**Национальный исследовательский университет**

**«МЭИ»**

**Институт радиотехники и электроники**

**Кафедра радиотехнических систем**

**Аппаратура потребителей спутниковых навигационных систем**

Курсовой проект

ФИО студента: Ткаченко Р.О.

Группа: Эр-15-15

Вариант №: 14

Дата:\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Подпись: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

ФИО преподавателя: Корогодин И.В.

Оценка: \_\_\_\_\_\_\_\_\_\_\_

Содержание

[Введение 3](#_Toc42001930)

[1 Использование сторонних средств 3](#_Toc42001931)

[1.1 Эфемериды всех полученных спутников по данным RTKLIB 5](#_Toc42001932)

[1.2 Эфемериды собственного спутника в gnav-файле RINEX 5](#_Toc42001933)

[1.3 График угла места собственного спутника от времени по данным Trimble GNSS Planning Online на заданный интервал времени 6](#_Toc42001934)

[Заключение 9](#_Toc42001935)

[2 Моделирование 10](#_Toc42001936)

[2.1 Результаты моделирования положения спутника ГЛОНАСС 10](#_Toc42001937)

[2.2 Построение SkyView 12](#_Toc42001938)

[2.3 Заключение по результатам моделирования 13](#_Toc42001939)

[3 Реализация 14](#_Toc42001940)

[3.1 Заключение по результатам реализации 14](#_Toc42001941)

[4 Заключение 15](#_Toc42001942)

[5 Список использованных источников 15](#_Toc42001943)

[6 Приложения 16](#_Toc42001944)

[6.1 Листинг кода этапа моделирования 16](#_Toc42001945)

[6.2 Листинг кода этапа реализации 19](#_Toc42001946)

# Введение

Техническая цель - добавление в программное обеспечение приемника функции расчета положения спутника ГЛОНАСС на заданное время по данным его эфемерид.

Для достижения цели выполняется ряд задач:

* обработка данных от приемника ГНСС в RTKLIB для проверки входных данных и формирования проверочных значений;
* обработка данных и моделирование в Matlab/Python для эскизного проектирования модуля;
* реализация программного модуля на С/С++, включая юнит-тестирование в Check.

Требования:

* отсутствие утечек памяти;
* малое время выполнения;
* низкий расход памяти;
* корректное выполнение при аномальных входных данных.

Среда взаимодействия: Взаимодействие осуществляется через github.

Курсовой проект разбит на три этапа, отличающиеся осваиваемыми инструментами.

Конечная цель проекта - получить библиотечные функции на Си++, позволяющие рассчитывать положение спутника ГЛОНАСС по эфемеридам.

# Использование сторонних средств

На крыше корпуса Е МЭИ установлена трехдиапазонная антенна [Harxon HX-CSX601A](https://en.harxon.com/u_file/product/18_08_08/Harxon%20HX-CSX601A%20Brochure.pdf). Она через 50-метровый кабель, [сплиттер, bias-tee и усилитель](https://www.srns.ru/wiki/Blog:Mikhaylova/25.06.2018_%D0%90%D0%BD%D1%82%D0%B5%D0%BD%D0%BD%D1%8B%D0%B9_%D0%BF%D0%BE%D1%81%D1%82_%D0%BD%D0%B0_8_%D0%BF%D1%80%D0%B8%D0%B5%D0%BC%D0%BD%D0%B8%D0%BA%D0%BE%D0%B2" \o "Blog:Mikhaylova/25.06.2018 Антенный пост на 8 приемников) подключена к трем навигационным приемникам:

* Javad Lexon LGDD,
* SwiftNavigation Piksi Multi,
* Clonicus разработки ЛНС МЭИ.

Приемники осуществляют первичную обработку сигналов, выдавая по интерфейсам соответствующие потоки данных - наблюдения псевдодальностей и эфемериды спутников. В этом году вы будете обрабатывать данные от приемника Clonicus, представленные в бинарном виде в формате NVS BINR.

На этом этапе мы получаем входные данные для этой функции - сами эфемериды.

Для этого воспользуемся пакетом [RTKLIB](https://github.com/Korogodin/RTKLIB_bin). RTKLIB - это программный пакет с открытым исходным кодом для стандартного и точного позиционирования с ГНСС. RTKLIB состоит из переносимой библиотеки программ и нескольких точек доступа (приложение программы) с использованием библиотеки. Особенности RTKLIB:

Он поддерживает стандартные и точные алгоритмы позиционирования с GPS, ГЛОНАСС, Galileo, QZSS, BeiDou и SBAS;

Он поддерживает различные режимы позиционирования с GNSS как в режиме реального времени, так и в режиме пост-обработки.

Он поддерживает множество стандартных форматов и протоколов для GNSS (RINEX /OBS / NAV / GNAV / HNAV / LNAV / QNAV и т.д.).

Скачиваем [RTKLIB](https://github.com/Korogodin/RTKLIB_bin) с официального сайта.

Первое, что нужно сделать - скачать из указанного репозитория файл в формате .bin, который содержит данные с приемника.

Из указанного репозитория скачиваем папку «RTKLIB\_bin\_master». В открывшейся папке находим приложение «RTKNAVI».

Далее бинарный файл эфемерид необходимо вывести в табличном виде с помощью программы RTKNAVI. В качестве входных данных (Input Streams) программы RTKNAVI подставлялись бинарные данные (соответствующие таблицы представлены на Рисунок 1 и Рисунок 2).

# Эфемериды всех полученных спутников по данным RTKLIB

Получим эфемериды спутника по данным RTKNAVI из RTKLIB. Программа RTKNAVI дает возможность таблицу текущих и предыдущих эфемерид.

|  |
| --- |
| Снимок1 |
| Рисунок 1 - Текущие эфемериды спутников по данным RTKNAVI |

# Эфемериды собственного спутника в gnav-файле RINEX

Эфемериды собственного спутника №14 (выделен серым цветом) получены из .nav файла.

|  |
| --- |
| Снимок2 |
| Рисунок 2 - Текущие эфемериды спутников в gnav-файле RINEX |

# График угла места собственного спутника от времени по данным Trimble GNSS Planning Online на заданный интервал времени

Для решения поставленной задачи воспользуемся интернет ресурсом Trimble (<https://www.gnssplanning.com>) строим график угла места для выбранного спутника на заданный период времени.

|  |
| --- |
| общий |
| Рисунок 3 - Общие установки в Trimble GNSS Planning Online |

Из доступных спутников выбрали необходимый (номер 14), при этом отключая все ненужные.

|  |
| --- |
| спутники |
| Рисунок 4 - Выбор спутника согласно варианту курсового проекта (спутник №14) |

|  |
| --- |
| график угла места |
| Рисунок 5 - Угол места спутника №14 на заданный интервал |
| Из рисунка 5 видно, что наилучше всего спутник был виден в районе 3:10. |
| Снимок3 |
| Рисунок 6 - Sky Plot для 14 космического аппарата системы ГЛОНАСС |

|  |
| --- |
| спутник2 |
| Рисунок 7 - Sky Plot для 14 космического аппарата системы ГЛОНАСС |

# Заключение

В ходе выполнения первого этапа были получены следующие результаты:

* обработаны данные от приёмника ГНСС с помощью RTKNAVI из пакета RTKLIB и преобразованы в таблицу эфемерид спутников ГЛОНАСС, в том числе и собственного спутника (№14), данные которого необходимы для следующих этапов в качестве проверочных значений;
* эфемериды собственного спутника в конвертированном gnav-файле RINEX с помощью RTKCONV из пакета RTKLIB;
* график угла места собственного спутника на заданный интервал времени, а также установлено количество появлений спутника в заданный промежуток времени;
* SkyView по данным Trimble GNSS Planning Online на заданный интервал времени.

# 2 Моделирование

На предыдущем этапе получено решение навигационной задачи с помощью программы вторичной обработки измерений – RTKLIB. В процессе работы она рассчитывает положение спутников на соответствующий момент сигнального времени. При этом используются эфемериды - параметры некоторой модели движения спутника. В разных ГНСС эти модели разные, а значит отличается и формат эфемерид, и алгоритмы расчета положения спутника.

Требуется реализовать на языке Matlab или Python функцию расчета положения спутника ГЛОНАСС на заданный момент по шкале времени UTC. В качестве эфемерид использовать данные, полученные на предыдущем этапе.

Для расчета положения спутника ГЛОНАСС по эфемеридным данным системы проводят численное интегрирование дифференциального уравнения.

Таблица 1. Эфемериды спутника ГЛОНАСС №14

|  |  |  |
| --- | --- | --- |
| Параметр | Размерность | Значение |
| T\_Omega | сек | 60318 |
|  | м | 12234925.293 |
|  | - 15420709.9609 |
|  | 16234023.9258 |
|  | м/с | -683.955193 |
|  | 2115.44323 |
|  | 2525.34008 |
|  | м/с2 | 0.000001863 |
|  | 0.000000931 |
|  | -0.000002794 |

# 2.1 Результаты моделирования положения спутника ГЛОНАСС

Алгоритм реализован на языке MATLAB, листинг программы приведен в приложении А.

 Рисунок 8 - Траектория движения спутника ГЛОНАСС №14 в системе координат ПЗ-90

 Рисунок 9 - Траектория движения спутника ГЛОНАСС №14 в системе координат CK ECI

# 2.2 Построение SkyView

На рисунке 10 изображен SkyView на заданный момент времени, полученный с помощью GNSS Planing Online SkyView, на рисунке 11 изрбражен SkyView, построенный в MATLAB:

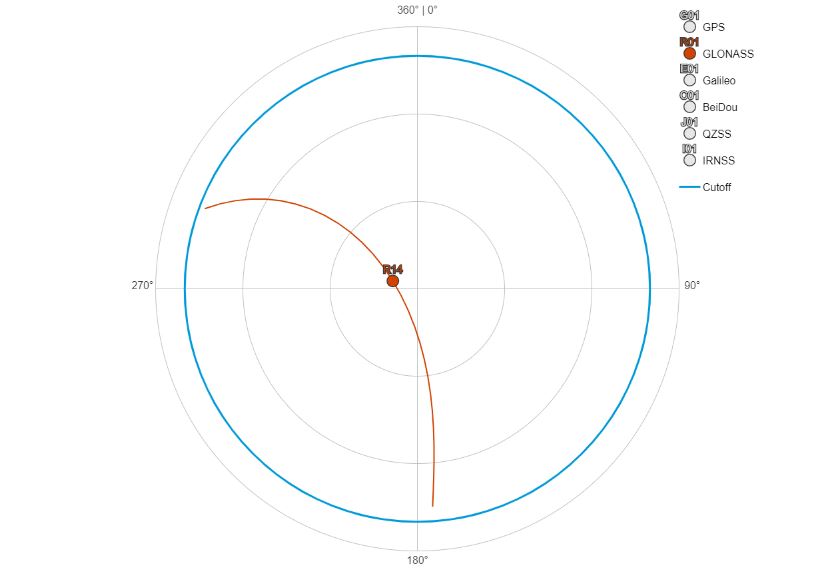
Рисунок 10 - SkyView на заданный момент времени для спутника №14

 Рисунок 11 – Полученная траектория спутника №14

По приведенным выше рисункам можно сказать, что заданный спутник в заданный интервал времени был виден и его сигнал принимался трехдиапазонной антенной [Harxon HX-CSX601A](https://en.harxon.com/u_file/product/18_08_08/Harxon%20HX-CSX601A%20Brochure.pdf), расположенной на крыше корпуса Е МЭИ. Видно, что полученный SkyView совпадает с построенным в MATLAB по эфемеридам НКА №14.

# 2.3 Заключение по результатам моделирования

На данном этапе была реализована на языке Matlab функция расчета положения спутника ГЛОНАСС №14. В качестве эфемерид использовались данные, полученные на предыдущем этапе. Использовались одни и те же эфемериды на весь рассматриваемый интервал.

В результате были получены графики траекторий движения спутника ГЛОНАСС №5 в системах координат: ПЗ-90 и ECI, а так же SkyView с графиком угла места для точки, в которой находился приемник.

# 3 Реализация

Требуется разработать на языке С/С++ функцию расчета положения спутника ГЛОНАСС на заданное время по шкале UTC, минимизируя время её исполнения и количество затрачиваемой оперативной памяти. Вызов функции не должен приводить к выбросу исключений или утечкам памяти при любом наборе входных данных. Допускается использовать одни и те же эфемериды на весь рассматриваемый интервал.

Функция расчета положения спутника в Matlab относительно проста, т.к. доступны библиотеки линейной алгебры и решения уравнений. В рассматриваемом примере потребуется, как минимум, выполнить свою реализацию решения дифура методом Рунге-Кутты.

Все указанные функции и тесты приведены в приложении.

# 3.1 Заключение по результатам реализации

На данном этапе была реализована функция расчета положения спутника ГЛОНАСС №14 на заданное время по шкале UTC на языке С/С++.

Из рисунка 12 видно, что полученная максимальная разница координат составляет 220 км. Эта разница может быть вызвана различиями реализаций метода Рунге-Кутта в Matlab и в написанной программе на С++. В Matlab функция решения системы дифференциальных уравнений методом Рунге-Кутта совмещает в себе методы 4 и 5 порядков, так что можно предположить, что результаты Matlab имеют более высокую точность.

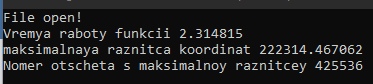


Рисунок 12 – Результат расчетов

Измеряли время исполнения функции измерением времени выполнения функции glns\_coord (оценка приближенная). Время выполнения функции расчета координат оценивается простым измерением времени на выполнение данной функции. Результат измерений времени в милисекундах показан на рисунке 13.

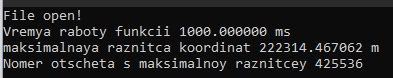


Рисунок 13 – Результат измерения времени

# 4 Заключение

При выполнение курсового проекта был приобретен опыт работы с использованием интерфейсных контрольных документов для СРНС ГЛОНАСС, и опыт извлечения эфемерид спутников из навигационного сообщения, моделирования алгоритма расчета положения спутника с использованием численного интегрирования системы дифференциальных уравнений. Так же получен первый бесценный опыт в использовании интернет-ресурса https://github.com/.

# 5 Список использованных источников

1. ИКД ГЛОНАСС. Общее описание системы с кодовым разделением.
2. GLONASS Satellite Coordinates Computation – Navipedia.
3. Материалы лекций по курсам: «Математическое моделирование РТУ и С», и «Аппаратура потребителей спутниковых радионавигационных систем».

# 6 Приложения

# 6.1 Листинг кода этапа моделирования

Файл KP\_etap2.m

clear all;

close all;

tic;

%Используемые константы

del\_t = 1; % Шаг расчетов

mu = 3.986004418E+14; % константа гравитационного поля Земли

we = 7.292115E-05; % угловая скорость вращения Земли

Rz = 6371000; % радиус Земли

C20 = -1082.62575E-06; % коэффициент при второй зональной гармонике разложения

%геопотенциала в ряд по сферическим функциям

%Эфемериды

NS = 14; % номер спутника

T\_Omega = .495000000000E+05 + 18 + 3\*3600; % время задания эфемерид

X = .122349252930E+08;

Y = -.154207099609E+08;

Z = .162340239258E+08;

Vx = -.683955192566E+03;

Vy = .211544322968E+04;

Vz = .252534008026E+04;

Ax = .186264514923E-05;

Ay = .931322574615E-06;

Az = -.279396772385E-05;

num = fix(12\*3600/del\_t); % Количество отсчетов за 12 часовой интервал расчета

num\_eph = fix((T\_Omega-12\*3600-3\*3600)/del\_t); % номер отсчета с эфемеридными данными

tt = del\_t.\*(1:1:num); %Вектор отсчетов времени

tt = tt + 12\*60\*60+3\*60\*60; %Вектор отсчетов времени

koord\_PZ = zeros(num,3);

koord\_ECI = zeros(num,3);

LL\_potr = [55.756735, 37.703177 170]; % координаты потребителя

t\_G0 = (9\*3600+18\*60+10.5009+3\*3600); %9:18:10.5009; Истинное звездное время на гринвичевскую полночь текущей даты

t\_G = t\_G0 + we\*(tt(num\_eph)- 3\*3600);

% пересчет координат из ПЗ-90 в ECI

Xa = X\*cos(t\_G) - Y\*sin(t\_G);

Ya = X\*sin(t\_G) + Y\*cos(t\_G);

Za = Z;

Vxa = Vx\*cos(t\_G) - Vy\*sin(t\_G) - we\*Ya;

Vya = Vx\*sin(t\_G) + Vy\*cos(t\_G) + we\*Xa;

Vza = Vz;

Yn = [Xa Ya Za Vxa Vya Vza];

% Интегрирование методом Рунге-Кутты

[t, Yn1] = ode45('proizv', T\_Omega:-del\_t:tt(1), Yn);

koord\_ECI(1:num\_eph,:) = Yn1(end:-1:1,1:3);

[t, Yn1] = ode45('proizv', T\_Omega:del\_t:tt(num), Yn);

koord\_ECI(num\_eph:end,:) = Yn1(1:end,1:3);

% Пересчет полученных координат из ECI в ПЗ-90

for i = 1:num

t\_G = t\_G0 + we\*(tt(i)- 3\*3600);

koord\_PZ(i,1) = koord\_ECI(i,1)\*cos(t\_G) + koord\_ECI(i,2)\*sin(t\_G);

koord\_PZ(i,2) = -koord\_ECI(i,1)\*sin(t\_G) + koord\_ECI(i,2)\*cos(t\_G);

koord\_PZ(i,3) = koord\_ECI(i,3);

end

% Пересчет координат из ПЗ-90 в WGS84 (для получения SkyView)

ppb = 1e-9;

mas = 1e-3/206264.8; % [рад]

M\_WGS84 = [-3\*ppb -353\*mas -4\*mas;

353\*mas -3\*ppb 19\*mas;

4\*mas -19\*mas -3\*ppb];

koord\_WGS84 = koord\_PZ.'; % Переход к вектору-столбцу

for i = 1:length(koord\_WGS84(1,:))

koord\_WGS84(:,i) = koord\_WGS84(:,i) + M\_WGS84 \* koord\_WGS84(:,i) + [0.07; -0; -0.77];

end

koord\_WGS84 = koord\_WGS84.'; % Переход к вектору-строке

% Пересчет координат из WGS84 в SkyView

X = zeros(num,1);

Y = zeros(num,1);

Z = zeros(num,1);

r = zeros(num,1);

teta = zeros(num,1);

phi = zeros(num,1);

for i = 1:length(koord\_WGS84(:,1))

[X(i), Y(i), Z(i)] = ecef2enu(koord\_WGS84(i,1),koord\_WGS84(i,2),koord\_WGS84(i,3),LL\_potr(1),LL\_potr(2),LL\_potr(3),wgs84Ellipsoid);

if Z(i) > 0

r(i) = sqrt(X(i)^2 + Y(i)^2 + Z(i)^2);

teta(i) = acos(Z(i)/r(i));

if X(i) > 0

phi(i) = -atan(Y(i)/X(i))+pi/2;

elseif (X(i)<0)&&(Y(i)>0)

phi(i) = -atan(Y(i)/X(i))+3\*pi/2;

elseif (X(i)<0)&&(Y(i)<0)

phi(i) = -atan(Y(i)/X(i))-pi/2;

end

else

teta(i) = NaN;

r(i) = NaN;

phi(i) = NaN;

end

end

[Xsf, Ysf, Zsf] = sphere(25);

alfa1 = pi/180.\*(1:359)';

beta1 = 85.\*ones(359,1);

plot3(koord\_PZ(:,1), koord\_PZ(:,2), koord\_PZ(:,3));

hold on

grid on

title('Положение спутника в СК ПЗ-90');

xlabel('OX, км');

ylabel('OY, км');

zlabel('OZ, км');

axis('square');

axis('equal');

surf(Xsf.\*Rz, Ysf\*Rz, Zsf\*Rz);

hold off

figure;

plot3(koord\_ECI(:,1), koord\_ECI(:,2), koord\_ECI(:,3));

hold on

grid on

title('Положение спутника в СК ECI');

xlabel('OX, км');

ylabel('OY, км');

zlabel('OZ, км');

axis('square');

axis('equal');

surf(Xsf.\*Rz, Ysf\*Rz, Zsf\*Rz);

hold off

figure;

ax = polaraxes;

hold on

polarplot(ax,phi,teta\*180/pi,'r')

polarplot(ax,alfa1,beta1,'b')

hold off

ax.ThetaDir = 'clockwise';

ax.ThetaZeroLocation = 'top';

title('Положение спутника SkyView ГЛОНАСС №14')

toc;

Файл proizv.m

function y = proizv(t, yy)

mu = 3.986004418E+14; % константа гравитационного поля Земли

Rz = 6378136; % экваториальный радиус Земли

we = 7.292115E-05; % угловая скорость вращения Земли

C20 = -1082.62575E-06;

t\_G0 = 4.627456713694580e+04;

t\_G = t\_G0 + we\*(t- 3\*3600);

% Ускорения, принимаем постоянными на всем интервале расчета

Ax = .186264514923E-05;

Ay = .931322574615E-06;

Az = -.279396772385E-05;

Jsum\_x = Ax\*cos(t\_G)-Ay\*sin(t\_G);

Jsum\_y = Ax\*sin(t\_G)+Ay\*cos(t\_G);

Jsum\_z = Az;

y = yy(:);

y(1) = yy(4);

y(2) = yy(5);

y(3) = yy(6);

r = sqrt(yy(1)\*yy(1)+yy(2)\*yy(2)+yy(3)\*yy(3));

mu\_ = mu/(r\*r);

x\_ = yy(1)/r;

y\_ = yy(2)/r;

z\_ = yy(3)/r;

ro = Rz/r;

y(4) = -mu\_\*x\_ + 3/2\*C20\*mu\_\*x\_\*ro^2\*(1-5\*z\_^2) + Jsum\_x;

y(5) = -mu\_\*y\_ + 3/2\*C20\*mu\_\*y\_\*ro^2\*(1-5\*z\_^2) + Jsum\_y;

y(6) = -mu\_\*z\_ + 3/2\*C20\*mu\_\*z\_\*ro^2\*(3-5\*z\_^2) + Jsum\_z;

end

# 6.2 Листинг кода этапа реализации

Исполняемый файл main.cpp

#include <iostream>

#include <fstream>

#include <include\libglnsvpos\glnsvpos.h>

#include <include\libglnsvpos\rungekutta.h>

#include <ctime>

using namespace std;

int main()

{

// 14 20 2 10 13 45 0.0 .479789450765E-04 .000000000000E+00 .495000000000E+05

// .122349252930E+05 -.683955192566E+00 .186264514923E-08 .000000000000E+00

// -.154207099609E+05 .211544322968E+01 .931322574615E-09 -.700000000000E+01

// .162340239258E+05 .252534008026E+01 -.279396772385E-08 .000000000000E+00

time\_t start, end;

double delt = 0.1;

int n = (int) 12\*3600/delt;

double \*\*koord = new double \* [n];

double \*koord\_matlab = new double[3];

double max\_del = 0;

int i\_max = 0;

for (int i = 0; i < n; i++)

{

koord[i] = new double [6];

}

ofstream out;

out.open("../res\_cpp.txt"); // путь сохранения результатов расчетов на С++

time(&start);

ifstream in("../res\_mat.txt"); // путь для чтения результатов матлаба

if (!in)

{

cout << "File not open!" << endl;

} else {

cout << "File open!" << endl;

}

time(&start);

glns\_coord(koord);

time(&end);

for (int i = 0; i < n; i++)

{

string koord\_str1 = to\_string(koord[i][0]);

string koord\_str2 = to\_string(koord[i][1]);

string koord\_str3 = to\_string(koord[i][2]);

out << koord\_str1 << "\t" << koord\_str2 << "\t" << koord\_str3 << endl;

in >> koord\_matlab[0] >> koord\_matlab[1] >> koord\_matlab[2];

for (int j = 0; j < 3; j++)

{

if (abs(koord[i][j]-koord\_matlab[j]) > max\_del)

{

max\_del = abs(koord[i][j]-koord\_matlab[j]);

i\_max = i;

}

}

delete [] koord[i];

koord[i] = nullptr;

}

in.close();

out.close();

delete[] koord;

koord = nullptr;

delete[] koord\_matlab;

koord\_matlab = nullptr;

double seconds = difftime(end, start);

string seconds1 = to\_string(seconds\*1000000/n);

cout << "Vremya raboty funkcii " << seconds1 << " s"<< endl;

string max\_del1 = to\_string(max\_del);

cout << "maksimalnaya raznitca koordinat " << max\_del1 << " m" << endl;

string imax = to\_string(i\_max);

cout << "Nomer otscheta s maksimalnoy raznitcey " << imax << endl;

}

Заголовочный файл glnsvpos.h

#ifndef GLNSVPOS\_H

#define GLNSVPOS\_H

void glns\_coord(double \*\*koord\_n);

#endif /\* #ifndef GLNSVPOS\_H \*/

Исполняемый файл glnsvpos.cpp

#include <include\libglnsvpos\glnsvpos.h>

#include <include\libglnsvpos\rungekutta.h>

#include <iostream>

#include <cmath>

#include <ostream>

using namespace std;

void glns\_coord(double \*\*koord\_n)

{

// 14 20 2 10 13 45 0.0 .479789450765E-04 .000000000000E+00 .495000000000E+05

// .122349252930E+05 -.683955192566E+00 .186264514923E-08 .000000000000E+00

// -.154207099609E+05 .211544322968E+01 .931322574615E-09 -.700000000000E+01

// .162340239258E+05 .252534008026E+01 -.279396772385E-08 .000000000000E+00

double t0 = 49500 + 18 + 3\*3600;

double ts = 12\*3600 + 3\*3600;

double te = ts + 12\*3600;

double dt = 1/10.0; //шаг по времени

double we = 7.292115E-05; // угловая скорость вращения Земли

int n = (int) abs((te-ts)/dt);

double \*t = new double [n];

double \*koord\_p = new double [6];

for (int i = 0; i < n; i++)

{

t[i] = ts + i\*dt;

}

int n\_eph = (int) (t0-ts)/dt;

// начальные условия (коорд и скорости)

koord\_p[0] = 0.122349252930E+08;

koord\_p[1] = -0.154207099609E+08;

koord\_p[2] = 0.162340239258E+08;

koord\_p[3] = -0.683955192566E+03;

koord\_p[4] = 0.211544322968E+04;

koord\_p[5] = 0.252534008026E+04;

double t\_G0 = (9\*3600+18\*60+10.5009+3\*3600); //9:18:10.5009; Истинное звездное время на гринвичевскую полночь текущей даты

double t\_G = t\_G0 + we\*(t[n\_eph-1]- 3\*3600);

double cosTg = cos(t\_G);

double sinTg = sin(t\_G);

// пересчет координат из ПЗ-90 в ECI

koord\_n[n\_eph-1][0] = koord\_p[0]\*cosTg - koord\_p[1]\*sinTg;

koord\_n[n\_eph-1][1] = koord\_p[0]\*sinTg + koord\_p[1]\*cosTg;

koord\_n[n\_eph-1][2] = koord\_p[2];

koord\_n[n\_eph-1][3] = koord\_p[3]\*cosTg - koord\_p[4]\*sinTg - we\*koord\_n[n\_eph-1][1];

koord\_n[n\_eph-1][4] = koord\_p[3]\*sinTg + koord\_p[4]\*cosTg + we\*koord\_n[n\_eph-1][0];

koord\_n[n\_eph-1][5] = koord\_p[5];

delete[] koord\_p;

koord\_p = nullptr;

//rungekutta(double t0, double tn, double \*koord0, double \*koord)

for (int i = n\_eph-1; i > 0; i--)

{

rungekutta(t[i], t[i-1], koord\_n[i], koord\_n[i-1]);

}

for (int i = n\_eph-1; i < n-1 ; i++)

{

rungekutta(t[i], t[i+1], koord\_n[i], koord\_n[i+1]);

}

delete[] t;

t = nullptr;

delete[] koord\_p;

koord\_p = nullptr;

}

Заголовочный файл rungekutta.h

#ifndef RUNGEKUTTA\_H

#define RUNGEKUTTA\_H

void rungekutta(double t0, double tn, double \*koord0, double \*koord);

#endif /\* #ifndef RUNGEKUTTA\_H \*/

Исполняемый файл rungekutta.cpp

#include <include\libglnsvpos\rungekutta.h>

#include <cmath>

#include <include\libglnsvpos\diffs.h>

#include <iostream>

#include <fstream>

using namespace std;

void rungekutta(double t0, double tn, double \*koord0, double \*koord){

double dt = tn-t0;

double \*proizv = new double [6];

double \*K1 = new double [6];

double \*K2 = new double [6];

double \*K3 = new double [6];

double \*K4 = new double [6];

double \*prom1 = new double [6];

diffs(t0,koord0,proizv);

for (int j = 0; j<=5; j++){

K1[j] = dt\*proizv[j];

prom1[j] = koord0[j]+K1[j]/2.0;

}

diffs(t0+dt/2.0,prom1,proizv);

for (int j = 0; j<=5; j++){

K2[j] = dt\*proizv[j];

prom1[j] = koord0[j]+K2[j]/2.0;

}

diffs(t0+dt/2.0,prom1,proizv);

for (int j = 0; j<=5; j++){

K3[j] = dt\*proizv[j];

prom1[j] = koord0[j]+K3[j];

}

diffs(tn,prom1,proizv);

for (int j = 0; j<=5; j++){

K4[j] = dt\*proizv[j];

koord[j] = koord0[j] + (K1[j]+2\*K2[j]+2\*K3[j]+K4[j])/6.0;

}

delete[] K1;

K1 = nullptr;

delete[] K2;

K2 = nullptr;

delete[] K3;

K3 = nullptr;

delete[] K4;

K4 = nullptr;

delete[] proizv;

proizv = nullptr;

delete[] prom1;

prom1 = nullptr;

}

Исполняемый файл diff.cpp

#include <cmath>

#include <include\libglnsvpos\diffs.h>

void diffs(double t, double \*koord, double \*proizv)

{

double mu = 3.986004418E+14; // конствнтва гравитационного поля Земли

double Rz = 6378136; // экваториальный радиус Земли

double we = 7.292115E-05; // угловая скорость вращения Земли

double C20 = -1082.62575E-06;

double t\_G0 = (9\*3600+18\*60+10.5009+3\*3600);

double t\_G = t\_G0 + we\*(t- 3\*3600);

// Ускорения, принимаем постоянными на всем интервале расчета

double Ax = .186264514923E-05;

double Ay = .931322574615E-06;

double Az = -.279396772385E-05;

double Jsum\_x = Ax\*cos(t\_G)-Ay\*sin(t\_G);

double Jsum\_y = Ax\*sin(t\_G)+Ay\*cos(t\_G);

double Jsum\_z = Az;

proizv[0] = koord[3];

proizv[1] = koord[4];

proizv[2] = koord[5];

double r = sqrt(koord[0]\*koord[0]+koord[1]\*koord[1]+koord[2]\*koord[2]);

double mu\_ = mu/(r\*r);

double x\_ = koord[0]/r;

double y\_ = koord[1]/r;

double z\_ = koord[2]/r;

double ro = Rz/r;

proizv[3] = -mu\_\*x\_ + 3/2\*C20\*mu\_\*x\_\*ro\*ro\*(1-5\*z\_\*z\_) + Jsum\_x;

proizv[4] = -mu\_\*y\_ + 3/2\*C20\*mu\_\*y\_\*ro\*ro\*(1-5\*z\_\*z\_) + Jsum\_y;

proizv[5] = -mu\_\*z\_ + 3/2\*C20\*mu\_\*z\_\*ro\*ro\*(3-5\*z\_\*z\_) + Jsum\_z;

}

Заголовочный файл diff.h

#ifndef DIFFS\_H

#define DIFFS\_H

void diffs(double t, double \*koord, double \*proizv);

#endif // DIFFS\_H