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

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

Направление подготовки: 11.05.01 - Радиоэлектронные системы и комплексы

ОТЧЕТ по курсовой работе

СТУДЕНТ

 / Хватов М.М. /

(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

ПОСТАНОВКА ЗАДАЧИ

В данной работе мы знакомимся с рядом инструментов и техник, используемых при разработке навигационных приемников.

Цель проекта - получить библиотеку функций на Си++, позволяющую рассчитывать положение спутника Beidou по его эфемеридам. На первом этапе подготовим вспомогательные данные для разработки: эфемериды и оценки положения спутника от сторонних сервисов (чтобы было с чем сравниваться на след. этапах)

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.

Этапы курсовой работы отличаются осваиваемыми инструментами.

ЭТАП 1. ИСПОЛЬЗОВАНИЕ СТОРОННИХ СРЕДСТВ

1.1 Описание работы

Конечная цель всего курсового проекта - получить библиотеку функций на Си++, позволяющую рассчитывать положение спутника Beidou по его эфемеридам. 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 Harxon HX-CSX601A. Она через 50-метровый кабель, сплиттер, bias-tee и усилительподключена 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, доступны в рабочем репозитории (директория logs) в нескольких форматах.

В моём варианты работы используется спутник C26.

Приведем параметры эфемерид системы в таблице 1.

Таблица 1 – Параметры эфемерид системы

Параметр	Описание
t_{oe}	Опорная эпоха эфемерид
$\sqrt{A}$	Корень из большой полуоси орбиты
e	Эксцентриситет орбиты
ω	Аргумент перигея
Δn	Поправка в среднее движение
M0	Средняя аномалия на опорную эпоху
Ω0	Долгота восходящего угла орбиты на опорную эпоху
Ω	Скорость прямого восхождения
i0	Угол наклона орбиты на опорную эпоху
IDOT	Скорость изменения наклона орбиты
Cuc	Амплитуда косинусной поправки к аргументу широты
Cus	Амплитуда синусной поправки к аргументу широты
Crc	Амплитуда косинусной поправки к радиусу орбиты
Crs	Амплитуда синусной поправки к радиусу орбиты
Cic	Амплитуда косинусной поправки к углу наклона
Cis	Амплитуда синусной поправки к углу наклона

1.2 Использование входных данных и определение номера спутника

Данные спутника берутся из текстового файла, полученного из дампа бинарного потока данных от приемника в формате NVS BINR. Воспользуемся данными для конкретного варианта и сведём их в таблицу 2.

Таблица 2 – Значения эфемерид спутника C26

Параметр	Значение
Satnum	26
toe (мс)	219600000.000
Crs (рад)	7.02187500000000000e+01
Dn (рад/мс)	4.31196528136168489e-12
M0 (рад)	5.78014959386014437e-01
Cuc (рад)	3.61073762178421021e-06
e	7.86192365922033787e-04
Cus (рад)	6.01261854171752930e-06
sqrtA ($м^{1/2}$)	5.28262158584594727e+03
Cic (рад)	4.28408384323120117e-08
Omega0 (рад)	1.79943690961005953e+00
Cis (рад)	-5.54136931896209717e-08
i0 (рад)	9.51213294811811605e-01
Crc (рад)	2.30484375000000000e+02
Omega (рад)	3.87711120604409931e-01
OmegaDot (рад/мс)	-7.17315593359416561e-12
iDot (рад/сек)	-1.67149819603767644e-13
Tgd (мс)	9.75000000000000000e+05
Toc (мс)	2.19600000000000000e+08
af2 ($мс/мс^2$)	1.48307593848345250e-22
af1 (мс/мс)	5.06794606280891458e-12
af0 (мс)	3.53220045566558838e-01
URA	0
IODE	257
IODC	1
codeL2	0
L2P	0
WN	789

Сравним эти данные с данными, приведёнными с сайта из альманаха спутников Бейдоу. На рисунке 2 покажем скриншот таблицы эфемерид с сайта glonass-iac.

PRN	IH	e	t_c	δ_i	$\dot{\Omega}$	A	Ω_0	ω	m	af_0	af_1	week
C26	000	0.00078773	192512	0.00874848	-6.88028659e-9	5282.5723	1.79960514	0.39429111	-3.09722181	0.0009479523	-3.63798e-12	789

Рисунок 2 – Таблицы эфемерид с сайта glonass-iac

Как видно, что данные предоставленные преподавателем действительно сходятся со спутником C26.

С помощью «Информационно-аналитического центра координатно-временного и навигационного обеспечения» определим номер НОРАД:

PRN	НОРАД	Тип КА	Тип системы	Дата запуска	Факт. сущ. (дней)	Примечание
C01	44231	GEO-8	BDS-2	17.05.19	657	Используется по ЦН
C13	41434	IGSO-6	BDS-2	30.03.16	1800	Используется по ЦН
C14	38775	MEO-6	BDS-2	19.09.12	3088	Используется по ЦН
C16	43539	IGSO-7	BDS-2	10.07.18	968	Используется по ЦН
C19	43001	MEO-1	BDS-3	05.11.17	1215	Используется по ЦН
C20	43002	MEO-2	BDS-3	05.11.17	1215	Используется по ЦН
C21	43208	MEO-3	BDS-3	12.02.18	1116	Используется по ЦН
C22	43207	MEO-4	BDS-3	12.02.18	1116	Используется по ЦН
C23	43581	MEO-5	BDS-3	29.07.18	949	Используется по ЦН
C24	43582	MEO-6	BDS-3	29.07.18	949	Используется по ЦН
C25	43603	MEO-11	BDS-3	25.08.18	922	Используется по ЦН
C26	43602	MEO-12	BDS-3	25.08.18	922	Используется по ЦН

Рисунок 3 - Состав и состояние системы BEIDOU с «Информационно-аналитического центра координатно-временного и навигационного обеспечения»

Теперь посмотрим номер НОРАД в Википедии:

№	Спутник	PRN	Дата (UTC)	Ракета	NSSDC ID	SCN	Орбита	Статус	Система
33	Бэйдоу-3 M9	C23	29.07.2018 01:48	CZ-3B/YZ-1	2018-062A	43581	СОО, ~21 500 км	действующий	Бэйдоу-3
34	Бэйдоу-3 M10	C24			2018-062B	43582	СОО, ~21 500 км	действующий	
35	Бэйдоу-3 M11	C26	24.08.2018, 23:37	CZ-3B/YZ-1	2018-067A	43602	СОО, ~21 500 км	действующий	
36	Бэйдоу-3 M12	C25			2018-067B	43603	СОО, ~21 500 км	действующий	
37	Бэйдоу-3 M13	C32	19.09.2018, 14:07	CZ-3B/YZ-1	2018-072A	43622	СОО, ~21 500 км	действующий	
38	Бэйдоу-3 M14	C33			2018-072B	43623	СОО, ~21 500 км	действующий	
39	Бэйдоу-3 M15	C35	15.10.2018, 04:23	CZ-3B/YZ-1	2018-078A	43647	СОО, ~21 500 км	действующий	
40	Бэйдоу-3 M16	C34			2018-078B	43648	СОО, ~21 500 км	действующий	
41	Бэйдоу-3 G1Q	C59	01.11.2018, 15:57	CZ-3B/E	2018-085A	43683	ГСО, 144.5° в. д.	действующий	
42	Бэйдоу-3 M17	C36	18.11.2018, 17:49	CZ-3B/YZ-1	2018-093A	43706	СОО, ~21 500 км	действующий	
43	Бэйдоу-3 M18	C37			2018-093B	43707	СОО, ~21 500 км	действующий	
44	Бэйдоу-3 IGSO-1	C38	20.04.2019, 14:41	CZ-3B/G2	2019-023A	44204	Геосинхронная, накл. 55°;	действующий	

Рисунок 4 - Состав и состояние системы BEIDOU с сайта “Википедия”

Номера спутника совпадают и равны 43602, название спутника - «BEIDOU-3 M11». Спутнику с PRN C26 соответствует спутник №35 - это нужно учитывать при выполнении следующих пунктов задания.

1.3 Определение формы орбиты и положения спутника на ней на начало рассматриваемого интервала времени по данным сервиса CelesTrak

Зайдем на сайт CelesTrak (https://celestrak.com) для выполнения этого пункта. Значение времени выставим 15:00, 16 февраля 2021 по UTC(0). Введем наше название спутника, сравним НОРАД и, убедившись, что он совпадает, запустим моделирование, результаты которого видим на 5-м рисунке:

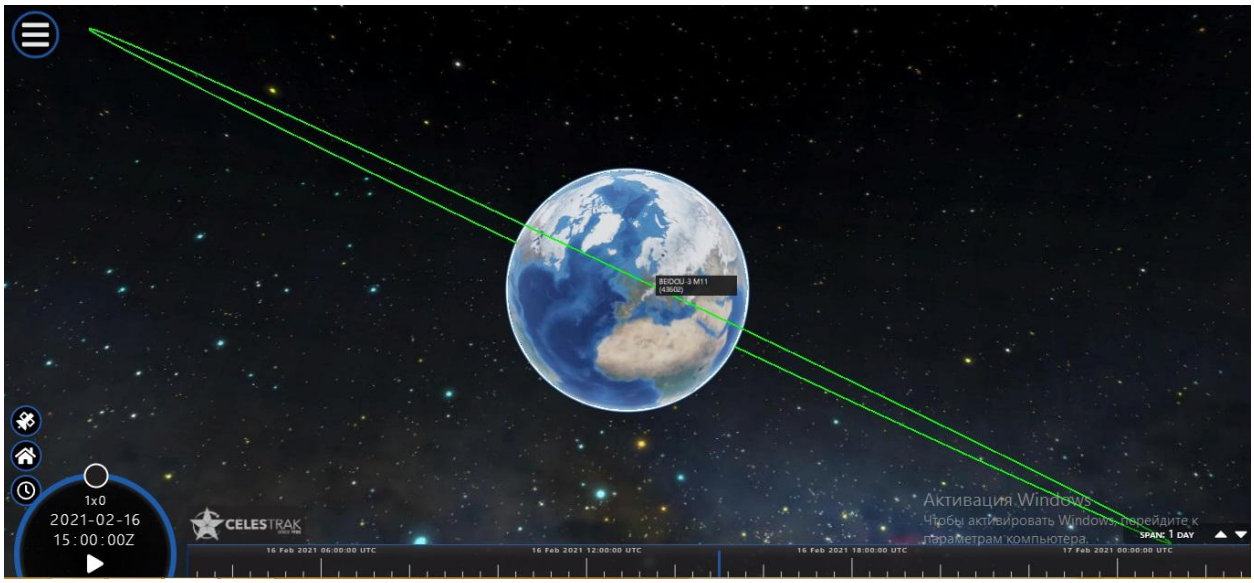

Рисунок 5 – Результат моделирования на CelesTrak. Общий вид+положение спутника.

Мы устанавливаем время 15:00, потому что нам по заданию требуется построить модель на момент времени 18:00 по Московскому времени. А в программе устанавливается время по часовому поясу UTC(0).

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

Заходим на сайт https://www.gnssplanning.com и устанавливаем координаты в соответствии с расположением антенны, т.е. координаты корпуса Е МЭИ, потому что на крыше этого здания установлена антенна. Начальное время 18:00 при часовом поясе UTC(+3) – это Московское время.

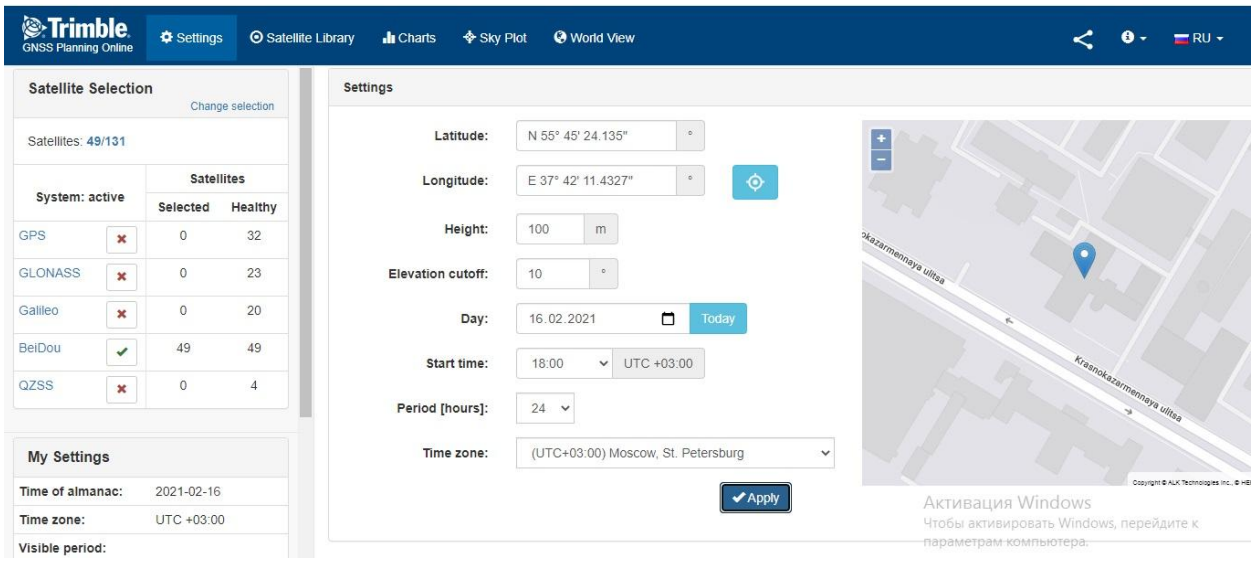

Рисунок 6 – Настройка Trimble GNSS Planning для дальнейшего моделирования.

Уберём все спутники кроме того, который дан в задании в моём варианте:

Рисунок 7 – единственный спутник, который нам нужен.

Теперь снимем график угла места для нашего спутника Beidou для времени 18:00 16 февраля 2021 года. Для этого откроем вкладку (Charts)

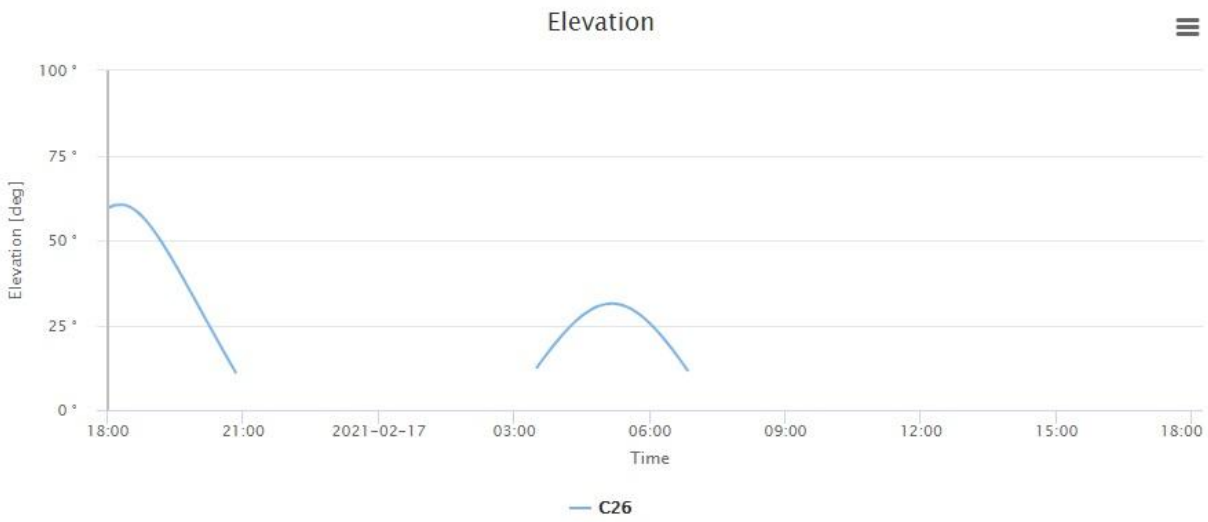

Рисунок 8 - График угла места собственного спутника от времени

В указанном интервале график видно 2 раза. Первый раз с 18:00 до 21:00. Второй раз с 03:30 до 07:00.

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

Теперь, для того, чтобы получить карту небосвода, перейдём во вкладку SkyPlot, оставив все предыдущие настройки. Моделирование проводим в том же временном интервале.

- 16 февраля в 18:00 по UTC +3

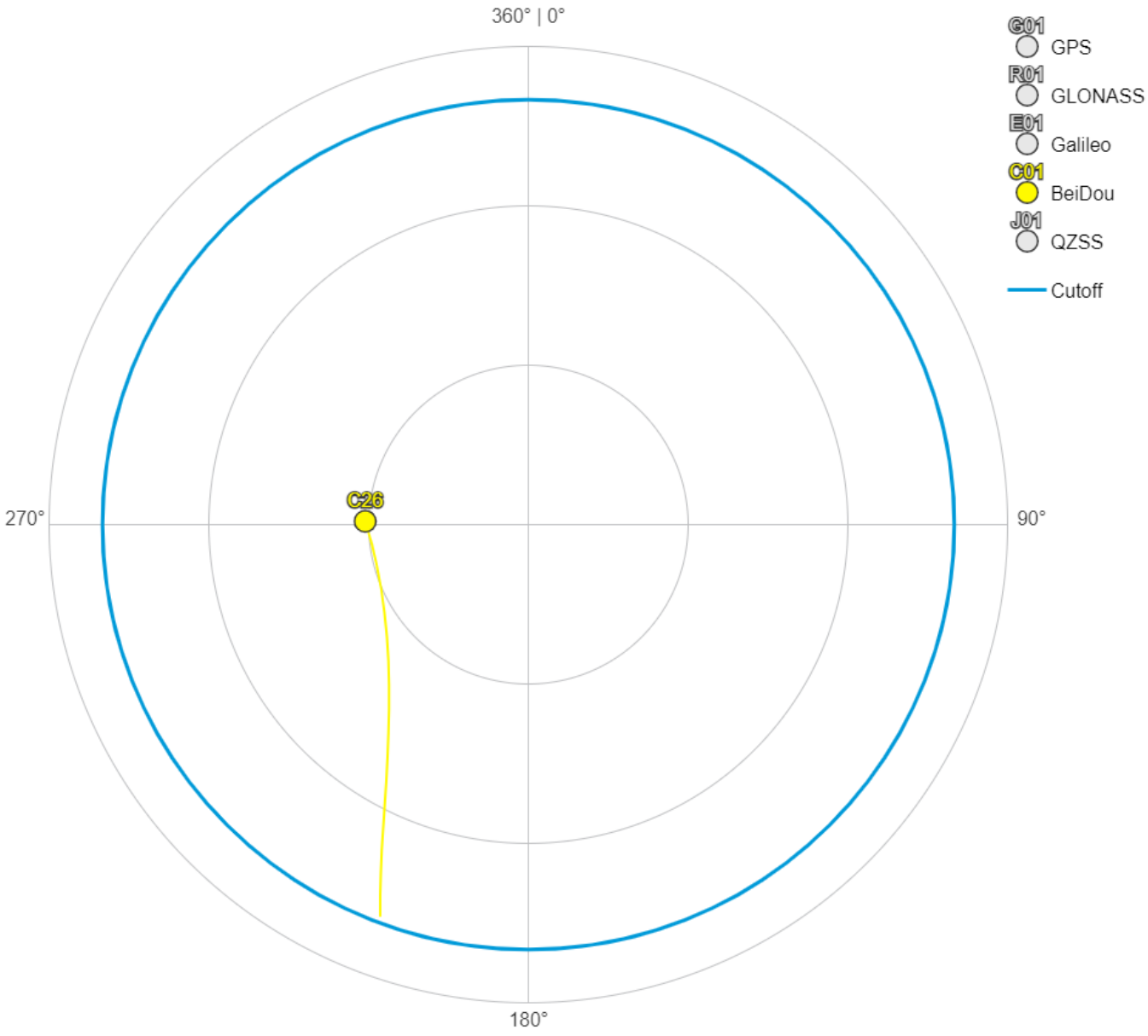

Рисунок 9 – SkyView спутника Beidou C26 16.02.21 18:00

- 17 февраля в 6:00 по UTC +3

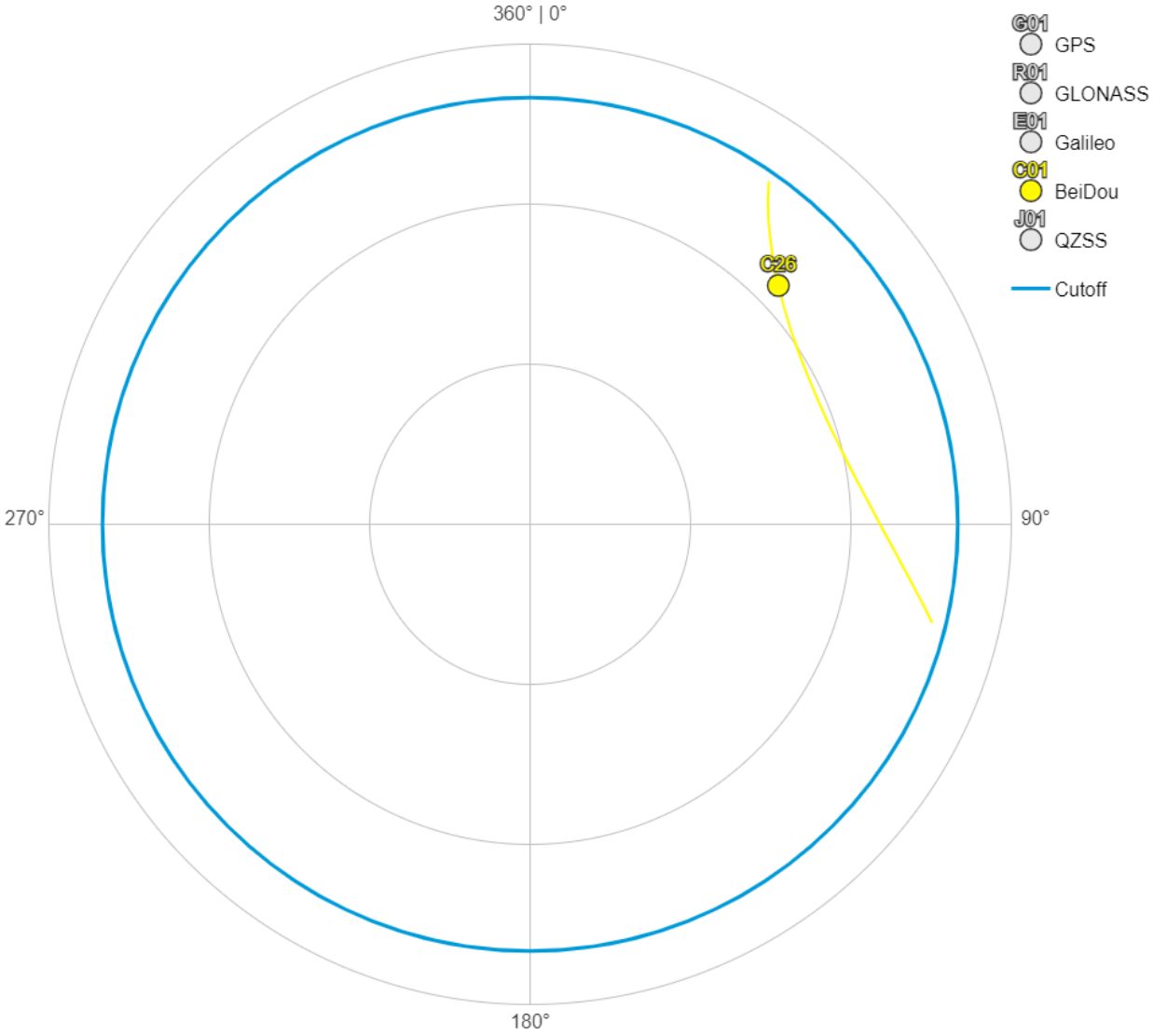

Рисунок 10 – SkyView спутника Beidou C26 17.02.21 6:00

Спутник движется снизу вверх, на следующем этапе мы будем проводить моделирование. На сайте мы моделировали с 18 часов 16 февраля до 18 часов 17 февраля, т.е. ровно сутки. На следующем этапе мы будем моделировать с 18:00 часов 16.02 по 6:00 17.02 с помощью пакета математического моделирования. Здесь траектория движения спутника прорисована полностью.

ЭТАП 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.

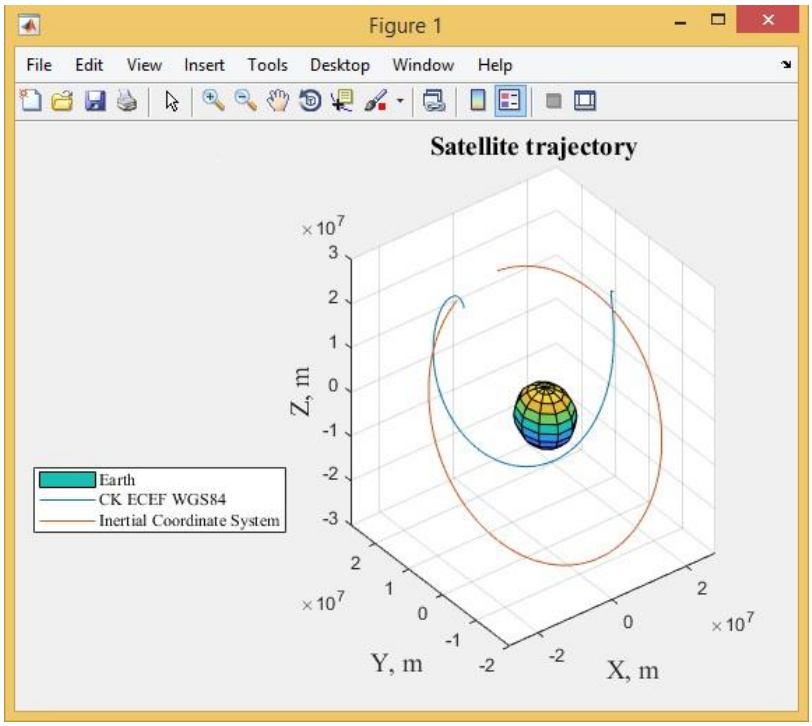

Рисунок 12 — Траектория движения спутника Beidou PRN №26 в системе координат WGS-84 (синяя линия) и инерциальной системе координат (красная линия).

На рисунке 12 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 9 и 10.

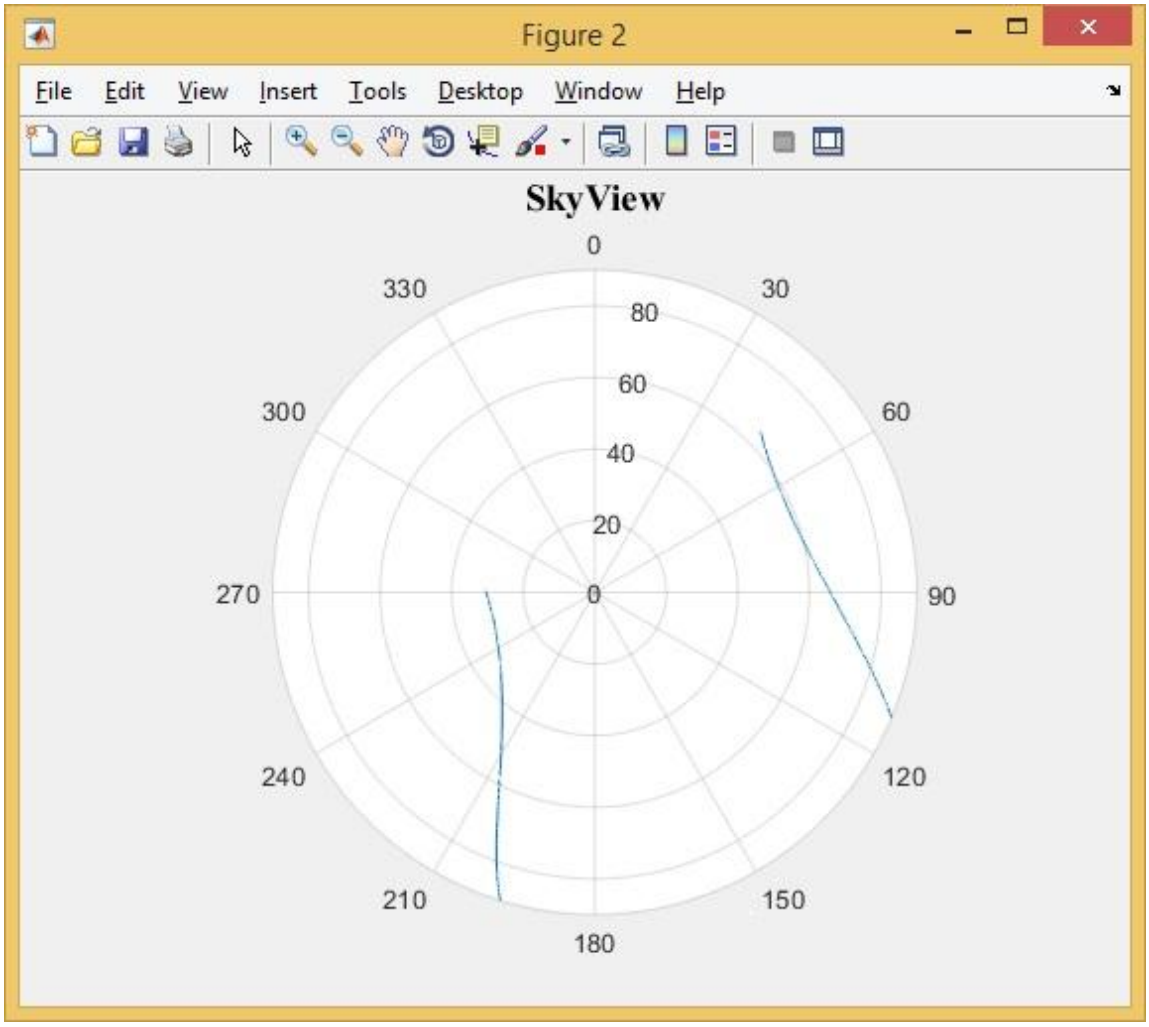

Рисунок 13 - SkyView спутника Beidou, полученный в результате моделирования.

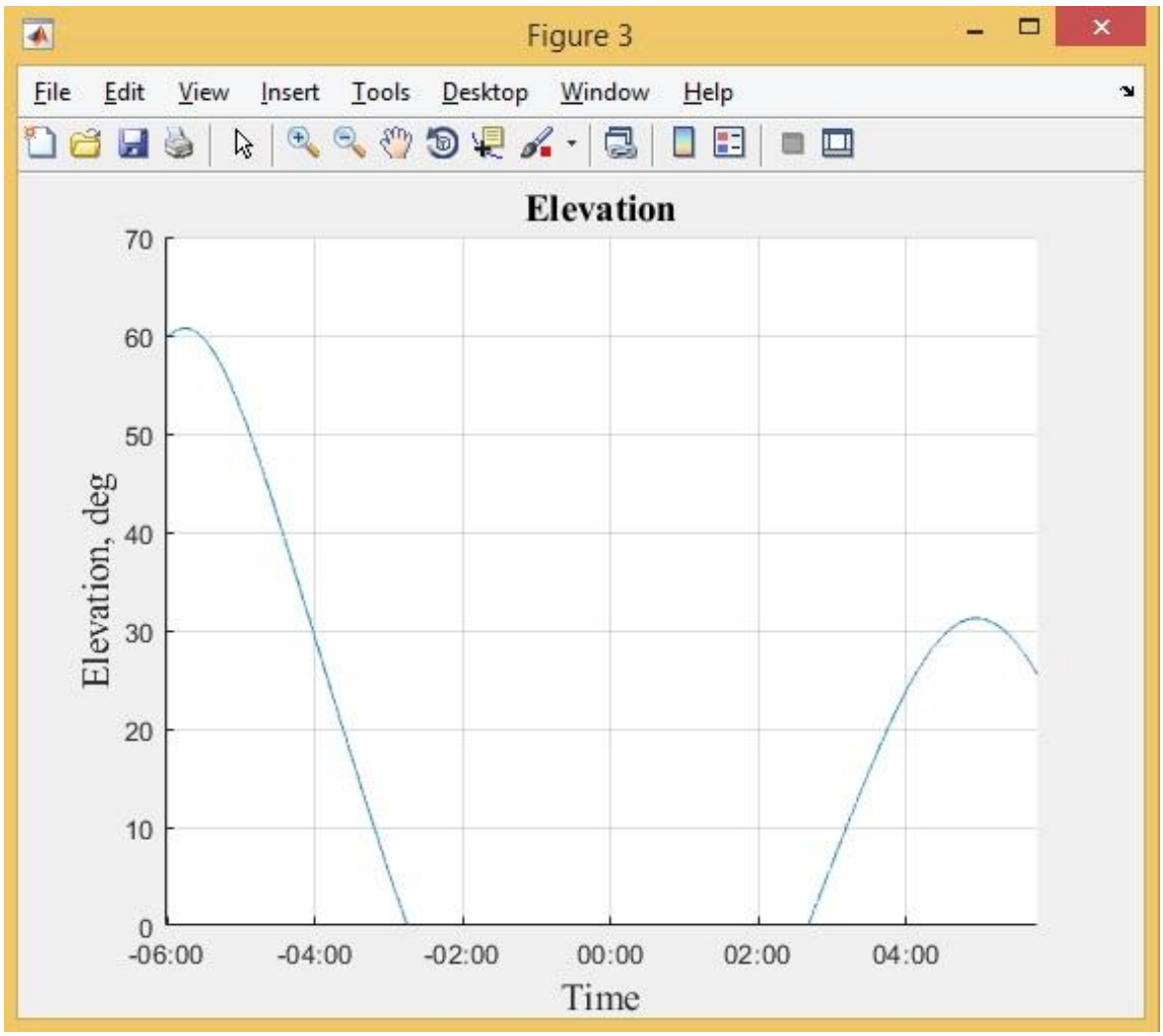

Рисунок 14 – График угла места спутника Beidou PRN №26

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.

ЭТАП 3. РЕАЛИЗАЦИЯ

3.1 Описание этапа

Требуется разработать на языке C/C++ функцию расчета положения спутника Beidou на заданное время по шкале UTC, минимизировать время её исполнения и количество затрачиваемой оперативной памяти. Вызов функции не должен приводить к выбросу исключений или утечкам памяти при любом наборе входных данных.

Функции расчета положения спутника в Matlab 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. Требуется выполнить свою реализацию решения трансцендентного уравнения.

Программный модуль должен сопровождаться unit-тестами под check:

- Тесты функций решения уравнения Кеплера.
- Тест расчетного положения спутника в сравнении с Matlab 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.

Требуется провести проверку на утечки памяти с помощью утилиты Valgrind.

3.2 Исходники программы

Функция была реализована на языке программирования C/C++ в среде разработке Microsoft Visual Studio. Так как функция расчета относительно проста, весь расчет 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:

```
	// 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 = 26;
		double toe = 219600000.000e-3;
		double Crs = 7.0218750000000000e+01;
		double Dn = 4.3119652813616848 9e-12;
		double M0 = 5.78014959386014437e-01;
		double Cuc = 3.61073762178421021e-06;
		double e = 7.86192365922033787e-04;
		double Cus = 6.01261854171752930e-06;
		double sqrtA = 5.28262158584594727e+03;
		double Cic = 4.28408384323120117e-08;
		double Omega0 = 1.79943690961005953e+00;
		double Cis = -5.54136931896209717e-08;
		double i0 = 9.51213294811811605e-01;
		double Crc = 2.30484375000000000e+02;
		double omega = 3.87711120604409931e-01;
		double OmegaDot = -7.17315593359416561e-12;
		double iDot = -1.67149819603767644e-13;
		double Tgd = 9.75000000000000000e+05;
		double toc = 2.19600000000000000e+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
		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);

			// Коэффиуиент 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
			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_matlab.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]);
				}
			}
		}

	// Очищение памяти
		delete[] *coord;
		delete[] coord;
		delete[] * coordMat;
		delete[] coordMat;

		time(&t_stop);

		double del_time = difftime(t_stop, t_start);

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

3.3 Результаты реализации

Как было отмечено в предыдущем разделе, функция выполняет несколько задач. Максимальная разница в координатах модели и функции составляет 1,49e-8м.

Время выполнения меньше одной секунды, в то время как в матлабе около 3 секунд.

Результаты выполнения функции приведены на рисунке 15.

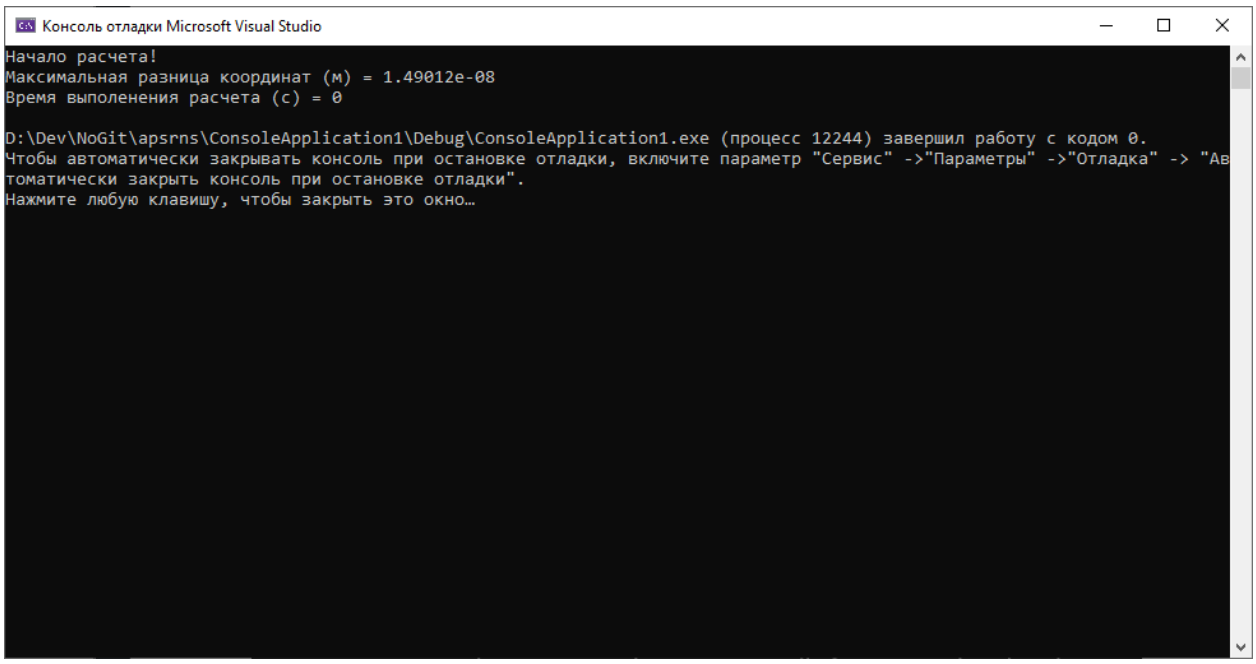

Рисунок 15 – Вывод функции в консоль отладки

3.4 Анализ работы программы

Воспользуемся встроенными инструментами отладки Microsoft Visual Studio. Так зайдя в «Отладка», далее в «Профилировщик производительности», и выбрав инструмент «Использование памяти», можно посмотреть потребляемую память программы и обнаружить утечки памяти. Результат работы инструмента «Использование памяти» показан на рисунке 16.

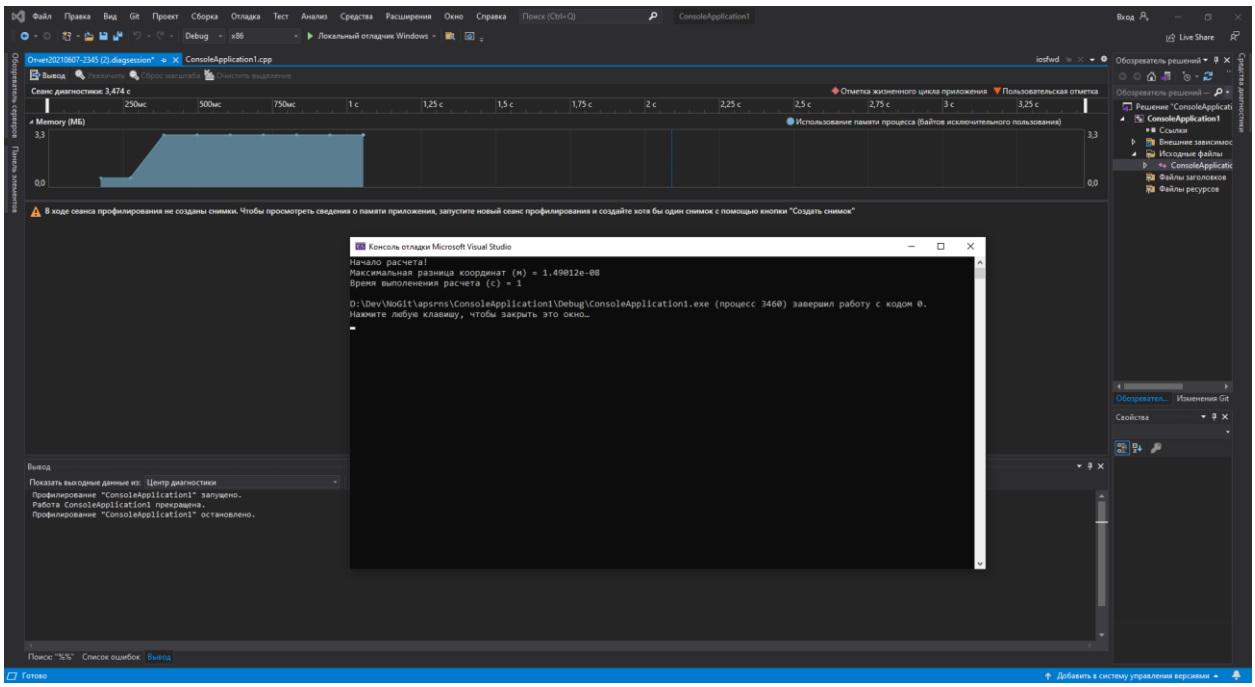

Рисунок 16 – Окно интерфейса пользователя при использовании инструмента «Использование памяти»

Можно отметить, что время выполнения программы увеличилось и составило примерно 1,05 секунды.

3.5 Выводы по этапу

В данном этапе была реализована на языке C/C++ функция расчета положения спутника Beidou на заданное время по шкале UTC. Функция сопровождается тестами решения уравнения Кеплера, проверкой на утечки памяти и профилированием.

Погрешность вычисления координат функцией и моделью составляет порядка 1,49e-8м, при использовании вещественного типа данных двойной точности double.

Время выполнение функции составляет менее одной секунды. Потребляемый объем оперативной памяти не превышает 3,3 МБ. Утечек памяти не обнаружено.

ПРИЛОЖЕНИЕ

```
close all;
clear all;
clc;
format long

%% Эфемериды
SatNum = 26;
toe = 219600000.000 * 10^-3;
Crs = 7.02187500000000000e+01;
Dn = 4.31196528136168489e-12;
M0 = 5.78014959386014437e-01;
Cuc = 3.61073762178421021e-06;
e = 7.86192365922033787e-04;
Cus = 6.01261854171752930e-06;
sqrtA = 5.28262158584594727e+03;
Cic = 4.28408384323120117e-08;
Omega0 = 1.79943690961005953e+00;
Cis = -5.54136931896209717e-08;
i0 = 9.51213294811811605e-01;
Crc = 2.30484375000000000e+02;
omega = 3.87711120604409931e-01;
OmegaDot = -7.17315593359416561e-12;
iDot = -1.67149819603767644e-13;
Tgd = 9.75000000000000000e+05;
toc = 2.19600000000000000e+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) = atan2(sqrt(1-e^2)*sin(E(k)),cos(E(k))-e);

    % Коэффиуиент 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 НКА 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

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
```

СПИСОК ЛИТЕРАТУРЫ

1. Электронный ресурс: www.glonass-iac.ru - инфомационно-аналитический центр коррдинатно-временного и навигационного обеспечения.
2. Сайт: https://www.celestrak.com - определение формы орбиты и положения спутника на ней.
3. Интернет-сервис: https://www.gnssplanningonline.com – программа, моделирующая положение спутников и их характеристики над заданной точкой земной поверхности.