**Национальный исследовательский университет**

**«МЭИ»**

**Институт радиотехники и электроники**

**Кафедра радиотехнических систем**

Курсовой проект

по дисциплине

Аппаратура потребителей спутниковых радионавигационных систем

«Разработка модуля расчёта координат спутника ГЛОНАСС»

ФИО студента: Юрьев Д.С.

Группа: ЭР-15-15

Вариант №:22

Дата:­ ­

Подпись:­ ­

ФИО преподавателя: Корогодин И.В.

Оценка: ­ ­

**Москва, 2020**

**Содержание**

[**Введение** 3](#_Toc43808170)

[**1 Использование сторонних средств** 3](#_Toc43808171)

[**1.1 Описание процесса использования RTKLIB** 3](#_Toc43808172)

[**1.2 Получение графика угла места и SkyView с помощью Trimble GNSS Planning** 6](#_Toc43808173)

[**2. Моделирование** 8](#_Toc43808174)

[**2.1 Результаты моделирования в Matlab** 9](#_Toc43808175)

[**2.2 Построение SkyView в Matlab** 10](#_Toc43808176)

[**3. Реализация** 11](#_Toc43808177)

[**3.1 Результат расчетов** 11](#_Toc43808178)

[**Заключение** 12](#_Toc43808179)

[**Использованные ресурсы** 13](#_Toc43808180)

# **Введение**

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

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

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.

Курсовой проект разбит на три этапа, отличающиеся осваиваемыми инструментами.

# **1 Использование сторонних средств**

# **1.1 Описание процесса использования RTKLIB**

На крыше корпуса Е МЭИ установлена трехдиапазонная антенна Harxon HX-CSX601A. Она через 50-метровый кабель, сплиттер, bias-tee и усилитель 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.

Необходимо обрабатывать данные от приемника Clonicus, представленные в бинарном виде в формате NVS BINR. Для этого воспользуемся пакетом RTKLIB, в состав которого входит парсер формата NVS BINR и удобные средства отображения данных.

При запуске программы RTKLIB получаем следующее окно (Рисунок 1):



Рисунок 1 – Окно программы RTKLIB v.2.4.2

В окне программы RTKLIB выбираем RTKCONV (Рисунок 2), чтобы конвертировать бинарный файл BINR.bin в текстовый формат NVS BINR.

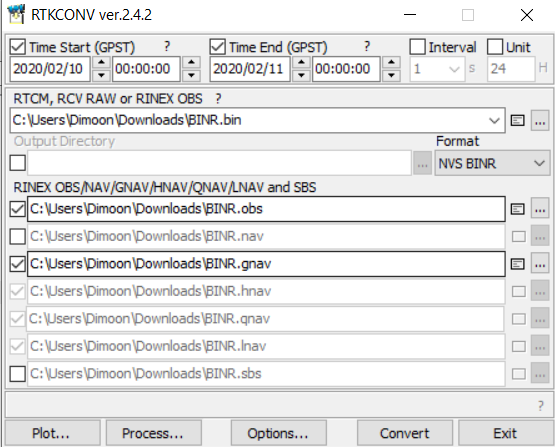


Рисунок 2 – Окно программы RTKCONV ver.2.4.2

В открывшемся окне выбираем Time Start (GPST), Time End (GPST), и ставим время интервала наблюдений с 00:00 10.02.20 до 00:00 11.02.20. В меню «Options» (Рисунок 3) выбираем спутниковую систему ГЛОНАСС и указываем в поле «Excluded Satellite» следующее: R3, R4, R5, R11, R12, R13, R14, R21, R23, тем самым исключая данные спутники из обработки. В первой строке RTKCONV указываем путь на файл бинарного потока .bin, указываем формат NVS BINR, и ставим галочки для конвертации файлов в форматы .obs и .gnav.

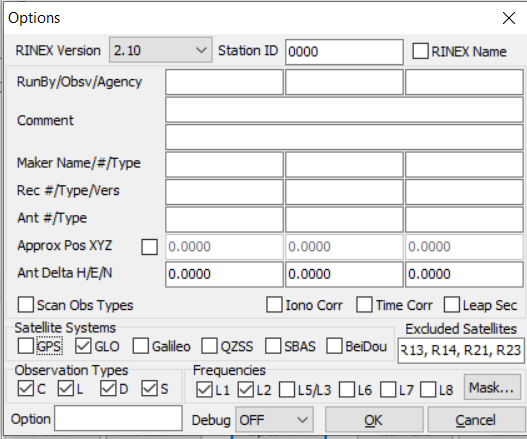


Рисунок 3 – Окно настроек программы RTKCONV ver.2.4.2

Затем нажимаем на кнопку «Convert» и получаем необходимые файлы. Для того, чтобы посмотреть содержимое открываем файл с расширением «gnav» и получаем эфемериды собственного спутника в gnav-файле RINEX (Рисунок 4).

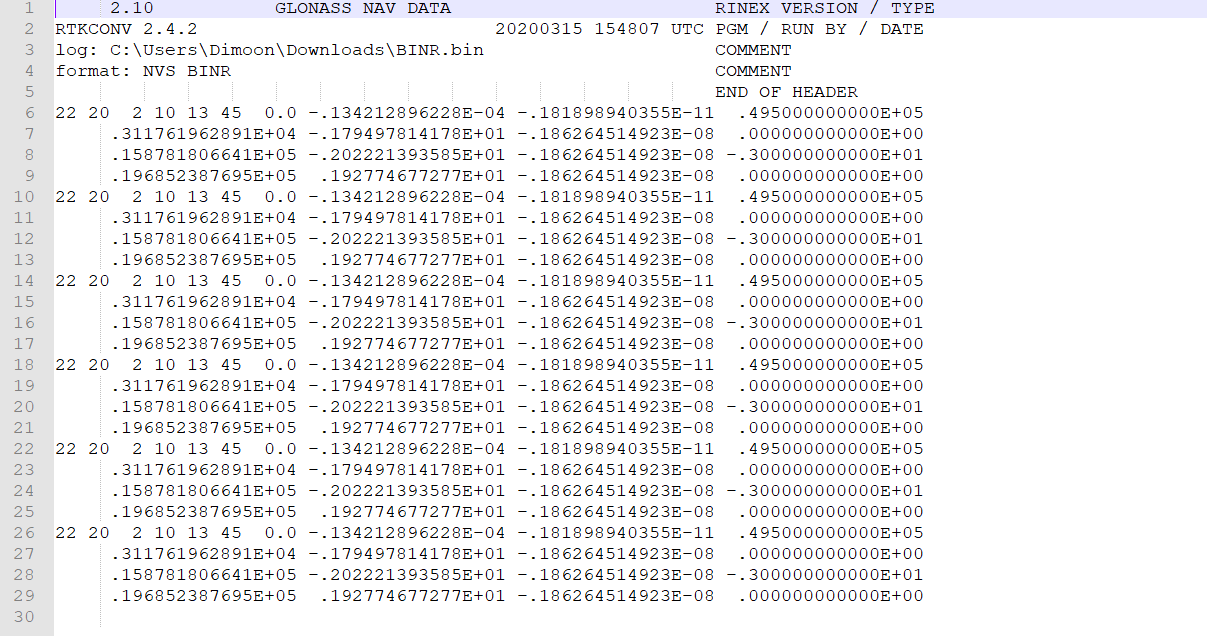


Рисунок 4 – Эфемериды спутника ГЛОНАСС №22 в .gnav файле

Для того, чтобы получить эфемериды собственного спутника по данным RTKNAVI из состава RTKLIB, в окне программы RTKLIB выбираем программу RTKNAVI, которая позволяет вывести таблицу текущих и предыдущих эфемерид (Рисунок 5).

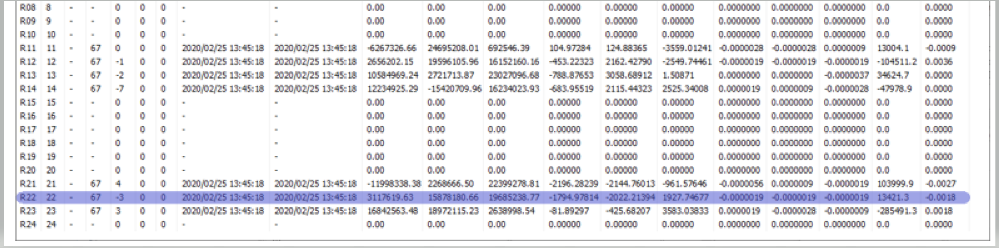


Рисунок 5 – Окно программы RTKNAVI ver.2.4.2

# **1.2 Получение графика угла места и SkyView с помощью Trimble GNSS Planning**

Для построения графика угла места собственного спутника от времени по данным Trimble GNSS Planning Online на заданный интервал времени, во вкладке настроек (Settings) указываем координаты корпуса Е МЭИ и время, когда производились замеры (Рисунок 6). Во вкладке библиотеки спутников (Satellite Library) отключаем отображение всех спутников, кроме заданного (Рисунок 7).

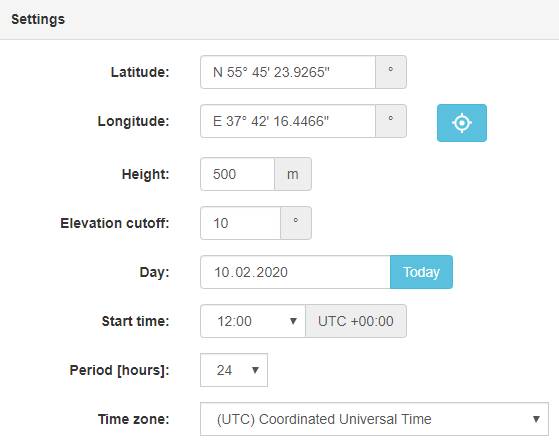


Рисунок 6 – Вкладка настроек (Settings) Trimble GNSS Planning

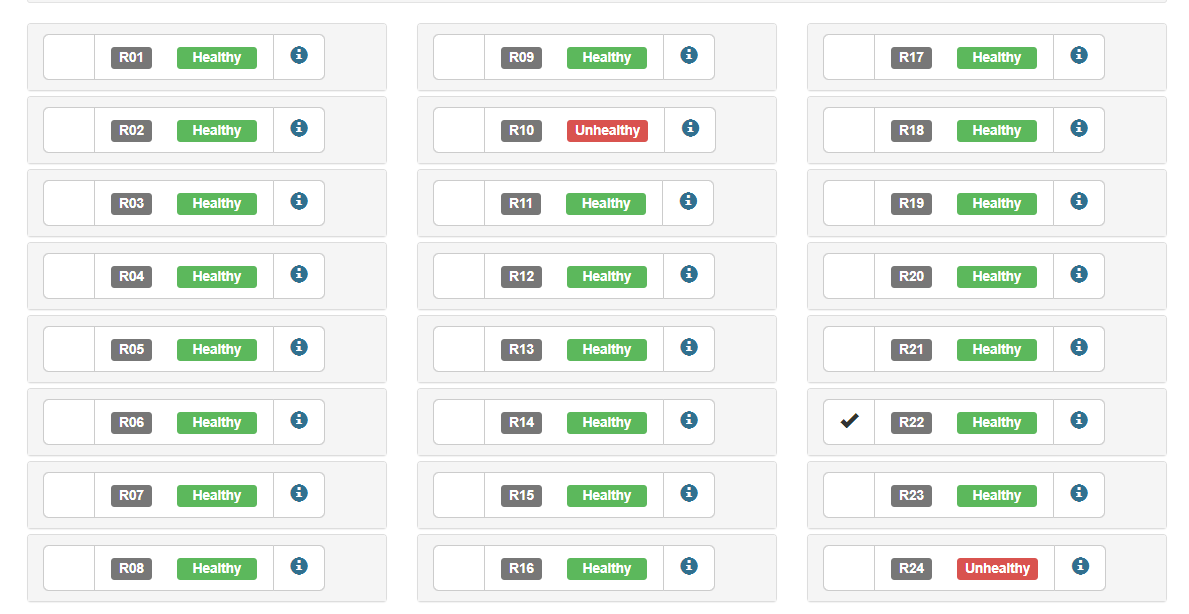


Рисунок 7 – Вкладка библиотеки спутников (Satellite Library) Trimble GNSS Planning

Далее переходим во вкладку «Charts» и получаем график угла места спутника (Рисунок 8). Полученные данные говорят нам о том, что спутник было видно 1 раз с 12:00 до 15:40.

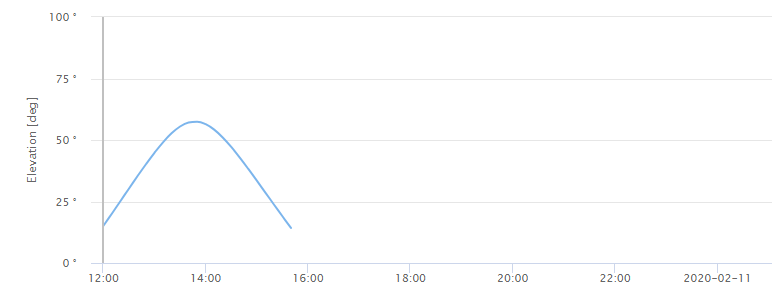


Рисунок 8 – График угла места спутника ГЛОНАСС №22

Перейдя во вкладку «Sky Plot», получаем карту небосвода (SkyView). Траектория движения спутника (Рисунок 9) соответствует его появлению.

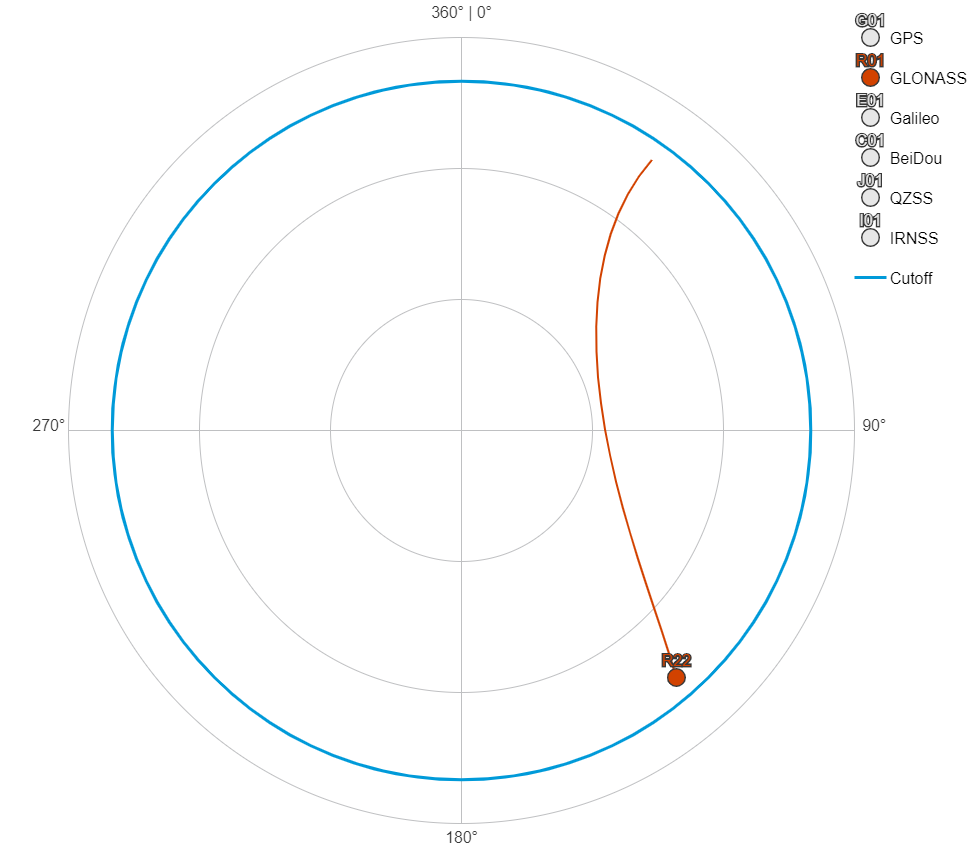


Рисунок 9 – SkyView спутника ГЛОНАСС №22

# **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 используются данные, полученные на предыдущем этапе.

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

Эфемериды, полученные на предыдущем этапе, будем брать из рисунка 5, так как они записаны в удобном варианте.

Необходимо построить трехмерные графики множества положений спутника №22 ГЛОНАСС. Графики в двух вариантах: в СК ECEF ПЗ-90.11 и соответствующей ей инерциальной СК. Положения должны соответствовать временному интервалу с 12:00 10.02.20 до 00:00 11.02.20. Допускается использовать одни и те же эфемериды на весь рассматриваемый интервал. Листинг программы моделирования приведен Приложении 1.

# **2.1 Результаты моделирования в Matlab**



Рисунок 10 – Траектория движения спутника ГЛОНАСС №22 в инерциальной системе координат (синяя линия) и системе координат ПЗ-90 (красная линия)

# **2.2 Построение SkyView в Matlab**

Необходимо построить SkyView за указанный временной интервал и сравнить результат с Trimble GNSS Planning Online, полученный на прошлом этапе.

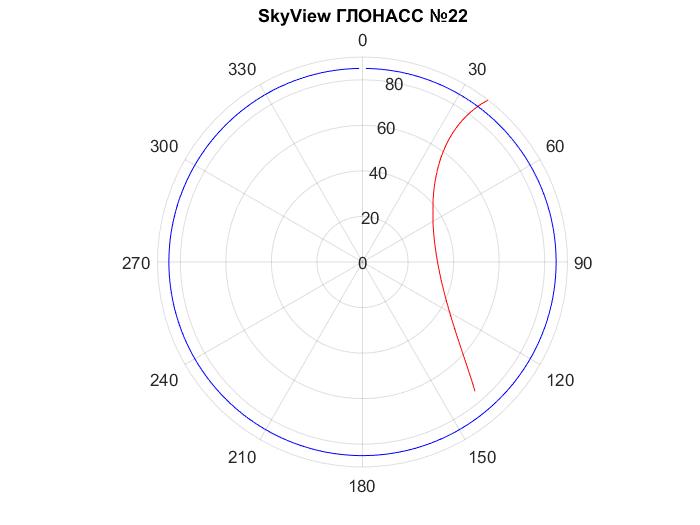


Рисунок 11 – SkyView спутника ГЛОНАСС №22

**3. Реализация**

Требуется разработать на языке С/С++ функцию расчета положения спутника ГЛОНАСС №22 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.

# **3.1 Результат расчетов**

На рисунке 12 приведены результаты расчетов положения спутника ГЛОНАСС №22. Среднее время выполнения составляет 273.15 [мкс]. Максимальная разница координат примерно 13 [м]. Также был получен номер отсчета с максимальной разницей координат. Код программы указан п Приложении 2.

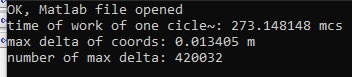


Рисунок 12 – Результат расчетов

Также была проведена проверка на утечки памяти. Результаты проверки представленны на рисунке 14:

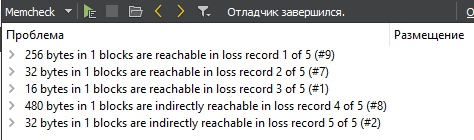


Рисунок 12 – Результаты проверки утечки памяти

# **Заключение**

С помощью RTKLIB были получены эфемериды спутника ГЛОНАСС №22 на интервале наблюдений с 12:00 10.02.20 до 00:00 11.02.20. Эфемериды получены в двух форматах: в виде таблицы текущих и предыдущих эфемерид, в которую входит 11 значений и текстового RINEX файла, в котором эти значения перечислены через пробел. Вторым удобно воспользоваться в модели на следующем этапе, первым - для проверки вручную.

Получены графики SkyPlot с помощью Trimble GNSS Planning Online. За время наблюдения спутник ГЛОНАСС №22 появляется один раз. Пролет наблюдается на интервале времени с 12:00 до 15:40. В момент времени 13:50 угол возвышения максимальный и составляет 57.46º. С помощью этих данных можно будет проверить работу алгоритма расчета положения НС.

В ходе второго этапа с использованием программы MATLAB R2017a, был реализован алгоритм расчета положения спутника ГЛОНАСС №22 на интервале времени с 12:00 10.02.2020 до 00:00 11.02.2020. В качестве исходных данных для алгоритма использовались полученные на первом этапе эфемериды. Получены графики траектории движения спутника ГЛОНАСС №22 в системе координат ПЗ-90 и инерциальной системе координат (рисунок 10), а также изображение в SkyView (рисунок 11). Полученное в ходе моделирования изображение спутника ГЛОНАСС №22 в SkyView, совпадает с изображением, полученным при помощи сервиса Trimble GNSS Planning Online.

В ходе третьего этапа была разработана функция расчета положения спутника ГЛОАНАСС №22. Были проведены все необходимые тесты и получены результаты расчета, а именно: среднее время выполнения, максимальная разница и номер отсчета с максимальной разницей координат. Полученные результаты расчета можно объяснить особенностью реализации метода Рунге-Кутты в среде С++.

В ходе выполнения курсового проекта были получены эфемериды спутника ГЛОНАСС №22 на заданный промежуток времени и его изображение в SkyView для последующей проверки результатов моделирования. Получен навык работы с пакетом RTKLIB, предназначенный для получения необходимых эфемерид. По полученным эфемеридам была реализована модель положения спутника в нескольких системах координат и изображение SkyView. Был улучшен навык по работе с программной средой Matlab. Заключительном этапом было написание в среде С++ функции для получения результатов расчета положения спутника.

# **Использованные ресурсы**

1. www.gnssplanningonline.com

2. Пакет RTKLIB.

3. ИКД ГЛОНАСС.

**ПРИЛОЖЕНИЕ 1**

Листинг программы для этапа 2

clear all;

close all;

dt = 0.1;

T\_eph = 49500 + 18 + 3\*3600;

X = .311761962891E+07;

Y = .158781806641E+08;

Z = .196852387695E+08;

Vx = -.179497814178E+04;

Vy = -.202221393585E+04;

Vz = .192774677277E+04;

Ax = -.186264514923E-05;

Ay = -.186264514923E-05;

Az = -.186264514923E-05;

coordpotr = [55.756657, 37.703288 190]; %55°45'24.0"N 37°42'11.8"E

n = fix(12\*3600/dt);

n\_eph = fix((T\_eph-(12+3)\*3600)/dt);

t12h = (12+3)\*60\*60 + dt.\*(1:1:n);

coordPZ90 = zeros(n,3);

coordECI = zeros(n,6);

w\_e = 7.292115E-05;

t\_G0 = 9\*3600+18\*60+10.5009; %9:18:10.5009;

t\_G = t\_G0 + w\_e\*(T\_eph - 3\*3600);

Yn = zeros(6,1);

Yn(1) = X\*cos(t\_G) - Y\*sin(t\_G);

Yn(2) = X\*sin(t\_G) + Y\*cos(t\_G);

Yn(3) = Z;

Yn(4) = Vx\*cos(t\_G) - Vy\*sin(t\_G) - w\_e\*Yn(2);

Yn(5) = Vx\*sin(t\_G) + Vy\*cos(t\_G) + w\_e\*Yn(1);

Yn(6) = Vz;

coordECI(n\_eph,:) = Yn;

% [t, Yn1] = ode45('difury', T\_eph:-dt:t\_12h(1), Yn);

% coordECI(1:n\_eph,:) = Yn1(end:-1:1,1:3);

%

% [t, Yn1] = ode45('difury', T\_eph:dt:t\_12h(n), Yn);

% coordECI(n\_eph:end,:) = Yn1(1:end,1:3);

for i = n\_eph+1:1:length(t12h)

K1 = difury(t12h(i),coordECI(i-1,:));

K2 = difury(t12h(i)+0.5\*dt, coordECI(i-1,:)+0.5\*dt\*K1');

K3 = difury(t12h(i)+0.5\*dt, coordECI(i-1,:)+0.5\*dt\*K2');

K4 = difury(t12h(i)+dt, coordECI(i-1,:)+dt\*K3');

coordECI(i,:) = coordECI(i-1,:) + dt\*(K1' + 2\*K2'+2\*K3'+K4')/6;

end

for i = n\_eph-1:-1:1

K1 = difury(t12h(i),coordECI(i+1,:));

K2 = difury(t12h(i)-0.5\*dt, coordECI(i+1,:)-0.5\*dt\*K1');

K3 = difury(t12h(i)-0.5\*dt, coordECI(i+1,:)-0.5\*dt\*K2');

K4 = difury(t12h(i)-dt, coordECI(i+1,:)-dt\*K3');

coordECI(i,:) = coordECI(i+1,:) - dt\*(K1'+2\*K2'+2\*K3'+K4')/6;

end

t\_G = t\_G0 + w\_e\*(t12h' - 3\*3600);

coordPZ90(:,1) = coordECI(:,1).\*cos(t\_G) + coordECI(:,2).\*sin(t\_G);

coordPZ90(:,2) = -coordECI(:,1).\*sin(t\_G) + coordECI(:,2).\*cos(t\_G);

coordPZ90(:,3) = coordECI(:,3);

X = zeros(n,1);

Y = zeros(n,1);

Z = zeros(n,1);

r = zeros(n,1);

teta = zeros(n,1);

phi = zeros(n,1);

parfor i = 1:length(coordPZ90(:,1))

[X(i), Y(i), Z(i)] = ecef2enu(coordPZ90(i,1), coordPZ90(i,2), coordPZ90(i,3), coordpotr(1), coordpotr(2), coordpotr(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));

phi(i) = -atan2(Y(i),X(i))+pi/2;

else

teta(i) = NaN;

r(i) = NaN;

phi(i) = NaN;

end

end

[X\_sf, Y\_sf, Z\_sf] = sphere(25);

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

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

R\_z = 6371;

% ECI & ПЗ-90

plot3(coordPZ90(:,1)/1000, coordPZ90(:,2)/1000, coordPZ90(:,3)/1000,'r');

hold on

plot3(coordECI(:,1)/1000, coordECI(:,2)/1000, coordECI(:,3)/1000,'b');

grid on

title('Траектория движения спутника ГЛОНАСС №22 в СК ПЗ-90 и ECI');

xlabel('OX, км');

ylabel('OY, км');

zlabel('OZ, км');

axis('square');

axis('equal');

surf(X\_sf\*R\_z, Y\_sf\*R\_z, Z\_sf\*R\_z);

hold off

% SkyView

figure;

axes = polaraxes;

hold on

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

polarplot(axes,cutoff\_grads,cutoff\_angle,'b')

hold off

axes.ThetaDir = 'clockwise';

axes.ThetaZeroLocation = 'top';

title('SkyView спутника ГЛОНАСС №22')

Листинг подпрограммы difury:

function dy = difury(t, y)

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

Rz = 6378136; % экваториальный радиус Земли

w\_e = 7.292115E-05; % угловая скорость вращения Земли

C20 = -1082.62575E-06; % коэффициент при второй зональной гармонике разложения геопотенциала в ряд по сферическим функциям

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

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

Ax = -.186264514923E-05;

Ay = -.186264514923E-05;

Az = -.186264514923E-05;

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

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

JsumZ = Az;

dy = y(:);

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

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

ro = Rz/r;

dy(1) = y(4);

dy(2) = y(5);

dy(3) = y(6);

dy(4) = -mu\_\*y(1) + 3/2\*C20\*mu\_\*y(1)\*ro^2\*(1-5\*(y(3)/r)^2) + JsumX;

dy(5) = -mu\_\*y(2) + 3/2\*C20\*mu\_\*y(2)\*ro^2\*(1-5\*(y(3)/r)^2) + JsumY;

dy(6) = -mu\_\*y(3) + 3/2\*C20\*mu\_\*y(3)\*ro^2\*(3-5\*(y(3)/r)^2) + JsumZ;

end

**ПРИЛОЖЕНИЕ 2**

Листинг программы для 3 этапа

Файл main.cpp

#include <iostream>

#include <fstream>

#include <cmath>

#include <ctime>

#include <Runge\_kutta.h>

using namespace std;

int **main**()

{

time\_t start, end;

double dt = 0.1, T\_eph = 49500 + 18 + 3\*3600;

int n = (int) 12\*3600/dt, n\_eph = (int) (T\_eph-15\*3600)/dt, i\_delta = 0;

double t\_G0 = 9\*3600+18\*60+10.5009, w\_e = 7.292115E-05, t\_G = t\_G0 + w\_e\*(T\_eph - 3\*3600);

double cosTg = cos(t\_G), sinTg = sin(t\_G);

double X = .311761962891E+07, Y = .158781806641E+08, Z = .196852387695E+08;

double Vx = -.179497814178E+04, Vy = -.202221393585E+04, Vz = .192774677277E+04;

double \*Y0 = new double [6] {X\*cosTg - Y\*sinTg, X\*sinTg + Y\*cosTg, Z, Vx\*cosTg - Vy\*sinTg - w\_e\*(X\*sinTg + Y\*cosTg), Vx\*sinTg + Vy\*cosTg + w\_e\*(X\*cosTg - Y\*sinTg),Vz};

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

double \*t\_12h = new double [n];

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

Yy[i] = new double [6];

t\_12h[i] = 15\*3600+i\*dt;

}

double \*\*Yy1 = new double \*[n\_eph];

for (int i = 0; i < n\_eph; i++){

Yy1[i] = new double [6];

}

double \*\*Yy2 = new double \*[n-n\_eph+1];

for (int i = 0; i < n-n\_eph+1; i++){

Yy2[i] = new double [6];

}

for (int j = 0; j < 6; j++){

Yy1[0][j] = Y0[j];

Yy2[0][j] = Y0[j];

}

time(&start);

RungeKutta(T\_eph, -dt, *Yy1*, n\_eph,6);

RungeKutta(T\_eph, dt, *Yy2*, n-n\_eph+1,6);

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

{

if (i <= n\_eph-1){

for (int j = 0; j < 6; j++)

{

Yy[i][j] = Yy1[n\_eph-i-1][j];

}

} else {

for (int j = 0; j < 6; j++)

{

Yy[i][j] = Yy2[i-n\_eph+1][j];

}

}

}

time(&end);

for (int i = 0; i < n\_eph; i++){

delete [] Yy1[i];

}

for (int i = 0; i < n-n\_eph+1; i++){

delete [] Yy2[i];

}

delete [] Yy1;

delete [] Yy2;

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

double max\_delta = 0;

ifstream input("D:\\MATLAB.txt");

if (!input){

cout << "Cant open file, check location of file" << endl;

} else {

cout << "OK, Matlab file opened" << endl;

}

ofstream output;

output.open("D:\\CPP.txt");

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

input >> Yy\_matlab[0] >> Yy\_matlab[1] >> Yy\_matlab[2];

string YY\_str1 = to\_string(Yy[i][0]), YY\_str2 = to\_string(Yy[i][1]), YY\_str3 = to\_string(Yy[i][2]);

output << YY\_str1 << "\t" << YY\_str2 << "\t" << YY\_str3 << endl;

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

if (abs(Yy[i][j] - Yy\_matlab[j]) > max\_delta){

max\_delta = abs(Yy[i][j] - Yy\_matlab[j]);

i\_delta = i;

}

}

}

time(&end);

input.close();

output.close();

delete [] t\_12h;

delete [] Y0;

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

delete [] Yy[i];

}

delete [] Yy;

delete [] Yy\_matlab;

double time\_RK = difftime(end, start);

string time\_RK1 = to\_string(time\_RK\*1000000/n), max\_delta1 = to\_string(max\_delta), idelta = to\_string(i\_delta);

cout << "time of work of one cicle~: " << time\_RK1 << " mcs" << endl;

cout << "max delta of coords: " << max\_delta1 << " m" << endl;

cout << "number of max delta: " << idelta << endl;

return 0;

}

Файл runge\_kutta.h

#ifndef RUNGE\_KUTTA\_H

#define RUNGE\_KUTTA\_H

void **RungeKutta**(double t\_start, double dt, double \*\*&Yy, int size, int size\_s);

#endif // RUNGE\_KUTTA\_H

Файл runge\_kutta.cpp

#include<difury.h>

#include<Runge\_kutta.h>

#include<cmath>

#include <iostream>

void **RungeKutta**(double t\_start, double dt, double \*\*&Yy, int size, int size\_s){

double \*K1 = new double[6];

double \*K2 = new double[6];

double \*K3 = new double[6];

double \*K4 = new double[6];

double \*y\_pr1 = new double[6];

double \*y\_pr2 = new double[6];

double \*y\_pr3 = new double[6];

for (int i = 0; i < size-1; i++)

{

double t\_tek = t\_start + i\*dt;

difury(t\_tek, Yy[i], K1);

for (int j = 0; j < size\_s; j++){

y\_pr1[j] = Yy[i][j] + 0.5\*dt\*K1[j];

}

difury(t\_tek + 0.5\*dt, y\_pr1, K2);

for (int j = 0; j < size\_s; j++){

y\_pr2[j] = Yy[i][j] + 0.5\*dt\*K2[j];

}

difury(t\_tek + 0.5\*dt, y\_pr2, K3);

for (int j = 0; j < size\_s; j++){

y\_pr3[j] = Yy[i][j] + dt\*K3[j];

}

difury(t\_tek + dt, y\_pr3, K4);

for (int j = 0; j < size\_s; j++){

Yy[i+1][j] = Yy[i][j] + dt/6 \* (K1[j]+2\*K2[j]+2\*K3[j]+K4[j]);

}

}

delete[] K1;

delete[] K2;

delete[] K3;

delete[] K4;

delete[] y\_pr1;

delete[] y\_pr2;

delete[] y\_pr3;

}

Файл difury.cpp

#include<difury.h>

#include<cmath>

#include <iostream>

void **difury**(double t, double \*y, double \*dy){

double mu = 3.986004418E+14;

double Rz = 6378136;

double w\_e = 7.292115E-05;

double C20 = -1082.62575E-06;

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

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

double Ax = -.186264514923E-05;

double Ay = -.186264514923E-05;

double Az = -.186264514923E-05;

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

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

double JZ = Az;

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

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

Rz = Rz/r;

dy[0] = y[3];

dy[1] = y[4];

dy[2] = y[5];

dy[3] = -mu\*y[0] + 3.0/2\*C20\*mu\*y[0]\*Rz\*Rz\*(1-5\*pow(y[2]/r,2)) + JX;

dy[4] = -mu\*y[1] + 3.0/2\*C20\*mu\*y[1]\*Rz\*Rz\*(1-5\*pow(y[2]/r,2)) + JY;

dy[5] = -mu\*y[2] + 3.0/2\*C20\*mu\*y[2]\*Rz\*Rz\*(3-5\*pow(y[2]/r,2)) + JZ;

}

Файл difury.h

#ifndef DIFURY\_H

#define DIFURY\_H

void **difury**(double t, double \*y, double \*dy);

#endif // DIFURY\_H