

```

clear all
close all
clc

%% Initialize Parameters
mu = 3.986004418e14; % m^3/s^2
GS_1= [-111.536, 35.097, 2.206]; % Geodetic coordinates (lat [deg], long [deg], alt [km])
GS_2= [-70.692, -29.016, 2.380]; % Geodetic coordinates (lat [deg], long [deg], alt [km])
arcsec2rad = (pi / 648000); % Conversion factor from arcsecs to radians
R_GS1= diag([1, 1, 0.01, 0.01].* (arcsec2rad^2)); % Covariance of GS1
R_GS2= diag([0.01, 0.01, 0.0001, 0.0001].* (arcsec2rad^2)); % Covariance of GS2
R_GPS= diag([5000^2, 5000^2, 5000^2, .5^2, .5^2, .5^2]); % Covariance of GPS
txt = '2024-11-24T05:04:30.000';
Epoch = datetime(txt, 'InputFormat', 'yyyy-MM-dd'T'HH:mm:ss.SSS');

%% Read CSV and Dictate Measurement Units (m, m/s)
filename = 'measurements.csv';
data = readtable(filename);

% Split data by sensor ID
sensor_ids = unique(string(data.sensor_id));
GPS_Data = data(string(data.sensor_id) == 'gps_measurement', :);
GS1_Data = data(string(data.sensor_id) == 'ground_observer_1', :);
GS2_Data = data(string(data.sensor_id) == 'ground_observer_2', :);

% Convert to matrices
GPS_Obs = table2array(GPS_Data(:, 3:8))*1000; %Convert from km to meters
t_gps = table2array(GPS_Data(:, 1));
GS1_Obs = deg2rad(table2array(GS1_Data(:, 9:end))); %Convert from Angles to Radians
t_gs1 = table2array(GS1_Data(:, 1));
GS2_Obs = deg2rad(table2array(GS2_Data(:, 9:end))); %Convert from Angles to Radians
t_gs2 = table2array(GS2_Data(:, 1));

%% Filters
% Batch Filter for Ground Station 1
[r_GS1, v_GS1, rv_cov_GS1, Ref_t_GS1] = batch_angles_filter(t_gs1, GS1_Obs, R_GS1, t_gps, GPS_Obs, R_GPS, Epoch);

% Batch Filter for Ground Station 2
[r_GS2, v_GS2, rv_cov_GS2, Ref_t_GS2] = batch_angles_filter(t_gs2, GS2_Obs, R_GS2, t_gps, GPS_Obs, R_GPS, Epoch);

% Expanded Sequential Filter for GPS Measurements
[r_GPS, v_GPS, rv_cov_GPS, Ref_t_GPS] = sequential_rv_filter(t_gps, GPS_Obs, R_GPS, mu, Epoch);
% [kep_est, rv_cov] = batch_gps_multiarc(t_gps, GPS_Obs, R_GPS, 40*60, Epoch);

% Keplerian Orbital Elements
[a_GPS, e_GPS, inc_GPS, RAAN_GPS, w_GPS, nu_GPS] = ijk2keplerian(r_GPS, v_GPS);
[a_GS1, e_GS1, inc_GS1, RAAN_GS1, w_GS1, nu_GS1] = ijk2keplerian(r_GS1, v_GS1);
[a_GS2, e_GS2, inc_GS2, RAAN_GS2, w_GS2, nu_GS2] = ijk2keplerian(r_GS2, v_GS2);

```

```

%% Fusing of Esimtates (Using Kalman Filter)
% Make sure they are all sorted
ref_times = [Ref_t_GS1, Ref_t_GS2, Ref_t_GPS];
All_r = [r_GS1, r_GS2, r_GPS];
All_v = [v_GS1, v_GS2, v_GPS];
All_P = cat(3, rv_cov_GS1, rv_cov_GS2, rv_cov_GPS);
[~, idx] = sort(ref_times);
All_r = All_r(:, idx);
All_v = All_v(:, idx);
All_P = All_P(:, :, idx);
ref_times = ref_times(idx);

% Extended Sequential filter
[r_fused, v_fused, rv_cov_fused, Ref_t_fused] = sequential_rv_filter(ref_times, ✓
[All_r' All_v'], All_P, mu, Epoch);
[a_fused, e_fused, inc_fused, RAAN_fused, w_fused, nu_fused] = ijk2keplerian(r_fused, ✓
v_fused);

% Estimated Orbit Calculatiosn
Period = 2*pi*sqrt(a_fused^3/mu);
perigee = a_fused*(1-e_fused);
apogee = a_fused*(1+e_fused);
J=cart2kepJac(r_fused, v_fused); % Jacobian car2kep
rv_cov_fused_oe = J * rv_cov_fused * J';

%% Plots
% Plot Orbit
[~, ~, r_prop, v_prop, time_prop] = Propagate(Period, r_fused, v_fused, 1);

% Plot Orbit and Change Colors of orbit pased on LOS
GS_Pos = plot_ground_station([GS_1; GS_2], 1);
plotOrbitVisibility(a_fused, e_fused, inc_fused, RAAN_fused, w_fused, nu_fused, GS_Pos);

% Plot Uncertainty Ellipsoid
cats=cat(3, diag(diag(rv_cov_GS1(1:3, 1:3))), diag(diag(rv_cov_GS2(1:3, 1:3))), diag ✓
(diag(rv_cov_GPS(1:3, 1:3))), diag(diag(rv_cov_fused(1:3, 1:3))));
plot_measurements_with_uncertainty([r_GS1'/1000; r_GS2'/1000; r_GPS'/1000; ✓
r_fused'/1000], cats)

%% Realign fused estiamtion to t=0 position
[r_start_fuse, v_start_fuse, ~, ~, ~] = Propagate(Period-12000, r_fused, v_fused, 0, ✓
1);
[~, ~, r_prop, v_prop, time_prop] = Propagate(Period, r_start_fuse, v_start_fuse, 0, ✓
1);

%% Range Calculations for GS1, GS2, General
relative_r_gs1 = r_prop - r_GS1';
relative_r_gs2 = r_prop - r_GS2';
relative_r_gs1_norm = vecnorm(relative_r_gs1, 2, 2)/1000;
relative_r_gs2_norm = vecnorm(relative_r_gs2, 2, 2)/1000;
relative_r_gs1_norm_meas = round(relative_r_gs1_norm((t_gs1(1)+1):(t_gs1(2)-t_gs1 ✓
(1)):(t_gs1(end)+1)));
relative_r_gs2_norm_meas = round(relative_r_gs2_norm((t_gs2(1)+1):(t_gs2(2)-t_gs2 ✓

```

```
(1):(t_gs2(end)+1)));
r_norm = vecnorm(r_prop, 2, 2)/1000;
r_norm_GPS_meas = round(r_norm((t_gps(1)+1):(t_gps(2)-t_gps(1)):(t_gps(end)+1)));

%% Azimuth and Elevation Plots
% Calculate
[el1, az1] = elevazim(r_prop,GS_Pos(1,:));
[el2, az2] = elevazim(r_prop,GS_Pos(2,:));
% Plot
result1 = EL_AZ_Ploter(el1',az1');
result2 = EL_AZ_Ploter(el2',az2');

%% UKF and MC Propagation
% Propagation
[x_mc, P_mc] = monte_carlo_propagation(r_fused, v_fused, rv_cov_fused, 5*3600, 100);
[x_ukf, P_ukf] = ukf_propagation([r_fused; v_fused] , rv_cov_fused, 5*3600);

% Kepler Elements
[a_ukf,e_ukf,inc_ukf,RAAN_ukf,w_ukf,nu_ukf] =ijk2keplerian(x_ukf(1:3), x_ukf(4:6));
[a_mc,e_mc,inc_mc,RAAN_mc,w_mc,nu_mc] =ijk2keplerian(x_mc(1:3), x_mc(4:6));

% Plot uncertainty
[~]= Propagate(Period, r_fused, v_fused, 1);
plot_measurements_with_uncertainty([x_ukf(1:3)'/1000;x_mc(1:3)'/1000], cats_prop,✓
[{'UKF Uncertainty'}, {'MC Uncertainty'}])
```