```
function [Pos, Vel, P_f, refrence_time] = sequential_rv_filter(times, measurements,↙
R_obs, mu, Epoch)
%-------------------------------------------------------------------------
% Extended Kalman Filter for orbit estimation from ECI position and
% velocity
%-------------------------------------------------------------------------
% Inputs:
%   times           [Nx1]   times in seconds of observations (assumes
%                           ascending order
%
%   measurements    [Nx4]   [r, v] in [m] and [v/s]
%
%   R_obs           [4x4]   covariance of each measurement (constant diagonal)
%
%   mu               [1]    Gravitational Parameter [m^3/s^2 ]
%
%   Epoch           datetime    The Epoch of all the measurements

% Outputs:
%   Pos     [3x1]  Estimated final position in ECI [m]
%
%   Vel     [3x1]  Estimated final velcoity in ECI [m]
%
%   P_f     [6x6] covariance of coodinate system [m^2 and (m/s)^2]
%
%   refrence_time  The time [s] of the final calcualted estimated trajectory
%
% Assumptions:
%   - initial state is in the ECI frame
%   - Keplerian two-body motion (no perturbations)

N = length(times);

% Initialization
X_0_star = measurements(1,:)';  % Refrence trajectory is first measurments
X_k_star = X_0_star;
P_k = R_obs(:,:,1);

% Initialize for History
X_k_hist = zeros(6, N);
X_k_hist(:, 1) = X_k_star;
P__k_hist = zeros(6, 6, N);
P__k_hist(:,:,1) = P_k;

% Initialize Times
t_prev = times(1);
Epoch_prev = Epoch;
dt_max = 5*60;  % 15 minutes in seconds largest propogation interval

for k = 2:N

    % Propagation Step
    dt_total = times(k) - t_prev;
```

```
    % Splitting if dt is to large:
    if dt_total<=dt_max
        %Single step propogation
        Phi = Kepler_STM_UV(dt_total, X_k_star, mu, 'seconds');
        X_pred = Phi * X_k_star;
        P_k = Phi* P_k *Phi';
    else
        N_steps = ceil(dt_total / dt_max);
        dt_step = dt_total / N_steps;
        X_temp = X_k_star;
        P_temp = P_k;
        for i = 1:N_steps
            Phi_step = Kepler_STM_UV(dt_step, X_temp, mu, 'seconds');
            X_temp = Phi_step * X_temp;
            P_temp = Phi_step * P_temp * Phi_step';
        end
        X_pred = X_temp;
        P_k = P_temp;
    end

    % Measurements Reading
    Y_k = measurements(k,:)';
    G = eye(length(measurements(1,:)));  % Measruements in r and v already
    y_k = Y_k - G*X_pred;
    H_k_tilda = eye(6);

    % Kalman gain
    if (isscalar(R_obs(1,1,:)))
        R = R_obs;
    else
        R = R_obs(:,:,k);
    end
    S = H_k_tilda * P_k * H_k_tilda' + R;
    K = P_k * H_k_tilda' / S;

    % Update
    x_k_hat = K * y_k;
    X_k_star = X_pred + x_k_hat;
    I = eye(6);
    P_k = (I - K*H_k_tilda) * P_k;
    Epoch_prev = Epoch_prev+seconds(dt_total);
    t_prev = times(k);

    % Save
    X_k_hist(:,k) = X_k_star;
    P__k_hist(:,:,k) = P_k;
end

X_f_star =  X_k_hist(:, end);
Pos = X_f_star(1:3);
Vel = X_f_star(4:6);
P_f = P__k_hist(:,:,end);
refrence_time = t_prev;
end
```