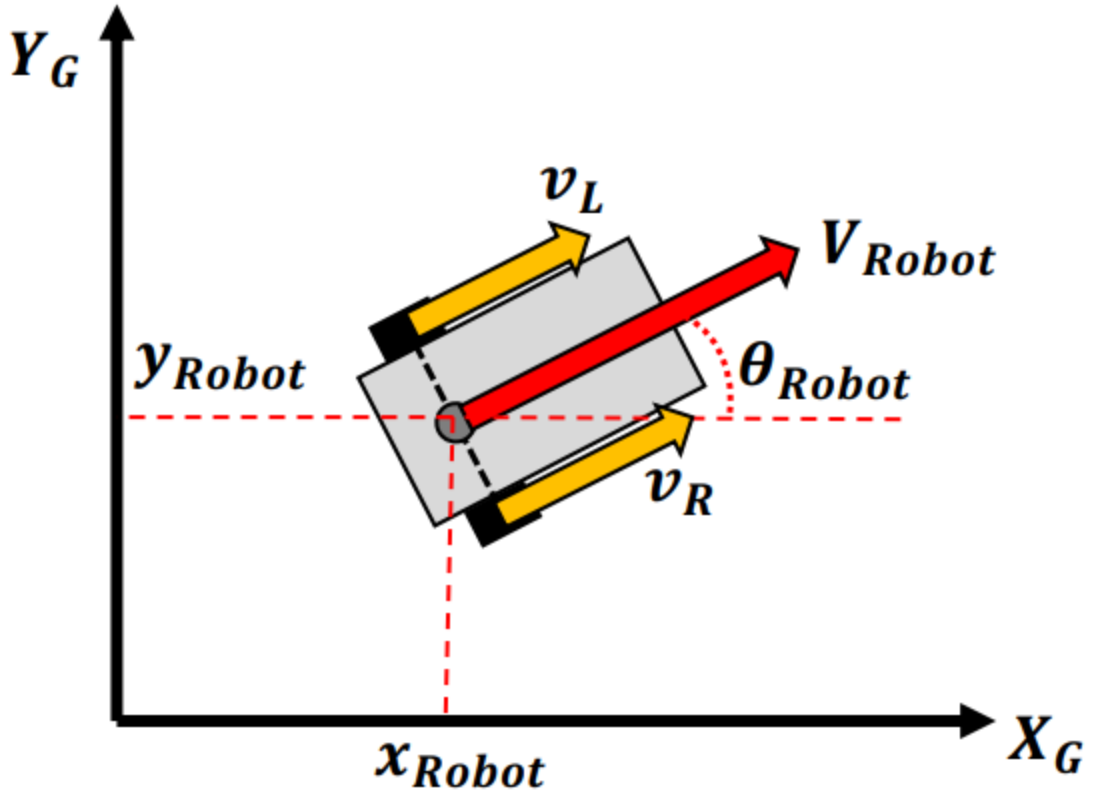


Autonomous 2-Wheel Differential Robot Modeling, Control, and Path Planning in Simulink

—

Muhammad Abdelsallam 201801356
University of Science and Technology - Zewail City
SPC 418 Control Systems Design for Autonomous Vehicles
Dr. Mohamed Elshalakani

Table of Content

Introduction

The autonomous 2-wheel differential robot has emerged as a versatile and flexible platform with numerous applications in diverse fields. Its ability to navigate through complex environments and perform intricate tasks makes it a valuable asset in industrial automation, surveillance, and search and rescue operations. In this project, we present a comprehensive study focused on the modeling, control, and path planning of such a robot using Simulink.

In the first phase of the project, a precise kinematics state-space model of the 2-wheel differential robot is developed. Then and to achieve precise and robust control, two distinct control strategies were designed and implemented: a state feedback controller and a Lyapunov controller. The state feedback controller is based on a linearized version of the developed model that utilizes feedback information from the error of the robot's actual states and the desired states (position and orientation) to regulate robot's motion and ensure its delivery and stability. On the other hand, the Lyapunov controller ensures asymptotic stability by employing developed Lyapunov functions for the nonlinear state space model. Then simulations with various inputs were conducted to analyze behavior and validate the accuracy and reliability of the robot controlled model.

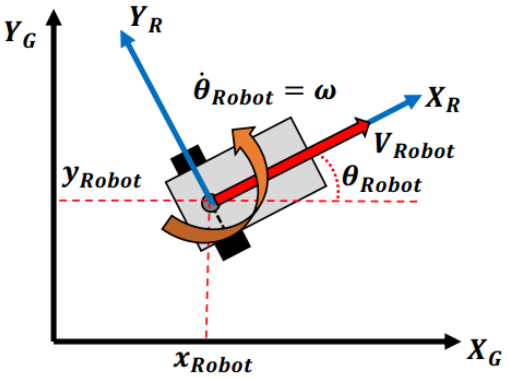
Furthermore, we integrate an Artificial Potential Field (APF) path planning and obstacle avoidance algorithm into the control system. The APF algorithm enables the robot to detect and avoid obstacles in its environment while planning a collision-free path toward a desired goal. By incorporating this algorithm, the robot demonstrates enhanced autonomy and safety during navigation. To improve the reliability of the system and reject noisy sensor data, an Extended Kalman Filter (EKF) is utilized. The EKF performs sensor fusion and estimation, enabling accurate localization and robust control. By effectively filtering the sensor measurements, the EKF enhances the overall performance and reliability of the autonomous 2-wheel differential robot.

In summary, this project encompasses the modeling and simulation of a 2-wheel differential robot, the design and implementation of state feedback and Lyapunov controllers, the integration of an APF path planning and obstacle avoidance algorithm, and the utilization of an extended Kalman filter for reliable localization and noise immunity. These contributions aim to enhance the precision, autonomy, and robustness of the autonomous 2-wheel differential robot, facilitating its applications in various real-world scenarios.

Modeling

Starting by constructing a kinematics model for a 2-wheel differential robot in the robot frame

Considering the system inputs as the heading velocity V_R, and angular velocity ω_R, and the outputs (states) as the position X_R, & Y_R and heading angle θ_R in the robot frame, the system becomes:



$$\dot{x} = \begin{bmatrix} \dot{X}_R \\ \dot{Y}_R \\ \dot{\theta}_R \end{bmatrix} = \begin{bmatrix} V_R \cos(\theta_R) \\ V_R \sin(\theta_R) \\ \omega_R \end{bmatrix}$$

With the possibility to translate it to the initial frame using the transformation matrix T that could be defined as

$$T = \begin{bmatrix} \cos(\theta_R) & -\sin(\theta_R) & 0 \\ \sin(\theta_R) & \cos(\theta_R) & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

Where

$$x_{inertial} = T.x_{robot}$$

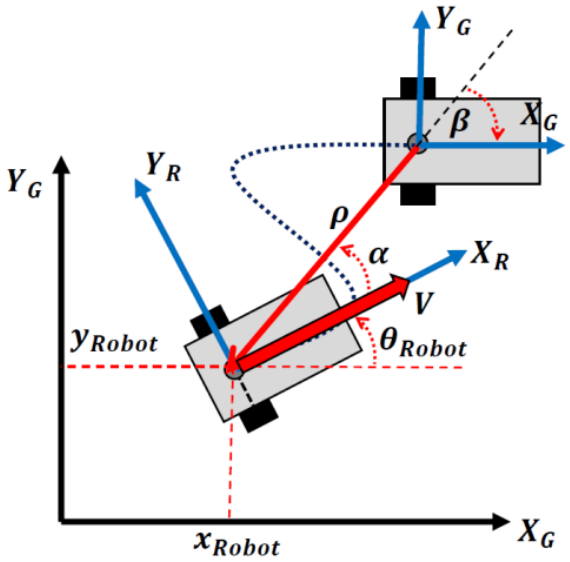
Noting that the stated transformation matrix assumes that the origin of the inertial frame coincides with the initial position of the robot. If the initial position is different, additional translation terms need to be included in the transformation matrix.

Controllers

Designing and implementing a state feedback controller and a Lyapunov controller

State Feedback Controller

Starting by defining the error signals ρ, α, & β of the robot's actual position and orientation from the desired position and orientation on the robot frame.



Where

$$\Delta X = X_G - X_R, \quad \Delta Y = Y_G - Y_R$$

$$\rho = \sqrt{(\Delta X)^2 + (\Delta Y)^2}$$

$$\alpha = atan2(\Delta Y/\Delta X) - \theta$$

$$\beta = -\alpha - \theta$$

After making various manipulations, it results

$$\dot{\rho} = -V\cos(\alpha)$$

$$\dot{\beta} = \frac{-V\cos(\alpha)}{\rho}$$

$$\dot{\alpha} = \frac{V\cos(\alpha)}{\rho} - \omega$$

Rearranging in a matrix form

$$\begin{bmatrix} \dot{\rho} \\ \dot{\alpha} \\ \dot{\beta} \end{bmatrix} = \begin{bmatrix} -\cos(\alpha) & 0 & 0 \\ \frac{\sin(\alpha)}{\rho} & 0 & -1 \\ -\frac{\sin(\alpha)}{\rho} & 0 & 0 \end{bmatrix} \begin{bmatrix} \rho \\ \alpha \\ \beta \end{bmatrix}$$

Note that this presented system can be considered in a transformation in the polar coordinates.

Taking the state feedback as a P-controller, the system will need to be linearized first.

So taking $sin(a) \simeq a,\ \&\ cos(a) \simeq 1$, results to

$$\begin{bmatrix} \dot{\rho} \\ \dot{\alpha} \\ \dot{\beta} \end{bmatrix} = \begin{bmatrix} -1 & 0 & 0 \\ \frac{\alpha}{\rho} & 0 & -1 \\ -\frac{\alpha}{\rho} & 0 & 0 \end{bmatrix} \begin{bmatrix} \rho \\ \alpha \\ \beta \end{bmatrix}$$

$$V = \rho$$
$$\omega = \dot{\alpha} + \beta$$

So taking the P-controllers using the gains $k_\rho, k_\alpha, \& k_\beta$ for $V,\ \&\ \omega$.

$$V = k_\rho \rho$$

$$\omega = k_\alpha \alpha + k_\beta \beta$$

Results

$$\begin{bmatrix} \dot{\rho} \\ \dot{\alpha} \\ \dot{\beta} \end{bmatrix} = \begin{bmatrix} -k_\rho & 0 & 0 \\ 0 & k_\rho - k_\alpha & -k_\beta \\ 0 & -k_\rho & 0 \end{bmatrix} \begin{bmatrix} \rho \\ \alpha \\ \beta \end{bmatrix}$$

To ensure stability matrix A has to have negative eigenvalues, so we evaluate the eigenvalues in trams of the gains and then put constraints on the gains to achieve negative eigenvalues. So we start by evaluating the characteristic equation for matrix A

$$|A - I\lambda| = 0$$

$$\begin{vmatrix} -k_\rho - \lambda & 0 & 0 \\ 0 & k_\rho - k_\alpha - \lambda & -k_\beta \\ 0 & -k_\rho & -\lambda \end{vmatrix} = 0$$

After some manipulation we reach that

$$\lambda_1 = -k_\rho$$

$$\lambda_2 = \frac{-(k_\rho - k_\alpha) + \sqrt{(k_\rho - k_\alpha)^2 + 4k_\beta k_\rho}}{2}$$

$$\lambda_3 = \frac{-(k_\rho - k_\alpha) - \sqrt{(k_\rho - k_\alpha)^2 + 4k_\beta k_\rho}}{2}$$

Then the constraints are

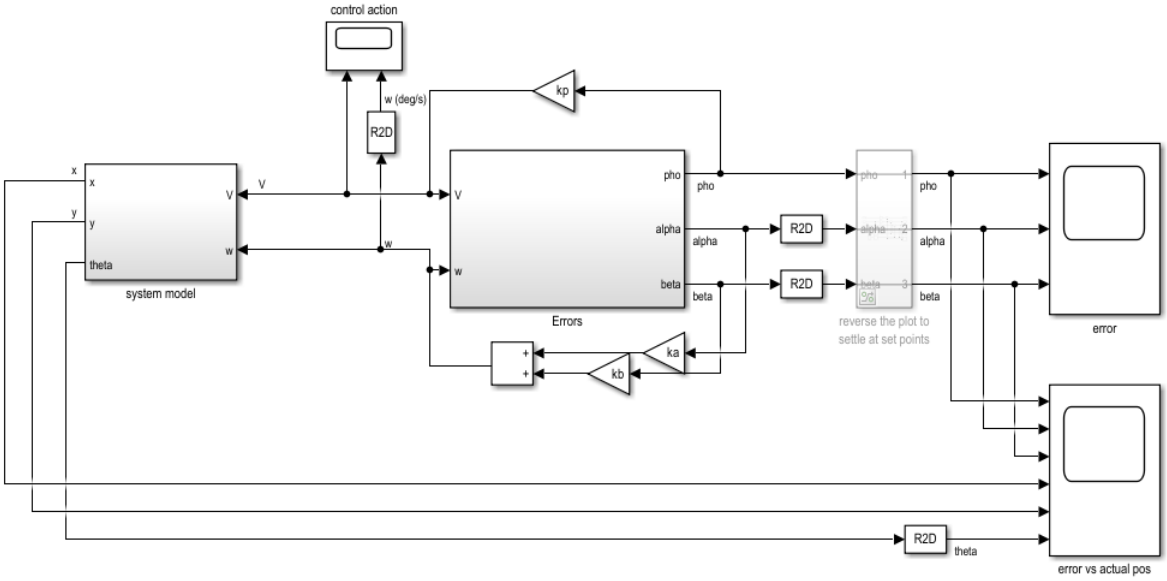
1. $k_\rho > 0$
2. $k_\alpha < k_\rho$
3. The discriminant $\sqrt{(k_\rho - k_\alpha)^2 + 4k_\beta k_\rho} > 0$

So choosing gains following these constraints ensures having all negative eigenvalues that guarantee to have a stable state feedback controller.

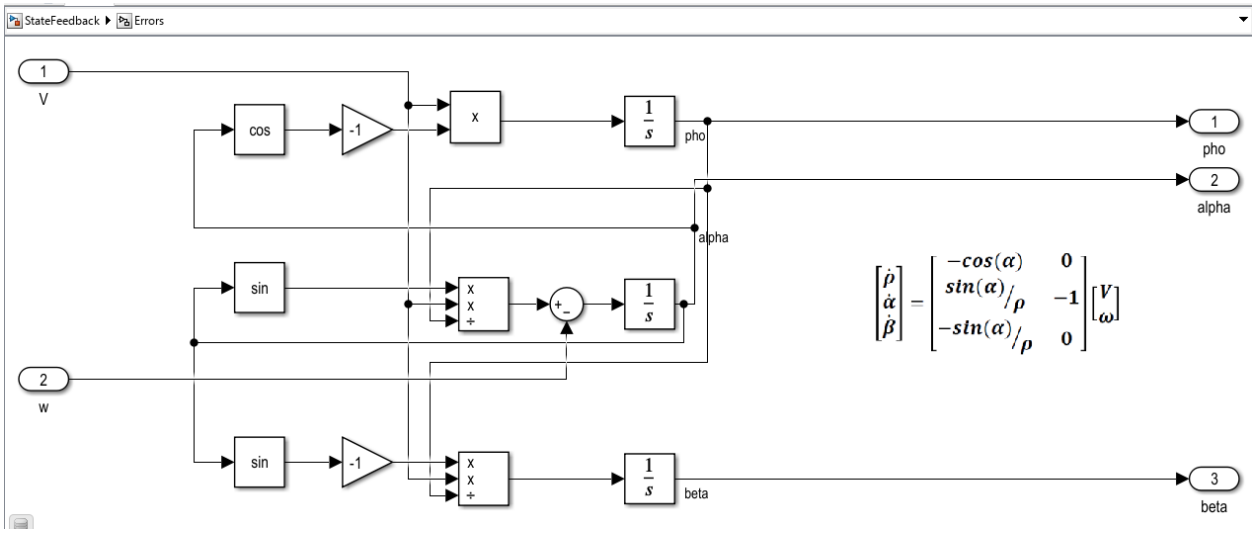
For instance, we choose $k_\rho = 5,\ k_\alpha = 10,\ k_\beta = -1.2$, that gives eigenvalues of $\lambda = -2, -3, \&\ -5.$

Simulink

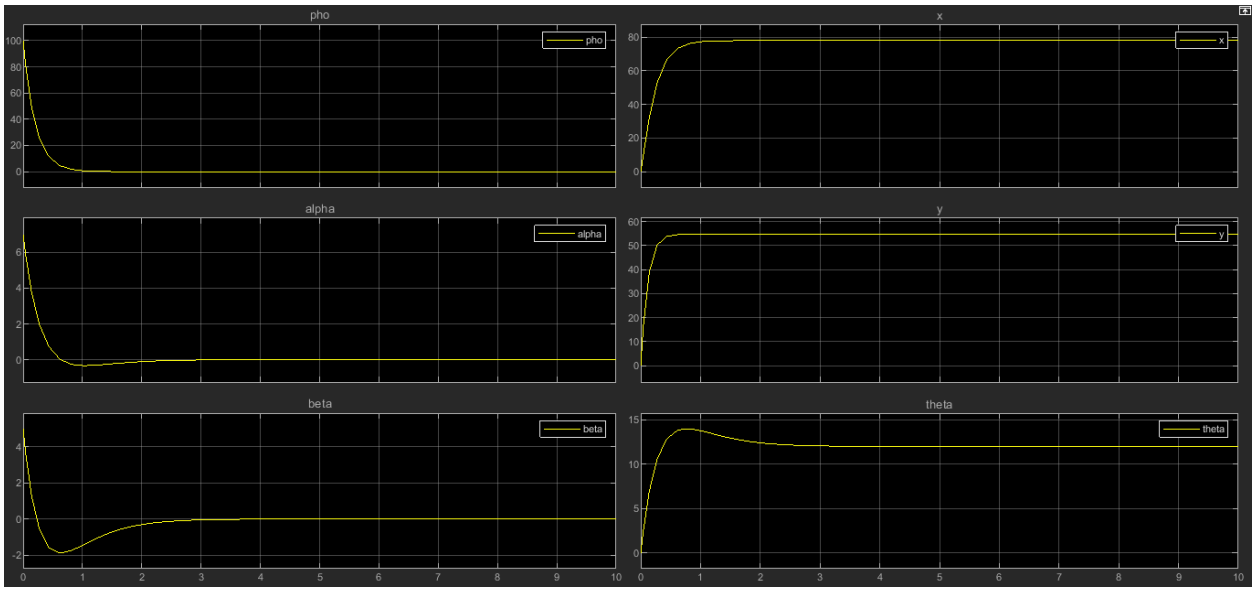
The system is now implemented on Simulink as



Where the errors block is

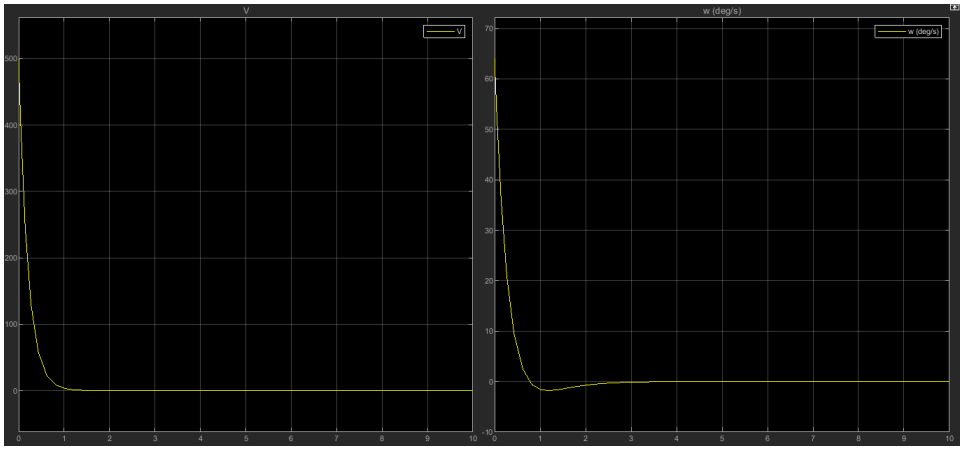


The system is now simulated for initial positions of zero ($X_i = 0, Y_i = 0, \theta_i = 0$), and a goal positions at $X_G = 80\,m,\ Y_G = 60\,m, \theta_G = 12\,deg$



The positions reached are $X = 77.8,\ Y = 54.6,\ \theta = 12$

With control actions of



Gaving errors in states of $e_x = 2.8\%,\ e_y = 9\%,\ e_\theta = 0$

Lyapunov Based Controller

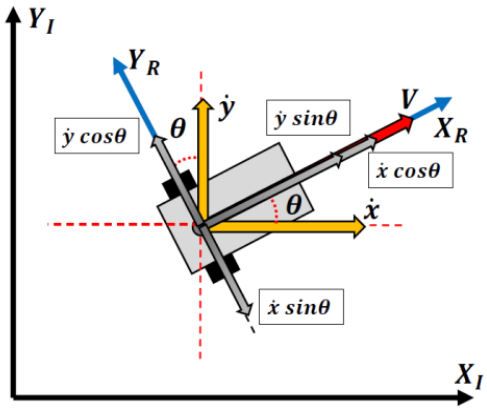
Due to the limitations on the state feedback controller, we needed to build another, more accurate and reliable controller.

To choose an appropriate Lyapunov function for the controller it has to satisfy

- $V(x),\ \&\ \dot{V}(x)$ *have to be continuous*
- $V(x)$ *has to be positive definite*
- $\dot{V}(x)$ *has to be negative definite*

- $V(0) = \dot{V}(0) = 0$

So we start by gathering the system physical constraints, the system non-holonomic constraints. The system non-holonomic constraints can be extracted from the following figure as



The no-sliding constraint

$$\dot{y}\cos(\theta) - \dot{x}\sin(\theta) = 0$$

The rolling constraint

$$\dot{x}\cos(\theta) + \dot{y}\sin(\theta) = V$$

We then define the system in the robot frame as

$$\begin{bmatrix} x_{e,R} \\ y_{e,R} \\ \theta_{e,R} \end{bmatrix} = \begin{bmatrix} \cos\theta & \sin\theta & 0 \\ -\sin\theta & \cos\theta & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x_e \\ y_e \\ \theta_e \end{bmatrix}$$

Where $x_e = x_d - x, y_e = y_d - y, \theta_e = \theta_d - \theta$

After some manipulations of differentiation the error signals and using the constraints evaluated to simplify the expressions as much as possible, we reach the following

$$\dot{x}_{e,R} = V_d \cos\theta_{e,R} - \boldsymbol{V} + y_{e,R}\,\boldsymbol{\omega}$$
$$\dot{y}_{e,R} = V_d \sin\theta_{e,R} - x_{e,R}\,\boldsymbol{\omega}$$
$$\dot{\theta}_{e,R} = \omega_d - \boldsymbol{\omega}$$

Choosing a suitable Lyapunov function according to the conditions states above

$$V(x) = \frac{1}{2}\left(x_{e,R}^2 + y_{e,R}^2\right) + (1 - \cos\theta_{e,R})$$
$$\dot{V}(x) = x_{e,R}\,\dot{x}_{e,R} + y_{e,R}\,\dot{y}_{e,R} + \sin\theta_{e,R}\dot{\theta}_{e,R}$$

Substituting back in the system to check and simplify, we reach

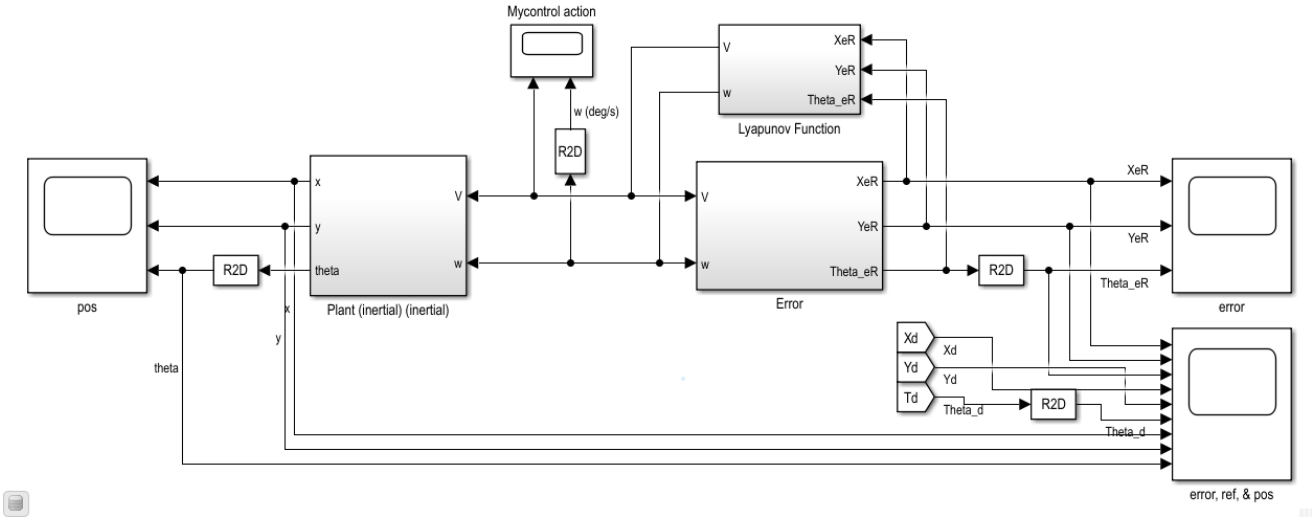
$$\boldsymbol{V} = V_d \cos\theta_{e,R} + \boldsymbol{k_x} x_{e,R}$$
$$\boldsymbol{\omega} = V_d\, y_{e,R} + \omega_d + \boldsymbol{k_\theta}\, \sin\theta_{e,R}$$

Where $k_{x'}$ & k_θ are the control law gains and are always positive constants

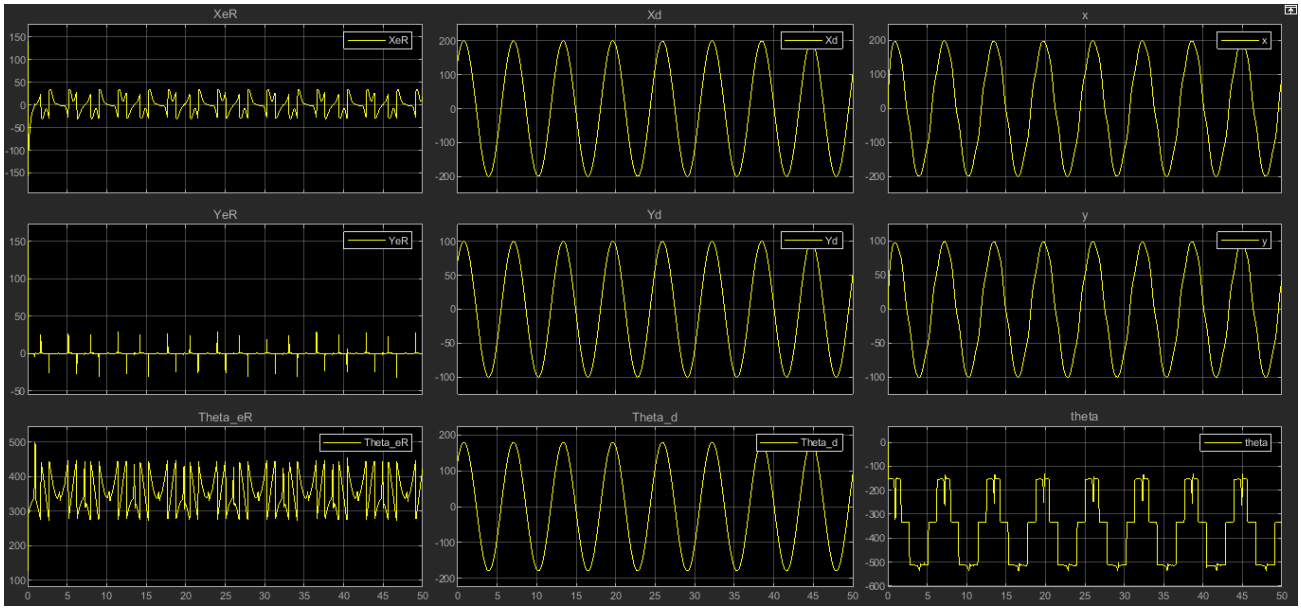
Using V, & ω values that follow these relations ensures having an asymptotically stable stable that's converging to the desired values.

Simulink

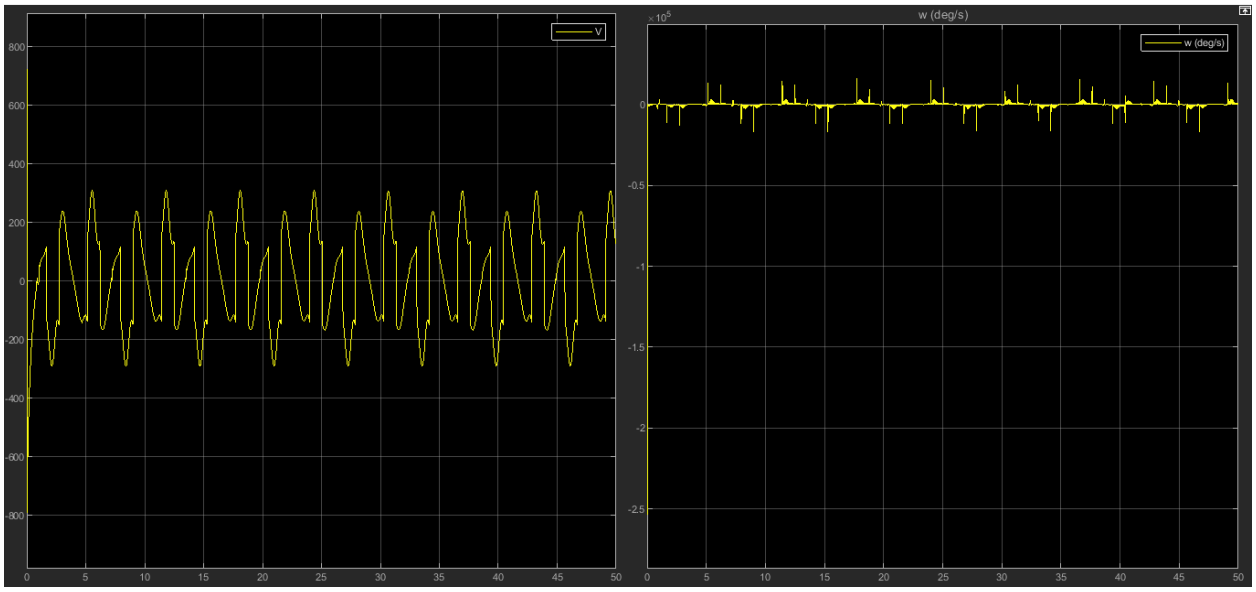
The system is now implemented on Simulink as



Simulating the system to follow a sinusoidal trajectory in the three states with a $pi/4$ phase shift and amplitudes of $A_x = 200,\ A_y = 100,\ A_\theta = 180\ degree$, with initial positions of zero $(X_i = 0, Y_i = 0, \theta_i = 0)$ and Lyapunov parameters of $k_x = 5;\ k_\theta = 10$



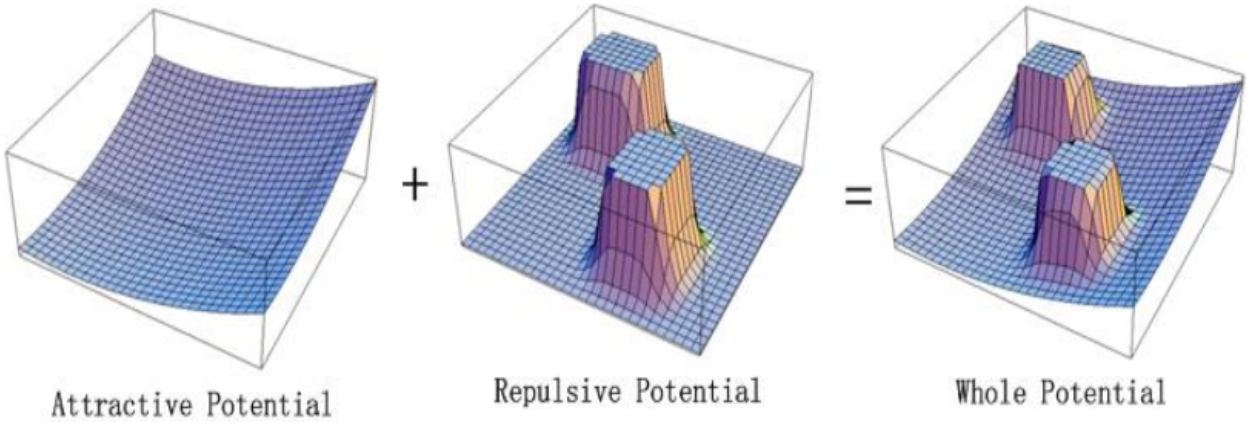
With control action of



The system reaches the given trajectory with a very slight error that is acceptable without the limitation of the small angle variations that the state feedback controller has.

Path Planning

In order to fully automate the system, it needs to be able to process and determine the path necessary to follow to reach the goal position while avoiding obstacles. The Artificial-Potential Field (AFP) algorithm is chosen to be used for path planning and obstacle avoidance. AFP approach is based upon the concepts of electrical potentials of attraction (goal point) and repulsion (obstacles). The total potential is expected to be as



The attractive potential is evaluated through

$$U_{att} = k_{att}(q - q_{goal})^2$$

where k_{att} is the gain for attractive potential, q is the robot position $x, \& \, y$ cooridenates, and q_{goal} is the goal position $x, \& \, y$ cooridenates

The repulsive potential

$$U_{rep} = k_{rep}(1/d(q) - 1/d_{lim})^2 \; for \; d(q) \leq d_{lim}, \; 0 \; otherwise$$

Where k_{rep} is the gain for repulsive potential, $d(q)$ is the distance between the robot and the nearest obstacle and d_{lim} is the limit at which the repulsive potential is triggered.

Then the total potential is evaluated as the sum of attractive and repulsive potentials

$$U_{tot} = U_{att} + U_{rep}$$

The force acting on the robot is the gradient of this potential is

$$F = - \nabla U_{tot}$$

$$F_{att} = k_{att}(q - q_{goal})$$

$$F_{rep} = k_{rep}(1/d(q) - 1/d_{lim})(1/d^2(q))(\frac{q - q_{goal}}{||q - q_{goal}||}) \quad for \; d(q) \leq d_{lim}, \; 0 \; otherwise$$

So the attractive and repulsive forces will both have two components, x-component, and a y-component so that we can use them to get the new x and y positions.

A Matlab function is built on that base but some enhancements are made for better path evaluation

- The attractive gain is made to be gradually increasing as the robot approaches the goal

This enhancement is made to overcome the problem of low gradient (and hence low force) acting on the robot near the goal. So the gain is made to be exponentially increasing as the robot approaches the goal following the formula

$$k_{att} = k_{att-max} + (k_{att-min} - k_{att-max})(1 - exp(- d/k_{att-d}))$$

Where $k_{att-max}$ is the maximum attraction gain, $k_{att-min}$ is the minimum attraction gain, k_{att-d} is the distance at which attractive force is maximum

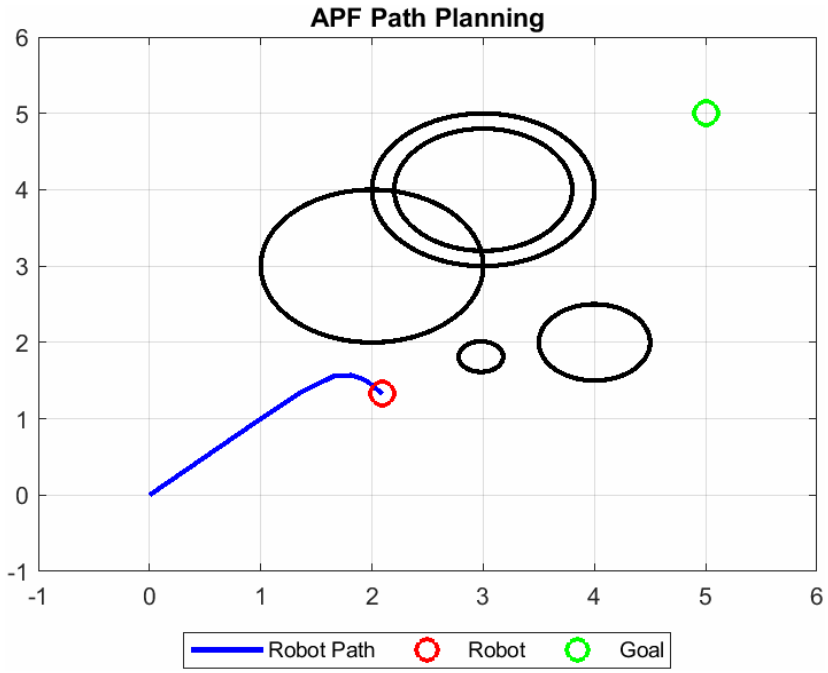
- Check if stuck in local minima

Local minima is the most critical problems of APF. it happens where attractive potential cancels out the repulsive potential (zero potential points). To overcome it, virtual obstacles are added at the points of zero total potential.

Simulation

Simulating the algorithm for the following parameters

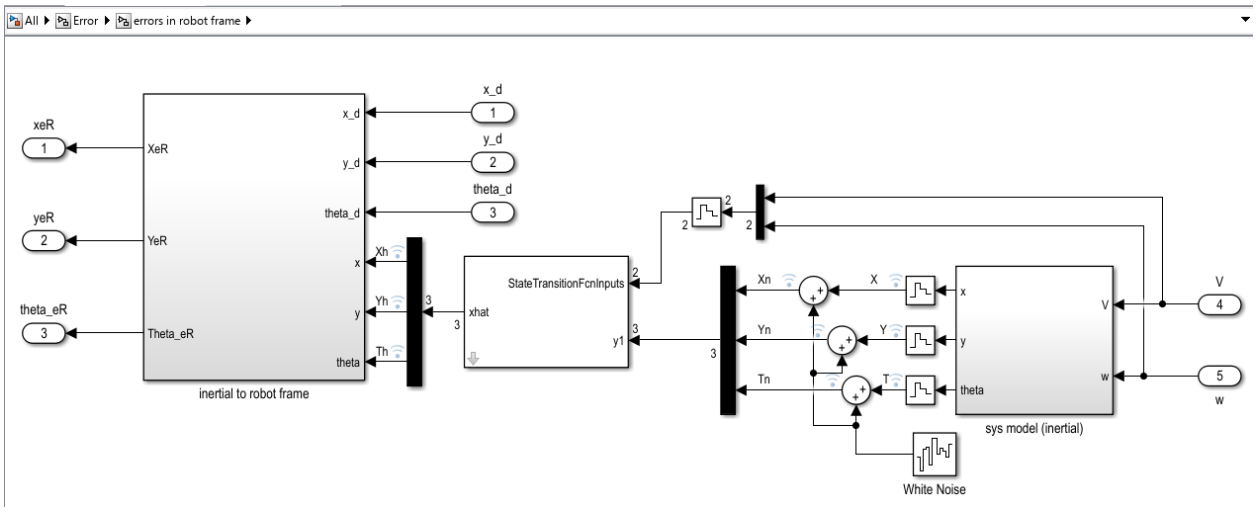
- Start position of $X_i = 0,\ Y_i = 0, \theta_i = 0$
- Goal position of $X_G = 5\ m,\ Y_G = 5\ m, \theta_G = 0$
- Obstacles at $[2,\ 3,\ 1;\ 3,\ 4,\ 1;\ 2,\ 3,\ 1;\ 3,\ 4,\ 0.8;\ 4,\ 2,\ 0.5]';\ \%\ [x;\ y;\ radius]$
- Time step 0.1s



The robot reaches the goal after making two virtual obstacles as this specific layout of obstacles, start, and goal are chosen to have two zero potential points

Extended Kalman Filter

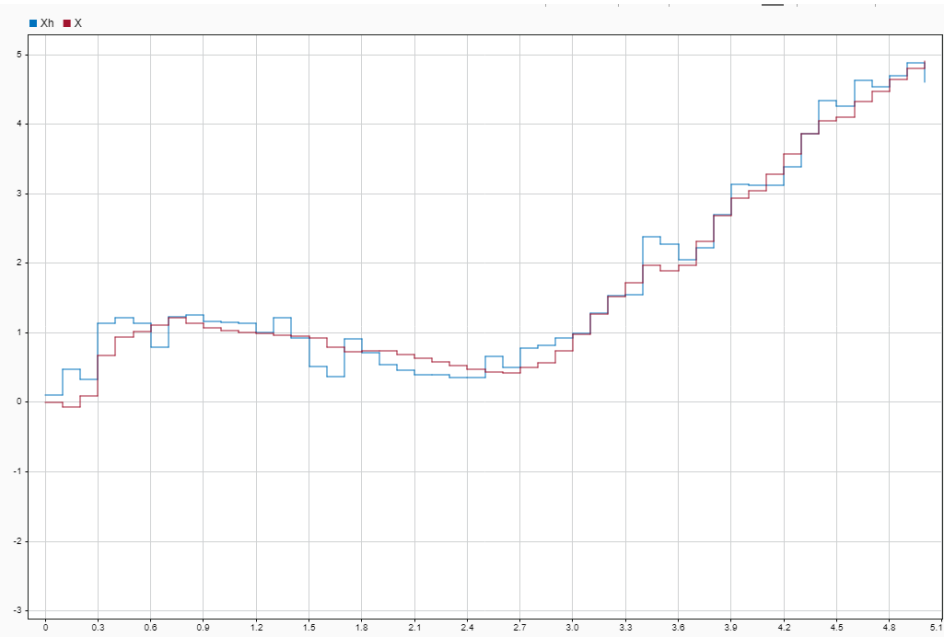
Due to the noise that could affect the system and the model and sensor readings uncertainty and as the heart of any autonomous system is the precise ability to localize the vehicle to make the appropriate decisions, an Extended Kalman Filter is added to the system so that the system gains some noise immunity, and accurately localize the vehicle.



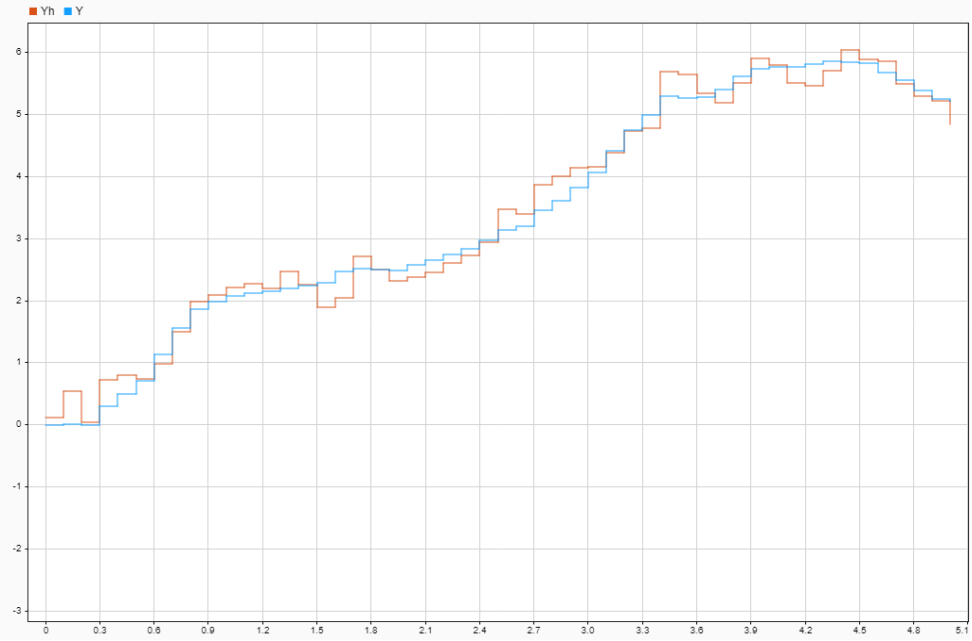
Simulation

Running the simulation for system states covariance of 0.01, and adding white noise to the inputs with a noise power of 0.01

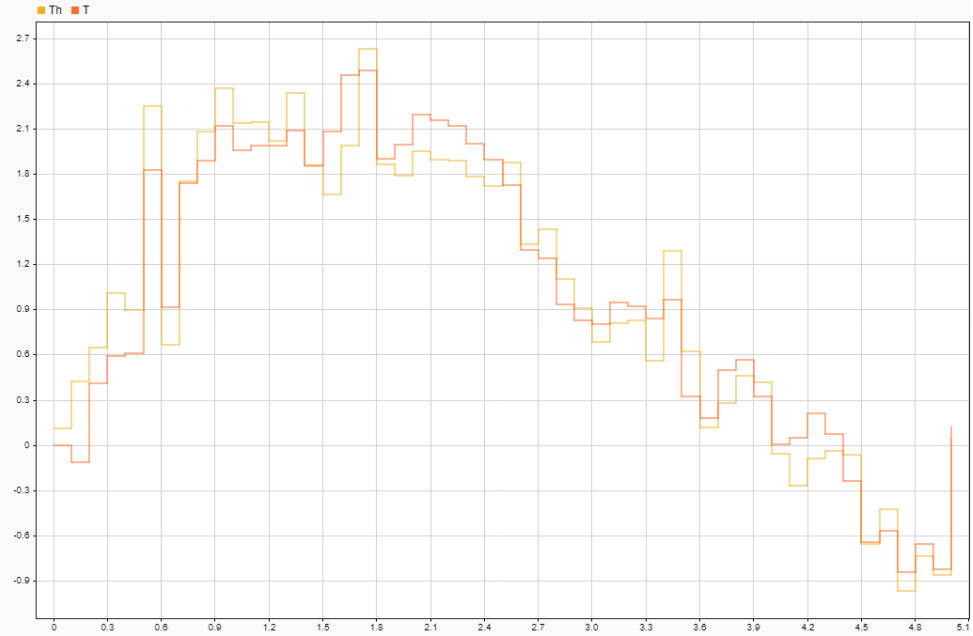
The filtered x position and the noisy x position



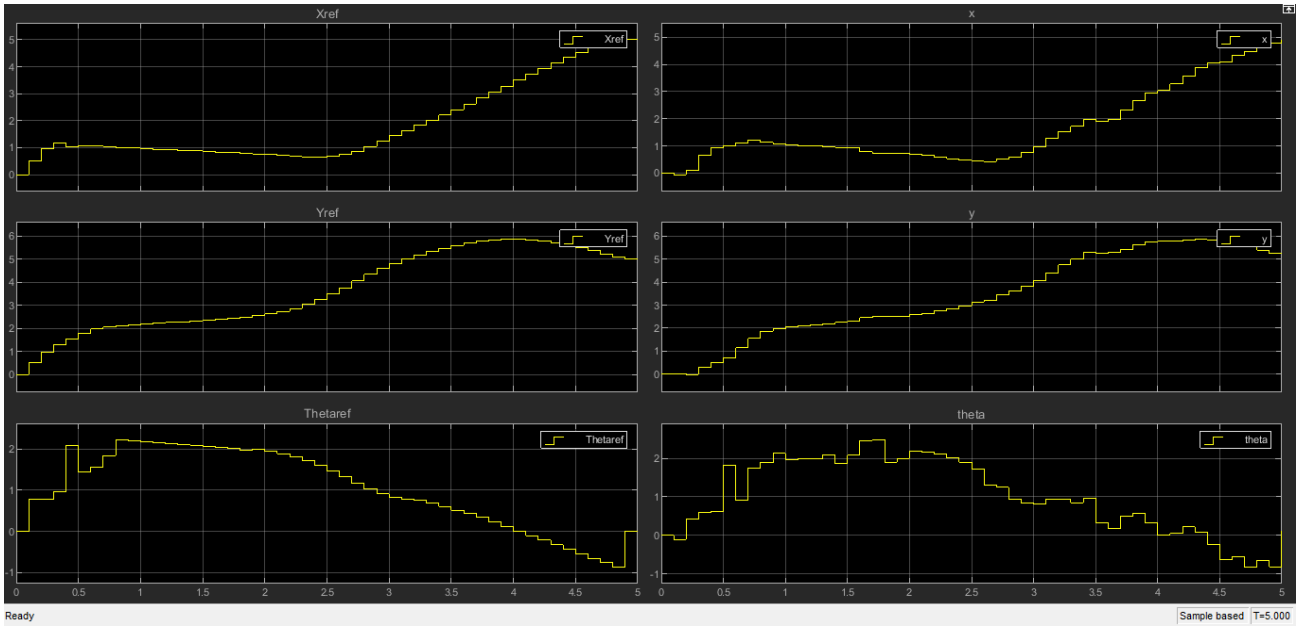
The filtered y position and the noisy y position



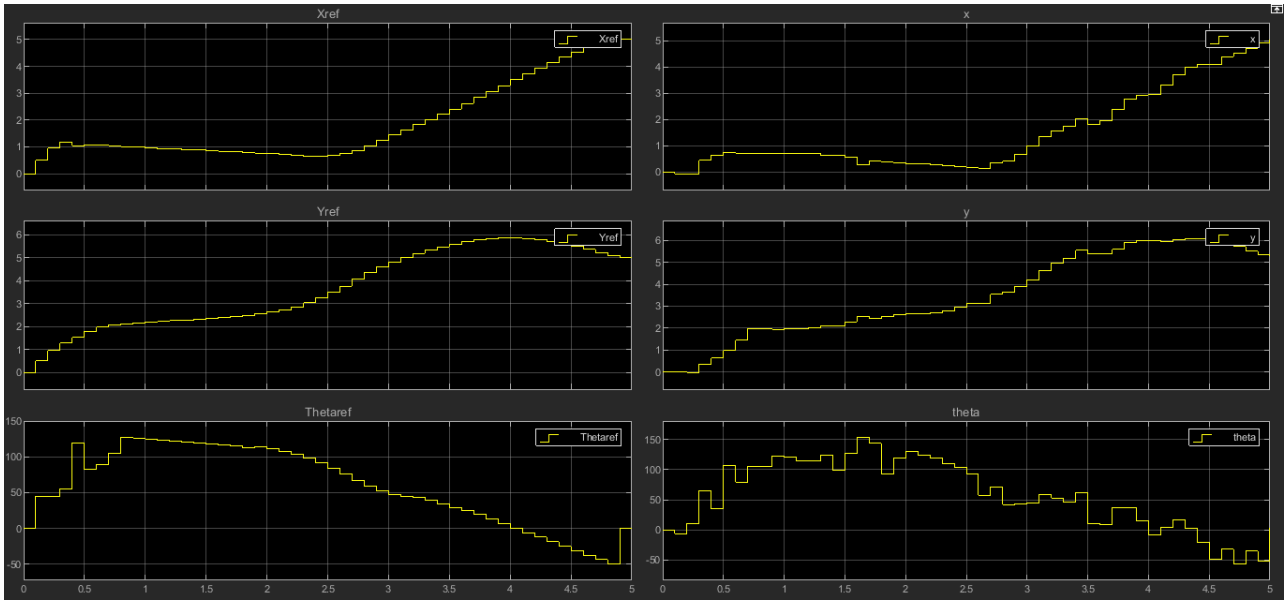
The filtered theta and the noisy theta



The system response with EKF



The system response without EKF



The EKF enhances the rejected some of the noise and made the system output to be more reliable.

References

1. Sabudin, E.N., Omar, R.B., & Melor, C.K. (2016). POTENTIAL FIELD METHODS AND THEIR INHERENT APPROACHES FOR PATH PLANNING.