НИУ «МЭИ»

Кафедра Радиотехнических систем

Расчетно-пояснительная записка к курсовому проекту по

дисциплине «Аппаратура потребителей СРНС»

Выполнил: студ. Бахолдин Н.В.

Группа: ЭР-15-17

Проверил: доц. Корогодин И.В.

Москва 2022

Оглавление

[Цель работы 3](#_Toc98878215)

[Дано 3](#_Toc98878216)

[Решение 3](#_Toc98878217)

# Цель работы

Изучение особенностей сигналов спутников GPS для определения положения спутника по данным с демодулятора его сигнала L1 C/A. На первом этапе происходит моделирование модуля разбора навигационного сообщения до структуры эфемерид.

# Дано

Файл “in.txt” с записанными в нем данными, зафиксированными навигационным приемником по сигналу GPS L1C/A. Файл содержит наблюдения псевдодальностей и прочих радионавигационных параметров, демодулированные и разобранные данные навигационного сообщения.

# Решение

Пункт 1

На первом этапе необходимо создать программу в среде С/ С++, выполняющую функции аналогичные модулю разбора навигационного сообщения. Листинг созданной части программы приведен в приложении.

На данном этапе реализован расчет всех параметров спутника.

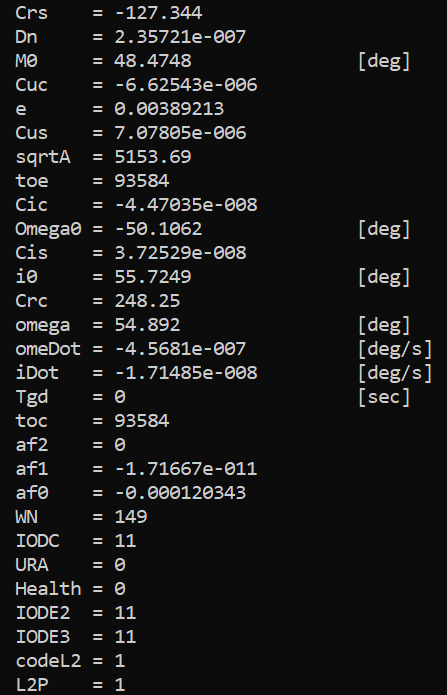


Рисунок 1 – Данные, полученные в ходе компиляции кода через командную строку

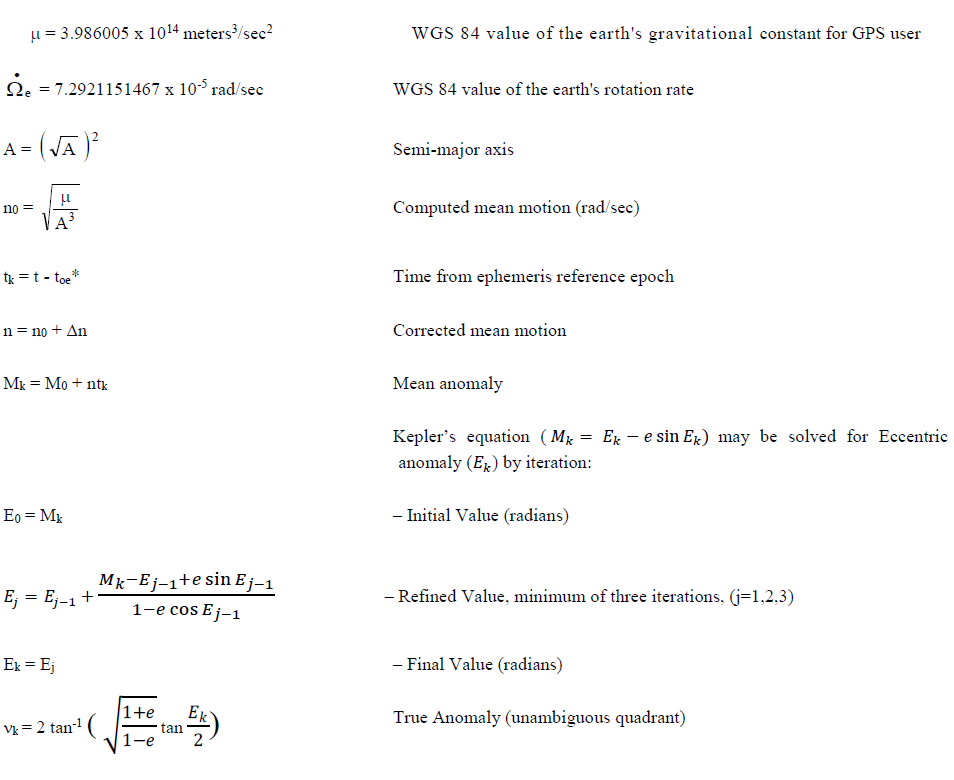
|  |  |
| --- | --- |
| IODE | 2827 |
| IODC | 11 |
| URA | 0 |
| Toe | 2022/02/14 01:59:44 |
| Toc | 2022/02/14 01:59:44 |
| A (√A) | 26560504.265 (5153.69) |
| e | 0.00389213 |
| I0 | 55.72485 |
| Ω0 | -50.10616 |
| ω | 54.89203 |
| M0 | 48.47481 |
| Δn | 2.3572e-07 |
| Ωdot | -4.5681e-07 |
| IDOT | -1.7149e-08 |
| Af0 | -120342.71 |
| Af1 | -0.0172 |
| Af2 | 0.0 |
| Cuc | -6.6254e-06 |
| Cus | 7.0781e-06 |
| Crc | 248.250 |
| Crs | -127.344 |
| Cic | -4.4703e-08 |
| Cis | 3.7253e-08 |
| Code | 001 |

Рисунок 2 – Данные, полученные в программе RTKNAVI

Вывод: Данные, полученные в программе RTKNAVI совпадают с данными, полученными в ходе компиляции кода. Первый этап курсового проекта выполнен.

Пункт 2.

Запишем время системы в формате GPS. Номер недели 2197, секунд от начала текущей недели = 86718 (с учетом секунд координации).



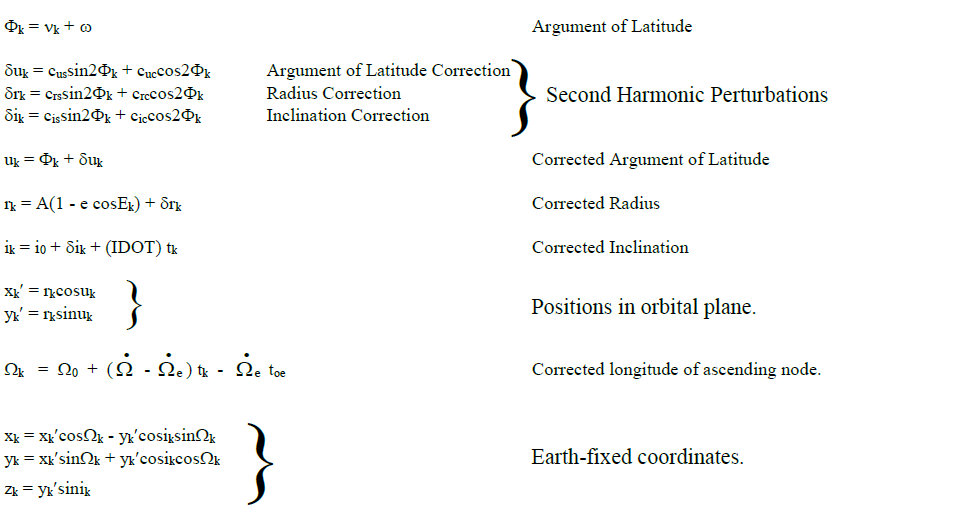


Рисунок 3 – Алгоритм расчета координат спутника

Результаты расчета:

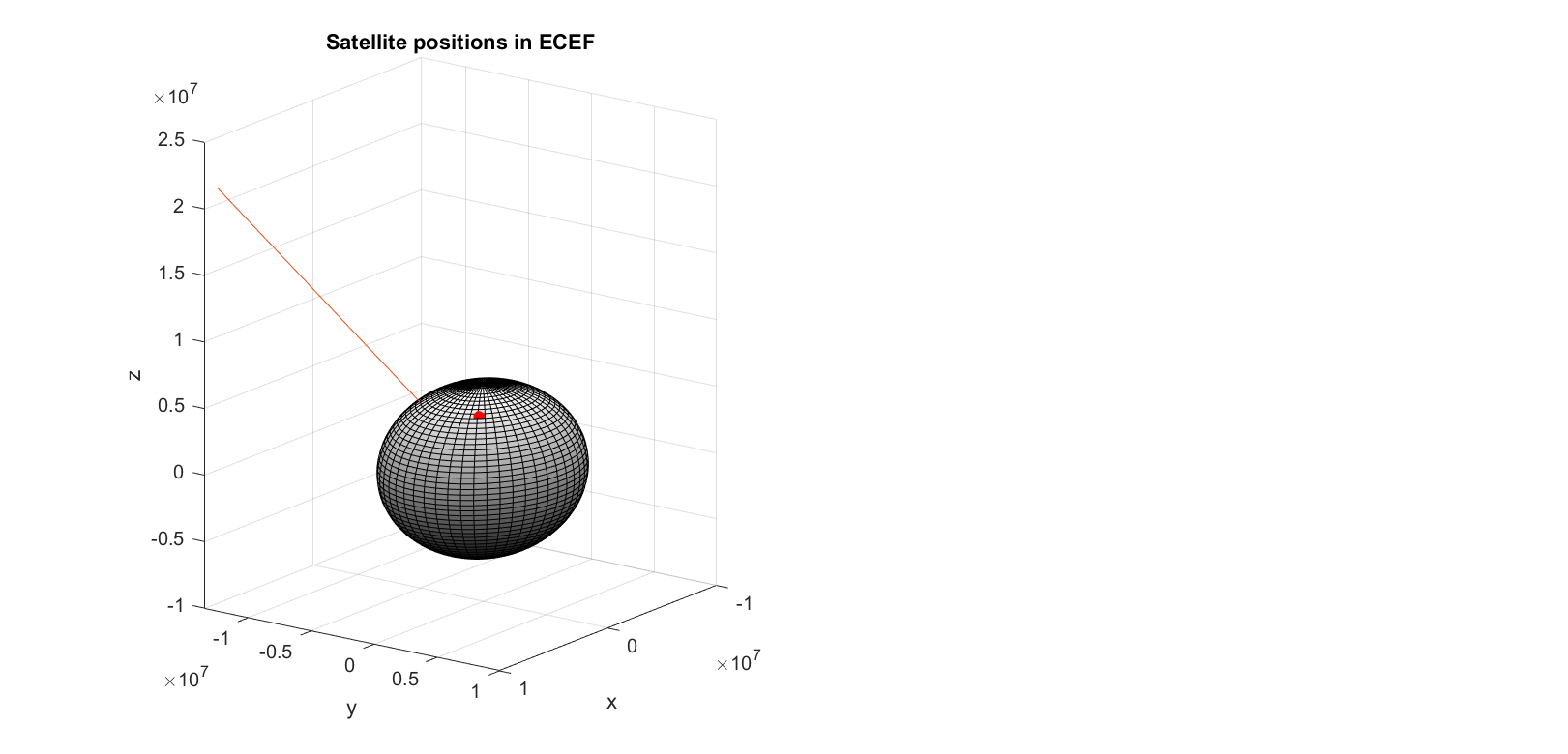


Рисунок 4 – Кривая положения спутника GPS в системе ECEF

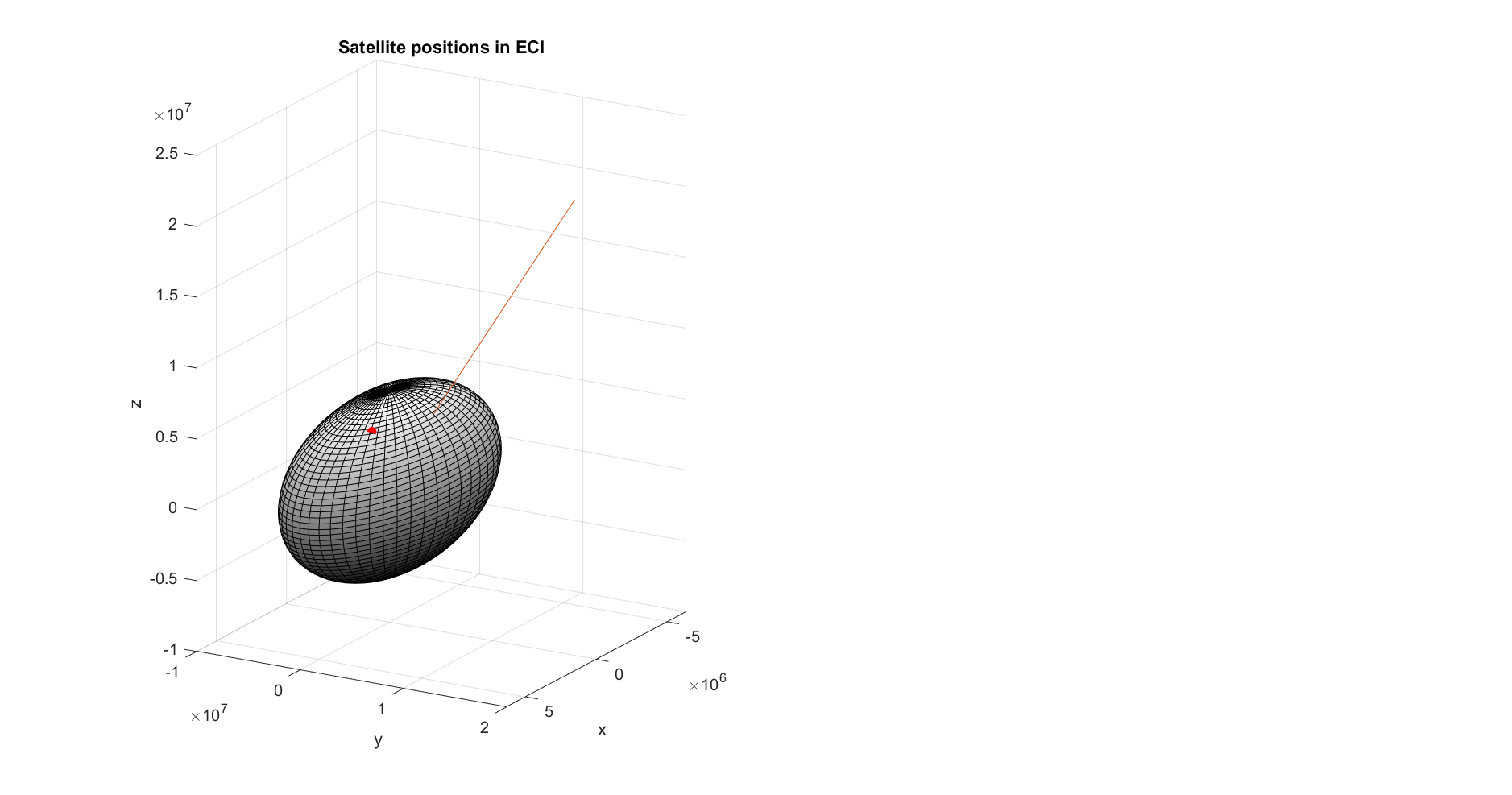


Рисунок 5 – Кривая положения спутника GPS в системе ECI

Вывод: На данный момент допущена вычислительная ошибка, которая не позволяет визуализировать траекторию движения спутника вокруг Земли. Предпринимаются попытки для исправления этой ошибки.

# Приложение

Листинг программы для пункта 1

#include <iostream>

#include <fstream>

#include <windows.h>

#include <string>

#include <stdlib.h>

#include <cmath>

#include <stdio.h>

#define SF\_2E5 pow(2,-5) //добавление scale factor-а

#define SF\_2E55 pow(2,-55)

#define SF\_2E43 pow(2,-43)

#define SF\_2E31 pow(2,-31)

#define SF\_2E4 pow(2,4)

#define SF\_2E19 pow(2,-19)

#define SF\_2E29 pow(2,-29)

#define SF\_2E33 pow(2,-33)

#define SemiCircles 180

using namespace std;

struct Ephemeris {

float Crs;

float Dn;

float M0;

float Cuc;

float e;

float Cus;

float sqrtA;

uint32\_t toe;

float Cic;

float Omega0;

float Cis;

float i0;

float Crc;

float omega;

float OmegaDot;

float iDot;

int16\_t Tgd;

uint32\_t toc;

float af2;

float af1;

float af0;

uint16\_t WN;

uint16\_t IODC;

uint32\_t URA;

uint32\_t Health;

uint16\_t IODE2;

uint16\_t IODE3;

bool codeL2;

bool L2P;

uint32\_t slot;

};

struct sub { // neccesary subframes (string - массив char-ов), достаем первые 3 сабфрейма

uint32\_t slot;

string sf1;

string sf2;

string sf3;

};

void sendStr(sub \*Rslot); //объявление функций (нужно для дальнейшей работы main-а)

uint32\_t razdel\_param(string sf, uint16\_t start1, int dlit1, uint16\_t start2, int dlit2);

void printEph(Ephemeris\* Eph);

void saveEph(Ephemeris\* Eph);

void razborsubframes(Ephemeris\* Eph, sub \*slot);

int32\_t sub2data(string sf, int32\_t start, int dlit);

int main(void)//главное тело функции

{

sub slot;//этой командой создаем структуру slot и передаем ее дальше

sendStr(&slot);//& - передача адреса

cout << slot.sf1 << endl << slot.sf2 << endl << slot.sf3 << endl;

Ephemeris \*Eph = (Ephemeris\*) calloc(1, sizeof(Ephemeris));

razborsubframes(Eph,&slot);//декодирование эфемерид из даты

printEph(Eph);

saveEph(Eph);

free(Eph);//освобождение памяти

}

void sendStr(sub \*Rslot)

{

SetConsoleOutputCP(CP\_UTF8);// подключение русского языка в консоль

string path = "in.txt";//создание строчки путь = название файла

ifstream fin;//объявляем класс и в стрим (выдуманная функция fin)

fin.open(path);//

if(fin.is\_open()) {

cout << "Файл открыт." << endl;

while (!fin.eof()) { //пока не конец файла

int N = 3;

string unknown\_data; //строка ненужных данных

int nmbr\_stlt; //номер спутника

uint32\_t slot;

uint32\_t subFrameNum;

string str;

uint32\_t slot\_SF1;

uint32\_t slot\_SF2;

uint32\_t slot\_SF3;

int mass[N];

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

{

fin >> mass[i];//из файла закидываем некоторые 3 значения

}

fin >> unknown\_data >> unknown\_data >> unknown\_data;//

fin >> nmbr\_stlt >> slot >> unknown\_data >> unknown\_data >> subFrameNum;//номер спутника - слот - 2 мусора - номер кадра

fin >> str;//строка данных

if (nmbr\_stlt == 3 and slot>= 604800/6){ //задаем номер спутника

if (subFrameNum == 1)

{

slot\_SF1 = slot;

Rslot->sf1 = str;

}

else if (subFrameNum == 2)

{

slot\_SF2 = slot;

Rslot->sf2 = str;

}

else if (subFrameNum == 3)

{

slot\_SF3 = slot;

Rslot->sf3 = str;//Rslot = real slot

}

if (slot\_SF1 + 1 == slot\_SF2 and slot\_SF2 + 1 == slot\_SF3) {

Rslot->slot = slot\_SF1;

return;

}

}

//

}

}

else

{

cout << "Ошибка открытия файла!!!" << endl;

}

fin.close();

}

int32\_t sub2data(string sf, int32\_t start, int dlit) {

int32\_t otvet = 0;

int32\_t Rotvet = 0;

for (int i = start; i < start+dlit; i++) {

otvet = (otvet | ((sf[i - 1] == '1') ? 1 : 0));//перевод текста в цифру

cout << sf[i-1];

if (i < start+dlit-1){

otvet = otvet<<1;

}

}

return otvet;

}

int32\_t compl2int(uint32\_t otvet, int dlit\_sub){ //сюда передаем побитовое значение otvet и сколько было бит и проверяем на дополнение до двух

int32\_t Rotvet = 0;

if (dlit\_sub == 8){

if (bool((1<<7) & otvet)){

otvet |= 0xFFFFFF00;

Rotvet = ~(otvet-1);

/\*cout<< endl << bitset<64>(Rotvet).to\_string() << endl;\*/

return -Rotvet;

}

}

if (dlit\_sub == 14){

if (bool((1<<13) & otvet)){

otvet |= 0xFFFFC000;

Rotvet = ~(otvet-1);

return -Rotvet;

}

}

if (dlit\_sub == 16){

if (bool((1<<15) & otvet)){//если 16й бит равен 1, тогда выполняем

otvet |= 0xFFFF0000;//0xFFFF0000 - с 32 по 17й бит единицы (заполняем единицами с 16 го бита

Rotvet = ~(otvet-1);

return -Rotvet;

}

}

if (dlit\_sub == 22){

if (bool((1<<21) & otvet)){

otvet |= 0xFFC00000;

Rotvet = ~(otvet-1);

return -Rotvet;

}

}

if (dlit\_sub == 24){

if (bool((1<<23) & otvet)){

otvet |= 0xFF000000;

Rotvet = ~(otvet-1);

return -Rotvet;

}

}

if (dlit\_sub == 32){

if (bool((1<<31) & otvet)){

otvet |= 0x00000000;

Rotvet = ~(otvet-1);

return -Rotvet;

}

}

return otvet;

}

uint32\_t razdel\_param(string sf, uint16\_t start1, int dlit1, uint16\_t start2, int dlit2) {

uint32\_t otvet = 0;//возвращаем 32 битное uint dlit1 - how much (выдумал)

for (int i = start1; i < start1+dlit1; i++) { // преобразование строки в набор бит

otvet = (otvet | ((sf[i-1] == '1')? 1 : 0)) << 1;// | - или, (функция вида true or false)

}

for (int i = start2; i < start2+dlit2; i++) {

otvet = otvet | ((sf[i-1] == '1')? 1 : 0);

if (i < start2+dlit2-1){//чтобы последний раз не сдвигать влево после окончания

otvet = otvet<<1;

}

}

return otvet;

}

void razborsubframes(Ephemeris\* Eph, sub \*slot){

Eph->slot = slot->slot;

Eph->Crs = compl2int(sub2data(slot->sf2,69,16),16)\*SF\_2E5;

Eph->Dn = compl2int(sub2data(slot->sf2,91,16),16)\*SF\_2E43\*SemiCircles;

Eph->M0 = compl2int(razdel\_param(slot->sf2,107, 8, 121, 24),32)\*SF\_2E31\*SemiCircles;

Eph->Cuc = compl2int(sub2data(slot->sf2,151,16),16)\*SF\_2E29;

Eph->e = razdel\_param(slot->sf2,167, 8, 181, 24) \* SF\_2E33;

Eph->Cus = compl2int(sub2data(slot->sf2,211,16),16)\*SF\_2E29;

Eph->sqrtA = razdel\_param(slot->sf2,227, 8, 241, 24) \* SF\_2E19;

Eph->toe = sub2data(slot->sf2,271,16)\*pow(2,4);

Eph->Cic = compl2int(sub2data(slot->sf3,61,16),16)\*SF\_2E29;

Eph->Omega0 = compl2int(razdel\_param(slot->sf3,77, 8, 91, 24),32)\*SF\_2E31\*SemiCircles;

Eph->Cis = compl2int(sub2data(slot->sf3,121,16),16)\*SF\_2E29;

Eph->i0 = compl2int(razdel\_param(slot->sf3,137, 8, 151, 24),32)\*SF\_2E31\*SemiCircles;

Eph->Crc = compl2int(sub2data(slot->sf3,181,16),16)\*SF\_2E5;

Eph->omega = compl2int(razdel\_param(slot->sf3,197, 8, 211, 24),32)\*SF\_2E31\*SemiCircles;

Eph->OmegaDot = compl2int(sub2data(slot->sf3,241,24),24)\*SF\_2E43\*SemiCircles;

Eph->iDot = compl2int(sub2data(slot->sf3,279,14),14)\*SF\_2E43\*SemiCircles;

Eph->Tgd = compl2int(sub2data(slot->sf1,197,8),8)\*SF\_2E31;

Eph->toc = compl2int(sub2data(slot->sf1,219,16),16)\*SF\_2E4;

Eph->af2 = compl2int(sub2data(slot->sf1,241,8),8)\*SF\_2E55;

Eph->af1 = compl2int(sub2data(slot->sf1,249,16),16)\*SF\_2E43;

Eph->af0 = compl2int(sub2data(slot->sf1,271,22),22)\*SF\_2E31;

Eph->WN = sub2data(slot->sf1,61,10);

Eph->IODC = razdel\_param(slot->sf1,83, 2, 211, 8);

Eph->URA = sub2data(slot->sf1,73,4);

Eph->Health = Eph->IODE2 = sub2data(slot->sf1,73,6);

Eph->IODE2 = sub2data(slot->sf2,61,8);

Eph->IODE3 = sub2data(slot->sf3,271,8);

Eph->codeL2 = sub2data(slot->sf1,71,2);

Eph->L2P = slot->sf1[90];

}

void saveEph(Ephemeris\* Eph)

{

ofstream fout;

string path = "out.txt";

fout.open(path);

if(fout.is\_open()) {

fout << endl << "LNAV Ephemeris (slot = " << Eph->slot << ") =" << endl;

fout << "\t\t Crs = " << Eph->Crs << endl;

fout << "\t\t Dn = " << Eph->Dn << endl;

fout << "\t\t M0 = " << Eph->M0 << "\t\t[deg]" << endl;

fout << "\t\t Cuc = " << Eph->Cuc << endl;

fout << "\t\t e = " << Eph->e << endl;

fout << "\t\t Cus = " << Eph->Cus << endl;

fout << "\t\t sqrtA = " << Eph->sqrtA << endl;

fout << "\t\t toe = " << Eph->toe << endl;

fout << "\t\t Cic = " << Eph->Cic << endl;

fout << "\t\t Omega0 = " << Eph->Omega0 << "\t\t[deg]" << endl;

fout << "\t\t Cis = " << Eph->Cis << endl;

fout << "\t\t i0 = " << Eph->i0 << "\t\t[deg]" << endl;

fout << "\t\t Crc = " << Eph->Crc << endl;

fout << "\t\t omega = " << Eph->omega << "\t\t[deg]" << endl;

fout << "\t\t omeDot = " << Eph->OmegaDot << "\t\t[deg/s]" << endl;

fout << "\t\t iDot = " << Eph->iDot << "\t\t[deg/s]" << endl;

fout << "\t\t Tgd = " << Eph->Tgd << "\t\t\t[sec]" << endl;

fout << "\t\t toc = " << Eph->toc << endl;

fout << "\t\t af2 = " << Eph->af2 << endl;

fout << "\t\t af1 = " << Eph->af1 << endl;

fout << "\t\t af0 = " << Eph->af0 << endl;

fout << "\t\t WN = " << Eph->WN << endl;

fout << "\t\t IODC = " << Eph->IODC << endl;

fout << "\t\t URA = " << Eph->URA << endl;

fout << "\t\t Health = " << Eph->Health << endl;

fout << "\t\t IODE2 = " << Eph->IODE2 << endl;

fout << "\t\t IODE3 = " << Eph->IODE3 << endl;

fout << "\t\t codeL2 = " << Eph->codeL2 << endl;

fout << "\t\t L2P = " << Eph->L2P << endl;

}

else

{

cout << "Ошибка открытия файла!!!" << endl;

}

fout.close();

cout << "Успешно!";

}

void printEph(Ephemeris\* Eph)

{

cout << endl << "LNAV Ephemeris (slot = " << Eph->slot << ") =" << endl;

cout << "\t\t Crs = " << Eph->Crs << endl;

cout << "\t\t Dn = " << Eph->Dn << endl;

cout << "\t\t M0 = " << Eph->M0 << "\t\t[deg]" << endl;

cout << "\t\t Cuc = " << Eph->Cuc << endl;

cout << "\t\t e = " << Eph->e << endl;

cout << "\t\t Cus = " << Eph->Cus << endl;

cout << "\t\t sqrtA = " << Eph->sqrtA << endl;

cout << "\t\t toe = " << Eph->toe << endl;

cout << "\t\t Cic = " << Eph->Cic << endl;

cout << "\t\t Omega0 = " << Eph->Omega0 << "\t\t[deg]" << endl;

cout << "\t\t Cis = " << Eph->Cis << endl;

cout << "\t\t i0 = " << Eph->i0 << "\t\t[deg]" << endl;

cout << "\t\t Crc = " << Eph->Crc << endl;

cout << "\t\t omega = " << Eph->omega << "\t\t[deg]" << endl;

cout << "\t\t omeDot = " << Eph->OmegaDot << "\t\t[deg/s]" << endl;

cout << "\t\t iDot = " << Eph->iDot << "\t\t[deg/s]" << endl;

cout << "\t\t Tgd = " << Eph->Tgd << "\t\t\t[sec]" << endl;

cout << "\t\t toc = " << Eph->toc << endl;

cout << "\t\t af2 = " << Eph->af2 << endl;

cout << "\t\t af1 = " << Eph->af1 << endl;

cout << "\t\t af0 = " << Eph->af0 << endl;

cout << "\t\t WN = " << Eph->WN << endl;

cout << "\t\t IODC = " << Eph->IODC << endl;

cout << "\t\t URA = " << Eph->URA << endl;

cout << "\t\t Health = " << Eph->Health << endl;

cout << "\t\t IODE2 = " << Eph->IODE2 << endl;

cout << "\t\t IODE3 = " << Eph->IODE3 << endl;

cout << "\t\t codeL2 = " << Eph->codeL2 << endl;

cout << "\t\t L2P = " << Eph->L2P << endl;

}

Листинг программы для пункта 2

clc

close all

clear

%Константы из ИКД GPS

pi = 3.1415326535898;%отношение длины окружности к ее диаметры (Ratio of a circle)s circumference to its diameter)

mu = 3.986004418e14;%геоцентрическая гравитационная постоянная

omegaE = 7.2921151467e-5;%средняя угловая скорость Земли

c = 299792458;%скорость света

%Константы из ИКД GPS

e\_nominal = 0;%номинальный эксцентриситет орбиты

i\_nominal = deg2rad(56);%градусов/номинальное наклонение орбиты относительно экваториальной плоскости

A\_nominal = 29600000;%метров/большая полуось номинальной орбиты

%Параметры из эфемерид (пункт 1)

A = 26560504.265 ;%большая полуось

n0 = sqrt(mu/A^3);%расчетное среднее движение (рад/с)

ecc = 0.00389213;%эксцентриситет

omega0 = -50.10616\*pi/180;%долгота восходящего узла

omegaDot = -4.5681e-07\*pi/180;%скорость изменения прямого восхождения

omega = 54.89203\*pi/180;%аргумент перигея

M0 = 48.47481\*pi/180;%ср. аномалия КА в контр. момент времени

af0 = -120342.71;%поправка времени КА

af1 = -0.0172;%поправка времени КА

t\_0e = 93602;%опорное время альманаха

%Расчет элементов Кеплера

%Correction terms (table 82)

delta\_n = 2.3572e-07;

i\_dot = -1.7149e-08;

C\_uc = -6.6254e-06;

C\_us = 7.0781e-06;

C\_rc = 248.250;

C\_rs = -127.344;

C\_ic = -4.4703e-08;

C\_is = 3.7253e-08;

n = n0+delta\_n;%скорректированное среднее движение

%% Расчет элементов Кеплера (table 61) (ECEF WGS 84 system)

for i = 1:86400 %(24 hours in sec)

t = 86700+i+18; %TOW (добавляем 18 секунд координации)

tk = t-t\_0e;%время от опорной эпохи эфемерид

Mk = M0+n\*tk;%средняя аномалия

E = zeros(1,4);

E(1,1) = Mk;

for j = 2:4

E(1,j) = E(1,j - 1) + (Mk - E(1,j - 1) + ecc\*sin(E(1,j - 1)))/(1 - ecc\*cos(E(1,j - 1))); % 3 iterations

E(1,j-1) = E(1,j);

end

E\_k = E(1,4);%эксцентрическая аномалия

chislitel = sqrt(1-ecc^2)\*sin(E\_k)/(1-ecc\*cos(E\_k));

znamenatel = (cos(E\_k)-ecc)/(1-ecc\*cos(E\_k));

nu = atan2(chislitel,znamenatel);%истинная аномалия

Fi = nu+omega;%аргумент широты

deltau = C\_us\*sin(2\*Fi)+C\_uc\*cos(2\*Fi);%аргумент скорректированной широты

deltar = C\_rs\*sin(2\*Fi)+C\_rc\*cos(2\*Fi);%радиальная коррекция

deltai = C\_is\*sin(2\*Fi)+C\_ic\*cos(2\*Fi);

%можно не использовать написанные выше формулы, так как они равны 0

uk = Fi+deltau;%скорректированный аргумент широты

rk = A\*(1-ecc\*cos(E\_k))+deltar;%скорректированный радиус

ik = i\_nominal +deltai + i\_dot\*tk;%скорректированный аргумент наклона

x\_shtrih = rk\*cos(uk);

y\_shtrih = rk\*sin(uk);%позиция в орбитальной плоскости

Omega = omega0 + (omegaDot - omegaE)\*tk - omegaE\*t\_0e;%скорректированная широта восходящего узла

x = x\_shtrih\*cos(Omega) - y\_shtrih\*cos(ik)\*sin(Omega);

y = x\_shtrih\*sin(Omega) + y\_shtrih\*cos(ik)\*cos(Omega);

z = y\_shtrih\*sin(ik);%координаты фазового центра антенны во время t

end

x\_coord(1,i) = x;

y\_coord(1,i) = y;

z\_coord(1,i) = z;

%% ECI (Earth-Centered Inertial coordinate system)

theta = omegaE\*tk;

x1 = x\*cos(theta)-y\*sin(theta);

y1 = x\*sin(theta)+y\*cos(theta);

z1 = z;

x1\_coord(1,i) = x1;

y1\_coord(1,i) = y1;

z1\_coord(1,i) = z1;

[x\_sphere, y\_sphere, z\_sphere] = sphere(50);

x\_Earth=Earth\_radius\*x\_sphere;

y\_Earth=Earth\_radius\*y\_sphere;

z\_Earth=Earth\_radius\*z\_sphere;%рисую Землю

figure (1)

subplot (1,2,1)

surf(x\_Earth,y\_Earth,z\_Earth); hold on;

colormap ('gray');

plot3(Coord\_x, Coord\_y, Coord\_z, 'r.','MarkerSize', 20 );

plot3(x\_coord(1,:), y\_coord(1,:), z\_coord(1,:));

xlabel('x');

ylabel('y');

zlabel('z');

title('Satellite positions in ECEF')

figure (2)

subplot (1,2,1)

surf(x\_Earth,y\_Earth,z\_Earth); hold on;

colormap ('gray');

plot3(Coord\_x, Coord\_y, Coord\_z, 'r.','MarkerSize', 20 );

plot3(x1\_coord(1,:), y1\_coord(1,:), z1\_coord(1,:));

xlabel('x');

ylabel('y');

zlabel('z');

title('Satellite positions in ECI')