


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

Институт: *Радиотехники и* Кафедра: *Радиотехнических систем*
электроники

Направление подготовки: 11.05.01 - Радиоэлектронные системы и
комплексы

Наименование курсовой работы: **ОТЧЕТ по курсовой работе**
Разработка модуля расчёта координат спутника
Beidou

СТУДЕНТ

 / Попов М.Г. /
(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 (СРНС) являются самыми точными системами по определению координат потребителя. Они стали важной частью в различных сферах нашей жизни. 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.

ГЛАВА 1. ИСПОЛЬЗОВАНИЕ СТОРОННИХ СРЕДСТВ

1.1. Описание задания

В задание дан номер спутника BEIDOU, в моем варианте – C16, а также бинарный и текстовый файл со значениями эфемерид для различных спутников, полученный от трехдиапазонной антенны Harxon HX-CSX601A, установленной на крыше корпуса Е МЭИ. Она через 50-метровый кабель, сплиттер, bias-tee и усилитель подключена к трем навигационным приемникам:

- Javad Lexon LGDD,
- SwiftNavigation Piksi Multi,
- Clonicus разработки ЛИС 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 с помощью «Информационно-аналитического центра координатно-временного и навигационного обеспечения» [1] номер НОРАД¹ и сравним его с номером из «Википедии» [2]:

¹ НОРАД(SCN) - номер по спутниковому каталогу представляет собой уникальный пятизначный идентификационный номер искусственных спутников Земли.

PRN	НОРАД	Тип КА	Тип системы	Дата запуска	Факт. сущ. (дней)	Примечание
C01	44231	GEO-8	BDS-2	17.05.19	653	Используется по ЦН
C02	38953	GEO-6	BDS-2	25.10.12	3048	Используется по ЦН
C03	41586	GEO-7	BDS-2	12.06.16	1722	Используется по ЦН
C04	37210	GEO-4	BDS-2	01.11.10	3772	Используется по ЦН
C05	38091	GEO-5	BDS-2	25.02.12	3291	Используется по ЦН
C06	36828	IGSO-1	BDS-2	01.08.10	3864	Используется по ЦН
C07	37256	IGSO-2	BDS-2	18.12.10	3725	Используется по ЦН
C08	37384	IGSO-3	BDS-2	10.04.11	3612	Используется по ЦН
C09	37763	IGSO-4	BDS-2	27.07.11	3504	Используется по ЦН
C10	37948	IGSO-5	BDS-2	02.12.11	3376	Используется по ЦН
C11	38250	MEO-3	BDS-2	30.04.12	3226	Используется по ЦН
C12	38251	MEO-4	BDS-2	30.04.12	3226	Используется по ЦН
C13	41434	IGSO-6	BDS-2	30.03.16	1796	Используется по ЦН
C14	38775	MEO-6	BDS-2	19.09.12	3084	Используется по ЦН
C16	43539	IGSO-7	BDS-2	10.07.18	964	Используется по ЦН
C19	43001	MEO-1	BDS-3	05.11.17	1211	Используется по ЦН
C20	43002	MEO-2	BDS-3	05.11.17	1211	Используется по ЦН
C21	43208	MEO-3	BDS-3	12.02.18	1112	Используется по ЦН
C22	43207	MEO-4	BDS-3	12.02.18	1112	Используется по ЦН
C23	43581	MEO-5	BDS-3	29.07.18	945	Используется по ЦН
C24	43582	MEO-6	BDS-3	29.07.18	945	Используется по ЦН

Рисунок 1 – Состав и состояние системы BEIDOU с «Информационно-аналитического центра координатно-временного и навигационного обеспечения»

№	Спутник	PRN	Дата (UTC)	Ракета	NSSDC ID	SCN	Орбита	Статус	Система
28	Бэйдоу-3 M5	C22	12.02.2018 05:10	CZ-3B/YZ-1	2018-018A	43207	COO, ~21 500 км	действующий	Бэйдоу-3
29	Бэйдоу-3 M6	C21			2018-018B	43208	COO, ~21 500 км	действующий	
30	Бэйдоу-3 M7	C29	29.03.2018 17:50	CZ-3B/YZ-1	2018-029A	43245	COO, ~21 500 км	действующий	
31	Бэйдоу-3 M8	C30			2018-029B	43246	COO, ~21 500 км	действующий	
32	Бэйдоу-2 IGSO-7	C16	09.07.2018 20:58	CZ-3A	2018-057A	43539	Геосинхронная, накл. 55°	действующий	Бэйдоу-2
33	Бэйдоу-3 M9	C23	29.07.2018 01:48	CZ-3B/YZ-1	2018-062A	43581	COO, ~21 500 км	действующий	
34	Бэйдоу-3 M10	C24			2018-062B	43582	COO, ~21 500 км	действующий	

Рисунок 2 – Состав и состояние системы BEIDOU с сайта Википедия

Из рисунков 1-2 видно, что номер спутника совпадает и равен 45539, название спутника - «Бэйдоу-2 IGSO-7»

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

Зайдем на официальный сайт CelesTrak [3] и настроим данный сервис для определения формы орбита и положения 16-го спутника на ней

Введем наше название спутника и сверим его по номеру NSSDC ID² и НОРАД (SCN).

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):

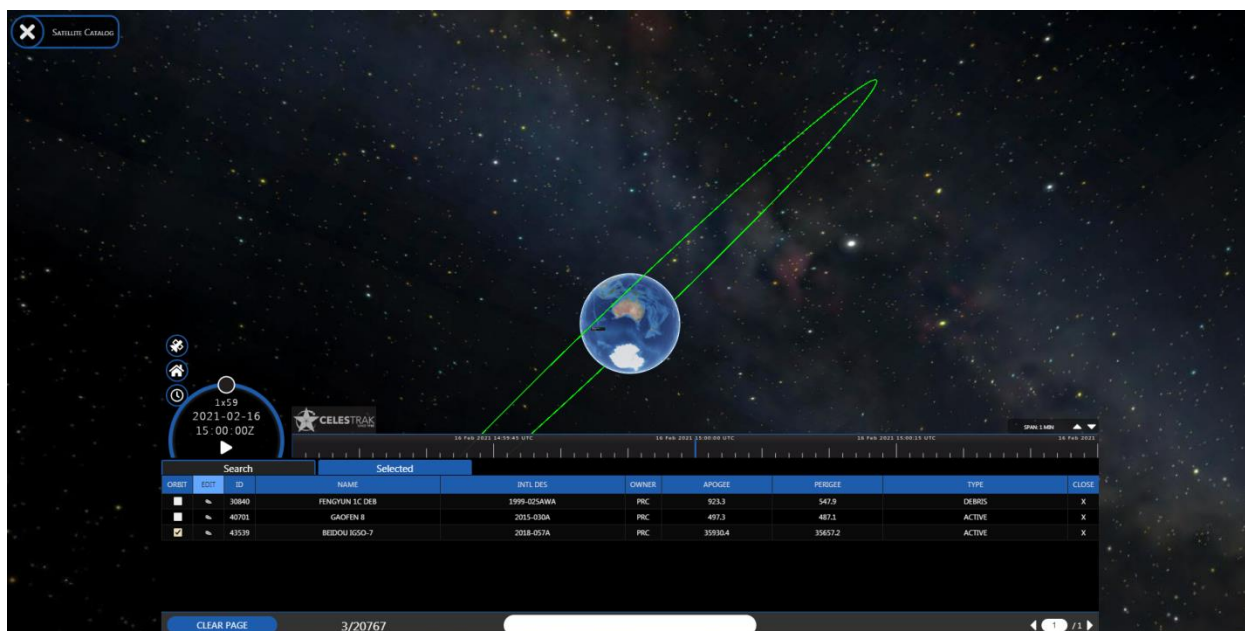


Рисунок 3 – Моделирование с помощью сервиса CelesTrak

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

Настроим для моделирования GNSS Planning Online [4], координаты установим в соответствии с расположением антенны – и они будут

² NSSDC ID - номер полёта представляет собой каталожный номер каждого летающего космического объекта, находящегося на орбите и зарегистрированного в COSPAR (Комитет по космическим исследованиям)

соответствовать значению корпуса Е МЭИ, также начальное время будет соответствовать 18:00, временной пояс будет равен +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:

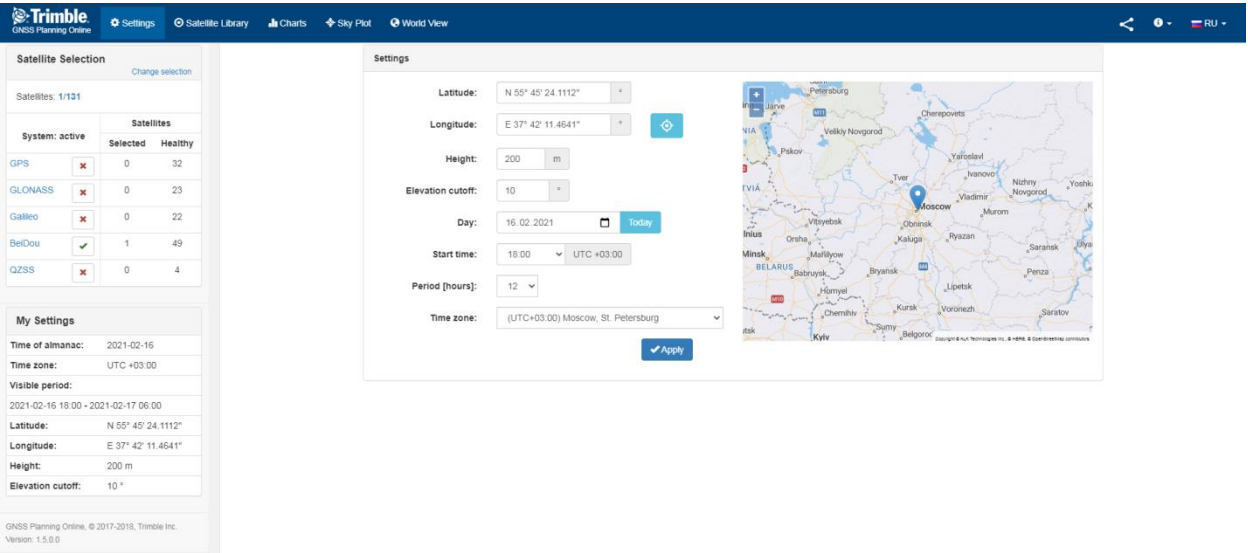


Рисунок 4 – Моделирование с помощью сервиса Trimble GNSS Planning

Далее ограничим количество отображаемых спутников и оставим в моделирование только нужный нам спутник – С16:

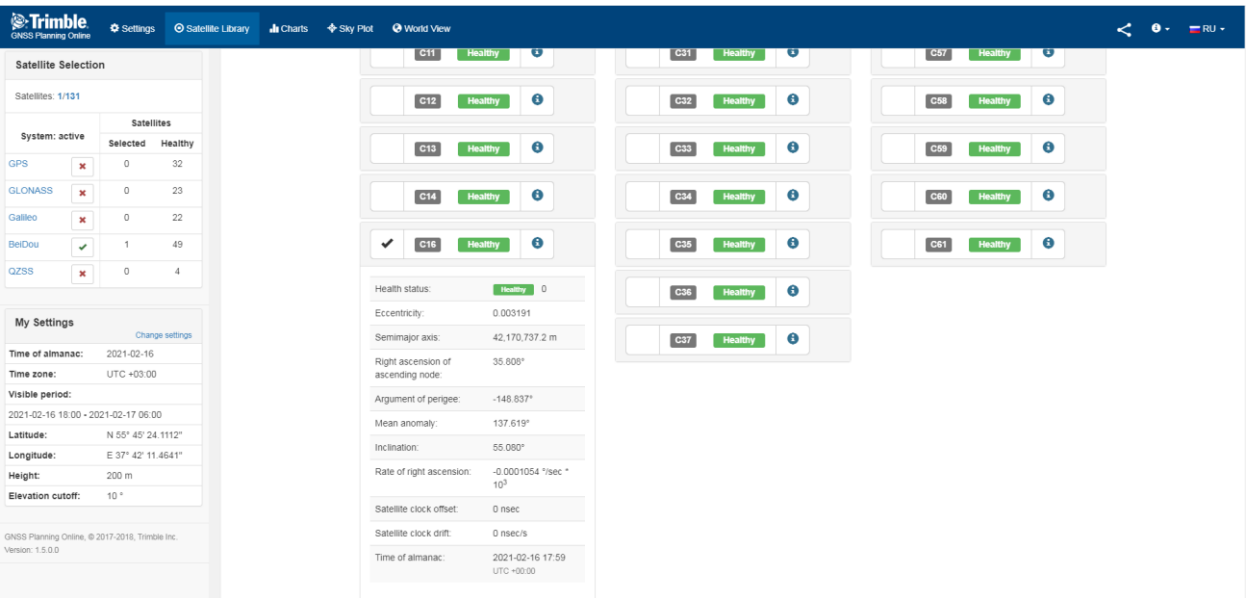


Рисунок 5 – Моделирование с помощью сервиса Trimble GNSS Planning

Получим график расчета угла места собственного спутника от времени:

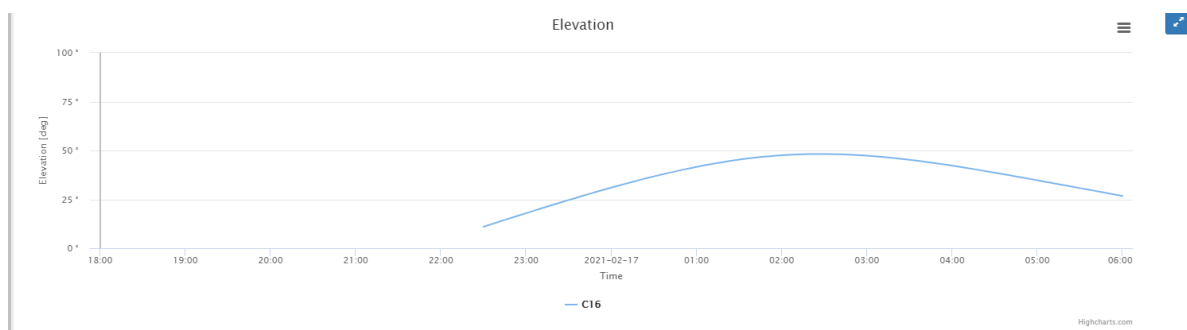


Рисунок 6 – График угла места собственного спутника от времени

По графику видно, что на указанном в задании интервале с 18:00 – 06:00, спутник был в области видимости 1 раз - с 22:30 16.02.2021 до 6:00 17.02.2021.

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

Так как сервис для определения Sky Plot используется тот же - Trimble GNSS Planning Online, то настройки оставим прежние, и проведем моделирование Sky Plot во временном интервале 18:00-06:00. Зафиксируем положение спутника на небосводе в критических точках, то есть когда он находился в области видимости - в 22:30 и 6:00.

Тогда получим два графика моделирования:

- 16 февраля 2021 в 22:30:

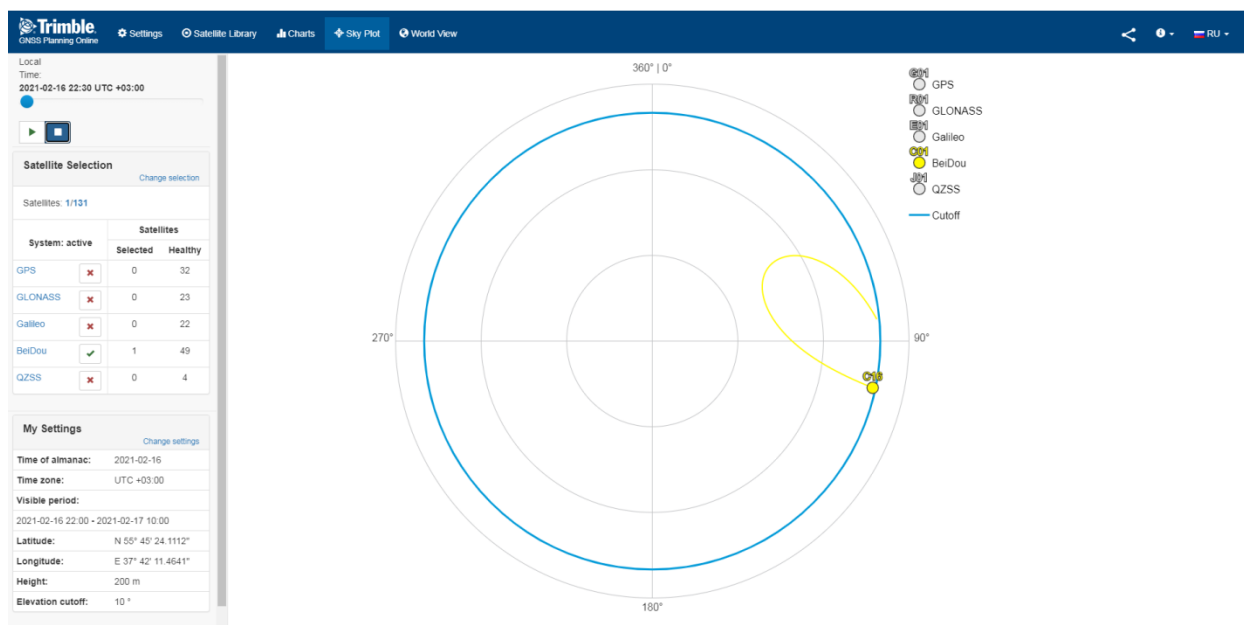


Рисунок 7 – Моделирование с помощью сервиса Trimble GNSS Planning

- 17 февраля 2021 в 6:00:

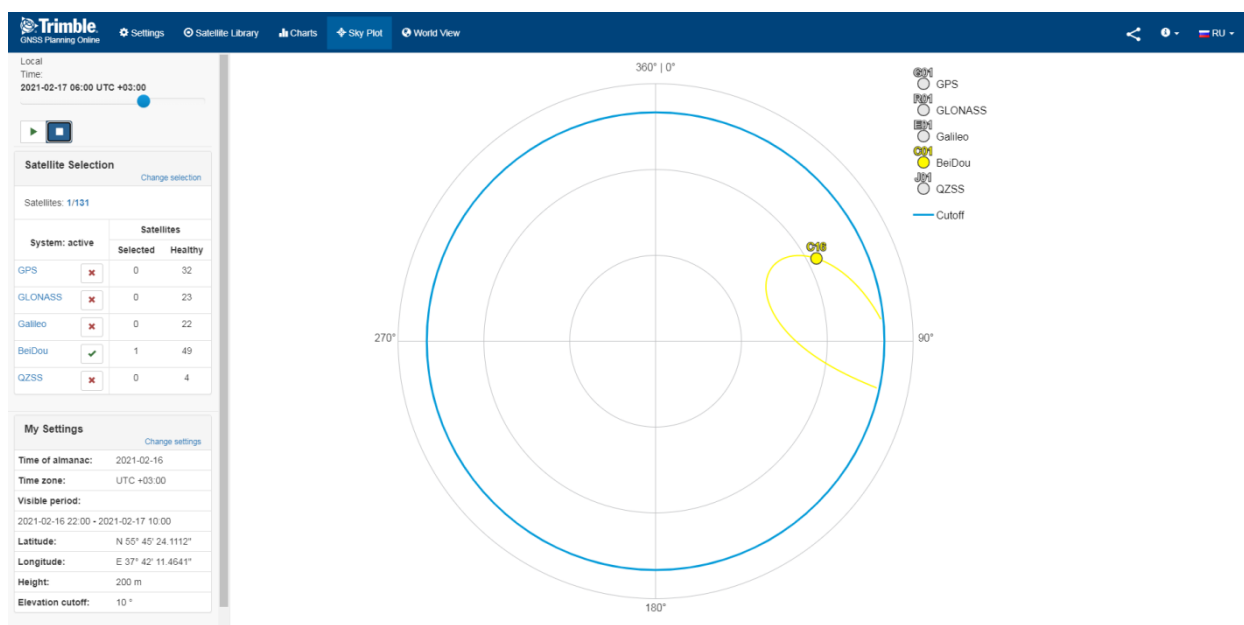


Рисунок 8 – Моделирование с помощью сервиса Trimble GNSS Planning

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

Таблица 1 – Описание параметров, входящих в состав эфемерид

Параметры	Определение
t_{oe}	Отсчет времени эфемерид
\sqrt{A}	Квадратный корень из большой полуоси орбиты
e	Эксцентриситет
ω	Аргумент перигея
Δn	Среднее отклонение движения от расчетного значения
M_0	Средняя аномалия в исходное время
Ω_0	Долгота восходящего узла орбитальной плоскости, вычисленная по опорному времени
$\dot{\Omega}$	Скорость прямого восхождения
i_0	Угол наклона в исходное время
$IDOT$	Скорость угла наклона
C_{uc}	Амплитуда косинусной поправки к аргументу широты
C_{us}	Амплитуда синусной поправки к аргументу широты
C_{rc}	Амплитуда косинусной поправки к радиусу орбиты
C_{rs}	Амплитуда синусной поправки к радиусу орбиты
C_{ic}	Амплитуда косинусной поправки к углу наклона
C_{is}	Амплитуда синусной поправки к углу наклона

1.6. Формирование таблицы эфемерид собственного спутника

Данные спутника берутся из текстового файла, полученного из дампа бинарного потока данных от приемника в формате NVS BINR.

Таблица 2 – Значения эфемерид спутника C16

Параметры	Значение	Размерность
SatNum	16	-
toe, t_{oe}	244800000.000	мс
Crs, C_{rs}	-9.5281250000000000e+01	м
Dn, Δn	7.70389232008367175e-13	рад/мс
M0, M_0	2.93507146527906437e+00	рад
Cuc, C_{uc}	-2.92435288429260254e-06	рад
e	3.17447888664901257e-03	-
Cus, C_{us}	2.32872553169727325e-05	рад
sqrtA, \sqrt{A}	6.49392874526977539e+03	м ^{1/2}
Cic, C_{ic}	-1.04308128356933594e-07	рад
Omega0, Ω_0	6.24949025738267050e-01	рад
Cis, C_{is}	7.07805156707763672e-08	рад
i0, i_0	9.61377831028401686e-01	рад
Crc, C_{rs}	-4.6468750000000000e+02	м
omega, ω	-2.60358371554086920e+00	рад
OmegaDot, $\dot{\Omega}$	-1.74471553154787378e-12	рад/мс
iDot, \dot{I}	-3.32156692802358755e-14	рад/мс
Tgd, T_{GD}	1.2100000000000000e+05	мс
toc, t_{oc}	2.4480000000000000e+08	мс
af2, a_{f2}	0.0000000000000000e+00	мс/мс ²
af1, a_{f1}	-1.50697232470520248e-11	мс/мс

af0, a_{f0}	-6.98594987392425537e-01	MC
URA	0	-
IODE	257	-
IODC	0	-
codeL2	0	-
L2P	0	-
WN	789	-

ГЛАВА 2. МОДЕЛИРОВАНИЕ

На данном этапе требуется реализовать функцию расчета положения спутника Beidou на заданный момент по шкале времени UTC на языке Matlab или Python. Значения, полученные на предыдущем этапе, нужны нам в качестве эфемерид для моделирования. 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. Для выполнения данного этапа я использовал пакет математического моделирования Matlab, а само моделирование проводил с помощью алгоритма с сайта Navipedia.

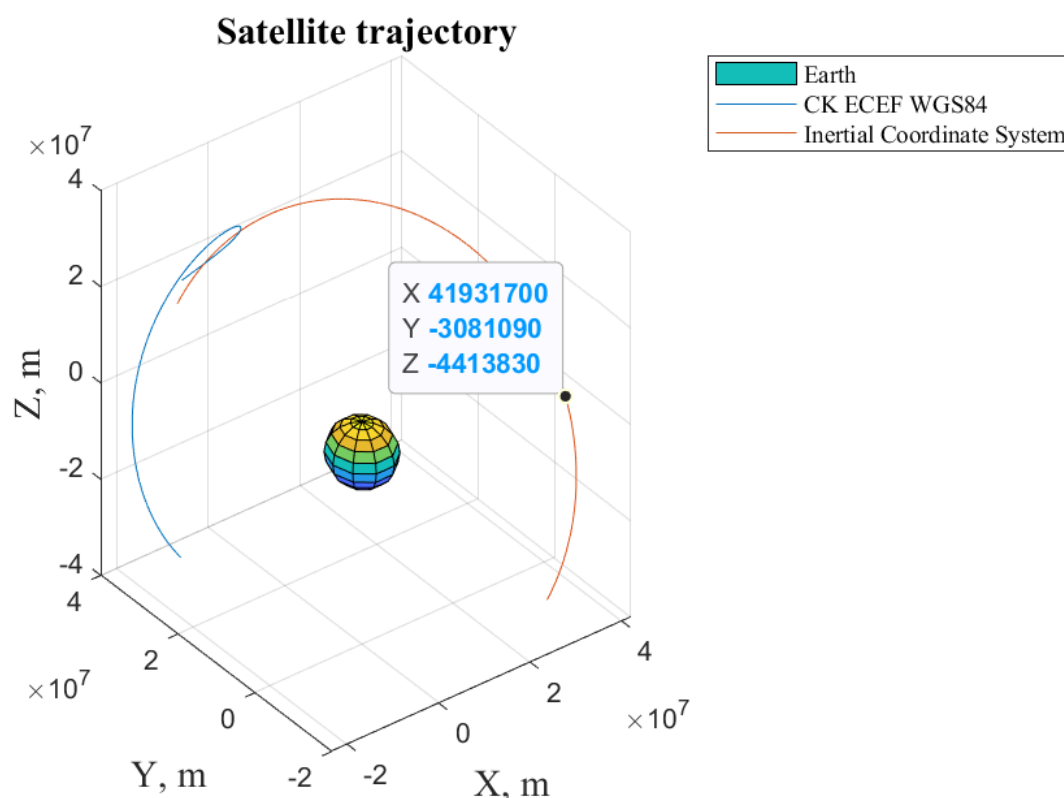


Рисунок 9 — Траектория движения спутника Beidou PRN №26 в системе координат WGS-84 (синяя линия) и инерциальной системе координат (красная линия).

На рисунке 9 помимо траектории спутника так же изображена модель Земли с нанесенной на нее координатой антенны на крыше корпуса Е НИУ «МЭИ».

Переход из системы ECEF в систему ECI был осуществлен также согласно алгоритму из ИКД. По полученному графику видно, что за установленный интервал времени спутник не успевает полностью пройти всю свою траекторию.

Помимо траектории спутников в трехмерном виде получим эту траекторию в полярной системе координат и сравним ее с результатом из Trimble GNSS Planning Online, изображенным на рисунках 7 и 8.

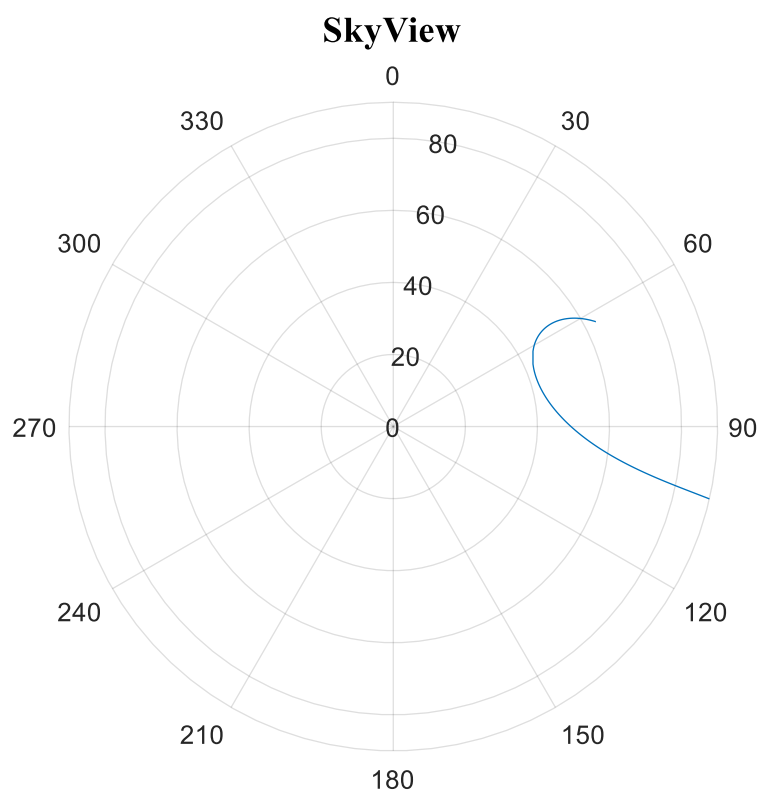


Рисунок 10 - SkyView спутника Beidou, полученный в результате моделирования.

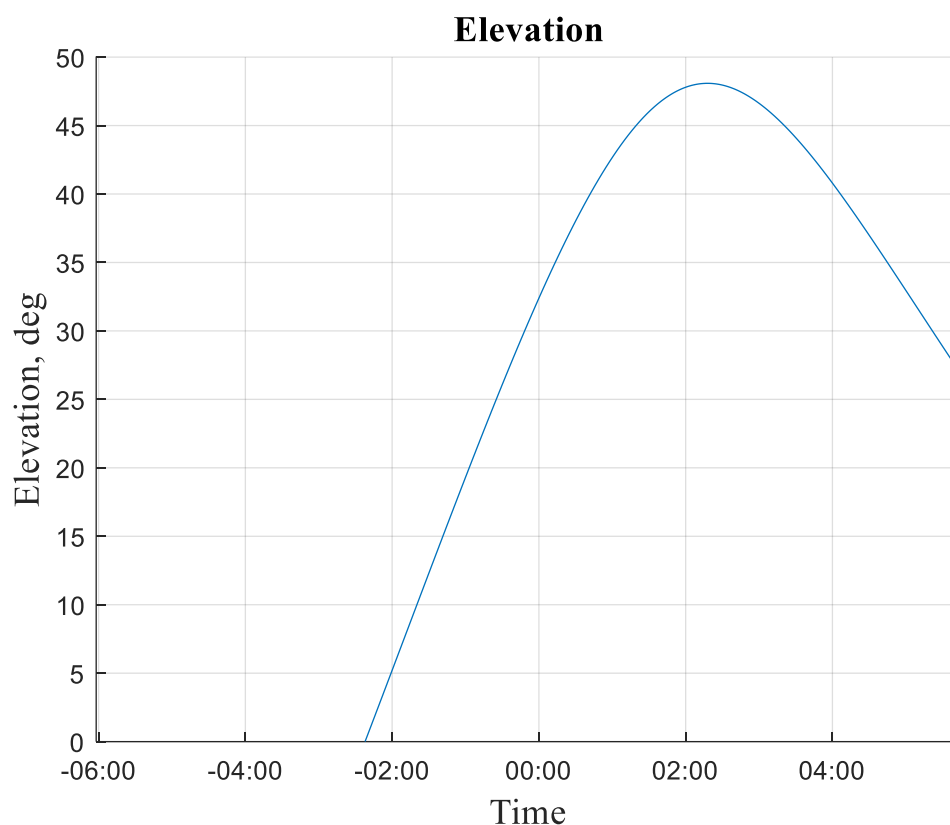


Рисунок 11 – График угла места спутника Beidou PRN №16

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 одни и те же параметры эфемерид на всём промежутке времени.

Код программы в приложении 1.

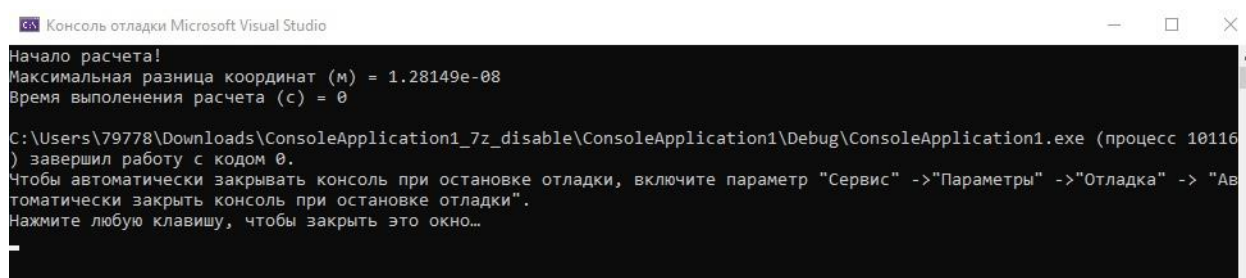
ГЛАВА 3. РЕАЛИЗАЦИЯ

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.

Код реализации представлен в приложении 2.

Результат запуска программы приведен на рисунке 12:



```
Консоль отладки Microsoft Visual Studio
Начало расчета!
Максимальная разница координат (м) = 1.28149e-08
Время выполнения расчета (с) = 0
C:\Users\79778\Downloads\ConsoleApplication1_7z_disable\ConsoleApplication1\Debug\ConsoleApplication1.exe (процесс 10116)
) завершил работу с кодом 0.
Чтобы автоматически закрывать консоль при остановке отладки, включите параметр "Сервис" ->"Параметры" ->"Отладка" -> "Автоматически закрыть консоль при остановке отладки".
Нажмите любую клавишу, чтобы закрыть это окно...
```

Рисунок 12 – Результат запуска программы

Критический путь		
Имя функции	Общее время ЦП [единицы, %]	Собственное время ЦП [единицы, %]
↳ [Внешний код]	79 (98,75%)	15 (18,75%)
↳ mainCRTStartup	64 (80,00%)	0 (0,00%)
↳ _scrt_common_main	64 (80,00%)	0 (0,00%)
↳ _scrt_common_main_seh	64 (80,00%)	0 (0,00%)
↳ invoke_main	63 (78,75%)	0 (0,00%)
↳ main	50 (62,50%)	8 (10,00%)
↳ [Внешний код]	37 (46,25%)	32 (40,00%)

Рисунок 13 – График загрузки ЦП

ЗАКЛЮЧЕНИЕ

В данной работе был произведен анализ ИКД Beidou, который состоял из трех этапов.

На первом этапе подготовили вспомогательные данные для разработки: эфемериды и оценки положения спутника от сторонних сервисов, с помощью которых была построена орбита заданного спутника Beidou, также получена траектория его движения на заданный промежуток времени.

На втором этапе был реализован расчет координат для заданного спутника, были получены сравнимые значения и графические отображения траекторий основных характеристик.

На третьем, заключительном, этапе была разработка функции расчета местоположения спутника Beidou на языке C++. В ходе выполнения данного этапа было произведено сравнение результатов с программой из Matlab.

В итоге был получен график использования памяти при проведении сеанса диагностики.

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

- [1]. Электронный ресурс: «Информационно-аналитического центра координатно-временного и навигационного обеспечения «www.glonass-iac.ru»»
- [2]. Электронный ресурс: «Википедия. Свободная энциклопедия «<https://ru.wikipedia.org/wiki/Бэйдоу>»»
- [3]. Электронный ресурс: «<https://www.celestrak.com>»
- [4]. Электронный ресурс: «<https://www.gnssplanningonline.com/>»

ПРИЛОЖЕНИЕ

Приложение 1

```
close all;
clear all;
clc;
format long
%% Эфемериды
SatNum = 16;
toe = 244800000.000 * 10^-3;
Crs = -9.5281250000000000e+01;
Dn = 7.70389232008367175e-13;
M0 = 2.93507146527906437e+00;
Cuc = -2.92435288429260254e-06;
e = 3.17447888664901257e-03;
Cus = 2.32872553169727325e-05;
sqrtA = 6.49392874526977539e+03;
Cic = -1.04308128356933594e-07;
Omega0 = 6.24949025738267050e-01;
Cis = 7.07805156707763672e-08;
i0 = 9.61377831028401686e-01;
Crc = -4.6468750000000000e+02;
omega = -2.60358371554086920e+00;
OmegaDot = -1.74471553154787378e-12;
iDot = -3.32156692802358755e-14;
Tgd = 1.2100000000000000e+05;
toc = 2.4480000000000000e+08;
af2 = 0.0000000000000000e+00;
af1 = -1.50697232470520248e-11;
af0 = -6.98594987392425537e-01;
URA = 0;
IODE = 257;
IODC = 0;
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 = sqrt(A^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)),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)+Cuc*cos_correction(k)+Cus*sin_correction(k);

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

    % Наклон
    i(k) = i0+iDot*t(k)+Cic*cos_correction(k)+Cis*sin_correction(k);

```

```

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

% Положение на орбите
x = r(k)*cos(u(k));
y = r(k)*sin(u(k));

% Координаты
X0(k) = x*cos(lambda(k))-y*cos(i(k))*sin(lambda(k));
Y0(k) = x*sin(lambda(k))+y*cos(i(k))*cos(lambda(k));
Z0(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,wgs84Elli
psoid , '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
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

```

// ConsoleApplication1.cpp : Этот файл содержит функцию "main". Здесь начинается и заканчивается выполнение программы.

//

#include <iostream>

#include <fstream>

using namespace std;

int main()

{

setlocale(LC_ALL, "rus");

std::cout << "Начало расчета!\n";

time_t t_start, t_stop;

time(&t_start);

double SatNum = 16;

double toe = 244800000.000 * 10⁻³;

double Crs = -9.5281250000000000e+01;

double Dn = 7.70389232008367175e-13;

double M0 = 2.93507146527906437e+00;

double Cuc = -2.92435288429260254e-06;

double e = 3.17447888664901257e-03;

double Cus = 2.32872553169727325e-05;

double sqrtA = 6.49392874526977539e+03;

double Cic = -1.04308128356933594e-07;

double Omega0 = 6.24949025738267050e-01;

double Cis = 7.07805156707763672e-08;

double i0 = 9.61377831028401686e-01;

double Crc = -4.6468750000000000e+02;

double omega = -2.60358371554086920e+00;

double OmegaDot = -1.74471553154787378e-12;

double iDot = -3.32156692802358755e-14;

double Tgd = 1.2100000000000000e+05;

double toc = 2.4480000000000000e+08;

double af2 = 0.0000000000000000e+00;

double af1 = -1.50697232470520248e-11;

double af0 = -6.98594987392425537e-01;

double URA = 0;

double IODE = 257;

double IODC = 0;

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

// Длина временного промежутка
int t_arr = end_time - begin_time;

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

// Среднее движение
double n0 = sqrt(mu / pow(A, 3));
double n = n0 + Dn;

// Координаты
double** coord = new double*[3];
for (int i = 0; i < 3; i++) {
    coord[i] = new double[t_arr];
}

// Координаты из матлаба
double** coordMat = new double*[3];
for (int i = 0; i < 3; i++) {
    coordMat[i] = new double[t_arr];
}

for (int k = 0; k < t_arr; k++) {
    // Время
    double t = begin_time + k - (int)toe;

    if (t > 302400)
        t = t - 604800;
    if (t < -302400)
        t = 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 (abs(E - E_old) > epsilon) {
    E_old = E;
    E = M + e * sin(E);
}
// Истинная аномалия
double nu = atan2(sqrt(1 - e*e) * 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 + Cus * 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
double Xk = x * cos(lambda) - y * cos(i) * sin(lambda);
double Yk = x * sin(lambda) + y * cos(i) * cos(lambda);
double Zk = y * sin(i);

coord[0][k] = Xk;
coord[1][k] = Yk;
coord[2][k] = Zk;
}

```

```

// Чтение координат из матлаба
ifstream file(" data_matlab3.txt");

if (!file.is_open())
    cout << "Файл не может быть открыт!" << endl;
else {
    for (int k = 0; k < t_arr; k++) {
        file >> coordMat[0][k] >> coordMat[1][k] >> coordMat[2][k];
    }
    file.close();
}

// Сравнение с матлабом
double max_del = 0;
for (int i = 0; i < 3; i++) {
    for (int k = 0; k < t_arr; k++) {
        if (abs(coord[0][k] - coordMat[0][k]) > max_del) {
            max_del = abs(coord[0][k] - coordMat[0][k]);
        }
    }
}
time(&t_stop);

double del_time = difftime(t_stop, t_start);

cout << "Максимальная разница координат (м) = " <<
    max_del << endl;
cout << "Время выполнения расчета (с) = " << del_time << endl;

delete[] *coord;
delete[] coord;
delete[] * coordMat;
delete[] coordMat;
}

```

СОДЕРЖАНИЕ

ВВЕДЕНИЕ	2
ГЛАВА 1. ИСПОЛЬЗОВАНИЕ СТОРОННИХ СРЕДСТВ	3
1.1. Описание задания	3
1.2. Определение формы орбиты и положения спутника на ней с помощью сервиса CelesTrak	5
1.3. Расчет графика угла места собственного спутника от времени по данным Trimble GNSS Planning Online	5
1.4. Расчет диаграммы угла места и азимута спутника (SkyView, он же SkyPlot) по данным Trimble GNSS Planning Online	7
1.5. Формирование списка и описание параметров, входящих в состав эфемерид	8
1.6. Формирование таблицы эфемерид собственного спутника	9
ГЛАВА 2. МОДЕЛИРОВАНИЕ	12
ГЛАВА 3. РЕАЛИЗАЦИЯ	15
ЗАКЛЮЧЕНИЕ	16
СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ	17
ПРИЛОЖЕНИЕ.....	18
СОДЕРЖАНИЕ	27