

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 курсовой  
работы:**

Разработка функции расчёта положения  
спутника Beidou

**СТУДЕНТ**

 / Серов К.М. /  
(подпись) (Фамилия и инициалы)

Группа ЭР-15-16  
(номер учебной группы)

**ПРОМЕЖУТОЧНАЯ АТТЕСТАЦИЯ ПО  
КУРСОВОЙ РАБОТЕ**

(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**

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 Си++, позволяющую рассчитывать положение спутника Beidou по его эфемеридам.

**Исходные данные:** PRN спутника Beidou - C21

## Этап 1. Использование сторонних средств

### Описание этапа

На первом этапе подготовим вспомогательные данные для разработки: эфемериды и оценки положения спутника от сторонних сервисов (чтобы было с чем сравниваться на след. этапах).

На крыше корпуса Е МЭИ установлена трехдиапазонная антенна Narxон НХ-СХ601А. 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.

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

Определим какому спутнику соответствует выданный PRN спутника.

№	Спутник	PRN	Дата (UTC)	Ракета	NSSDC ID	SCN	Орбита	Статус	Система
29	Бэйдоу-3 М6	C21	12.02.2018 05:10	CZ-3В/YZ-1	2018-018В	43208	СОО, ~21 500 км	действующий	Бэйдоу-3

Рисунок 1 — Состав орбитальной группировки космической навигационной системы Beidou на 10 марта 2020 года [1]

Номер спутника C21 соответствует спутнику Beidou – 3 М6, номер по спутниковому каталогу НОРАД (или SCN) равен 43208.

Проверим эту информацию, для этого воспользуемся данными о состоянии космических аппаратов Beidou на 02.03.21 из «Информационно-аналитического центра координатно-временного и навигационного обеспечения» [2].

PRN	НОРАД	Тип КА	Тип системы	Дата запуска	Факт. суш. (дней)	Примечание
C21	43208	МЕО-3	BDS-3	12.02.18	1114	Используется по ЦН

Рисунок 2 — Данные о состоянии космических аппаратов Beidou на 02.03.21 (источник «Информационно-аналитического центра координатно-временного и навигационного обеспечения»)

Информация с рисунков 1 и 2 совпадает.

### 1.1. Определение формы орбиты и положения спутника

Определим формы орбиты и положения спутника на ней на начало рассматриваемого интервала времени по данным сервиса Celestrak: общий вид + положение спутника на 18:00 МСК 16 февраля 2021, так, чтобы было видно подспутниковую точку и время.

18:00 по МСК соответствует 15:00 по UTC (UTC +3). Так как сервис Celestrak работает в формате времени UTC, установим время 15:00 UTC 16 февраля 2021.

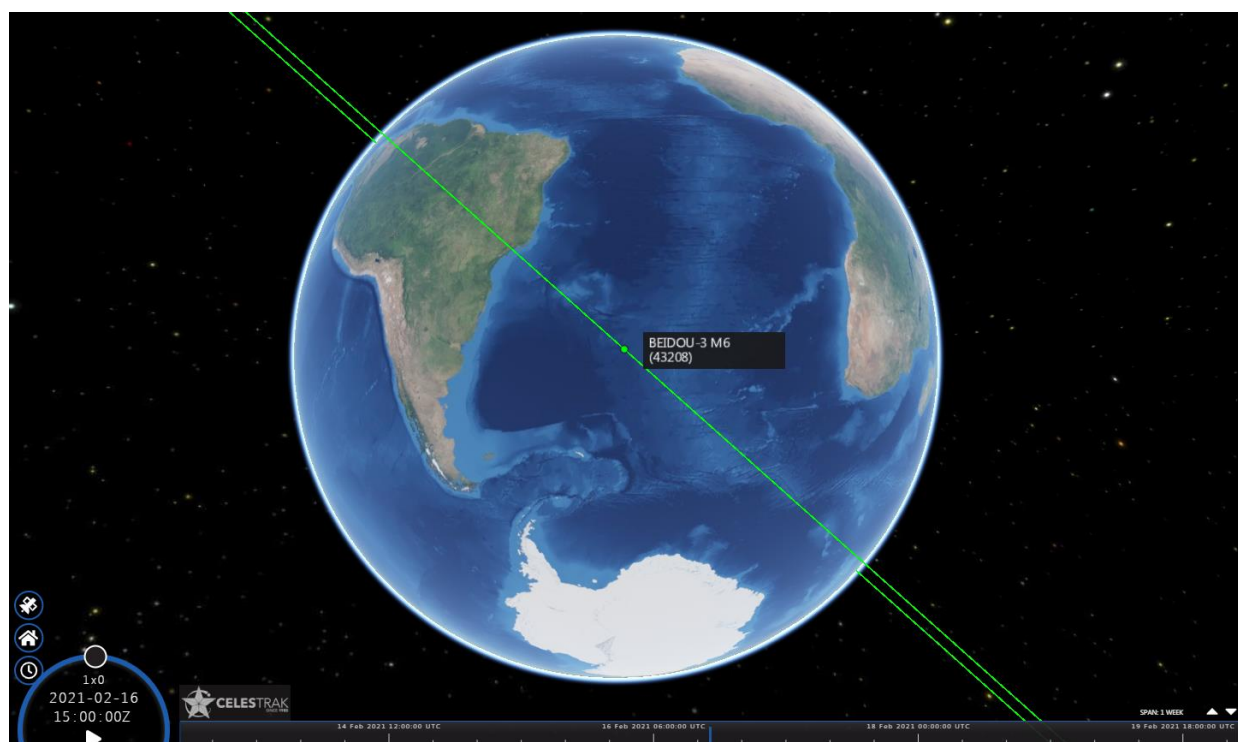


Рисунок 3 — Модель сервиса Celestrak, видно подспутниковую точку и время

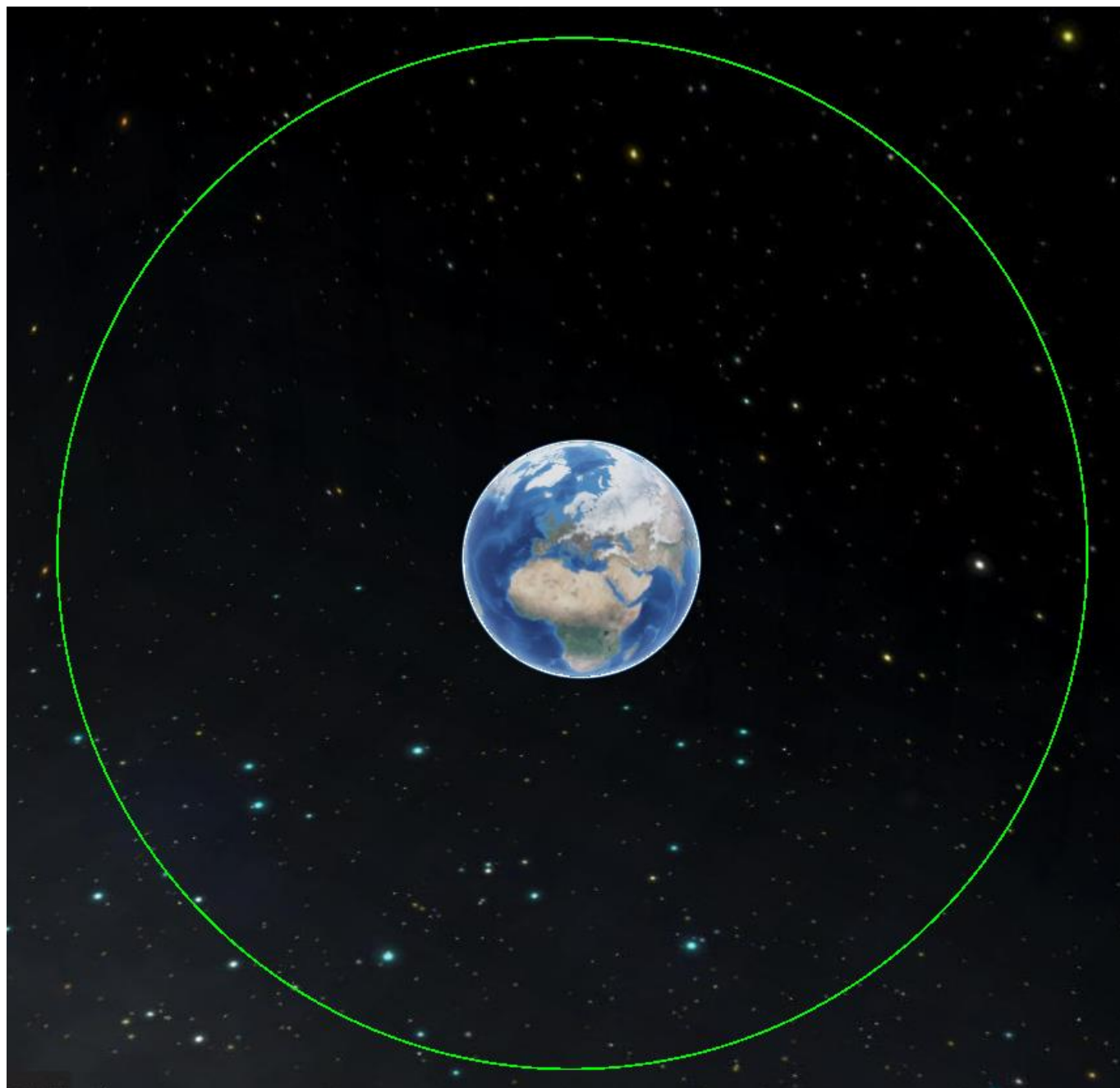


Рисунок 4 — Модель сервиса Celestrak, общий вид орбиты спутника

## 1.2. Расчет графика угла места собственного спутника от времени

Рассчитаем график угла места собственного спутника от времени по данным Trimble GNSS Planning Online на интервал времени с 18:00 МСК 16 февраля до 06:00 МСК 17 февраля 2021 года.

Установили приблизительные координаты местоположения антенны и границы времени, также выбрали конкретный, интересующий нас, спутник C21.

Рисунок 5 — Экран настроек Trimble GNSS Planning Online

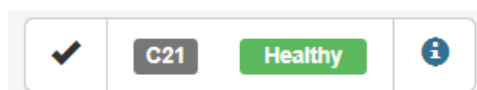


Рисунок 6 — Выбор собственного спутника

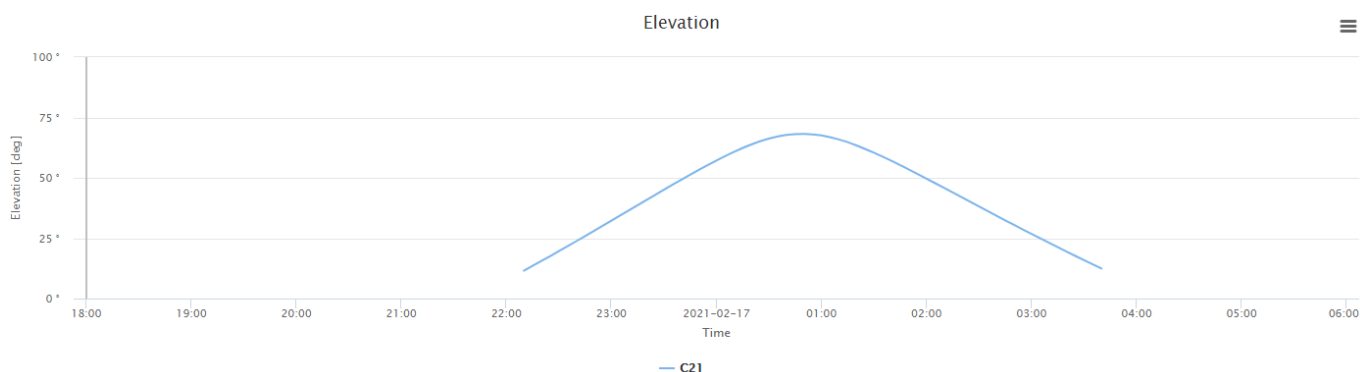


Рисунок 7 — График угла места спутника C21 от времени

По рисунку 6 видно, что спутник находился в зоне видимости в промежутке времени с 22:10 до 3:50.

### 1.3. Расчет диаграммы угла места и азимута спутника

Рассчитаем диаграммы угла места и азимута спутника (SkyView, он же SkyPlot) по данным Trimble GNSS Planning Online на интервал времени с 18:00 МСК 16 февраля до 06:00 МСК 17 февраля 2021 года.

Зафиксируем моменты появления спутника в зоне видимости и его исчезновения, пронаблюдаем траекторию движения спутника.

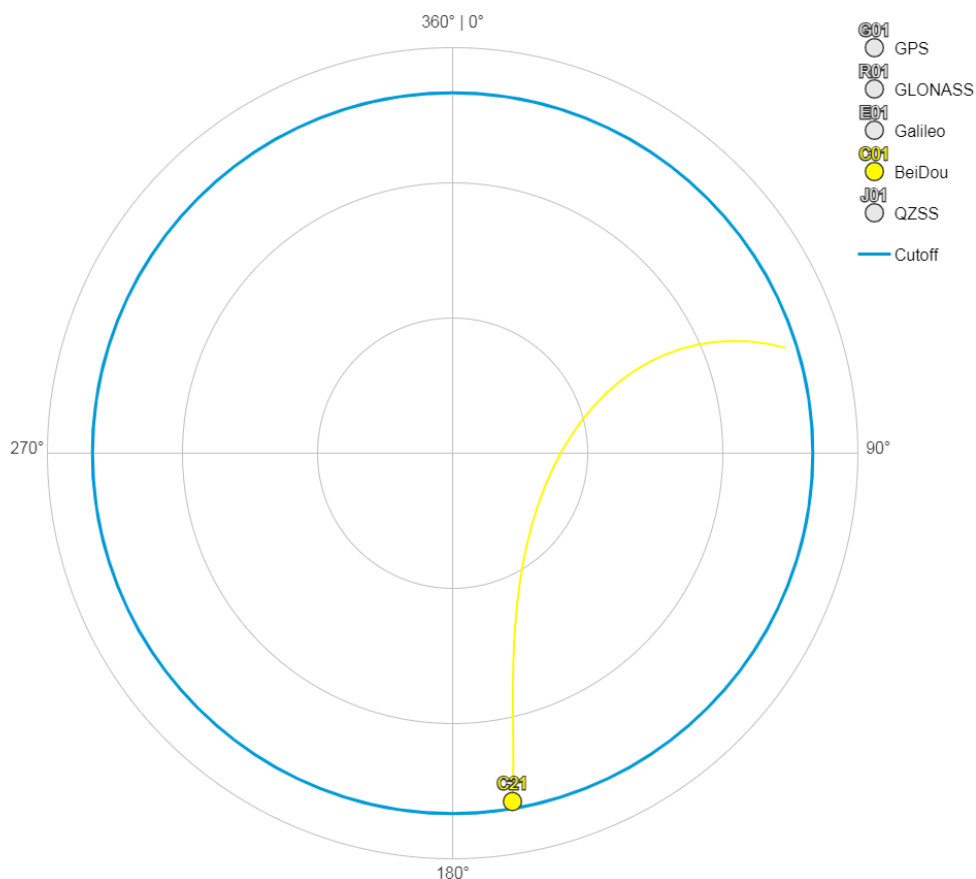


Рисунок 8 — Момент появления спутника в зоне видимости (Время: 2021-02-16 22:10 UTC +03:00)

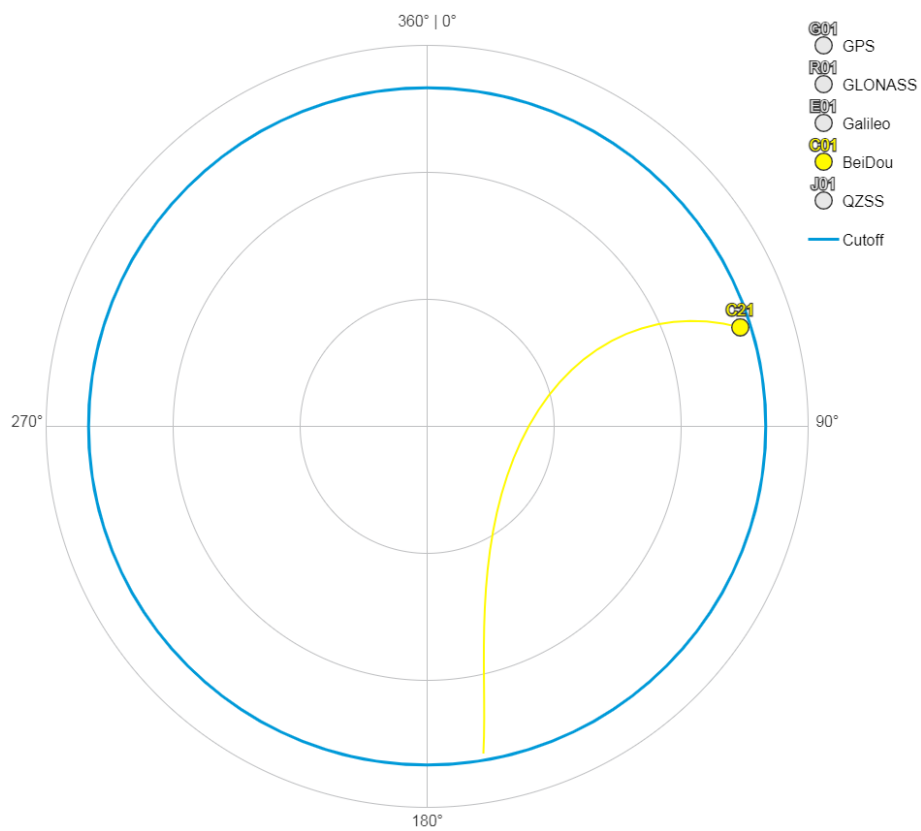


Рисунок 9 — Момент перед исчезновением спутника из зоны видимости (Время:  
2021-02-17 03:40 UTC +03:00)



## 1.4. Формирование списка и описание параметров

Сформируем список и описание параметров, входящих в состав эфемерид в сигнале B1I Beidou. Сформируем список эфемерид [3]:

Таблица 1 — Описание параметров эфемерид

Параметры	Определение
$t_{oe}$	Исходное время эфемерид
$\sqrt{A}$	Квадратный корень из большой полуоси
$e$	Эксцентриситет
$\omega$	Аргумент перигея
$\Delta n$	Среднее отклонение движения от расчетного значения
$M_0$	Средняя аномалия в исходное время
$\Omega_0$	Долгота восходящего узла орбитальной плоскости, вычисленная по исходному времени
$\dot{\Omega}$	Скорость прямого восхождения
$i_0$	Угол наклона в исходное время
$IDOT$	Скорость угла наклона
$C_{uc}$	Амплитуда косинусного гармонического корректирующего члена к аргументу широты
$C_{us}$	Амплитуда синусного гармонического корректирующего члена к аргументу широты
$C_{rc}$	Амплитуда косинусного гармонического корректирующего члена к радиусу орбиты
$C_{rs}$	Амплитуда синусного гармонического корректирующего члена к радиусу орбиты
$C_{ic}$	Амплитуда косинусного гармонического корректирующего члена к углу наклона
$C_{is}$	Амплитуда синусного гармонического корректирующего члена к углу наклона

Таблица 2 — Значения параметров эфемерид спутника C21

Параметр	Обозначение	Значение	Размерность
SatNum	PRN	21	-
toe	$t_{oe}$	241200000.000	мс
Crs	$C_{rc}$	-6.6843750000000000e+01	рад
Dn	$\Delta n$	3.86908994426393704e-12	рад/мс
M0	$M_0$	3.86908994426393704e-12	рад
Cuc	$C_{uc}$	-3.16184014081954956e-06	рад
e	$e$	6.40757265500724316e-04	-
Cus	$C_{us}$	6.28596171736717224e-06	рад
sqrtA	$\sqrt{A}$	5.28262227249145508e+03	м <sup>1/2</sup>
Cic	$C_{ic}$	1.76951289176940918e-08	рад
Omega0	$\Omega_0$	-2.80692060956725220e-01	рад
Cis	$C_{is}$	-6.79865479469299316e-08	рад
i0	$i_0$	9.64946480705556331e-01	рад
Crc	$C_{rc}$	2.33203125000000000e+02	рад
omega	$\omega$	-9.96705605657731697e-01	рад
OmegaDot	$\dot{\Omega}$	-6.91350226083361201e-12	рад/мс
iDot	$\dot{I}$	-1.40362989539061277e-13	рад/с
Tgd	$T_{GD}$	1.4100000000000000e+05	мс
toc	$t_{oc}$	2.4120000000000000e+08	мс
af2	$a_{f2}$	0.0000000000000000e+00	мс/мс <sup>2</sup>
af1	$a_{f1}$	-1.83684178978182899e-11	мс/мс
af0	$a_{f0}$	-8.41504871845245361e-01	мс
URA	-	0	-
IODE	-	257	-
IODC	-	0	-
codeL2	-	0	-
L2P	-	0	-
WN	-	789	-

## **Этап 2. Моделирование**

### **Описание этапа**

Эфемериды - параметры некоторой модели движения спутника. В разных ГНСС эти модели разные, а значит отличается и формат эфемерид, и алгоритмы расчета положения спутника.

Одна из самых простых и удобных моделей - в системе GPS. Beidou наследует данную модель.

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, соответствующим номеру студента по списку. 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.

Приведём таблицу эфемеридов, используемых в данном этапе:

Таблица 3 — Значения параметров эфемерид спутника C21 для 3 этапа

Параметр	Обозначение	Значение	Размерность
SatNum	PRN	21	-
toe	$t_{oe}$	241200000.000	мс
Crs	$C_{rc}$	-6.6843750000000000e+01	рад
Dn	$\Delta n$	3.86908994426393704e-12	рад/мс
M0	$M_0$	3.86908994426393704e-12	рад
Cuc	$C_{uc}$	-3.16184014081954956e-06	рад
e	$e$	6.40757265500724316e-04	-
Cus	$C_{us}$	6.28596171736717224e-06	рад
sqrtA	$\sqrt{A}$	5.28262227249145508e+03	м <sup>1/2</sup>
Cic	$C_{ic}$	1.76951289176940918e-08	рад
Omega0	$\Omega_0$	-2.80692060956725220e-01	рад
Cis	$C_{is}$	-6.79865479469299316e-08	рад
i0	$i_0$	9.64946480705556331e-01	рад
Crc	$C_{rc}$	2.33203125000000000e+02	рад
omega	$\omega$	-9.96705605657731697e-01	рад
OmegaDot	$\dot{\Omega}$	-6.91350226083361201e-12	рад/мс
iDot	$\dot{IDOT}$	-1.40362989539061277e-13	рад/с

## 2.1. Алгоритм расчёта

В таблице 3 приведены параметры эфемерид вещания Beidou для вычисления их спутниковых координат в любую эпоху наблюдения. Эти параметры периодически обновляются и не должны использоваться вне установленного времени (около четырех часов), поскольку ошибка экстраполяции экспоненциально растет за пределами срока ее действия.

Для вычисления спутниковых координат, по навигационному сообщению, необходимо использовать следующий алгоритм.

- Вычислим время  $t_k$  из исходного времени эфемерид  $t_{oe}$  ( $t$  и  $t_{oe}$  выражаются в секундах в неделе GPS):

$$t_k = t - t_{oe}$$

Если  $t_k > 302400$  секунд, вычитаем 604 800 секунд из  $t_k$ . Если  $t_k < -302400$  секунд, то добавляем 604800 секунд.

- Вычислим среднюю аномалию для  $t_k$ ,

$$M_k = M_0 + \left( \frac{\sqrt{\mu}}{\sqrt{a^3}} + \Delta n \right) t_k$$

- Решим (итеративно) уравнение Кеплера для аномалии эксцентриситета  $E_k$ :

$$M_k = E_k - e \sin(E_k)$$

- Вычислим истинную аномалию  $\nu_k$ :

$$\nu_k = \arctan \left( \frac{\sqrt{1-e^2} \sin(E_k)}{\cos(E_k - e)} \right)$$

- Вычислим аргумент широты  $u_k$  из аргумента перигея  $\omega$ , истинной аномалии  $\nu_k$  и поправок  $C_{uc}$  и  $C_{us}$ :

$$u_k = \omega + \nu_k + C_{uc} \cos(2(\omega + \nu_k)) + C_{us} \sin(2(\omega + \nu_k))$$

- Вычислим радиально расстояние  $r_k$  с учётом поправок  $C_{rc}$  и  $C_{rs}$ :

$$r_k = a(1 - e \cos(E_k)) + C_{rc} \cos(2(\omega + \nu_k)) + C_{rs} \sin(2(\omega + \nu_k))$$

- Вычислим наклон  $i_k$  орбитальной плоскости по наклону  $i_o$  в исходное время  $t_{oe}$  и поправкам  $C_{ic}$  и  $C_{is}$ :

$$i_k = i_o + IDOT \cdot t_k + C_{ic} \cos(2(\omega + \nu_k)) + C_{is} \sin(2(\omega + \nu_k))$$

- Вычислим долготу восходящего узла  $\lambda_k$  (относительно Гринвича). В этом расчете используется долгота восходящего узла по исходному времени ( $\Omega_0$ ), поправка от видимого изменения звездного времени по Гринвичу между началом недели и опорным временем  $t_k = t - t_{oe}$ , а также изменение долготы восходящего узла по сравнению с исходным моментом  $t_{oe}$ :

$$\lambda_k = \Omega_0 + (\dot{\Omega} - \omega_E)t_k - \omega_E t_{oe}$$

$\omega_E = 7.2921151467 \cdot 10^{-5}$  - скорость вращения Земли

- Расчетные положения спутников в орбитальной плоскости:

$$\begin{cases} x_k = r_k \cos(u_k) \\ y_k = r_k \sin(u_k) \end{cases}$$

- Расчетные GEO / MEO / IGSO координаты спутника в BDCS:

$$\begin{cases} X_k = x_k \cos(\lambda_k) - y_k \cos(i_k) \sin(\lambda_k) \\ Y_k = x_k \sin(\lambda_k) + y_k \cos(i_k) \cos(\lambda_k) \\ Z_k = y_k \sin(i_k) \end{cases}$$

Построим трехмерный график множества положений спутника Beidou с системным номером 21:

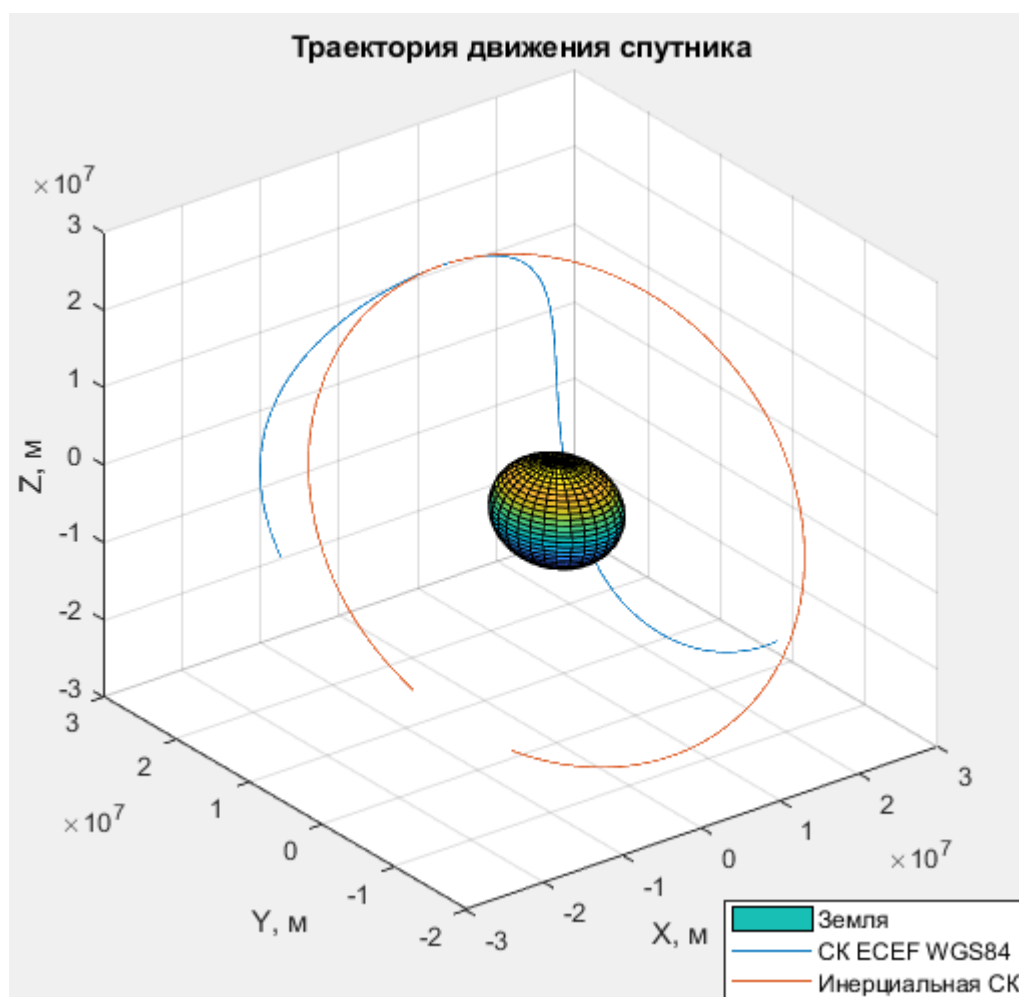


Рисунок 10 – Траектория движения спутника

Построим SkyView за интервал времени с 18:00 МСК 16 февраля до 06:00 МСК 17 февраля 2021 года:

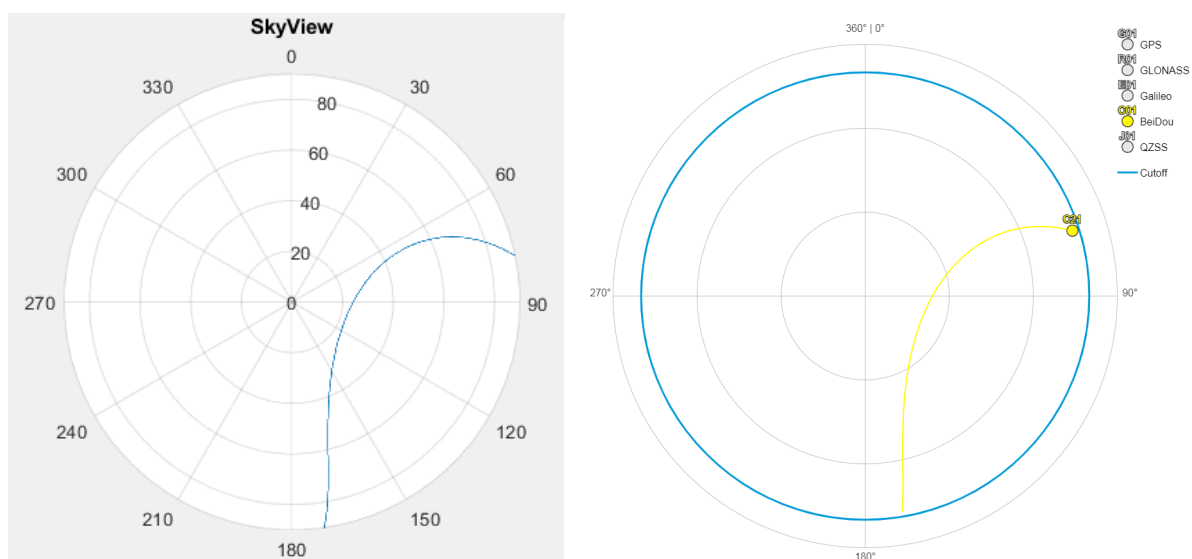


Рисунок 11 — Модель SkyView (слева), SkyView с 1 этапа (справа)

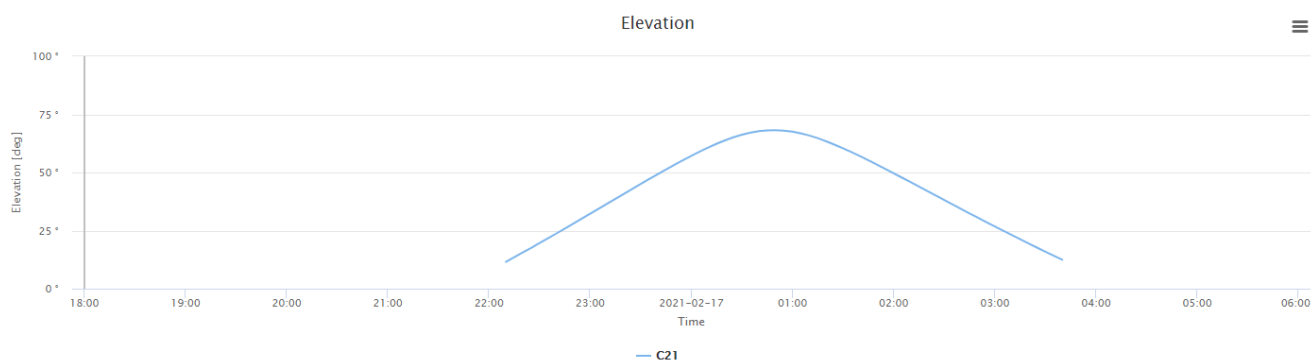
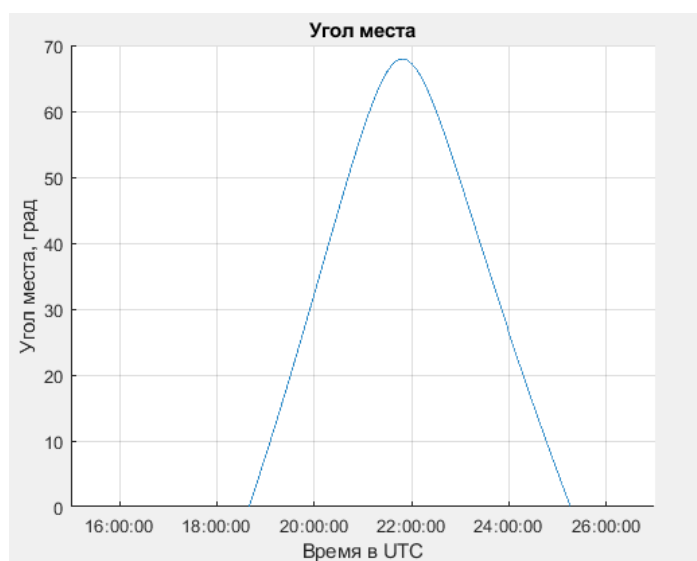


Рисунок 12 — График угла места 2 этапа (сверху), 1 этапа (снизу)



При сравнении рисунков, полученных на 2 этапе, с рисунками, полученными на 1 этапе, видно, что они практически совпадают. Однако, имеется погрешность, связанная с тем, что мы используем параметры эфемерид в установленном промежутке времени (около четырех часов).

Код программы представлен в приложении 1.

### Этап 3. Реализация

Требуется разработать на языке C/C++ функцию расчета положения спутника Beidou 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.

Функция расчета положения спутника в Matlab/Python относительно проста, т.к. доступны библиотеки линейной алгебры и решения уравнений. Но при разработке встраиваемого ПО приходится сохранять лицензионную частоту, минимизировать вычислительную нагрузку и затраты памяти. Поэтому отобразить модель из Matlab/Python в прошивку приемника дословно, как правило, не получается. В рассматриваемом примере потребуется, как минимум, выполнить свою реализацию решения трансцендентного уравнения.

Программный модуль должен сопровождаться unit-тестами под check:

- Тесты функции решения уравнения Кеплера
- Тест расчетного положения спутника в сравнении с Matlab/Python с шагом 0.1 секунды.

Во время второго теста должно вычисляться и выводиться средняя длительность исполнения функции. Допускается использовать одни и те же эфемериды на весь рассматриваемый интервал (как на предыдущем этапе).

Требуется провести проверку на утечки памяти с помощью утилиты valgrind.

### 3.1. Исходники программы.

Функция была реализована на языке программирования C/C++ в среде разработки Microsoft Visual Studio. Так как функция расчёта относительно проста, весь расчёт выполняется внутри главной функции программы `main()`. Функция, помимо вычислений координат, так же считывает координаты из файла, сравнивает их с рассчитанными, находит максимальную разницу, а также вычисляет общее время выполнения.

Код реализации представлен в приложении 2.

### 3.2. Результат реализации



Рисунок 13 — Результат реализации

После реализации проведём профилирование, которое представляет из себя сбор характеристик программы во время её выполнения.

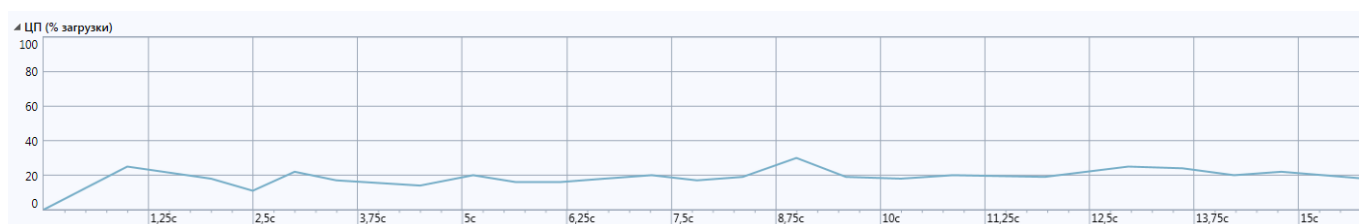


Рисунок 14 — График загрузки ЦП

Информация о критическом пути:

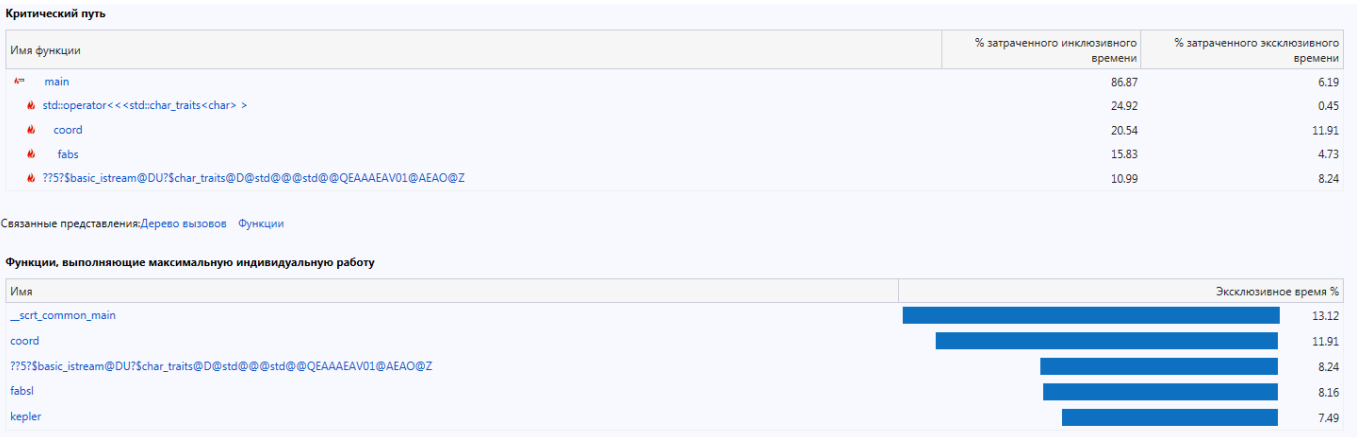


Рисунок 15 — Результаты инструментирования

Результаты подробного профилирования приведены в приложении 3.

Был получен график использования памяти при сеансе диагностики.

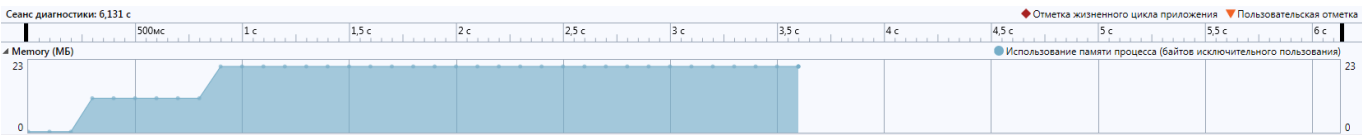


Рисунок 16 — Сеанс диагностики по использованию памяти

Также была проведена отладка



Рисунок 17 — Сеанс отладки

## **ЗАКЛЮЧЕНИЕ**

В ходе выполнения данной курсовой работы были получены эфемериды и оценки положения спутника от сторонних сервисов. Затем на языке Matlab была реализована функция расчета положения спутника Beidou на заданный момент по шкале времени UTC. Также эта функция была реализована на языке C++, при помощи которой производилось сравнение полученных результатов. Все результаты полученные в ходе работы соответствуют ожиданиям.

```

clear all;
clc;
close all;
%% Эфемериды
SatNum = 21;
toe = 241200000.000* 10^-3;
Crs = -6.6843750000000000e+01;
Dn = 3.86908994426393704e-12;
M0 = 7.11567786397328206e-01;
Cuc = -3.16184014081954956e-06;
e = 6.40757265500724316e-04;
Cus = 6.28596171736717224e-06;
sqrtA = 5.28262227249145508e+03 ;
Cic = 1.76951289176940918e-08;
Omega0 = -2.80692060956725220e-01 ;
Cis = -6.79865479469299316e-08;
i0 = 9.64946480705556331e-01 ;
Crc = 2.33203125000000000e+02 ;
omega = -9.96705605657731697e-01 ;
OmegaDot = -6.91350226083361201e-12;
iDot = -1.40362989539061277e-13 ;
Tgd =1.4100000000000000e+05 ;
toc = 2.4120000000000000e+08 ;
af2 = 0.0000000000000000e+00;
af1 = -1.83684178978182899e-11 ;
af0 = -8.41504871845245361e-01 ;
URA = 0;
IODE = 257;

IODC = 1;
codeL2 = 0;
L2P = 0;
WN = 789;
%% Константы
mu = 3.986004418e14; % гравитационная постоянная
Omega_E = 7.2921151467e-5; % скорость вращения
%% Расчет
tstart = (24*2 + 18 - 3)*60*60; % время старта 18:00 МСК 16 февраля
tstop = (24*3 + 6 - 3)*60*60; % время окончания 6:00 МСК 17 февраля
% Массив времени
t_arr = tstart:1:tstop;
% Большая полуось
A = sqrtA^2;
% Среднее движение
n0 = sqrt(mu/A^3);
n = n0 + Dn;
for k = 1:length(t_arr)
    % Время
    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) = atan2((sqrt(1 - e^2) * sin(E(k)))/(1 - e*cos(E(k))), (cos(E(k)) - e)/(1 - e*cos(E(k))));

% Коэффициенты коррекции
Phi(k) = omega + nu(k);
cor_cos(k) = cos(2*Phi(k));
cor_sin(k) = sin(2*Phi(k));

% Аргумент широты
delta_u(k) = Phi(k) + Cuc*cor_cos(k) + Cus*cor_sin(k);

% Радиус
delta_r(k) = A * (1 - e * cos(E(k))) + Crc*cor_cos(k) + Crs*cor_sin(k);
% Наклон
delta_i(k) = i0 + iDot * t(k) + Cic*cor_cos(k) + Cis*cor_sin(k);

% Положение на орбите
x = delta_r(k) * cos(delta_u(k));
y = delta_r(k) * sin(delta_u(k));

% Долгота восходящего угла
Omega(k) = Omega0 + (OmegaDot - Omega_E) * t(k) - Omega_E*toe;

% Координаты
coordx(k) = x * cos(Omega(k)) - y * cos(delta_i(k)) * sin(Omega(k));
coordy(k) = x * sin(Omega(k)) + y * cos(delta_i(k)) * cos(Omega(k));
coordz(k) = y * sin(delta_i(k));

%%
coordx1(k) = coordx(k)*cos(Omega(k)) + coordy(k)*sin(Omega(k));
coordy1(k) = - coordx(k)*sin(Omega(k)) + coordy(k)*cos(Omega(k));
coordz1(k) = coordz(k);
end
%% Пересчет координат центра масс НКА в систему координат WGS-84
ppb = 1e-9;
mas = 1e-3/206264.8; % [рад]
MATRIX_WGS_84 = [-3*ppb -353*mas -4*mas;
    353*mas -3*ppb 19*mas;
    4*mas -19*mas -3*ppb];
crd_WGS_84 = [coordx; coordy; coordz];
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.'; % Переход к вектору-строки
%% построение графиков
R_Earth = 6371e3;
[XE,YE,ZE] = sphere(30);
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(coordx1, coordy1, coordz1)
title('Траектория движения спутника', 'FontName', 'Arial')
xlabel('X, м', 'FontName', 'Arial')
ylabel('Y, м', 'FontName', 'Arial')
zlabel('Z, м', 'FontName', 'Arial')
hold off
lgd = legend('Земля', 'СК ЕСЕС WGS84', 'Инерциальная СК');
lgd.FontName = 'Arial';
%% Координаты корпуса Е и их перевод в систему WGS-84
Earth_radius = 6378136;
H = 300; % высота [м]
a = Earth_radius;
B = deg2rad(55.45241346); % широта
N = a/sqrt((1-e^2*(sin(B))^2));
L = deg2rad(37.42114473); % долгота
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), B, L, H, wgs84Ellipsoid, 'radian
s');
    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')
% Угол места
th = hours(t_arr/60/60 - 2*24); % Перевод временной оси в формат hh:mm:ss
figure
grid on
hold on
plot(th, (-teta)*180/pi+90, 'DurationTickFormat', 'hh:mm:ss')
xlim([th(1) th(end)])
title('Угол места', 'FontName', 'Arial')
xlabel('Время в UTC', 'FontName', 'Arial')
ylabel('Угол места, град', 'FontName', 'Arial')

```



```

// Kursovoy_proekt.cpp
//

#include <iostream>
#include <fstream>
#include <cmath>
#include "Kepler.h"
#include "Rasch_coord.h"
using namespace std;
int main()
{
    long double e, mu, Omega_e, toe, A, n0, i0, Omega0, omega, M0, Dn, n, OmegaDot, IDOT, Crs,
    Cuc, Cus, Cic, Cis, Crc;
    long double t;
    e = 6.40757265500724316E-04;
    mu = 3.98600442E+14;
    Omega_e = 7.2921151467E-5;
    toe = 241200000E-3;
    A = pow(5.28262227249145508E+03, 2);
    n0 = pow(mu / (pow(A, 3)), 0.5);
    i0 = 9.64946480705556331e-01;
    Omega0 = -2.80692060956725220e-01;
    omega = -9.96705605657731697e-01;
    M0 = 7.11567786397328206e-01;
    Dn = 3.86908994426393704e-12;
    n = n0 + Dn;
    OmegaDot = -6.91350226083361201e-12;
    IDOT = -1.40362989539061277e-13;
    Crs = -6.68437500000000000e+01;
    Cuc = -3.16184014081954956e-06;
    Cus = 6.28596171736717224e-06;
    Cic = 1.76951289176940918e-08;
    Cis = -6.79865479469299316e-08;
    Crc = 2.33203125000000000e+02;

    long double t_begin = (24 * 2 + 15) * 3600;
    long double t_finish = (24 * 3 + 3) * 3600;
    int N_stop = 432000;
    long double *x = new long double[N_stop];
    long double *y = new long double[N_stop];
    long double *z = new long double[N_stop];
    long double xx, yy, zz;
    int k = 0;
    for (long double i = t_begin; i < t_finish; i+=0.1)
    {
        coord(e, mu, Omega_e, toe,
            A, n0, i0, Omega0, omega, M0,
            Dn, n, OmegaDot, IDOT, Crs, Cuc, Cus,
            Cic, Cis, Crc, i, &xx, &yy, &zz);
        x[k] = xx;
        y[k] = yy;
        z[k] = zz;
        k++;
    }

    ofstream f;
    f.open("Znach.txt");

```

```

k = 0;
for (int i = t_begin; i < t_finish; i += 1)
{
f << "k = " << k << "\t" << "t =" << i << "\t" << x[k] << "\t" << y[k] << "\t" << z[k] << endl;
k++;
}
f.close();

long double* x_e2 = new long double[N_stop];
long double* y_e2 = new long double[N_stop];
long double* z_e2 = new long double[N_stop];
int i = 0;

ifstream fin("cord.txt");
if (!fin.is_open())
    cout << "Файл не открылся!" << endl;
else
{
    while (fin >> x_e2[i] >> y_e2[i] >> z_e2[i])
    {
        i++;
    }
    fin.close();
}

double sumdx = 0;
double sumdy = 0;
double sumdz = 0;
double dxmax = 0;
double dymax = 0;
double dzmax = 0;
for (int i = 0; i < N_stop; i++)
{
    sumdx += (fabs(x[i] - x_e2[i]));
    sumdy += (fabs(y[i] - y_e2[i]));
    sumdz += (fabs(z[i] - z_e2[i]));
}
double raznost_x = sumdx / N_stop;
double raznost_y = sumdy / N_stop;
double raznost_z = sumdz / N_stop;

cout << "raznost_x=" << raznost_x << endl;
cout << "raznost_y=" << raznost_y << endl;
cout << "raznost_z=" << raznost_z << endl;

delete[] x;
delete[] y;
delete[] z;
return 0;
}

```

```

//Rasch_coord.cpp
#include <math.h>
#include "Rasch_coord.h"
#include "Kepler.h"
void coord(long double e, long double mu, long double Omega_e, long double toe,
long double A, long double n0, long double i0, long double Omega0, long double omega, long
double M0,
long double Dn, long double n, long double OmegaDot, long double IDOT, long double Crs,
long double Cuc, long double Cus,
long double Cic, long double Cis, long double Crc, long double t, long double *x, long
double *y, long double *z)
{
    long double tk, Mk, E_k, nu_k, Phi_k, delta_u_k, delta_r_k, delta_i_k, u_k, r_k, i_k;
    long double x_k, y_k, Omega_k, x_k1, y_k1, z_k1;
    tk = t - toe;
    if (tk > 302400)
    {
        tk = 604800 - tk;
    }
    else if (tk < -302400)
    {
        tk = 604800 + tk;
    }
    Mk = M0 + n * tk;
    E_k = kepler(e, Mk);
    nu_k = atan2((sqrt(1 - pow(e, 2)) * sin(E_k)) / (1 - e * cos(E_k)), (cos(E_k) - e) / (1 - e *
cos(E_k)));
    Phi_k = nu_k + omega;
    delta_u_k = Cus * sin(2 * Phi_k) + Cuc * cos(2 * Phi_k);
    delta_r_k = Crs * sin(2 * Phi_k) + Crc * cos(2 * Phi_k);
    delta_i_k = Cis * sin(2 * Phi_k) + Cic * cos(2 * Phi_k);
    u_k = Phi_k + delta_u_k;
    r_k = A * (1 - e * cos(E_k)) + delta_r_k;
    i_k = i0 + delta_i_k + IDOT * tk;
    x_k = r_k * cos(u_k);
    y_k = r_k * sin(u_k);
    Omega_k = Omega0 + (OmegaDot - Omega_e) * tk - Omega_e * toe;
    // earth-fixed
    x_k1 = x_k * cos(Omega_k) - y_k * cos(i_k) * sin(Omega_k);
    y_k1 = x_k * sin(Omega_k) + y_k * cos(i_k) * cos(Omega_k);
    z_k1 = y_k * sin(i_k);
    *x = x_k1;
    *y = y_k1;
    *z = z_k1;
}

```

```

// Kepler.cpp
#include <math.h>
#include "Kepler.h"
long double kepler(long double e, long double Mk)
{
    long double E[] = { 0, 0, 0, 0 };
    E[0] = Mk;
    for (int j = 1; j < 4; j++)
    {
        E[j] = E[j - 1] + (Mk - E[j - 1] + e * sin(E[j - 1])) / (1 - e * cos(E[j - 1]));
        E[j - 1] = E[j];
    }
    return E[3];
}

```

## Приложение 3

▲ Kursovoy_proekt...	0	100,00%	0,00%	0,00	0,00	
▲ mainCRTStartup	1	100,00%	0,00%	42 282 017 719,00	38 099,00	Kursovoy_proekt.exe
▲ _sclr_commo...	1	100,00%	13,12%	42 281 979 620,00	5 549 434 732,00	Kursovoy_proekt.exe
▲ _sclr_comm...	1	86,87%	0,00%	36 732 413 362,00	121 880,00	Kursovoy_proekt.exe
▲ invoke_m...	1	86,87%	0,00%	36 731 528 202,00	39 634,00	Kursovoy_proekt.exe
▲ main	1	86,87%	6,19%	36 731 488 568,00	2 617 194 956,00	Kursovoy_proekt.exe
▲ std::o...	259 203	24,92%	0,45%	40 654,00	732,00	Kursovoy_proekt.exe
▷ std::...	259 203	10,29%	5,25%	16 784,00	8 564,00	Kursovoy_proekt.exe
▷ std::b...	259 203	6,10%	1,94%	9 956,00	3 159,00	Kursovoy_proekt.exe
▷ std::b...	259 203	1,69%	1,67%	2 758,00	2 720,00	Kursovoy_proekt.exe
_Che...	259 203	1,57%	1,57%	2 556,00	2 556,00	Kursovoy_proekt.exe
_RTC...	259 203	1,56%	1,56%	2 546,00	2 546,00	Kursovoy_proekt.exe
?widt...	259 203	1,45%	1,45%	2 367,00	2 367,00	MSVCP140D.dll
_sec...	259 203	1,37%	1,37%	2 229,00	2 229,00	Kursovoy_proekt.exe
▷ std::c...	259 203	0,28%	0,26%	452,00	421,00	Kursovoy_proekt.exe
?setst...	259 203	0,16%	0,16%	264,00	264,00	MSVCP140D.dll
?sput...	3	0,00%	0,00%	544 354,00	544 354,00	MSVCP140D.dll
?widt...	3	0,00%	0,00%	570,00	570,00	MSVCP140D.dll
?flags...	3	0,00%	0,00%	414,00	414,00	MSVCP140D.dll
?rdbu...	3	0,00%	0,00%	278,00	278,00	MSVCP140D.dll
▷ coord	432 000	20,54%	11,91%	20 102,00	11 653,00	Kursovoy_proekt.exe
▷ fabs	1 296 000	15,83%	4,73%	5 164,00	1 544,00	Kursovoy_proekt.exe
▷ ??5?\$ba...	129 606	10,99%	8,24%	35 857,00	26 881,00	MSVCP140D.dll
▷ ??6?\$ba...	43 203	4,54%	0,03%	44 438,00	336,00	MSVCP140D.dll
▷ ??6?\$ba...	86 400	1,80%	1,66%	8 787,00	8 130,00	MSVCP140D.dll
▷ std::basi...	1	1,34%	0,00%	564 980 709,00	8 057,00	Kursovoy_proekt.exe
▷ ??6?\$ba...	129 600	0,50%	0,29%	1 630,00	954,00	MSVCP140D.dll
▷ operato...	6	0,13%	0,00%	9 498 369,00	537,00	Kursovoy_proekt.exe
▷ operato...	3	0,04%	0,00%	5 411 918,00	1 279,00	Kursovoy_proekt.exe
??Bios_...	43 202	0,03%	0,03%	298,00	298,00	MSVCP140D.dll
??6?\$ba...	3	0,02%	0,02%	2 374 636,00	2 374 636,00	MSVCP140D.dll
▷ std::basi...	1	0,00%	0,00%	1 880 333,00	2 612,00	Kursovoy_proekt.exe
▷ std::basi...	1	0,00%	0,00%	258 326,00	1 935,00	Kursovoy_proekt.exe
▷ std::basi...	1	0,00%	0,00%	227 344,00	65 181,00	Kursovoy_proekt.exe
▷ pow	1	0,00%	0,00%	119 606,00	31 828,00	Kursovoy_proekt.exe

▷ pow	1	0,00%	0,00%	119 606,00	31 828,00	Kursovoy_proekt.exe
▷ std::basi...	1	0,00%	0,00%	93 770,00	64 148,00	Kursovoy_proekt.exe
_Check...	1	0,00%	0,00%	25 474,00	25 474,00	Kursovoy_proekt.exe
▷ std::basi...	1	0,00%	0,00%	18 703,00	6 639,00	Kursovoy_proekt.exe
▷ std::basi...	1	0,00%	0,00%	17 334,00	11 655,00	Kursovoy_proekt.exe
▷ pow<lo...	1	0,00%	0,00%	2 182,00	425,00	Kursovoy_proekt.exe
_RTC_C...	1	0,00%	0,00%	1 943,00	1 943,00	Kursovoy_proekt.exe
_securi...	1	0,00%	0,00%	1 545,00	1 545,00	Kursovoy_proekt.exe
▷ std::basi...	1	0,00%	0,00%	1 259,00	854,00	Kursovoy_proekt.exe
std::basi...	1	0,00%	0,00%	549,00	549,00	Kursovoy_proekt.exe
std::basi...	1	0,00%	0,00%	542,00	542,00	Kursovoy_proekt.exe
▷ pow	1	0,00%	0,00%	535,00	339,00	Kursovoy_proekt.exe
▷ pre_c_initi...	1	0,00%	0,00%	361 319,00	82 592,00	Kursovoy_proekt.exe
▷ _sclr_initi...	1	0,00%	0,00%	156 357,00	65 446,00	Kursovoy_proekt.exe
pre_cpp_initi...	1	0,00%	0,00%	116 054,00	116 054,00	Kursovoy_proekt.exe
▷ _sclr_is_ma...	1	0,00%	0,00%	69 909,00	65 200,00	Kursovoy_proekt.exe
▷ post_pgo_ini...	1	0,00%	0,00%	35 229,00	170,00	Kursovoy_proekt.exe
▷ std::dynami...	1	0,00%	0,00%	20 176,00	1 472,00	Kursovoy_proekt.exe
▷ std::dynami...	1	0,00%	0,00%	2 177,00	125,00	Kursovoy_proekt.exe
_RTC_Termi...	1	0,00%	0,00%	853,00	853,00	Kursovoy_proekt.exe
▷ _sclr_acquir...	1	0,00%	0,00%	791,00	450,00	Kursovoy_proekt.exe
▷ _sclr_releas...	1	0,00%	0,00%	415,00	322,00	Kursovoy_proekt.exe
▷ _security_init...	1	0,00%	0,00%	131 526,00	89 230,00	Kursovoy_proekt.exe

### **Список литературы и источников**

1. Википедия. Бэйдоу - [https://ru.wikipedia.org/wiki/Бэйдоу#Список\\_спутников](https://ru.wikipedia.org/wiki/Бэйдоу#Список_спутников)
2. «Информационно-аналитический центр координатно-временного и навигационного обеспечения» - <https://www.glonass-iac.ru/BEIDOU/>

3. BeiDou Navigation Satellite System Signal In Space Interface Control Document  
Open Service Signal B1I (Version 3.0) - China Satellite Navigation Office  
February 2019