

Федеральное государственное бюджетное образовательное  
учреждение высшего образования  
«Национальный исследовательский университет «МЭИ»

Институт: ИРЭ Кафедра: Радиотехнических систем

Специальность: 11.05.01 Радиоэлектронные системы и  
комплексы

**ОТЧЕТ по курсовой работе**

Наименование Разработка модуля расчёта координат спутника  
курсовой работы: Beidou

**СТУДЕНТ**

 / Дворецкий И.А.  
(подпись) (Фамилия и инициалы)  
Группа ЭР-15-16  
(Номер учебной группы)

**ЗАЩИТА КУРСОВОЙ РАБОТЫ**

\_\_\_\_\_  
(отлично, хорошо, удовлетворительно, неудовлетворительно,  
зачтено, не зачтено)

\_\_\_\_\_/ Корогодин И.В.  
(подпись) (Фамилия и инициалы члена комиссии)

\_\_\_\_\_/ Шатилов А.Ю.  
(подпись) (Фамилия и инициалы члена комиссии)

**Москва, 2021**

## **ВВЕДЕНИЕ**

Спутниковые радионавигационные системы время являются неотъемлемой частью нашей жизни. Они используются в различных сферах начиная от телефона до ракет. Наиболее распространенными являются системы ГЛОНАСС (Россия), GPS (США), Galileo (Евросоюз), Beidou (Китай).

**Цель проекта** - добавление в программное обеспечение приемника функции расчета положения спутника Beidou на заданное время по данным его эфемерид.

Требования к разрабатываемому программному модулю:

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

Для достижения цели выполняется ряд задач, соответствующих этапам проекта и контрольным мероприятиям:

- обработка данных от приемника, работа со сторонними сервисами для подготовки входных и проверочных данных для разрабатываемого модуля;
- моделирование модуля в Matlab/Python;
- реализация программного модуля на C/C++, включая юнит-тестирование в Check.

Конечная цель всего курсового проекта - получить библиотеку функций на «C++», позволяющую рассчитывать положение спутника Beidou по его эфемеридам.

## ЭТАП 1. ИСПОЛЬЗОВАНИЕ СТОРОННИХ СРЕДСТВ

### 1.1. Описание задания

Дан номер спутника BEIDOU, вариант – C06, значения эфемерид для спутников указаны в бинарном и текстовом файлах. Значения получены от антенны Narxон HX-CSX601A, установленной на крыше корпуса Е МЭИ. Она через 50-метровый кабель, сплиттер, bias-tee и усилитель подключена к трем навигационным приемникам:

- Javad Lexion LGDD,
- SwiftNavigation Piksi Multi,
- Clonicus разработки ЛНС МЭИ.

Эти приемники осуществляют первичную обработку сигналов Beidou B1I, выдавая по интерфейсам соответствующие потоки данных - наблюдения псевдодальностей и эфемериды спутников. Данные от приемника Clonicus, записанные вечером 16 февраля 2021 года.

C06	36828	IGSO-1	BDS-2	01.08.10	3867	Используется по ЦН
-----	-------	--------	-------	----------	------	--------------------

Рисунок 1 – Состояние 6-го спутника BEIDOU с «Информационно-аналитического центра координатно-временного и навигационного обеспечения»

5	Компас IGSO-1	C06	31.07.2010, 20:50	CZ-3A	2010-036A	36828	Геосинхронная, накл. 55°; 118° в. д.	действующий
---	---------------	-----	-------------------	-------	-----------	-------	---	-------------

Рисунок 2 – Состояние 6-го спутника BEIDOU с сайта Википедия.

По рисункам 1 и 2 видно номер спутника – 36828, название спутника – «Компас IGSO-1».

## 1.2. Определение орбиты и положения спутника на ней с помощью сервиса CelesTrak

Для выполнения данного пункта нужно перейти на сайт CelesTrak (<https://celestrak.com>) , настроить параметры и выбрать нужный спутник, после чего будет определена орбита и его положение.

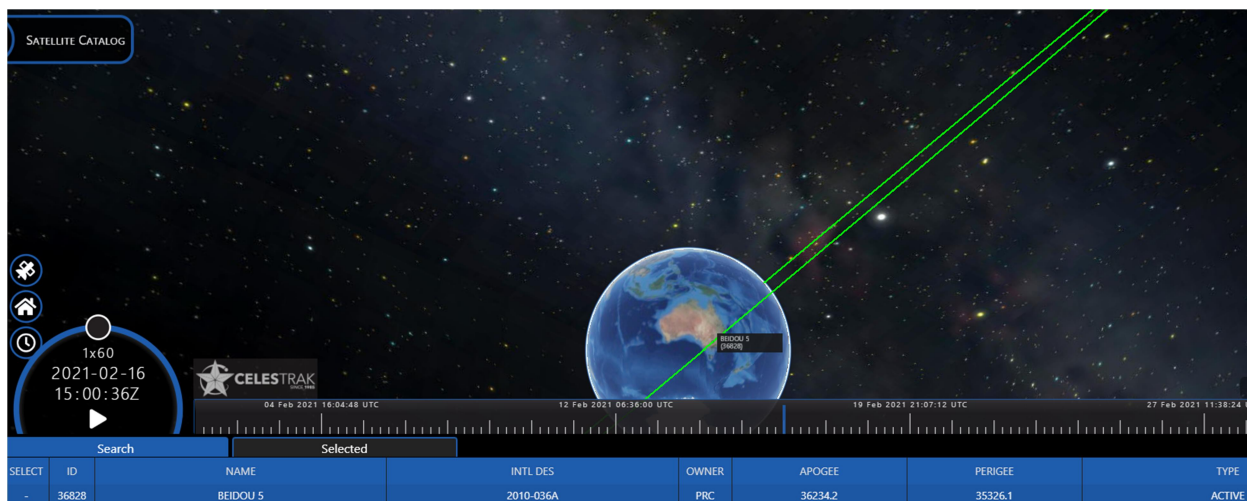


Рисунок 3 – Моделирование с помощью CelesTrak.

Значения совпадают, значит это действительно нужный нам спутник, проведем моделирование на момент времени 15:00, 16 февраля 2021, так как на данном сервисе отсчет времени происходит по UTC(0).

## 1.3. Расчёт графика угла места собственного спутника от времени по данным Trimble GNSS Planning Online

Введём параметры для моделирования GNSS Planning Online, координаты установим в соответствии с расположением антенны соответственно значению корпуса Е МЭИ, также начальное время будет соответствовать 18:00, временной пояс +3 (UTC +3) на всем этапе моделирования в сервисе GNSS Planning Online, высота выбирается из суммы высоты над уровнем моря (146 м) и примерной высотой здания (25 м) и округляется до сотен.

Satellite Selection Change selection

Satellites: 1/131

System: active	Satellites	
	Selected	Healthy
GPS	<input type="checkbox"/>	0 32
GLONASS	<input type="checkbox"/>	0 23
Galileo	<input type="checkbox"/>	0 22
BeiDou	<input checked="" type="checkbox"/>	1 49
QZSS	<input type="checkbox"/>	0 4

My Settings

Time of almanac:	2021-02-16
Time zone:	UTC +03:00
Visible period:	2021-02-16 18:00 - 2021-02-17 06:00
Latitude:	N 55° 45' 23.5491"
Longitude:	E 37° 42' 13.4571"
Height:	200 m
Elevation cutoff:	10 °

Settings

Latitude: N 55° 45' 23.5491"
Longitude: E 37° 42' 13.4571"
Height: 200 m
Elevation cutoff: 10 °
Day: 16.02.2021 Today
Start time: 18:00 UTC +03:00
Period [hours]: 12
Time zone: (UTC+03:00) Moscow, St. Petersburg
Apply

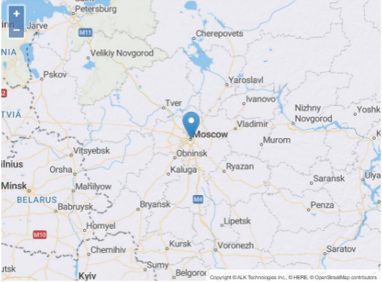


Рисунок 4 – Моделирование с помощью сервиса Trimble GNSS Planning.

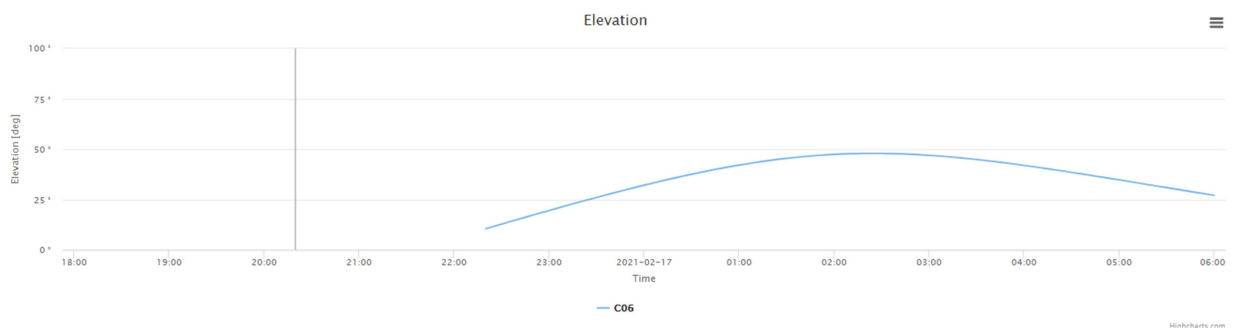


Рисунок 5 – График угла места собственного спутника от времени.

Из графика видно, что спутник на указанном временном интервале с 18:00 до 06:00 был в области видимости с 22:20 до 06:00.

#### 1.4. Расчет диаграммы угла места и азимута спутника (SkyView, он же SkyPlot) по данным Trimble GNSS Planning Online

Проведем моделирование Sky Plot во временном интервале 18:00-06:00 и зафиксируем положение спутника в критических точках, то есть когда он находился в области видимости - в 22:20, 00:50, 03:30 и 06:00.

4 графика положения спутника:

- 16 февраля 2021 в 22:20:

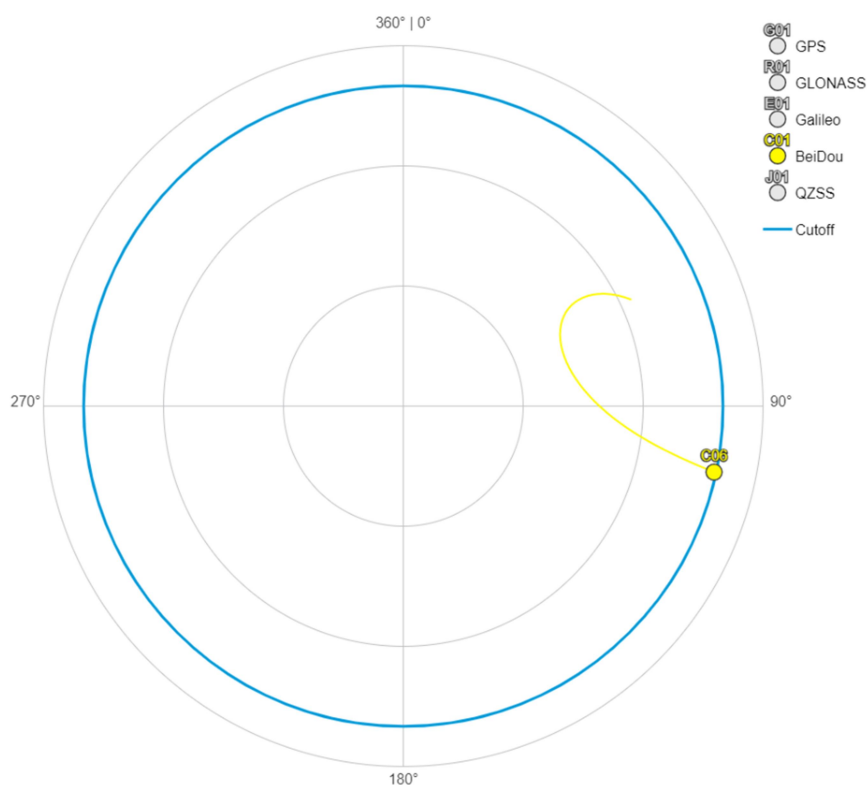


Рисунок 7 - Моделирование с помощью сервиса Trimble GNSS Planning.

- 17 февраля в 00:50:

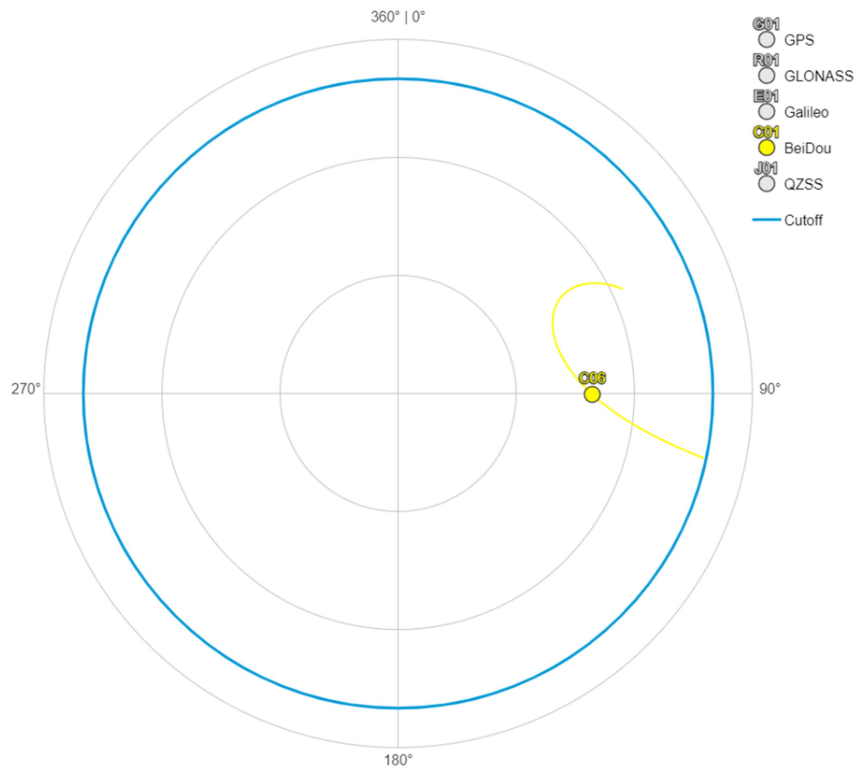


Рисунок 8 - Моделирование с помощью сервиса Trimble GNSS Planning.

- 17 февраля в 03:30:

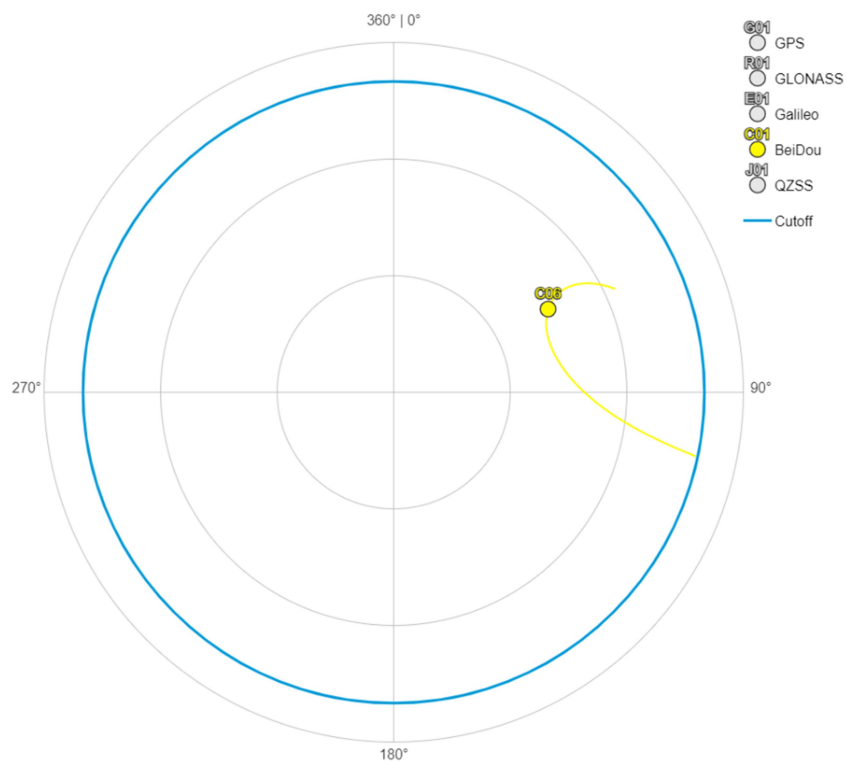


Рисунок 9 - Моделирование с помощью сервиса Trimble GNSS Planning.

- 17 февраля в 06:00:

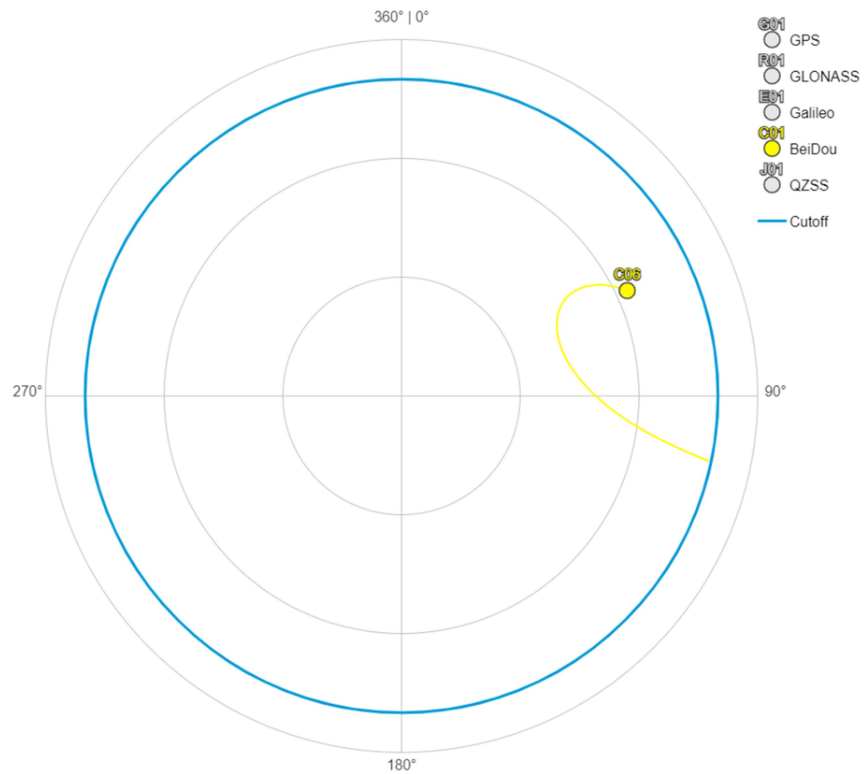


Рисунок 10 - Моделирование с помощью сервиса Trimble GNSS Planning.

### 1.5. Формирование списка и описание параметров, входящих в состав эфемерид.

Таблица 1 – Значения эфемерид спутника C06

Параметр	Значение	Размерность
SatNum	6	-
$t_{oe}, t_{oe}$	241200000.000	мс
$Crs, C_{rs}$	-9.2875000000000000e+01	м
$Dn, \Delta n$	8.71107710704449589e-13	рад/мс
$M_0, M_0$	2.32726368913121773e+00	рад
$Cuc, C_{uc}$	-2.62074172496795654e-06	рад
$e$	1.05765871703624725e-02	-



Cus, $C_{us}$	2.34702602028846741e-05	рад
sqrtA, $\sqrt{A}$	6.49287138557434082e+03	м <sup>1/2</sup>
Cic, $C_{ic}$	-1.12690031528472900e-07	рад
Omega0, $\Omega_0$	6.36759199965142852e-01	рад
Cis, $C_{is}$	3.25962901115417480e-09	рад
i0, $i_0$	9.46015118241178121e-01	рад
Crc, $C_{rc}$	-4.82140625000000000e+02	м
Omega, $\omega$	-2.20504767262928070e+00	рад
OmegaDot, $\Omega$	-1.77328815028356074e-12	рад/мс
iDot, $IDOT$	-2.00008331149807446e-14	рад/мс
Tgd, $T_{GD}$	8.70000000000000000e+04	мс
toc, $t_{oc}$	2.41200000000000000e+08	мс
af2, $a_{f2}$	0.00000000000000000e+00	мс/мс <sup>2</sup>
af1, $a_{f1}$	3.37445626996668580e-11	мс/мс
af0, $a_{f0}$	9.76731553673744202e-02	мс
URA	0	-
IODE	257	-
IODC	0	-
codeL2	0	-
L2P	0	-
WN	789	-

## Этап 2. Моделирование.

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

Построить трехмерные графики множества положений спутника Beidou с системным номером. Графики в двух вариантах: в СК ECEF WGS84 и соответствующей ей инерциальной СК. Положения должны соответствовать временному интервалу с 18:00 МСК 16 февраля до 06:00 МСК 17 февраля 2021 года. Допускается использовать одни и те же эфемериды на весь рассматриваемый интервал.

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

Рассчитаем количество секунд от начала текущей недели:

$$(24 \cdot 3 + 15) \cdot 3600 = 313200 \text{ с}$$

Моделирование производим в программе MatLab.

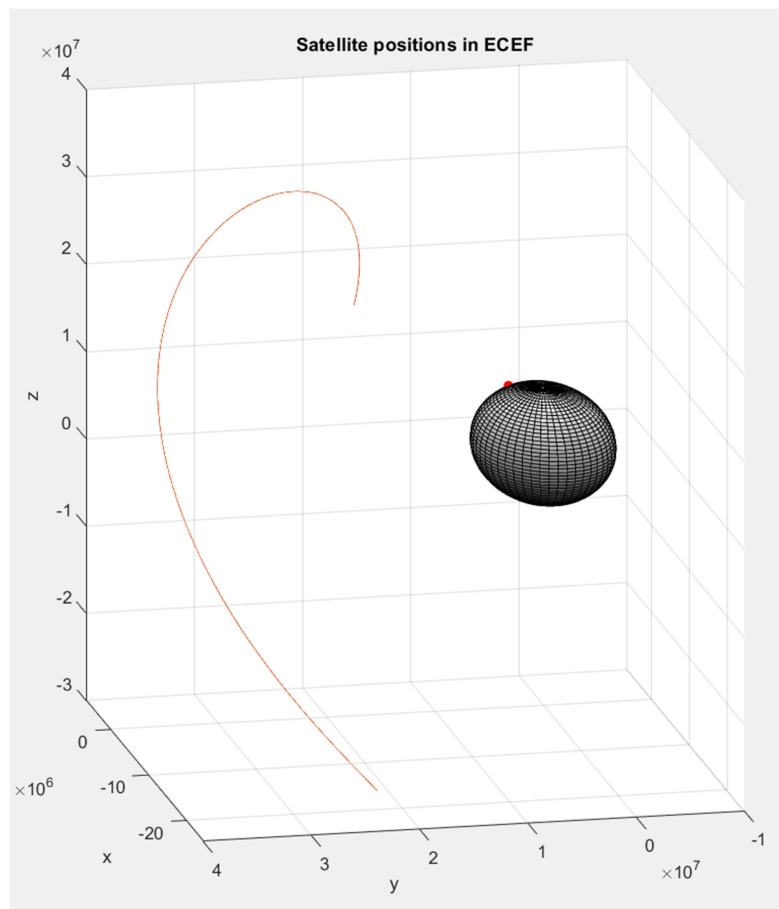


Рисунок 11 - Траектория движения спутника Beidou в системе ECEF.

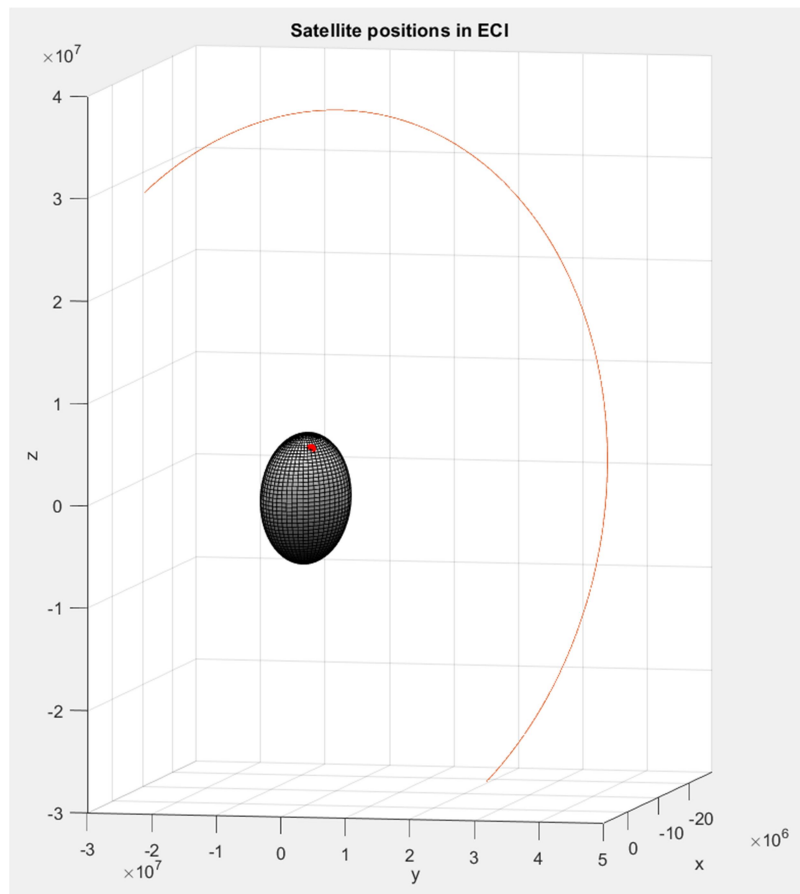


Рисунок 12 - Траектория движения спутника Beidou в системе ECI.

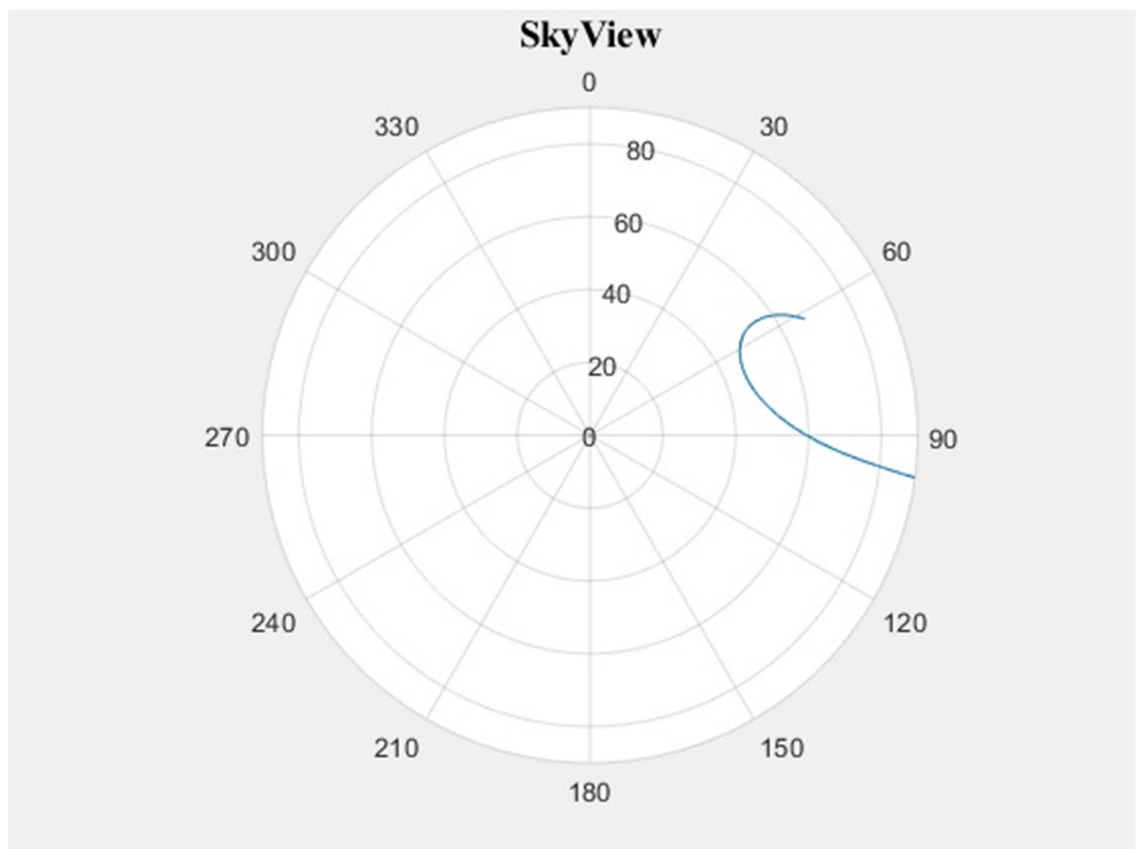


Рисунок 13 - SkyView спутника Beidou.

Сравним графики Sky View на рисунках 6 и 13, можно сказать что результаты совпадают с малейшими расхождениями, это связано с использованием одних эфемерид и особенностями алгоритмов программы MatLab.

Код программы в приложении.

### **Приложение.**

```
close all;
```

```
clear all;
```

```
clc;
```

```
format long
```

```
%% Эфемериды
```

```
SatNum = 6;
```

```
toe = 241200;
```

```
Crs = -9.287500000000000e+01;
```

```
Dn = 8.71107710704449589e-13;
```

```
M0 = 2.32726368913121773e+00;
```

```
Cuc = -2.62074172496795654e-06;
```

```
e = 1.05765871703624725e-02;
```

```
Cus = 2.34702602028846741e-05;
```

```
sqrtA = 6.49287138557434082e+03;
```

```
Cic = -1.12690031528472900e-07;
```

```
Omega0 = 6.63759799965142852e-01;
```

```
Cis = 3.25962901115417480e-09;
```

```
i0 = 9.46015118241178121e-01;
```

```
Crc = -4.8214062500000000e+02;
```

```
omega = -2.20504767262928070e+00;
```

OmegaDot = -1.773288508356074e-12;

iDot = -2.00008331149807446e-14;

Tgd = 9.750000000000000e+05;

toc = 2.196000000000000e+08;

af2 = 1.48307593848345250e-22;

af1 = 5.06794606280891458e-12;

af0 = 3.53220045566558838e-01;

URA = 0;

IODE = 257;

IODC = 1;

codeL2 = 0;

L2P = 0;

WN = 789;

%% Значения констант

mu = 3.986004418e14; % гравитационная постоянная

omega\_e = 7.2921151467e-5; % скорость вращения

%% Временной промежуток

begin\_time = (24\*2+18-3)\*60\*60; % время начала 8:00 по МСК 16 февраля

end\_time = (24\*3+6-3)\*60\*60; % время окончания 6:00 по МСК 17 февраля

%% Длина временного промежутка

t\_arr = begin\_time:1:end\_time;

%% Большая полуось

A = sqrtA^2;

```

%% Среднее движение

n0 = sqrt(mu/A^3);

n = n0+Dn;

for k = 1:length(t_arr)

    % Vremya

    t(k) = t_arr(k)-toe;

    if t(k) > 302400

        t(k) = t(k)-604800;

    end

    if t(k) < -302400

        t(k) = t(k)+604800;

    end

    % Средняя аномалия

    M(k) = M0+n*t(k);

    % Решение уравнения Кеплера

    E(k) = M(k);

    E_old(k) = M(k)+1;

    epsilon = 1e-6;

    while abs(E(k)-E_old(k)) > epsilon

        E_old(k) = E(k);

        E(k) = M(k)+e*sin(E(k));

    end

```

% Истинная аномалия

$$\nu(k) = \text{atan2}(\sqrt{1-e^2} \sin(E(k)), \cos(E(k)) - e);$$

% Коэффициент коррекции

$$\cos\_correction(k) = \cos(2 * (\omega + \nu(k)));$$

$$\sin\_correction(k) = \sin(2 * (\omega + \nu(k)));$$

% Аргумент широты

$$u(k) = \omega + \nu(k) + C_{uc} * \cos\_correction(k) + C_{us} * \sin\_correction(k);$$

% Радиус

$$r(k) = A * (1 - e * \cos(E(k))) + C_{rc} * \cos\_correction(k) + C_{rs} * \sin\_correction(k);$$

% Наклон

$$i(k) = i_0 + \dot{i} * t(k) + C_{ic} * \cos\_correction(k) + C_{is} * \sin\_correction(k);$$

% Долгота восходящего угла

$$\lambda(k) = \Omega_0 + (\Omega_{dot} - \omega_e) * t(k) - \omega_e * t_0;$$

% Положение на орбите

$$x = r(k) * \cos(u(k));$$

$$y = r(k) * \sin(u(k));$$

% Координаты

$$X_0(k) = x * \cos(\lambda(k)) - y * \cos(i(k)) * \sin(\lambda(k));$$

$$Y_0(k) = x * \sin(\lambda(k)) + y * \cos(i(k)) * \cos(\lambda(k));$$

$$Z_0(k) = y * \sin(i(k));$$

```

%
X(k) = X0(k)*cos(lambda(k))+Y0(k)*sin(lambda(k));
Y(k) = -X0(k)*sin(lambda(k))+Y0(k)*cos(lambda(k));
Z(k) = Z0(k);
end

%% Из HKA в WGS84

ppb = 1e-9;
mas = 1e-3/206264.8; % [radian]

MATRIX_WGS_84 = [-3*ppb -353*mas -4*mas;
    353*mas -3*ppb 19*mas;
    4*mas -19*mas -3*ppb];

crd_WGS_84 = [X0; Y0; Z0];

for i = 1:length(crd_WGS_84(1,:))
    crd_WGS_84(:,i) = crd_WGS_84(:,i) + MATRIX_WGS_84 * crd_WGS_84(:,i) + [0.07; -0; -0.77];
end

crd_WGS_84 = crd_WGS_84.'; % perekhod k vektoru-stroke

%% postroenie grafikov

R_Earth = 6371e3;
[XE,YE,ZE] = sphere(10);

figure

surf(XE*R_Earth,YE*R_Earth,ZE*R_Earth)

```



```

hold on

grid on

plot3(crd_WGS_84(:,1), crd_WGS_84(:,2), crd_WGS_84(:,3))

plot3(X, Y, Z)

title('Satellite trajectory', 'FontName', 'Times New Roman', 'FontSize',14)

xlabel('X, m', 'FontName', 'Times New Roman', 'FontSize',14)

ylabel('Y, m', 'FontName', 'Times New Roman', 'FontSize',14)

zlabel('Z, m', 'FontName', 'Times New Roman', 'FontSize',14)

hold off

lgd = legend('Earth','CK ECEF WGS84','Inertial Coordinate System');

lgd.FontName = 'Times New Roman';

```

```

%% Перевод координат корпуса E в систему WGS84

```

```

% Широта(сначала идут значения - затем перевод)

```

```

N_gr = 55;

```

```

N_min = 45;

```

```

N_sec = 23.8178;

```

```

N = N_gr*pi/180+N_min/3437.747+N_sec/206264.8;

```

```

% Долгота(сначала идут значения - затем перевод)

```

```

E_gr = 37;

```

```

E_min = 42;

```

```

E_sec = 12.2608;

```

```

E = E_gr*pi/180+E_min/3437.747+E_sec/206264.8;

```

```

H = 500; % Приблизительное значение высоты расположения антенны на корпусе E(высота над
уровнем моря + высота корпуса E)

```

```

llh = [N E H];

crd_PRM = llh2xyz(llh)';

%% Построение SkyPlot

for i = 1:length(crd_WGS_84(:,1))

    [X(i) Y(i) Z(i)] =
    ecef2enu(crd_WGS_84(i,1),crd_WGS_84(i,2),crd_WGS_84(i,3),N,E,H,wgs84Ellipsoid,'radians');

    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

%% skyplot

figure

ax = polaraxes;

polarplot(ax,phi,teta*180/pi)

```

```

ax.ThetaDir = 'clockwise';

ax.ThetaZeroLocation = 'top';

title('SkyView', 'FontName', 'Times New Roman', 'FontSize',14)

%% Построение графика угла места

th = hours(t_arr/3660 - 68); % перевод временной оси в формат hh:mm

figure

grid on

hold on

plot(th,(-teta)*180/pi+90,'DurationTickFormat','hh:mm') % временная ось

xlim([th(1) th(end)])

title('Elevation', 'FontName', 'Times New Roman', 'FontSize',14) % отображение названия графика

xlabel('Time', 'FontName', 'Times New Roman', 'FontSize',14) % отображение названия
горизонтальной оси

ylabel('Elevation, deg', 'FontName', 'Times New Roman', 'FontSize',14) % отображение названия
вертикальной оси

%% функция преобразования координат из WGS84 в ECEF

function xyz = llh2xyz(llh)

phi = llh(1); % llh(1) = широта в радианах

lambda = llh(2); % llh(2) = долгота в радианах

h = llh(3); % llh(3) = высота над уровнем моря в метрах

a = 6378137.0000; % полуось земли в метрах

b = 6356752.3142; % полуось земли в метрах

e = sqrt(1-(b/a).^2);

sinphi = sin(phi);

```

```

cosphi = cos(phi);
coslam = cos(lambda);
sinlam = sin(lambda);
tan2phi = (tan(phi))^2;
tmp = 1-e*e;
tmpden = sqrt(1+tmp*tan2phi);

x = (a*coslam)/tmpden+h*coslam*cosphi;

y = (a*sinlam)/tmpden+h*sinlam*cosphi;

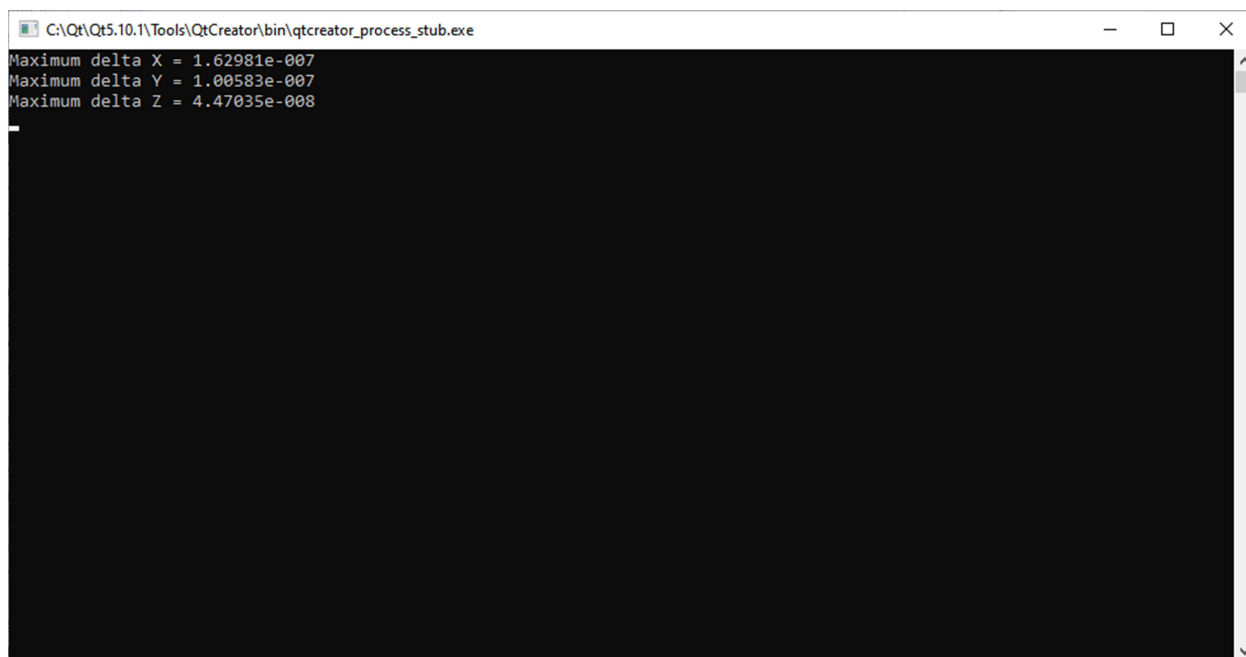
tmp2 = sqrt(1-e*e*sinphi*sinphi);
z = (a*tmp*sinphi)/tmp2+h*sinphi;

xyz(1) = x; %    xyz(1) = ECEF x-координата в метрах
xyz(2) = y; %    xyz(2) = ECEF y-координата в метрах
xyz(3) = z; %    xyz(3) = ECEF z-координата в метрах
end

```

### Этап 3. Реализация.

На этом этапе работы надо разработать на языке C++ функцию расчёта положения спутника Beidou на заданное время по шкале UTC. Написание программы производилось в Visual studio 2019. Алгоритм расчёта местоположения спутника включает в себя уравнение Кеплера. Необходимо протестировать программу.

A screenshot of a Qt Creator console window. The title bar shows the file path 'C:\Qt\Qt5.10.1\Tools\QtCreator\bin\qtcreator\_process\_stub.exe'. The console output displays three lines of text: 'Maximum delta X = 1.62981e-007', 'Maximum delta Y = 1.00583e-007', and 'Maximum delta Z = 4.47035e-008'. The rest of the console area is black.

```
C:\Qt\Qt5.10.1\Tools\QtCreator\bin\qtcreator_process_stub.exe
Maximum delta X = 1.62981e-007
Maximum delta Y = 1.00583e-007
Maximum delta Z = 4.47035e-008
```

Рисунок 14 – Результаты теста программы.

### Заключение.

При выполнении данного курсового проекта была построена орбита заданного спутника Beidou, получена траектория движения этого спутника на заданный промежуток времени.

Так же были изучены все этапы нахождения местоположения спутника и реализованы в Matlab.

Последним этапом курсового проекта была разработка функции расчёта местоположения на языке программирования C++ и сравнение результатов с программой из Matlab.

## Приложение.

```
#include <iostream>
#include <math.h>

using namespace std;

int main()
{
    // Эфемериды
    double SatNum = 6;
    double toe = 241200;
    double Crs = -9.2875000000000000e+01;
    double Dn = 8.71107710704449589e-13;
    double M0 = 2.32726368913121773e+00;
    double Cuc = -2.62074172496795654e-06;
    double e = 1.05765871703624725e-02;
    double Cus = 2.34702602028846741e-05;
    double sqrtA = 6.49287138557434082e+03;
    double Cic = -1.12690031528472900e-07;
    double Omega0 = 6.63759799965142852e-01;
    double Cis = 3.25962901115417480e-09;
    double i0 = 9.46015118241178121e-01;
    double Crc = -4.82140625000000000e+02;
    double omega = -2.20504767262928070e+00;
    double OmegaDot = -1.773288508356074e-12;
    double iDot = -2.00008331149807446e-14;
    double Tgd = 9.7500000000000000e+05;
    double toc = 2.1960000000000000e+08;
    double af2 = 1.48307593848345250e-22;
    double af1 = 5.06794606280891458e-12;
    double af0 = 3.53220045566558838e-01;
    double URA = 0;
    double IODE = 257;
    double IODC = 1;
    double codeL2 = 0;
    double L2P = 0;
    double WN = 789;

    // Значения констант
    double mu = 3.986004418e14; // гравитационная постоянная
    double omega_e = 7.2921151467e-5; // скорость вращения

    // Временной промежуток
    double begin_time = (24*2+18-3)*60*60; // время начала 8:00 по МСК 16
    // февраля
    double end_time = (24*3+6-3)*60*60; // время окончания 6:00 по МСК 17
    // февраля

    // Длина временного промежутка
    double step_time = 1;
    int t_len = 1 + (end_time - begin_time) / step_time;

    double *X0 = new double[t_len];
    double *Y0 = new double[t_len];
    double *Z0 = new double[t_len];

    // Большая полуось
    double A = pow(sqrtA, 2);

    // Среднее движение
    double n0 = sqrt(mu/pow(A, 3));
```

```

double n = n0+Dn;

for (int t_ = begin_time, k = 0; t_ <= end_time; t_ += step_time, k++)
{
    double t = t_ - toe;

    // Vremya
    if (t > 302400) {
        t -= 604800;
    }
    if (t < -302400) {
        t += 604800;
    }

    // Средняя аномалия
    double M = M0+n*t;

    // Решение уравнения Кеплера
    double E = M;
    double E_old = M+1;
    double epsilon = 1e-6;

    while (fabs(E-E_old) > epsilon) {
        E_old = E;
        E = M+e*sin(E);
    }

    // Истинная аномалия
    double nu = atan2(sqrt(1-pow(e,2))*sin(E),cos(E)-e);

    // Коэффициент коррекции
    double cos_correction = cos(2*(omega+nu));
    double sin_correction = sin(2*(omega+nu));

    // Аргумент широты
    double u = omega+nu+Cuc*cos_correction+Cus*sin_correction;

    // Радиус
    double r = A*(1-
e*cos(E))+Crc*cos_correction+Crs*sin_correction;

    // Наклон
    double i = i0+iDot*t+Cic*cos_correction+Cis*sin_correction;

    // Долгота восходящего угла
    double lambda = Omega0+(OmegaDot-omega_e)*t-omega_e*toe;

    // Положение на орбите
    double x = r*cos(u);
    double y = r*sin(u);

    // Координаты
    X0[k] = x*cos(lambda)-y*cos(i)*sin(lambda);
    Y0[k] = x*sin(lambda)+y*cos(i)*cos(lambda);
    Z0[k] = y*sin(i);
}

// Считываем значения из матлаба
double *X0m = new double[t_len];
double *Y0m = new double[t_len];
double *Z0m = new double[t_len];

```

```

FILE *file;
if ((file = fopen("../data_matlab.txt", "rb+")) == NULL) {
    printf("Cannot open file.\n");
}
else {
    for (int i = 0; i < t_len; i++) {
        fscanf(file, "%le %le %le\n", &X0m[i], &Y0m[i],
&Z0m[i]);
    }
    fclose(file);
}

double deltaX, deltaY, deltaZ;
double maxDeltaX = 0, maxDeltaY = 0, maxDeltaZ = 0;

// Сравниваем значения
for (int i = 0; i < t_len; i++) {
    // Определение максимальной разницы по X
    deltaX = fabs(X0[i] - X0m[i]);
    if (deltaX > maxDeltaX)
        maxDeltaX = deltaX;

    // Определение максимальной разницы по Y
    deltaY = fabs(Y0[i] - Y0m[i]);
    if (deltaY > maxDeltaY)
        maxDeltaY = deltaY;

    // Определение максимальной разницы по Z
    deltaZ = fabs(Z0[i] - Z0m[i]);
    if (deltaZ > maxDeltaZ)
        maxDeltaZ = deltaZ;
}

cout << "Maximum delta X = " << maxDeltaX << endl;
cout << "Maximum delta Y = " << maxDeltaY << endl;
cout << "Maximum delta Z = " << maxDeltaZ << endl;

return 0;
}

```