var point\_x1 = 1200/6\*3;*

*var point\_y1 = 300;*

*var point\_x2 = 1200/6\*4;*

*var point\_y2 = 300;*

*var sP = 1;*

*var k = 0;*

*var flag = 0;*

При открытии страницы и загрузке холста, отрисовываются первоначальные планеты с отношением масс, заданным по умолчанию.

*function firstPaint(){*

 *var canvas = document.getElementById('pole');*

 *if (canvas.getContext){*

 *var ctx = canvas.getContext('2d');*

 *}*

 *var img1 = new Image();*

 *img1.src = "planet1.jpg";*

 *img1.onload = function(){*

 *var img = new Image();*

 *img.src = "space.jpg";*

 *img.onload = function(){*

 *var pattern = ctx.createPattern(img,"repeat");*

 *ctx.fillStyle = pattern;*

 *ctx.beginPath();*

 *ctx.clearRect(0,0,1500,800);*

 *ctx.moveTo(1,1);*

 *ctx.fillRect(0, 0, 1500, 800);*

 *ctx.closePath();*

 *ctx.beginPath();*

 *ctx.fillStyle = "#F6F231";*

 *ctx.arc(point\_x1,point\_y1,10,0,Math.PI\*2);*

 *ctx.fill();*

 *var img2 = new Image();*

 *img2.src = "planet2.jpg";*

 *img2.onload = function(){*

 *var pattern2 = ctx.createPattern(img2,"repeat");*

 *ctx.beginPath();*

 *ctx.fillStyle = pattern2;*

 *ctx.moveTo(point\_x2,point\_y2);*

 *ctx.arc(point\_x2,point\_y2,10,0,Math.PI\*2);*

 *ctx.fill();*

 *}*

 *}*

*}*

При запуске программы функция *makePlanets* при помощи *AddPlanet* создает планеты и рассчитывает вспомогательные параметры, необходимые для расчета траекторий движения планет.

*}*

*function AddPlanet(x,y,mass,speed){*

 *this.x = x;*

 *this.y = y;*

 *this.mass = mass;*

 *this.M = this.mass\*gravitationalConstant;*

 *this.speed = speed;*

*}*

*function makePlanets(){*

 *point\_x1 = 1200/6\*3;*

 *point\_y1 = 300;*

 *point\_x2 = 1200/6\*4;*

 *point\_y2 = 300;*

 *var mass =document.getElementById('firstMass').value\*Math.pow(10,20);*

 *var speed = 1;*

 *var firstPlanet = new AddPlanet(point\_x1,point\_y1,mass,speed);*

 *mass = document.getElementById('secondMass').value\*Math.pow(10,20);*

 *speed = 1;*

 *var secondPlanet = new AddPlanet(point\_x1,point\_y1,mass,speed);*

 *firstPlanet.r = Math.pow(10,7)\*(point\_x2 - point\_x1 - (point\_x2-point\_x1)\*firstPlanet.mass/(firstPlanet.mass + secondPlanet.mass));//радиус вектор 1 тела относительно центра масс системы*

 *secondPlanet.r = Math.pow(10,7)\*((point\_x2-point\_x1)\*firstPlanet.mass/(firstPlanet.mass + secondPlanet.mass));//радиус вектор 1 тела относительно центра масс системы*

 *firstPlanet.massCenterX = point\_x2 - (point\_x2-point\_x1)\*firstPlanet.mass/(firstPlanet.mass + secondPlanet.mass);//рассчет x координыты центра масс*

 *firstPlanet.c = firstPlanet.speed\*firstPlanet.r;//константы закона площадей*

 *secondPlanet.c = secondPlanet.speed\*secondPlanet.r;//константы закона площадей*

 *firstPlanet.f = -firstPlanet.M + firstPlanet.speed\*firstPlanet.c;//векторы Лапласса*

 *secondPlanet.f = -secondPlanet.M + secondPlanet.speed\*secondPlanet.c;//векторы Лапласса*

 *firstPlanet.p = Math.pow(firstPlanet.c, 2)/firstPlanet.M/Math.pow(10,6); //фокальный параметр*

 *secondPlanet.p = Math.pow(secondPlanet.c, 2)/secondPlanet.M/Math.pow(10,6); //фокальный параметр*

 *firstPlanet.e = firstPlanet.f/firstPlanet.M;//эксцентриситет*

 *secondPlanet.e = secondPlanet.f/secondPlanet.M;//эксцентриситет*

 *console.log('e1=',firstPlanet.e,'e2=',secondPlanet.e);*

 *drawPlanets(firstPlanet,secondPlanet);*

*}*

После получения даннных о траекториях, вызывается функция *drawPlanets,*которая определяет концы болшой и малой полуосей траектории, а затем визуализирует движение планет.

*function drawPlanets(firstPlanet,secondPlanet){*

 *var canvas = document.getElementById('pole');*

 *if (canvas.getContext){*

 *var ctx = canvas.getContext('2d');*

 *}*

 *var img = new Image();*

 *img.src = "space.jpg";*

 *var img2 = new Image();*

 *img2.src = "planet2.jpg";*

 *var pattern2 = ctx.createPattern(img2,"repeat");*

 *img.onload = function() {*

 *var pattern = ctx.createPattern(img,"repeat");*

 *ctx.fillStyle = pattern;*

 *var angle = Math.PI;*

 *var angle2 = 0;*

 *firstPlanet.a = firstPlanet.p/(1 - firstPlanet.e)\*6;*

 *firstPlanet.b = firstPlanet.p/Math.pow((1 - firstPlanet.e),0.5)\*3;*

 *firstPlanet.c = Math.abs(firstPlanet.e\*firstPlanet.a);//фокольное расстояние*

 *secondPlanet.a = secondPlanet.p/(1 - secondPlanet.e)\*6;*

 *secondPlanet.b = secondPlanet.p/Math.pow((1 - secondPlanet.e),0.5)\*3;*

 *secondPlanet.c = Math.abs(secondPlanet.e\*secondPlanet.a);//фокольное расстояние*

 *firstPlanet.leftScope = firstPlanet.a\*Math.cos(180);*

 *secondPlanet.scope = secondPlanet.a\*Math.cos(0);*

 *point\_x1 = firstPlanet.massCenterX + firstPlanet.c;*

 *point\_x2 = firstPlanet.massCenterX - secondPlanet.c;*

*var x = 10, y = 390;*

 *var timerId = setInterval( function(){*

 *if (k==2){*

 *var a =1;*

 *}*

 *else{*

 *angle = angle + Math.PI/360;*

 *if (angle >= 2\*Math.PI){*

 *angle = 0;*

 *}*

 *angle2 = angle2 + Math.PI/360;*

 *if (angle2 >= 2\*Math.PI){*

 *angle2 = 0;*

 *}*

 *firstPlanet.x = point\_x1 + firstPlanet.a\*Math.cos(angle);*

 *firstPlanet.y = point\_y1 + firstPlanet.b\*Math.sin(angle);*

 *secondPlanet.x = point\_x2 + secondPlanet.a\*Math.cos(angle2);*

 *secondPlanet.y = point\_y2 + secondPlanet.b\*Math.sin(angle2);*

 *ctx.fillStyle = pattern;*

 *ctx.beginPath();*

 *ctx.clearRect(0,0,1500,800);*

 *ctx.moveTo(1,1);*

 *ctx.fillRect(0, 0, 1500, 800);*

 *ctx.closePath();*

 *ctx.beginPath();*

 *ctx.moveTo(firstPlanet.x,firstPlanet.y);*

 *ctx.fillStyle = "#F6F231";*

 *ctx.arc(firstPlanet.x,firstPlanet.y,10,0,2\*Math.PI);*

 *ctx.fill();*

 *ctx.beginPath();*

 *ctx.moveTo(firstPlanet.x,firstPlanet.y);*

 *ctx.fillStyle = pattern2;*

 *ctx.arc(secondPlanet.x ,secondPlanet.y ,10,0,2\*Math.PI);*

 *ctx.fill();*

 *ctx.beginPath();*

 *ctx.moveTo(firstPlanet.massCenterX,point\_y1);*

 *ctx.fillStyle = "#1D067F";*

 *ctx.arc(firstPlanet.massCenterX,point\_y1 ,4,0,2\*Math.PI);*

 *ctx.fill();*

 *y = graph1(firstPlanet,secondPlanet,x,y);*

 *x = x + 1;*

 *if (k==1){*

 *clearTimeout(timerId);*

 *k = 0;*

 *}*

 *}*

 *},sP);*

 *};*

*}*

*function breakk(){*

 *k = 1;*

*}*

*function pausee(){*

 *if (k==0){*

 *k = 2;*

 *}*

 *else{*

 *k = 0;*

 *}*

*}*

Функция *graph1* принимает информацию о положении обеих планет и строит график зависимости расстояния от времени, а так же показывает величину расстояния в данный момент отрисовки.

*function graph1(firstPlanet,secondPlanet,x,y){*

 *var canvas = document.getElementById('pole2');*

 *if (canvas.getContext){*

 *var canvas = canvas.getContext('2d');*

 *}*

*if ( flag == 0){*

 *canvas.beginPath();*

 *canvas.strokeStyle = "#000000";*

 *canvas.moveTo(10,395);*

 *canvas.lineTo(10,10);*

 *canvas.lineTo(5,15);*

 *canvas.moveTo(10,10);*

 *canvas.lineTo(15,15);*

 *canvas.moveTo(5,390);*

 *canvas.lineTo(690,390);*

 *canvas.lineTo(685,395);*

 *canvas.moveTo(690,390);*

 *canvas.lineTo(685,385);*

 *canvas.moveTo(550,20);*

 *canvas.lineTo(550,100);*

 *canvas.lineTo(690,100);*

 *canvas.lineTo(690,20);*

 *canvas.lineTo(550,20);*

 *canvas.strokeStyle = "#000000";*

 *canvas.font = "20px serif";*

 *canvas.fillText('R(t)',20,20);*

 *canvas.stroke();*

 *flag = 1;*

*}*

 *return lengAndDraw(firstPlanet,secondPlanet,canvas,x,y)*

*}*

*function lengAndDraw(firstPlanet,secondPlanet,canvas,x,y){*

 *var l = Math.sqrt( Math.pow(firstPlanet.x-secondPlanet.x,2) + Math.pow(firstPlanet.y - secondPlanet.y,2))/3.5;*

*// console.log(l);*

 *canvas.beginPath();*

 *canvas.clearRect(550,20,140,80);*

 *canvas.strokeStyle = "#000000";*

 *canvas.moveTo(x,y);*

 *canvas.lineTo(x + 1, 390 - l);*

 *canvas.font = "20px serif";*

 *canvas.fillText('r = ' + Math.floor(l\*3.5) + '\*10^7', 570, 60);*

 *canvas.stroke();*

 *return (390 - l);*

*}*

**Результаты**

При помощи скриптового языка программирования был реализован инструмент для визуализации движения двух планет.

Ниже представлен пример результата работы программы.





**Вывод**

В ходе работы была выполнена поставленная задача. Использование множества различных функций и методов дало большой опыт программирования.

**Заключение**

В программу заложена возможность дополнения её новыми методами и более глубокой визуализацией.

**Список используемой литературы**

1.Кочетков А.В., Федотов П.В. Метод решения задачи двух тел // Интернет-журнал «НАУКОВЕДЕНИЕ» Том 7, №6 (2015) http://naukovedenie.ru/PDF/70TVN615.pdf (доступ свободный). Загл. с экрана. Яз. рус., англ. DOI: 10.15862/70TVN615

**Форма задания на выполнение
курсового проекта**

Санкт-Петербургский политехнический университет Петра Великого

**ЗАДАНИЕ**

**НА ВЫПолнение курсового проекта**

студенту группы 13632/1 Шелых Е.И.

 ***1. Тема проекта:*** Визуализация численных методов интегрирования

 ***2. Срок сдачи студентом законченного проекта***

 ***3. Исходные данные к проекту***: курс лекций по математическому моделированию за первый и второй семестры

 ***4. Содержание пояснительной записки*** (перечень подлежащих разработке вопросов): введение, основная часть (раскрывается структура основной части), заключение, список использованных источников, приложения.

Примерный объём пояснительной записки 12 страниц печатного текста.

***5. Перечень графического материала*** (с указанием обязательных чертежей и плакатов): отсутствует

***6. Консультанты***

 ***7. Дата получения задания***: «\_\_\_».\_\_\_\_\_\_\_\_\_\_\_\_ 20\_\_ г.

Руководитель \_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

 *(подпись) (инициалы, фамилия)*

Задание принял к исполнению \_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

 *(подпись студента) (инициалы, фамилия)*

\_\_\_\_\_\_\_\_\_\_\_\_*(дата)*