НИУ «МЭИ»

Кафедра Радиотехнических систем

Расчетно-пояснительная записка к курсовому проекту по дисциплине «Аппаратура потребителей СРНС»

Выполнил: Антропов Е.А.

Группа: ЭР-15-17

Проверил: доц. Корогодин И.В.

Москва 2022

Этап 2. Моделирование траектории движения

На данном этапе требуется реализовать функцию расчета положения спутника GPS 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 GPS 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 00:00 МСК 14 февраля до 00:00 МСК 15 февраля 2022 года. Допускается использовать одни и те же эфемериды на весь рассматриваемый интервал. Для выполнения данного этапа я использовал пакет математического моделирования Matlab.

Исходные данные:

Значения эфемерид, полученные в исходном сигнале и обработанные в первом пункте. Уточнение и проверка данных выполнено в RTKNAVI.

```
LNAV Ephemeris (slot = 221472010) =
        Crs     = 1.171875e+01
        Dn      = 1.368335e-09  [deg/s]
        M0      = 5.455674e+01  [deg]
        Cuc     = 7.729977e-07
        e       = 1.137974e-02
        Cus     = 1.152977e-06
        sqrtA   = 5.153669e+03
        toe     = 86400
        Cic     = -1.173466e-07
        Omega0  = -1.094355e+02         [deg]
        Cis     = 1.136214e-07
        i0      = 5.653057e+01  [deg]
        Crc     = 3.719688e+02
        omega   = 5.051432e+01  [deg]
        OmegaDot= -4.812432e-07         [deg/s]
        iDot    = -6.773462e-09         [deg/s]
        Tgd     = 5.122274e-09
        toc     = 86400
        af2     = 0.000000e+00
        af1     = -9.549694e-12
        af0     = 4.321518e-04
        WN      = 149
        IODC    = 22
        URA     = 0
        Health  = 0
        IODE2   = 23
        IODE3   = 23
        codeL2  = 1
        L2P     = 0
```

Рисунок 1 — Значения эфемерид на заданную дату

На прошлом этапе эфемериды получены для даты 14.02.2022 0:00:00, Далее на суточном интервале (86400 секунд) по алгоритму из ИКД рассчитываются координаты спутника в системе ECEF WGS 84. Перевод координат в инерциальную систему осуществляется по заданным формулам:

$$x' = x\cos(\theta) - y\sin(\theta)$$

$$y' = x\sin(\theta) + y\cos(\theta)$$

$$z' = z$$

where

$$\theta = \dot{\Omega}_e\,(t - t_0)$$

Далее с помощью встроенной функции ecef2enu центр декартовой системы координат перемещается в точку приема, местоположение которой определяется с помощью RTKNAVI (рисунок 5). Полученные координаты спутника относительно приемника с учетом их знака пересчитываются в полярную систему координат с помощью известных соотношений:

$$r = \sqrt{x^2 + y^2 + z^2}\,,\quad \cos\theta = \frac{z}{\sqrt{x^2 + y^2 + z^2}},\quad \operatorname{tg}\varphi = \frac{y}{x}.$$

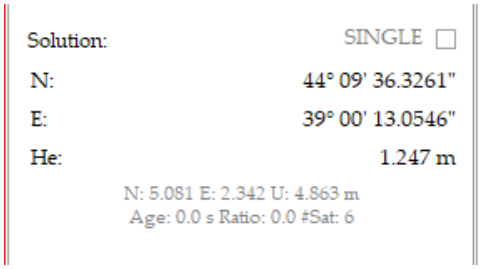

Рисунок 2 – Координаты приемника

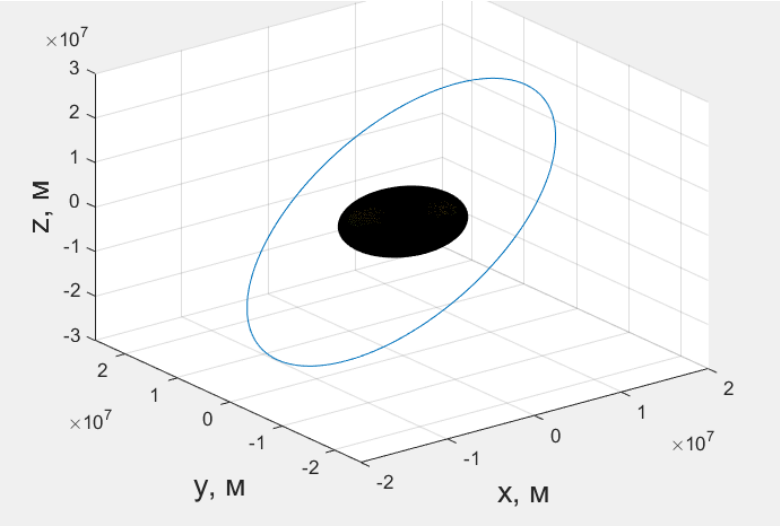

Рисунок 3 — Множество положений спутника GPS в системе ECI

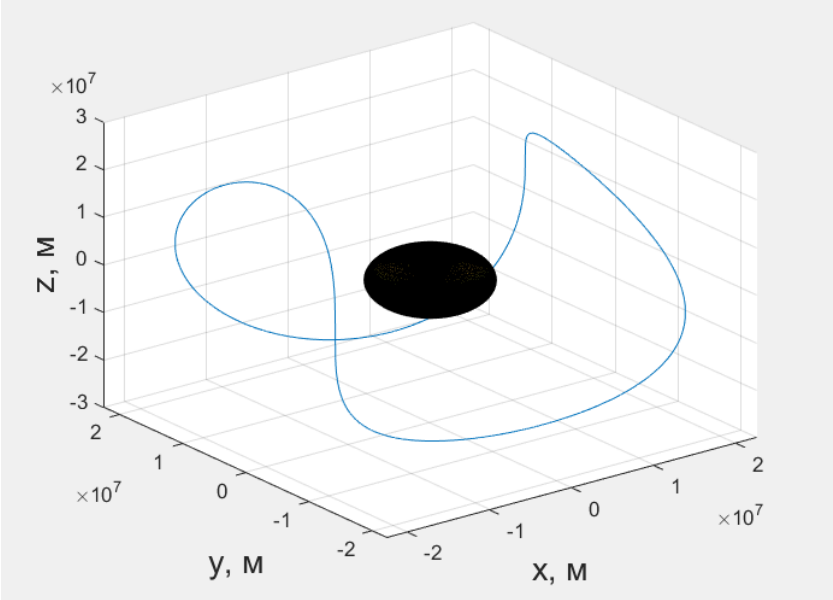

Рисунок 4 — Множество положений спутника GPS в системе ECEF WGS84

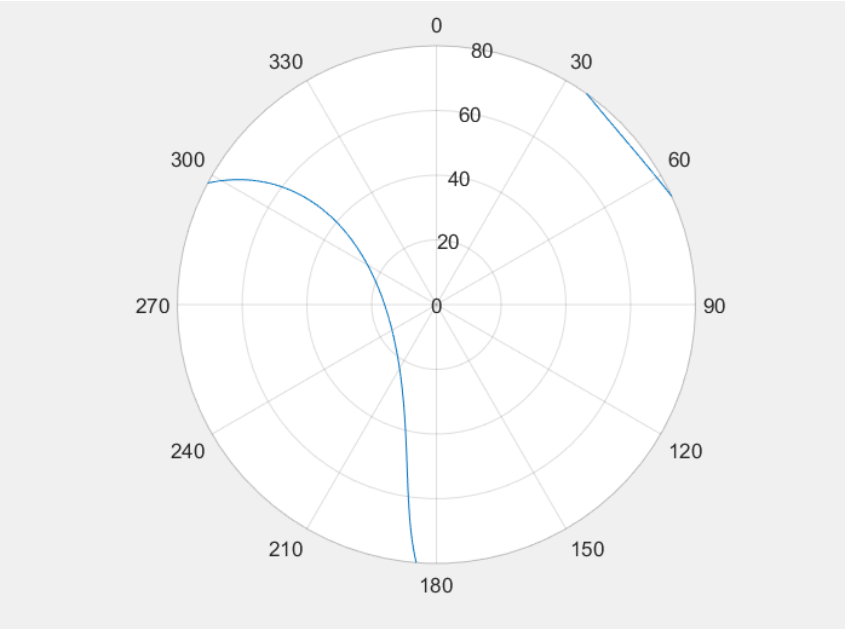

Рисунок 5 — Sky Plot

Для построения SkyView с помощью Trimble GNSS Planning Online необходимо выбрать заданный спутник, дату, рассматриваемый период, а также координаты точки приема. Полученные настройки приведены на рисунке 6.

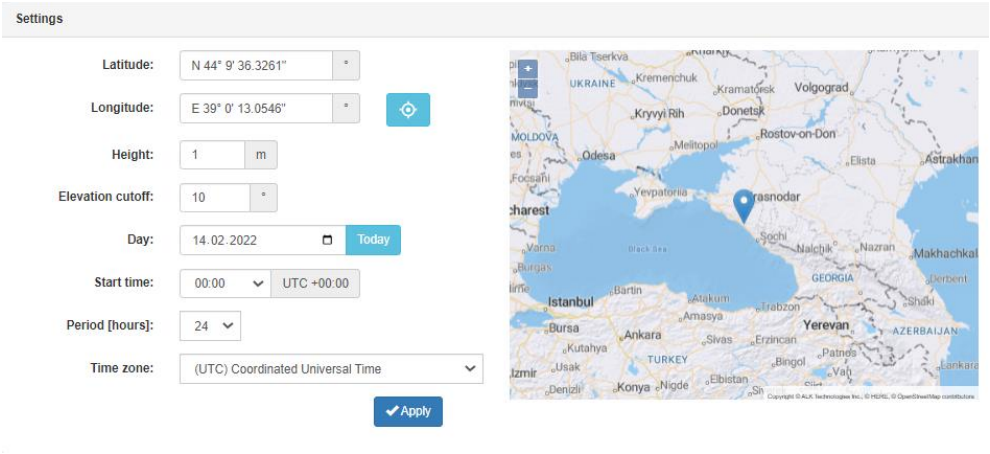

Рисунок 6 — Исходные данные для Trimble GNSS Planning Online

На рисунке 7 приведен SkyView спутника GPS №1 на интервале времени 14.02.2022 00:00 - 15.02.2022 00:00.

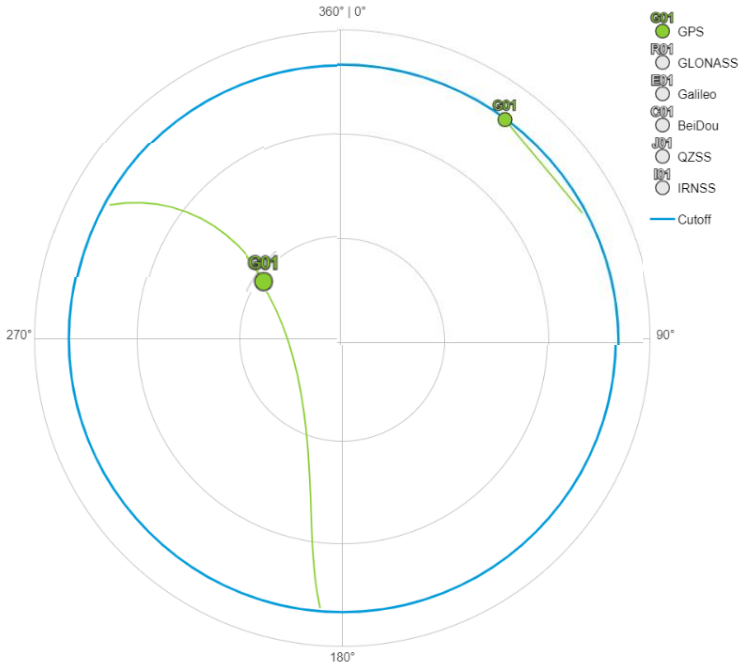

Рисунок 7 — SkyView из Trimble GNSS Planning Online

На втором этапе курсового проекта была разработана программа, выполняющая расчет положения спутника GPS, выводящая в файл out.txt значения координат спутника в заданном формате. При сравнении графиков на рисунках 5 и 7 замечается сходство рассчитанного SkyView и полученного с помощью Trimble GNSS Planning Online.

Приложение

```
clc
clear all
close all

% Константы из ИКД
pi = 3.1415326535898;
mu = 3.986005e+14; % Гравитационная постоянная
Omega_e_dot = 7.2921151467e-5; % Скорость вращения Земли

% Эфемериды
C_rs = 1.171875e+01; % Амплитуда поправочного члена синусоидальной гармоники к
радиусу орбиты
delta_n = deg2rad(2.4630e-7); % Среднее отклонение движения от вычисленного значения
M_0 = deg2rad(5.455674e+01); % Средняя аномалия
C_uc = 7.729977e-07; % Амплитуда поправочного члена косинусной гармоники к аргументу
широты
eks = 1.137974e-02; % Эксцентриситет
C_us = 1.152977e-06; % Амплитуда поправочного члена синусоидальной гармоники к
аргументу широты
rootA = 5.153669e+03; % Корень из большой полуоси
t_oe = 86400; % Опорное время эфемерид
C_ic = -1.173466e-07; % Амплитуда поправочного члена косинусной гармоники к углу
наклона
Omega_0 = deg2rad(-1.094355e+02); % Долгота узла
C_is = 1.136214e-07; % Амплитуда поправочного члена синусоидальной гармоники к углу
наклона
i_0 = deg2rad(5.653057e+01); % Наклонение
C_rc = 3.719688e+02; % Амплитуда поправочного члена косинусной гармоники к радиусу
орбиты
omega = deg2rad(5.051432e+01); % Аргумент перигея
Omega_dot = deg2rad(-4.812432e-07); % Скорость прямого восхождения
IDOT = deg2rad(-6.773462e-09); % Скорость угла наклона

% Расчет значений элементов Кеплера
A = rootA^2; % Большая полуось
n_0 = sqrt(mu/A^3); % Расчетное среднее движение

for i = 1:86400 % Суточный интервал рассмотрения
t = 86400+18 + i; % Количество секунд от начала текущей недели при начале
рассмотрения в 00:00:01
% Время от опорной эпохи эфемерид
t_k = t - t_oe;

if t_k>302400
 t_k = t_k - 604800;
elseif t_k<-302400
 t_k = t_k + 604800;
end

n = n_0 + delta_n; % Скорректированное среднее движение
M_k = M_0 + n * t_k; % Средняя аномалия
% Эксцентрическая аномалия
E_0 = M_k;
E_k = 0;
k = 0;

while abs(E_0 - E_k)>1e-8
 E_k = E_0;
 E_0 = E_0 + (M_k - E_0 + eks * sin(E_0))/(1 - eks * cos(E_0));
```

```
 k = k + 1;
end

nu_k = atan2((sqrt(1 - eks^2) * sin(E_k)/(1 - eks * cos(E_k))), ((cos(E_k) - eks)/(1
- eks * cos(E_k)))); % Истинная аномалия
Phi_k = nu_k + omega; % Аргумент широты
delta_u_k = C_us*sin(2*Phi_k) + C_uc*cos(2*Phi_k); % Аргумент поправки на широту
delta_r_k = C_rs*sin(2*Phi_k) + C_rc*cos(2*Phi_k); % Коррекция радиуса
delta_i_k = C_is*sin(2*Phi_k) + C_ic*cos(2*Phi_k); % Коррекция наклона
u_k = Phi_k + delta_u_k; % Скорректированный аргумент широты
r_k = A * (1 - eks * cos(E_k)) + delta_r_k; % Скорректированный радиус
i_k = i_0 + delta_i_k + IDOT * t_k; % Скорректированное наклонение

% Позиция в орбитальной плоскости
x_k_sh = r_k * cos(u_k);
y_k_sh = r_k * sin(u_k);

Omega_k = Omega_0 + (Omega_dot - Omega_e_dot) * t_k - Omega_e_dot * t_oe; %
Скорректированная широта восходящего узла

% Расчет координат спутника
x_k = x_k_sh * cos(Omega_k) - y_k_sh * cos(i_k) * sin(Omega_k);
y_k = x_k_sh * sin(Omega_k) + y_k_sh * cos(i_k) * cos(Omega_k);
z_k = y_k_sh * sin(i_k);

x_current(1,i) = x_k;
y_current(1,i) = y_k;
z_current(1,i) = z_k;

% Инерциальная система координат
theta = Omega_e_dot * t_k;
x_k_1 = x_k * cos(theta) - y_k * sin(theta);
y_k_1 = x_k * sin(theta) + y_k * cos(theta);
z_k_1 = z_k;

x_converted(1,i) = x_k_1;
y_converted(1,i) = y_k_1;
z_converted(1,i) = z_k_1;

% Данные из RTKNAVI (координаты приемника)
B = deg2rad(44.09363261); % Широта
L = deg2rad(39.00130546); % Долгота
H = 1.247; % Высота

[x(i), y(i), z(i)] =
ecef2enu(x_current(1,i),y_current(1,i),z_current(1,i),B,L,H,wgs84Ellipsoid,'radians')
; % Отображение координат спутника относительно точки приема

% Полярная система координат
if z(i) > 0
rho(i) = sqrt(x(i)^2 + y(i)^2 + z(i)^2);
theta_1(i) = acos(z(1,i)/rho(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 theta_1(i) = NaN;
     rho(i) = NaN;
     phi(i) = NaN;
```

```
end
end

% Вывод значений координат
out = fopen('../out.txt', 'w+');
for i = 1:length(x_current)
    fprintf(out, '%6.0f %10.10f %10.10f %10.10f\n', i, x_current(1,i),
y_current(1,i), z_current(1,i));
end
fclose(out);

% Вывод графиков
R = 6378136; % Радиус Земли
[x_sphere, y_sphere, z_sphere] = sphere(100);
x_Earth=R*x_sphere;
y_Earth=R*y_sphere;
z_Earth=R*z_sphere;

figure (1)
subplot (1,1,1)
surf(x_Earth,y_Earth,z_Earth);
hold on;
plot3(x_current(1,:), y_current(1,:), z_current(1,:));
xlabel('x, м','FontSize',16);
ylabel('y, м','FontSize',16);
zlabel('z, м','FontSize',16);

figure (2)
subplot (1,1,1)
surf(x_Earth,y_Earth,z_Earth);
hold on;
plot3(x_converted(1,:), y_converted(1,:), z_converted(1,:));
xlabel('x, м','FontSize',16);
ylabel('y, м','FontSize',16);
zlabel('z, м','FontSize',16);

figure (3)
axes = polaraxes;
polarplot(axes,phi,rad2deg(theta_1))
axes.ThetaDir = 'clockwise';
axes.ThetaZeroLocation = 'top';
rlim([0 80]);
```