Robot Vision and Navigation : Coursework 1

Integrated Navigation for a Robotic Lawnmower

Thomas Luo

1 Method Description

In this coursework, I were provided with 3 files : one file containing the pseudo ranges of the different satellites at different time t, another file containing the pseudo range rates of the different satellites at different time t and one last file composed of measurements from the sensors of the lawnmower (wheels, gyroscope and compass). With the first and second files, I could compute the GNSS positions and velocities of the lawnmower. With the third file, I could compute the different positions and velocities of the lawnmower relatively to its initial positions and velocities. Then, using these 2 results, I computed an integrated horizontal-only DR/GNSS navigation solution using Kalman filtering : I used the GNSS solution to correct the Dead Reckoning solution.

To compute the heading, I used the gyroscope and compass data provided in the third file : I used the Magnetic heading solution to correct the gyroscope derived heading solution by using a 2 state Kalman filter.

All these solutions were computed using the algorithm presented during the lectures or the workshops.

2 Algorithm

First, I computed the GNSS positions and velocities solutions of the lawnmower. To do that, I used a least-square estimation method to initialize the lawnmower's positions (x_0) and velocities (v_0)[1] :

Algorithm 1 Initialize GNSS solution (x_2, v_0)

procedure INITIALISATION
    $x_0 \leftarrow (0, 0, 0)$
    $v_0 \leftarrow (0, 0)$
    For each satellite, computes its positions and velocities
    distance $\leftarrow$inf
    while distance $\geq \epsilon$
        for each satellites[2]
            compute its Sagnac effect compensation matrix
            compute its predicted range and predicted range rate from the user position
            compute its line-of-sight unit vector from the user position
        end for
        Compute predicted state vector positions x'[3] *and velocities v'*[4]
        Compute measurement innovation vector range dz[3] *and range rate dz'*[4]
        Compute measurement matrix[3] H
        $x \leftarrow x' + (H^T H)^{-1} H^T dz$
        $v \leftarrow v' + (H^T H)^{-1} H^T dz'$
        $distance \leftarrow norm(x_0 - x)$
        $x_0 \leftarrow x$
        $v_0 \leftarrow v$
    end while

This algorithm stop when 2 predicted positions of the lawnmower are close enough to each other. Then, we will say that there is convergeance. Then, I apply a GNSS Kalman filter[5] to the initialized positions and velocities :

Algorithm 2 compute GNSS solution with 8 state Kalman filter

procedure INITIALISATION

x_0,v_0

$\tau \leftarrow 0.5$

initialise state vector $X_0 = (x_0, v_0, d\rho_a, d\rho_b)^T$

initialise error covariance matrix P

for each epoch k=2..n

$$\Phi_{k-1} \leftarrow \begin{pmatrix} I_3 & \tau I_3 & 0_{3,1} & 0_{3,1} \\ 0_3 & I_3 & 0_{3,1} & 0_{3,1} \\ 0_{1,3} & 0_{1,3} & 1 & \tau \\ 0_{1,3} & 0_{1,3} & 0 & 1 \end{pmatrix}$$

$$Q_{k-1} \leftarrow \begin{pmatrix} \frac{1}{3}S_a\tau^3 I_3 & \frac{1}{2}S_a\tau^2 I_3 & 0_{3,1} & 0_{3,1} \\ \frac{1}{2}S_a\tau^2 I_3 & S_a\tau I_3 & 0_{3,1} & 0_{3,1} \\ 0_{1,3} & 0_{1,3} & S_{c\phi}\tau + \frac{1}{3}S_{cf}\tau^3 & \frac{1}{2}S_{cf}\tau^2 \\ 0_{1,3} & 0_{1,3} & \frac{1}{2}S_{cf}\tau^2 & S_{cf}\tau \end{pmatrix}$$

$x_k \leftarrow \Phi_{k-1} X_{k-1}$

$P_k \leftarrow \Phi_{k-1} P \Phi_{k-1}^T + Q_{k-1}$

for each satellites[2] $j = 1..m$

compute its Sagnac effect compensation matrix

compute its predicted range r_j and predicted range rate r'_j from the user position

compute its line-of-sight unit vector u_j from the user position

end for

$$H_k \leftarrow \begin{pmatrix} -u_{1,x} & -u1,y & -u_{1,z} & 0 & 0 & 0 & 1 & 0 \\ -u_{2,x} & -u2,y & -u_{2,z} & 0 & 0 & 0 & 1 & 0 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ -u_{m,x} & -um,y & -u_{m,z} & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & -u_{1,x} & -u1,y & -u_{1,z} & 0 & 1 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ 0 & 0 & 0 & -u_{m,x} & -um,y & -u_{m,z} & 0 & 1 \end{pmatrix}$$

$$R_k \leftarrow \begin{pmatrix} \sigma_p^2 & 0 & \cdots & 0 & 0 & 0 & \cdots & 0 \\ 0 & \sigma_p^2 & \cdots & 0 & 0 & 0 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & \sigma_p^2 & 0 & 0 & \cdots & 0 \\ 0 & 0 & \cdots & 0 & \sigma_r^2 & 0 & \cdots & 0 \\ 0 & 0 & \cdots & 0 & 0 & \sigma_r^2 & \cdots & 0 \\ 0 & 0 & \cdots & 0 & 0 & 0 & \cdots & \sigma_r^2 \end{pmatrix}$$

$K_k \leftarrow P_k H_k^T (H_k P_k H_k^T + R_k)^{-1}$

$$dz \leftarrow \begin{pmatrix} \rho_1 - r_1 - d\rho_a \\ \vdots \\ \rho_m - r_m - d\rho_a \\ \rho'_1 - r'_1 - d\rho_b \\ \vdots \\ \rho'_m - r'_m - d\rho_b \end{pmatrix}$$

$X_k \leftarrow x_k + K_k dz$

$P \leftarrow (I - K_k H_k) P_k$

end for

with ρ_a and ρ_b clock offset and clock drift, τ propagation time, S_a acceleration power spectral density (PSD), $S_{c\phi}$ clock phase PSD, S_{cf} clock frequency PSD, ρ_j measured pseudo range of satellite j and ρ'_j

measured pseudo range rate of satellite j.

Secondly, I computed the lawnmower positions and velocities using by dead car reckoning before corrected it with the GNSS solutions. Since, the dead reckoning solution depend on the heading, I corrected the gyroscope derived heading solution using the magnetic heading[5]. To do that, I used a 2 state Kalman Filter :

Algorithm 3 Corrected gyro-derived heading solution with 2 state Kalman filter

procedure INITIALISATION
- $X_0 = (\delta\Psi \ b_g)^T = (0 \ 0)$
- $(\Psi^G)_i$ gyro-derived heading solution
- $\tau \leftarrow 0.5$
- *initialise error covariance matrix P*
- *for each epoch k=2..n*
  - $\Phi_{k-1} \leftarrow \begin{pmatrix} 1 & \tau_s \\ 0 & 1 \end{pmatrix}$
  - $Q_{k-1} \leftarrow \begin{pmatrix} S_{rg}\tau + \frac{1}{3}S_{bgd}\tau^3 & \frac{1}{2}S_{bgd}\tau^2 \\ \frac{1}{2}S_{bgd}\tau^2 & S_{bgd}\tau \end{pmatrix}$
  - $x_k \leftarrow \Phi_{k-1}X_{k-1}$
  - $P_k \leftarrow \Phi_{k-1}P\Phi_{k-1}^T + Q_{k-1}$
  - $H_k \leftarrow \begin{pmatrix} -1 & 0 \end{pmatrix}$
  - $R_k \leftarrow \sigma_M^2$
  - $K_k \leftarrow P_k H_k^T (H_k P_k H_k^T + R_k)^{-1}$
  - $dz \leftarrow (\Psi^M - \Psi^G) - H_k x_k$
  - $X_k \leftarrow x_k + K_k dz$
  - $P \leftarrow (I - K_k H_k)P_k$
  - $\Psi_k^G \leftarrow \Psi_k^G - \delta\Psi$
- *end for*

with $\delta\Psi$ the heading measurement error, S_{rg} the gyroscope random noise with PSD, S_{bgd} gyroscopte bias variation, σ_M the magnetic heading noise variance, Ψ^M the magnetic heading solution and Ψ^G the gyro-derived heading solution.

with these heading solutions, I compute the dead reckoning solutions6[6] position x and velocities v:

Algorithm 4 Dead reckoning navigation solution

procedure INITIALISATION
- $(\Psi)_i$ gyro-derived heading solution
- $x_0 = (L_0, \lambda_0) = GNSSinitialization$
- $v_0 = GNSSinitialization$
- $(v_{N,0}, v_{E,0}) \leftarrow (v_0 cos(\Psi_0), v_0 sin(\Psi_0))$
- $\tau \leftarrow 0.5$
- *for each epoch k=2..n*
  - $\begin{pmatrix} v'_{N,k} \\ v'_{E,k} \end{pmatrix} \leftarrow \frac{1}{2} \begin{pmatrix} cos(\Psi_k) + cos(\Psi_{k-1}) & 0 \\ 0 & sin(\Psi_k) + sin(\Psi_{k-1}) \end{pmatrix} \begin{pmatrix} v'_{N,k-1} \\ v'_{E,k-1} \end{pmatrix}$
  - compute meridian radius of curvature R_N and transverse radius of curvature R_E
- $L_k \leftarrow L_{k-1} + \frac{v'_{N,k}\tau}{R_N}$
- $\lambda_k \leftarrow \lambda_{k-1} + \frac{v'_{E,k}\tau}{R_E cos(L_k)}$
- $v_{N,k} \leftarrow 2v'_{N,k} - v_{N,k-1}$
- $v_{E,k} \leftarrow 2v'_{E,k} - v_{E,k-1}$
- *end for*

Once I have computed the GNSS and dead reckoning navigation solutions, I can now compute the corrected dead reckoning navigation solution using a 4 state Kalman Filter[7] :

Algorithm 5 Corrected dead reckoning solution with 4 state Kalman filter

procedure INITIALISATION

$X_0 = (\delta v_N \ \delta v_E \ \delta L \ \delta\lambda)^T = (0\ 0\ 0\ 0)$

$$P_0 \leftarrow \begin{pmatrix} \sigma_v^2 & 0 & 0 & 0 \\ 0 & \sigma_v^2 & 0 & 0 \\ 0 & 0 & \frac{\sigma_r^2}{R_N} & 0 \\ 0 & 0 & 0 & \frac{\sigma_r^2}{R_E cos(L_0)} \end{pmatrix}$$

$\tau \leftarrow 0.5$

for each epoch k=2..n

$$\Phi_{k-1} \leftarrow \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ \frac{\tau_s}{R_N} & 0 & 1 & 0 \\ 0 & \frac{\tau}{R_E cos(L_{k-1})} & 0 & 1 \end{pmatrix}$$

$$Q_{k-1} \leftarrow \begin{pmatrix} S_{DR}\tau & 0 & \frac{1}{2}\frac{S_{DR}\tau^2}{R_N} & 0 \\ 0 & S_{DR}\tau & 0 & \frac{1}{2}\frac{S_{DR}\tau^2}{R_E cos(L_{k-1})} \\ \frac{1}{2}\frac{S_{DR}\tau^2}{R_N} & 0 & \frac{1}{3}\frac{S_{DR}\tau^3}{R_N^2} & \\ 0 & \frac{1}{2}\frac{S_{DR}\tau^2}{R_E cos(L_{k-1})} & 0 & \frac{1}{3}\frac{S_{DR}\tau^3}{R_E cos(L_{k-1})} \end{pmatrix}$$

$x_k \leftarrow \Phi_{k-1} X_{k-1}$

$P_k \leftarrow \Phi_{k-1} P \Phi_{k-1}^T + Q_{k-1}$

$$H_k \leftarrow \begin{pmatrix} 0 & 0 & -1 & 0 \\ 0 & 0 & 0 & -1 \\ -1 & 0 & 0 & 0 \\ 0 & -1 & 0 & 0 \end{pmatrix}$$

$$R_k \leftarrow \begin{pmatrix} \frac{\sigma_{GR}^2}{R_N^2} & 0 & 0 & 0 \\ 0 & \frac{\sigma_{GR}^2}{R_E^2 cos(L_K)^2} & 0 & 0 \\ 0 & 0 & \sigma_{Gv}^2 & 0 \\ 0 & -1 & 0 & \sigma_{Gv}^2 \end{pmatrix}$$

$K_k \leftarrow P_k H_k^T (H_k P_k H_k^T + R_k)^{-1}$

$$dz \leftarrow \begin{pmatrix} L_k^G - L_k^D \\ \lambda_k^G - \lambda_k^D \\ v_{N,k}^G - v_{N,k}^D \\ v_{E,k}^G - v_{E,k}^D \end{pmatrix} - H_k x_k$$

$X_k \leftarrow x_k + K_k dz$

$P \leftarrow (I - K_k H_k) P_k$

$L_k^D \leftarrow L_k^D - \delta L_k$

$\lambda_k^D \leftarrow \lambda_k^D - \delta\lambda_k$

$v_{N,k}^D \leftarrow v_{N,k}^D - \delta v_{N,k}$

$v_{E,k}^D \leftarrow v_{E,k}^D - \delta v_{E,k}$

end for

with σ_v the initial velocity uncertainty, σ_r the initial position uncertainty, S_{DR} the dead reckoning PSD, σ_{Gr} the GNSS positions measurement error standard deviation, σ_{Gv} the GNSS velocity measurements error standard deviation.

3 Results

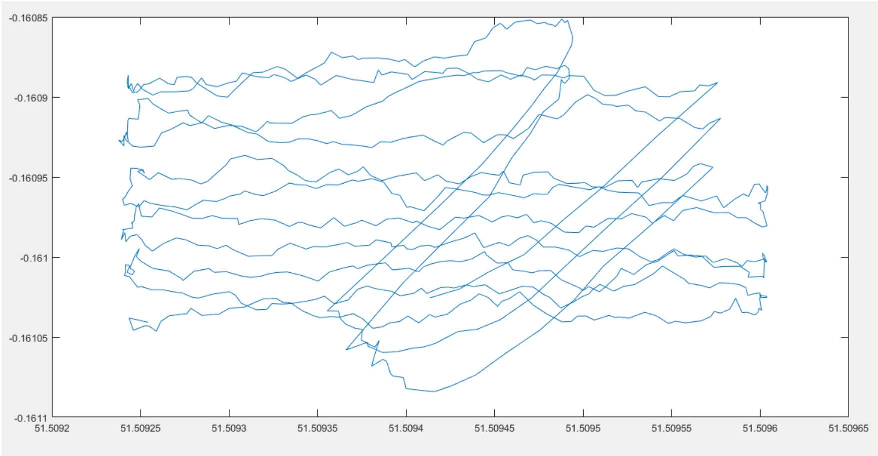

Figure 1: Figure showing horizontal trajectory of the lawnmower

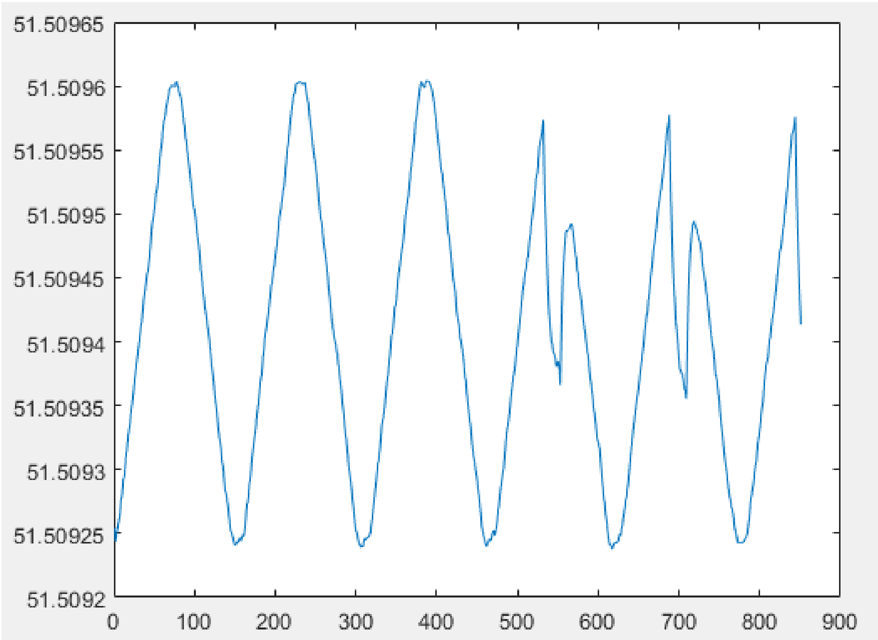

Figure 2: Figure showing the latitude trajectory of the lawnmower

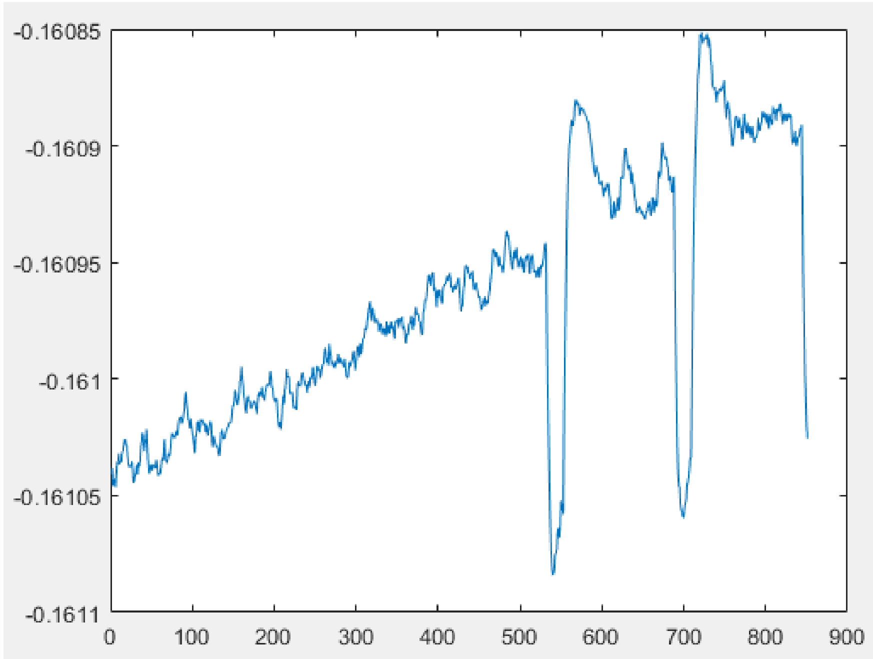

Figure 3: Figure showing the longitude trajectory of the lawnmower

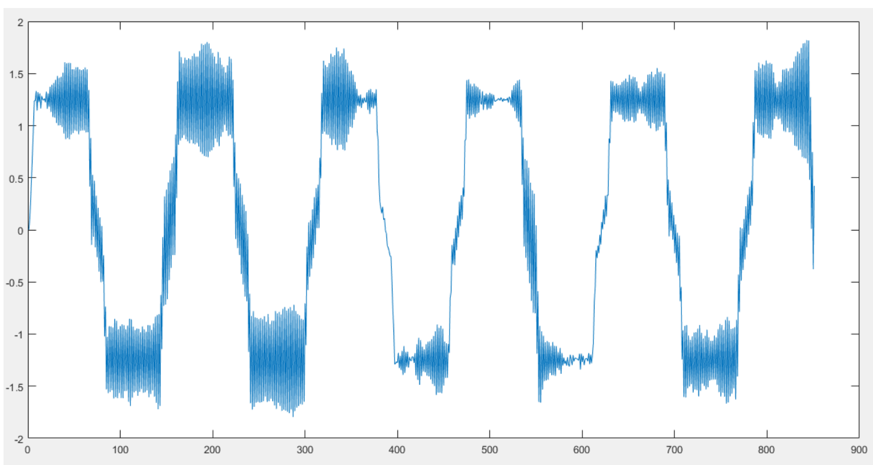

Figure 4: Figure showing the evolution of the north direction velocity of the lawnmower

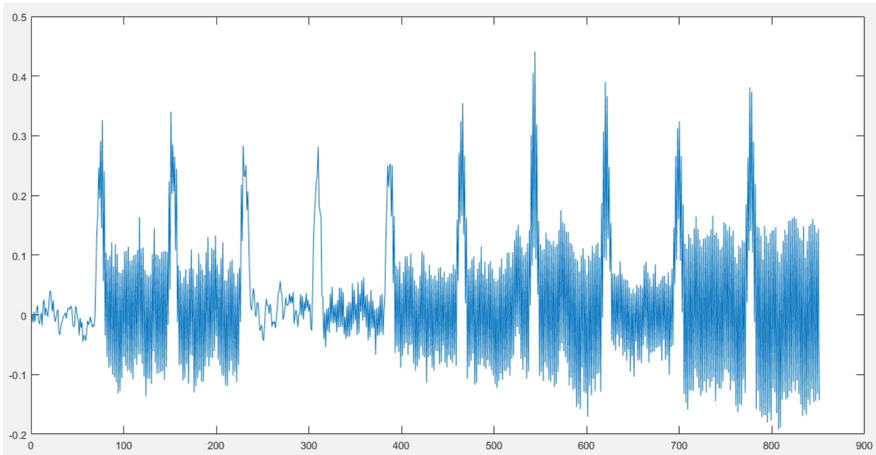

Figure 5: Figure showing the evolution of the east direction velocity of the lawnmower

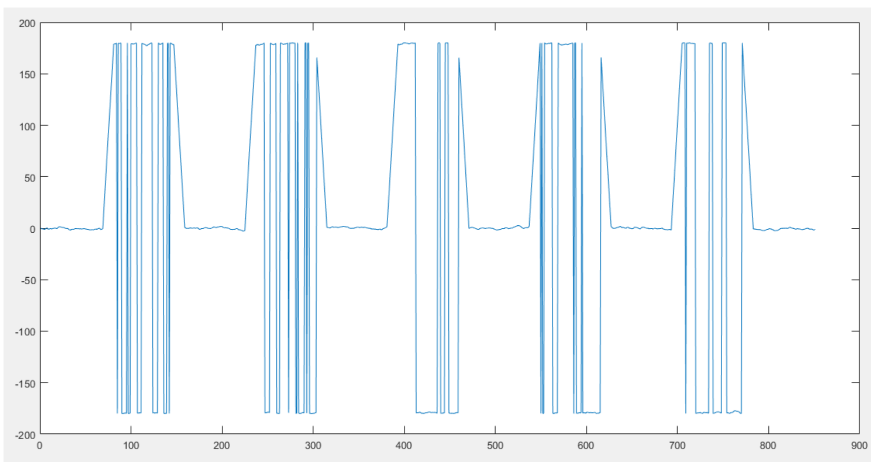

Figure 6: Figure showing the evolution of the heading direction of the lawnmower

4 Code

These files have been provided in the .zip.
Matlab Code :

```
% This routine determine the position of the object at all of the epochs
% using GNSS and Kalman Filter
function GNSSresults = GNSScomputation
Define_Constants

% import data for determining positions and Velocity
pseudo_ranges = csvread('Data/Pseudo_ranges.csv');
pseudo_range_rates = csvread('Data/Pseudo_range_rates.csv');

% n is the number of epoch, m the number of satellites
[n,~] = size(pseudo_ranges);

% Array to store result
GNSSresults = zeros(n-1, 7);
% 1st column is the time in seconds
GNSSresults(:,1) = pseudo_ranges(2:end,1);
% 2nd column contains Geodetic latitude in degrees
% 3rd column contains Geodetic longitude in degrees
% 4th column contains North Velocity in m/s
% 5th column contains East velocity in m/s
% 6th column contains heading in degrees
% 7th column contains height

%% Initialise the Kalman filter state vector estimate and error covariance matrix
[x_est,P_matrix] = Initialise_GNSS(pseudo_ranges, pseudo_range_rates);

% x_est correspond to the initial positions (Which will be the initialized Kalman filter state vector estimate)
% P_matrix is the initial error covariance matrix
% positions and velocities are the positions/Velocities computed without using Kalman Filter

%% Apply Kalman Filter to initial state vector/error covariance matrix

S_a_cphi = 0.01;% clock phase PSD
S_a_cf = 0.04;%clock frequency PSD
sigma_p = 2; % error standart deviation of pseudo range measurements in m
sigma_r = 0.02; % error standart deviation of pseudo range rate measurement in m/s

result = Kalman_Filter(x_est, P_matrix, pseudo_ranges, pseudo_range_rates, S_a_cphi, S_a_cf, sigma_p, sigma_r);

GNSSresults(:,2:3) = result(:,2:3);
GNSSresults(:, 4:5) = result(:,5:6);
GNSSresults(:,7) = result(:,4);
end

%% This subroutine is used to initialised the Kalman Filter
function [x_est,P_matrix] = Initialise_GNSS(pseudo_ranges, pseudo_range_rates)
    % first run Define_Constants.m
    Define_Constants

    % Extract data
    pseudo_ranges_data = pseudo_ranges(2:end, 2:end);
    pseudo_range_rates_data = pseudo_range_rates(2:end,2:end);
    [n,m] = size(pseudo_range_rates_data); % m number au satellite, n number of epoch
```

```
function DR_result = SensorComputation
% this routine determines the trajectory of an object using dead reckoning
% navigation.

% Define constants
Define_Constants

% load data
dead_reckoning = csvread('Data/Dead_reckoning.csv');
[n,~] = size(dead_reckoning); % n is the number of epoch

% result
DR_result = zeros(n,6);
DR_result(:,1) = dead_reckoning(:,1); % store time in s

% let's use GNSScomputation to initialize the position
GNSSmeasurement = GNSScomputation;

% the rear wheels are the driving wheels, so we can assume that the average
% of the rear wheel speeds correspond to the forward speed of the lawnmower
rear_wheel_L = dead_reckoning(:,4); % m/s
rear_wheel_R = dead_reckoning(:,5);
forward_speed = (rear_wheel_L + rear_wheel_R)/2; % m/s

% take height as calculated by GNSS
h = GNSSmeasurement(1,7);

time = dead_reckoning(:,1);

% first, we will compute the heading by correcting the gyroscope
% measurement with the compass measurement

%% Compute heading by combining gyrosope-derived heading and magnetic heading
% In this section, We will use a Gyro-Magnetometer Kalman Filter States to compute heading
dead_reckoning = csvread('Data/Dead_reckoning.csv');
gyroscope_heading_rate = dead_reckoning(:,6);
compassHeading = dead_reckoning(:,7)*deg_to_rad;

% Compute heading from gyroscope
gyroscopeHeading = zeros(n,1);
gyroscopeHeading(1) = gyroscope_heading_rate(1)*0.5;
for i=2:n
    gyroscopeHeading(i) = gyroscopeHeading(i-1)+0.5*gyroscope_heading_rate(i);
end

% store corrected heading
heading = zeros(n,1);
heading(1) = gyroscopeHeading(1); % initialize

% Initialize 2 states Kalman filter vector
h_est = zeros(2,1); %

% Initiaize state estimation error covariance matrix
sigma_gyroBias = 1*deg_to_rad; % bias standar deviation of 1 degree per second
Ph_matrix = [10^-4 0;... % noise standar deviation of 10-4 rad/s
             0 sigma_gyroBias];

for i=2:n
```

```
% This routine compute an integrated horizontal DR/GNQQ navigation solution
% using Kalman filtering

% Define constants
Define_Constants

% Compute GNSS solution
GNSSsolutions = GNSScomputation;
[n,~] = size(GNSSsolutions); % n is the number of epoch

% Compute DR solution
DRsolution = SensorComputation;

% separate the data
time = GNSSsolutions(:,1);

GNSSlatitude = GNSSsolutions(:,2)*deg_to_rad;
GNSSlongitude = GNSSsolutions(:,3)*deg_to_rad;
GNSSheight = GNSSsolutions(:,7);
GNSSnorth_velocity = GNSSsolutions(:,4);
GNSSeast_velocity = GNSSsolutions(:,5);

DRlatitude = DRsolution(:,2)*deg_to_rad;
DRlongitude = DRsolution(:,3)*deg_to_rad;
DRnorth_velocity = DRsolution(:,4);
DReast_velocity = DRsolution(:,5);

% initialize latitude with DR measurement
latitude = DRlatitude;

% Since it's a lawnmower, we assume that height is 0
height = zeros(n,1);

% Store result
correctedResults = zeros(n, 6);
correctedResults(:,1) = time;
correctedResults(1,:) = DRsolution(1,:); % initalization of latitude,
    longitude, velocity, heading and height

% Initialize 4-state Kalman filter vector
x_est = zeros(4,1);

% Initialise state estimation error covariance matrix
P_matrix = zeros(4);
sigma_v = 0.02;
sigma_r = 10;
[R_N,R_E]= Radii_of_curvature(latitude(1));
P_matrix(1,1) = sigma_v^2;
P_matrix(2,2) = sigma_v^2;
P_matrix(3,3) = sigma_r^2/(R_N + height(1))^2;
P_matrix(4,4) = sigma_r^2/(R_E + height(1))^2*cos(latitude(1))^2;

for i=2:n
    %% Step 1 : Compute transition matrix
    tau_s = 0.5; % propagation time in s
    [R_N,R_E]= Radii_of_curvature(latitude(i-1));
    Phi = eye(4);
    Phi(3,1) = tau_s/(R_N + height(i-1));
    Phi(4,2) = tau_s/((R_E + height(i-1))*cos(latitude(i-1)));
```

5 References

[1]P.D. Groves, Principles of GNSS, Inertial, ans Multisensor Integrated Navigation Systems, 2nd Edition, Artech House, 2013, COMPGX04:ROBOT VISION AND NAVIGATION, Workshop1

[2]P.D. Groves, COMPGX04:ROBOT VISION AND NAVIGATION, Workshop1, Equation (1), (2) and (3)

[3]P.D. Groves, COMPGX04:ROBOT VISION AND NAVIGATION, Workshop1, Equation (4)

[4]P.D. Groves, COMPGX04:ROBOT VISION AND NAVIGATION, Workshop1, Equation (9),

[4]P.D. Groves, COMPGX04:ROBOT VISION AND NAVIGATION, Workshop2, task 2A,

[5]P.D. Groves, COMPGX04:ROBOT VISION AND NAVIGATION, Lecture 6A, Gyro-Magnetometer Integration

[6]P.D. Groves, COMPGX04:ROBOT VISION AND NAVIGATION, Workshop3, task 1,

[7]P.D. Groves, COMPGX04:ROBOT VISION AND NAVIGATION, Workshop3, task 2,