Project 1-Kalman Filter

Instruction

Load data

- **Sample Matlab Code**

```
%Initialization
clc, close all, clear

%Load data and correct IMU initial offset
[time, data] = rtpload(‘EKF_DATA3.txt'); %data of the circle in front of Engineering Building
```

Read data

```
%Get Odometry IMU and GPS data (x, y, theta, covariance)
Odom_x = data.O_x;
Odom_y = data.O_y;
Odom_theta = data.O_t;
Gps_x = data.G_x;
Gps_y = data.G_y;
IMU_heading = data.I_t;
```

IMU calibration and plot data

```
% Calibrate IMU to match with the robot's heading initially
IMU_heading = IMU_heading +(0.32981-0.237156)*ones(length(IMU_heading),1); %For EKF_DATA3

 figure(1), plot(time, data.O_t)
 figure(2), plot(time, data.I_t)
 figure(3), plot(data.O_x, data.O_y, 'x')
 figure(4), plot(time, IMU_heading)
```

```
%Velocity of the robot
V = 0.14;%0.083;
%Distance between 2 wheels
L = 1; %meter
%Angular Velocity
Omega = V*tan(Odom_theta(1))/L;
%set time_step
delta_t = 0.001;
%total_step
total=1:length(Odom_x);
```

$$\mathbf{X}(k) = \mathbf{A}(k)\mathbf{X}(k-1)$$

$$\begin{bmatrix} x_r(k) \\ y_r(k) \\ v_r(k) \\ \theta_r(k) \\ \omega(k) \end{bmatrix} = \begin{bmatrix} 1 & 0 & \Delta t cos(\theta(k-1)) & 0 & 0 \\ 0 & 1 & \Delta t sin(\theta(k-1)) & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & \Delta t \\ 0 & 0 & 0 & 0 & 1 \end{bmatrix} \times \begin{bmatrix} x_r(k-1) \\ y_r(k-1) \\ v_r(k-1) \\ \theta_r(k-1) \\ \omega(k-1) \end{bmatrix}$$

Initialization

```
s.x = [Odom_x(1); Odom_y(1); V; Odom_theta(1); Omega]; %Enter State (1x5)

%Enter transistion matrix A (5x5)

s.A = [1 0 delta_t*cos(Odom_theta(1)) 0 0;
       0 1 delta_t*sin(Odom_theta(1)) 0 0;
       0 0 1                  0 0;
       0 0 0                  1 delta_t;
       0 0 0                  0 1];
```

```
%Define a process noise (stdev) of state: (Student can play with this number)

%Enter covariance matrix Q (5x5) for state x

s.Q = [.00004  0   0   0   0;
       0  .00004  0   0   0;
       0   0   .0001  0   0;
       0   0    0  .0001  0;
       0   0    0   0  .0001];
```

Initialization

```
%Define the measurement matrix H:
%Enter measurement matrix H (5x5) for measurement model z
s.H = [ 1   0   0   0   0;
      0   1   0   0   0;
      0   0   1   0   0;
      0   0   0   1   0;
      0   0   0   0   1];
```

Initialization

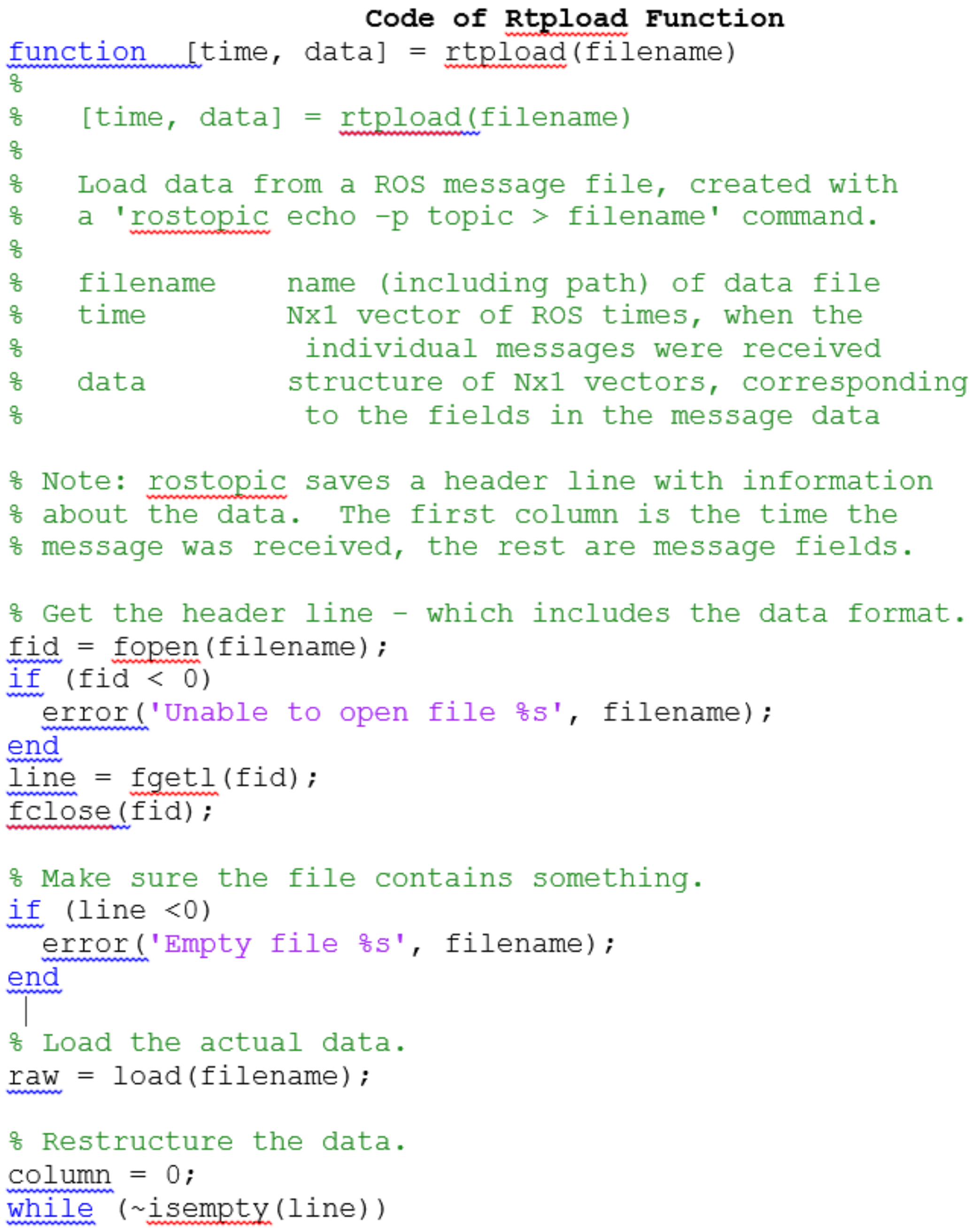
```
%Define a measurement error (stdev)
%Enter covariance matrix R (5x5) for measurement model Z
s.R = [.04  0   0   0   0;
       0  .04  0   0   0;
       0   0  .01  0   0;
       0   0   0   0.01  0;
       0   0   0   0  .01];
```

Initialization

```
%B matrix initialization:
s.B = [ 1   0   0   0   0;
        0   1   0   0   0;
        0   0   1   0   0;
        0   0   0   1   0;
        0   0   0   0   1];
```

Initialization

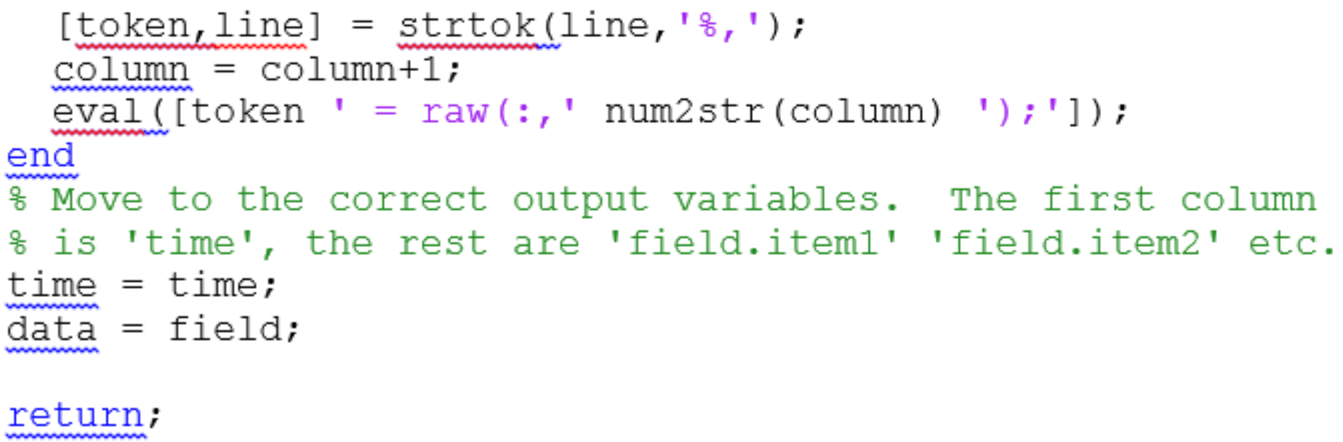
```
%Enter initial value of u (5x5)
s.u = [0; 0; 0; 0; 0];

%Enter initial covariance matrix P (5x5)
s.P = [.01  0   0   0   0;
         0  .01  0   0   0;
         0   0  .01  0   0;
         0   0   0  .01  0;
         0   0   0   0  .01];
```



Code of Rtpload Function

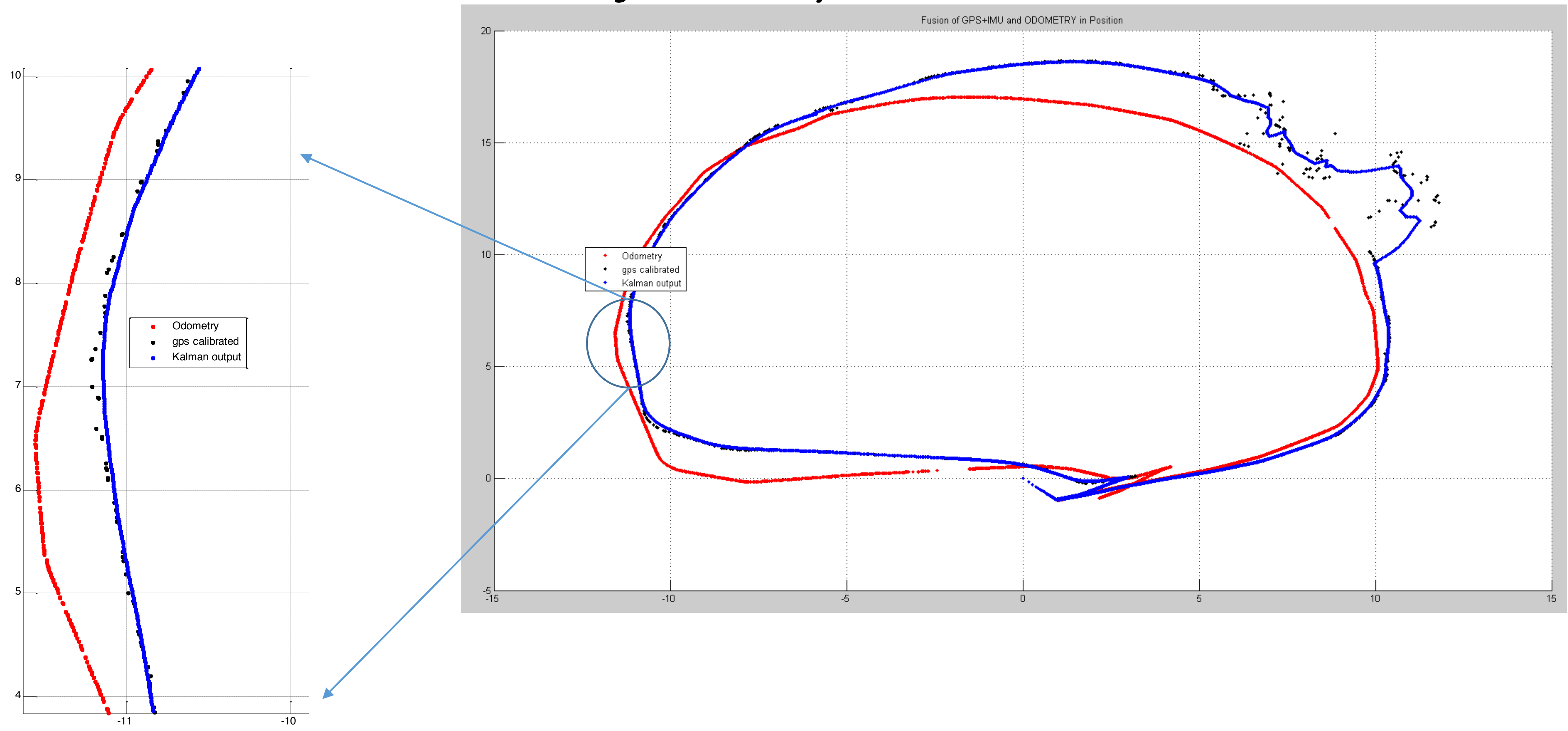
```
function  [time, data] = rtpload(filename)
%
%   [time, data] = rtpload(filename)
%
%   Load data from a ROS message file, created with
%   a 'rostopic echo -p topic > filename' command.
%
%   filename    name (including path) of data file
%   time        Nx1 vector of ROS times, when the
%                individual messages were received
%   data        structure of Nx1 vectors, corresponding
%                to the fields in the message data

% Note: rostopic saves a header line with information
% about the data.  The first column is the time the
% message was received, the rest are message fields.

% Get the header line - which includes the data format.
fid = fopen(filename);
if (fid < 0)
  error('Unable to open file %s', filename);
end
line = fgetl(fid);
fclose(fid);

% Make sure the file contains something.
if (line <0)
  error('Empty file %s', filename);
end

% Load the actual data.
raw = load(filename);

% Restructure the data.
column = 0;
while (~isempty(line))
```

Rtpload (Continued)

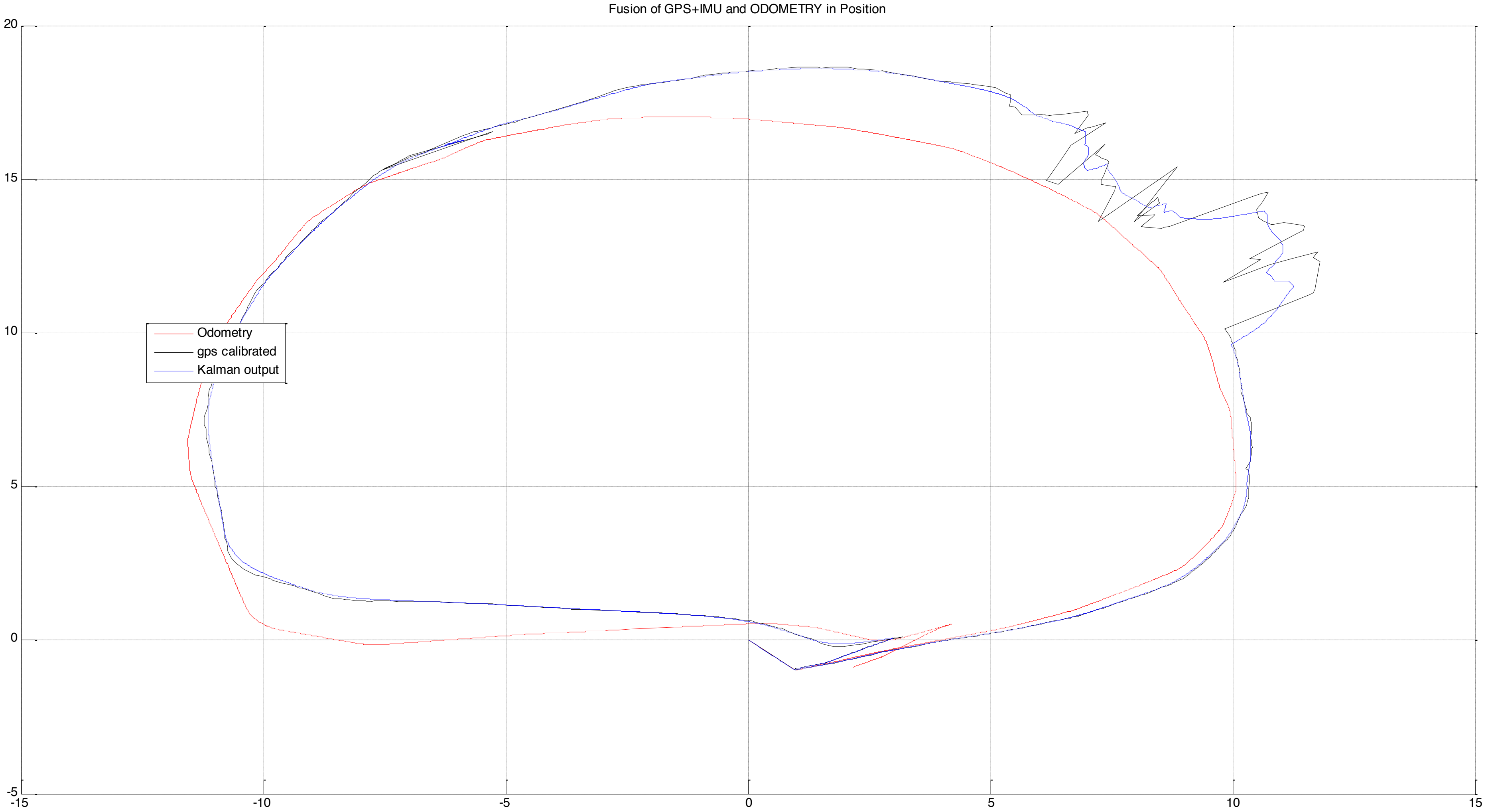


```
  [token,line] = strtok(line,'%,');
  column = column+1;
  eval([token ' = raw(:,' num2str(column) ');']);
end
% Move to the correct output variables.  The first column
% is 'time', the rest are 'field.item1' 'field.item2' etc.
time = time;
data = field;

return;
```

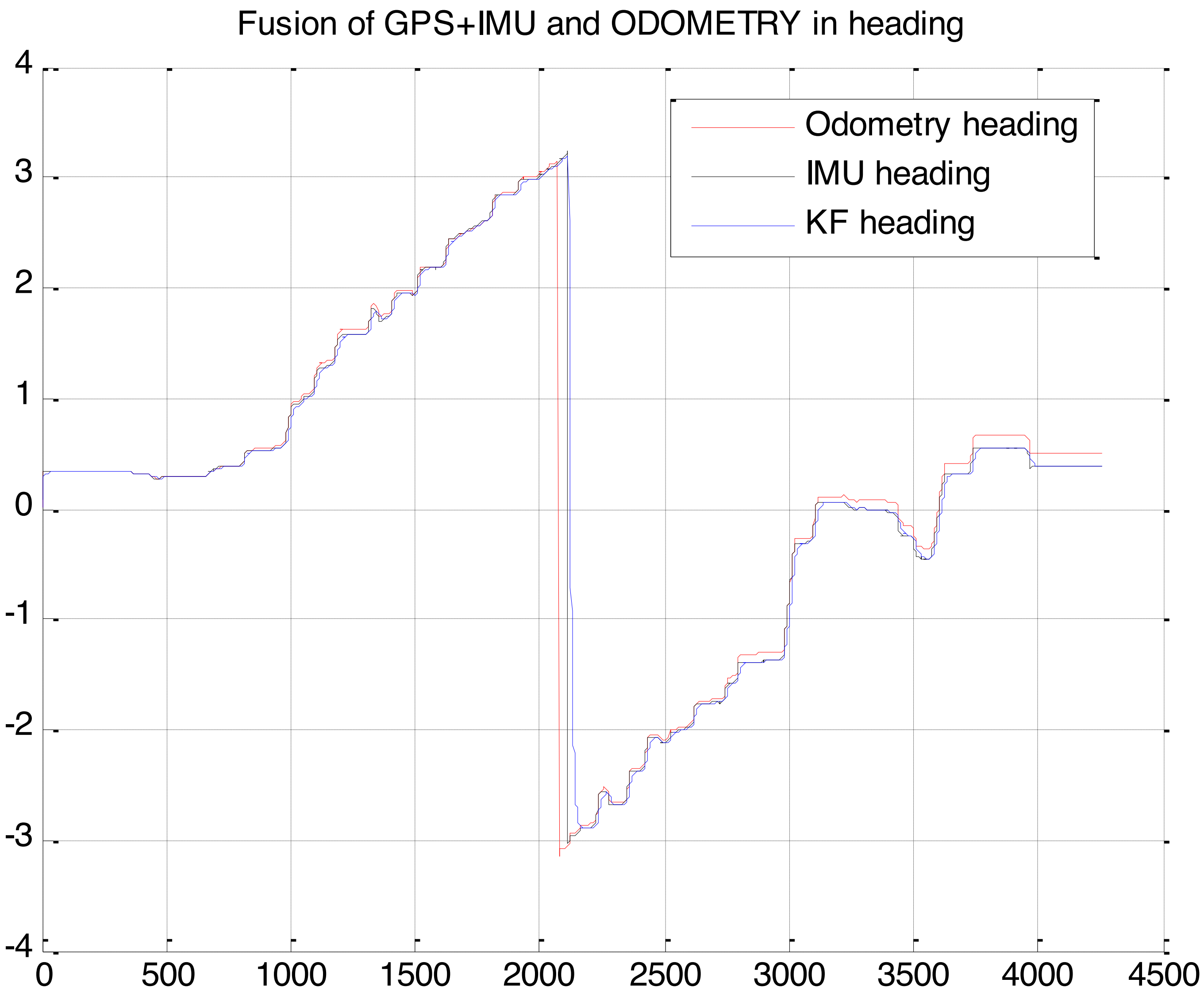
Results: Position/Trajectory



Results: Position/Trajectory



Heading


Fusion of GPS+IMU and ODOMETRY in heading
4
3
2
1
0
-1
-2
-3
-4
0
500
1000
1500
2000
2500
3000
3500
4000
4500
Odometry heading
IMU heading
KF heading