MOBILE ROBOT - LAB 1
April 09, 2016
Truong Giang Vo & Subodh Mishra

Contents

Part I: Mobile Robot Type (2,0)

I.1 Complete Kinematic Study

I.1.1 Parameterization

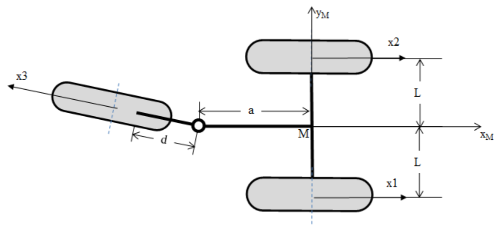

Figure I.1.1: Schematic of mobile robot type (2,0)

Table I.1: Parameters of type (2,0) mobile robot

W	L	α	d	β	γ	φ	ψ
1_f	L	$-\pi/2$	0	0	$\pi/2$	φ_1	0
2_f	L	$\pi/2$	0	0	$-\pi/2$	φ_2	0
3_c	a	π	d	β_3	0	φ_3	$\beta_3+\pi$

The configuration vector is: $q = \begin{bmatrix} x & y & \theta & \beta_3 & \varphi_1 & \varphi_2 & \varphi_3 \end{bmatrix}^T$

I.1.2 Posture Kinematic Model

We have:

$$\mathbf{J_1} = \begin{bmatrix} 1 & 0 & L \\ 1 & 0 & -L \\ -\cos\beta_3 & -\sin\beta_3 & a\sin\beta_3 \end{bmatrix} \qquad \mathbf{J_2} = -r\mathbf{I}_3$$

$$\mathbf{C_1} = \begin{bmatrix} 0 & 1 & 0 \\ 0 & 1 & 0 \\ \sin\beta_3 & -\cos\beta_3 & d + a\cos\beta_3 \end{bmatrix} \qquad \mathbf{C_2} = \begin{bmatrix} 0 & 0 & d \end{bmatrix}^T$$

Since:

$$\mathbf{C}_1^*(\beta_s) = \begin{bmatrix} \mathbf{C}_{1f} \\ \mathbf{C}_{1s} \end{bmatrix} = \begin{bmatrix} 0 & 1 & 0 \\ 0 & 1 & 0 \end{bmatrix} \qquad \Rightarrow \quad \text{rank}(\mathbf{C}_1^*) = 1$$

Degree of mobility of the robot: $\delta_m = 3 - \text{rank}(\mathbf{C}_1^*) = 2$
Degree of steerability of the robot: $\delta_s = 0$
We have: $\dot{\xi} = [\dot{x} \;\; \dot{y} \;\; \dot{\theta}]^T \in \text{Ker}(\mathbf{C}_1^*)$ iff: $\mathbf{C}_1^*.{}^m\mathbf{\Omega}_0(\theta)\dot{\xi} = \mathbf{C}_1^*.{}^m\dot{\xi} = 0$
Therefore Ker$(\mathbf{C}_1^*)$ is: $\Sigma = \begin{bmatrix} 1 & 0 \\ 0 & 0 \\ 0 & 1 \end{bmatrix}$
The robot has no steerable wheels, there is no input u_s, and the dimension of input u_m is

$$\dim(u_m) = \dim(\delta_m) = 2$$

The posture kinematic equation is:

$$
\begin{aligned}
{}^0\dot{\xi} &= {}^0\mathbf{\Omega}_m(\theta)\Sigma u_m \\
\Rightarrow \quad \begin{bmatrix} \dot{x} \\ \dot{y} \\ \dot{\theta} \end{bmatrix} &= \begin{bmatrix} \cos\theta & -\sin\theta & 0 \\ \sin\theta & \cos\theta & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 \\ 0 & 0 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} V \\ \omega \end{bmatrix} = \begin{bmatrix} V\cos\theta \\ V\sin\theta \\ \omega \end{bmatrix}
\end{aligned} \tag{I.1}
$$

I.1.3 Configuration Kinematic Model

We have:

$$
\mathbf{D}(\beta_c) = -\mathbf{C}_{2c}^{-1}\mathbf{C}_{1c}(\beta_c) = \begin{bmatrix} -\sin\beta_3/d & \cos\beta_3/d & (d + a\cos\beta_3)/d \end{bmatrix}
$$

$$
\mathbf{E}(\beta_s, \beta_c) = -\mathbf{J}_2^{-1}\mathbf{J}_1(\beta_s, \beta_c) = \begin{bmatrix} 1/r & 0 & L/r \\ 1/r & 0 & -L/r \\ -\cos\beta_3/r & -\sin\beta_3/r & a\sin\beta_3/r \end{bmatrix}
$$

Thus:

$$
\mathbf{S}(q) = \begin{bmatrix} {}^0\mathbf{\Omega}_m(\theta)\Sigma(\beta_s) & 0_{3\times 2} \\ \mathbf{D}(\beta_c)\Sigma(\beta_s) & 0_{1\times 2} \\ \mathbf{E}(\beta_s, \beta_c)\Sigma(\beta_s) & 0_{3\times 2} \end{bmatrix} = \begin{bmatrix} \cos\theta & 0 \\ \sin\theta & 0 \\ 0 & 1 \\ -\sin\beta_3/d & -(d + a\cos\beta_3)/d \\ 1/r & L/r \\ 1/r & -L/r \\ -\cos\beta_3/r & a\sin\beta_3/r \end{bmatrix}
$$

I.1.4 Wheel Motorization

From the configuration kinematic model, we have:

$$
\begin{aligned}
\begin{bmatrix} \dot{\beta}_c \\ \dot{\varphi} \end{bmatrix} &= \begin{bmatrix} \mathbf{D}(\beta_c)\Sigma(\beta_s) \\ \mathbf{E}(\beta_s, \beta_c)\Sigma(\beta_s) \end{bmatrix} u_m = \mathbf{F}(\beta_c) u_m \\
\Rightarrow \quad \mathbf{F}(\beta_c) &= \begin{bmatrix} -\sin\beta_3/d & -(d + a\cos\beta_3)/d \\ 1/r & L/r \\ 1/r & -L/r \\ -\cos\beta_3/r & a\sin\beta_3/r \end{bmatrix}
\end{aligned}
$$

In this study it is decided that the spinning axes of 2 fixed wheels φ_1 and φ_2 will be motorized. Therefore, we have the equation:

$$
\dot{q}_m = \begin{bmatrix} \dot{\varphi}_1 \\ \dot{\varphi}_2 \end{bmatrix} = \mathbf{F}_{23} u_m = \begin{bmatrix} 1/r & L/r \\ 1/r & -L/r \end{bmatrix} \begin{bmatrix} V \\ \omega \end{bmatrix} \tag{I.2}
$$

$$
\text{and} \quad u_m = \begin{bmatrix} V \\ \omega \end{bmatrix} = \mathbf{F}_{23}{}^{-1}\dot{q}_m = \begin{bmatrix} r/2 & r/2 \\ r/2L & -r/2L \end{bmatrix} \begin{bmatrix} \dot{\varphi}_1 \\ \dot{\varphi}_2 \end{bmatrix} \tag{I.3}
$$

I.1.5 Modeling - Validation and Simulation

From the input $\dot{q}_m = \begin{bmatrix} \dot{\varphi}_1 & \dot{\varphi}_2 \end{bmatrix}^T$, we can use the Direct Kinematic Nodel to find u_m: $u_m = \mathbf{F}_{23}{}^{-1}\dot{q}_m$. Then the posture of the mobile robot can be found using localization, as shown in the Simulink model in Figure I.1.2 below:

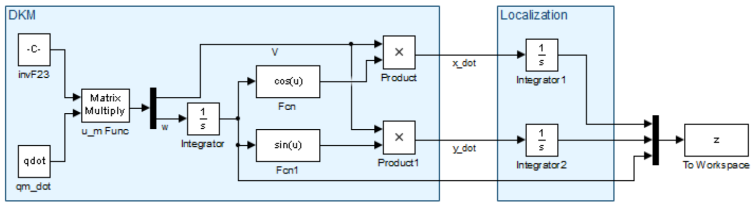

Figure I.1.2: Direct Kinematic Model and Localization for O_m of Robot Type (2,0)

Several sets of $\dot{q}_m$ inputs are tested on the model. For each pair of $[\dot{\varphi}_1\ \dot{\varphi}_2]$, the posture and trajectory of the robot within the first 10 seconds of the simulation are plotted:

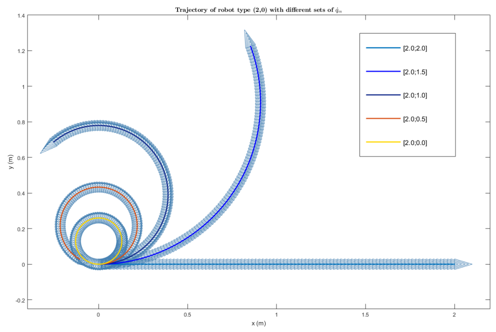

Figure I.1.3: Trajectory of the robot in 10s using Direct Kinematic Model (Robot Geometry is not to scale)

I.2 Static Decoupling Control Law

I.2.1 Theoretical Study

Since the largest linearizable subsystem has dimension 2 ($\delta_m + \delta_s = 2$), the system is not fully linearizable by static feedback. The static decoupling control law will be used for position tracking of point P not lying on the spinning axis of the fixed wheel.
The desired trajectory of point P is a circle with radius $R = 2m$ and origin $[0\ 0]$, therefore the its desired position at time t is defined by:

$$\mathbf{h}^d = \begin{bmatrix} R\cos(\omega^d t) \\ R\sin(\omega^d t) \end{bmatrix} \tag{I.4}$$

$$\text{and} \qquad \dot{\mathbf{h}}^d = \begin{bmatrix} -R\omega^d \sin(\omega^d t) \\ R\omega^d \cos(\omega^d t) \end{bmatrix} \tag{I.5}$$

We have position of point P in robot frame $\mathcal{R}_m$: ${}^mP = [d\ 0]^T$
Thus, its position in fixed frame $\mathcal{R}_0$ is:

$${}^0P = \mathbf{h} = \begin{bmatrix} x + d\cos\theta \\ y + d\sin\theta \end{bmatrix} \tag{I.6}$$

$$\Rightarrow \quad \dot{\mathbf{h}} = \begin{bmatrix} \dot{x} - d\sin(\theta)\dot{\theta} \\ \dot{y} + d\cos(\theta)\dot{\theta} \end{bmatrix}$$

Subsitututing the expressions of $\dot{x}$, $\dot{y}$ and $\dot{\theta}$ from equation (I.1):

$$\begin{aligned}\dot{\mathbf{h}} &= \begin{bmatrix} V\cos\theta - d\sin(\theta)\omega \\ V\sin\theta + d\cos(\theta)\omega \end{bmatrix} \\ &= \begin{bmatrix} \cos\theta & -d\sin\theta \\ \sin\theta & d\cos\theta \end{bmatrix} \begin{bmatrix} V \\ \omega \end{bmatrix} \\ &= \mathbf{K}(\theta)\mathbf{u}\end{aligned} \tag{I.7}$$

Introducting the auxiliary control $\mathbf{W}$ such that:

$$\begin{aligned} & \mathbf{W} = \dot{\mathbf{h}} = \mathbf{K}(\theta)\mathbf{u} \\ \Rightarrow \quad & \mathbf{u} = \mathbf{K}^{-1}(\theta)\mathbf{W}\end{aligned} \tag{I.8}$$

With the control law:

$$\begin{aligned} & \mathbf{W} = \dot{\mathbf{h}}^d + K_p(\mathbf{h}^d - \mathbf{h}) \\ \Rightarrow \quad & \dot{\mathbf{h}} = \dot{\mathbf{h}}^d + K_p(\mathbf{h}^d - \mathbf{h}) \\ \Rightarrow \quad & 0 = (\dot{\mathbf{h}}^d - \dot{\mathbf{h}}) + K_p(\mathbf{h}^d - \mathbf{h})\end{aligned} \tag{I.9}$$

Let $\mathbf{e}(t) = \mathbf{h}^d - \mathbf{h}$ be the error between the desired and actual positions, equation (I.9) becomes:

$$\begin{aligned} & \frac{d\mathbf{e}}{dt} + K_p\mathbf{e}(t) = 0 \\ \Rightarrow \quad & \frac{d\mathbf{e}}{\mathbf{e}} = -K_p dt \\ \Rightarrow \quad & \int_0^t \frac{d\mathbf{e}}{\mathbf{e}} = \int_0^t -K_p dt \\ \Rightarrow \quad & \ln\left(\frac{\mathbf{e}(t)}{\mathbf{e}(0)}\right) = -K_p t \\ \Rightarrow \quad & \mathbf{e}(t) = \mathbf{e}(0)\mathrm{e}^{-K_p t} \\ \Rightarrow \quad & \lim_{t\to\infty} \mathbf{e}(t) = 0\end{aligned}$$

Therefore the system is asymptotically exponentially stable. Which means the tracking errors $\mathbf{e}(t) = [e_x(t)\ e_y(t)]^T$ will converge to 0 over time.
As for the orientation of the robot, it can be proven that even though it is uncontrollable, the error will also converge to a constant value: We recall that the state vector of the system is $\mathbf{z}$ where $\dim(\mathbf{z}) = 3$, while it is trivial to show that the relative degree of the system is 2. Since the relative degree is less than the number of state variables, there exists an internal dynamic in the system, which is the orientation θ. When $t \to \infty$, we have:

$$\begin{aligned} & \lim_{t\to\infty} \mathbf{e}(t) = \lim_{t\to\infty} (\mathbf{h}^d - \mathbf{h}) = 0 \\ \Rightarrow\ & \lim_{t\to\infty} \mathbf{W} = \dot{\mathbf{h}}^d + K_p(\mathbf{h}^d - \mathbf{h}) = \dot{\mathbf{h}}^d \\ \text{From Eqn. (I.5)}\ \Rightarrow\ & \lim_{t\to\infty} \mathbf{W} = \dot{\mathbf{h}}^d = \begin{bmatrix} -R\omega^d \sin(\omega^d t) \\ R\omega^d \cos(\omega^d t) \end{bmatrix} \\ \text{From Eqn. (I.8)}\ \Rightarrow\ & \mathbf{u}_\infty = \mathbf{K}^{-1}(\theta)\mathbf{W}_\infty = \begin{bmatrix} \cos\theta & \sin\theta \\ \dfrac{-\sin\theta}{d} & \dfrac{\cos\theta}{d} \end{bmatrix} \begin{bmatrix} -R\omega^d \sin(\omega^d t) \\ R\omega^d \cos(\omega^d t) \end{bmatrix} \\ \Rightarrow\ & \begin{bmatrix} V_\infty \\ \omega_\infty \end{bmatrix} = \begin{bmatrix} R\omega^d \sin(\theta - \omega^d t) \\ \dfrac{R\omega^d}{d} \cos(\theta - \omega^d t) \end{bmatrix}\end{aligned}$$

When the system has stabilized, we have $\omega_\infty = \omega^d$, therefore:

$$\begin{aligned}
\omega_\infty = \omega^d &= \frac{R\omega^d}{d}\cos(\theta - \omega^d t) \\
\Rightarrow \quad \cos(\theta - \omega^d t) &= \frac{d}{R} \\
\Rightarrow \quad \sin\left(\frac{\pi}{2} - (\theta - \omega^d t)\right) &= \frac{d}{R} \\
\Rightarrow \quad \sin(\theta^d - \theta) &= \frac{d}{R} \\
\Rightarrow \quad \theta^d - \theta &= \arcsin\left(\frac{d}{R}\right) \\
\Rightarrow \quad e_\theta &= \arcsin\left(\frac{d}{R}\right)
\end{aligned}$$

Thus, it is proven that when the system become stable, the error in orientation e_θ is also stable and converges to a constant value. By substituting $d = 0.15$m and $R = 2$m, we can calculate $e_\theta = 0.075$rad.

I.2.2 Modeling

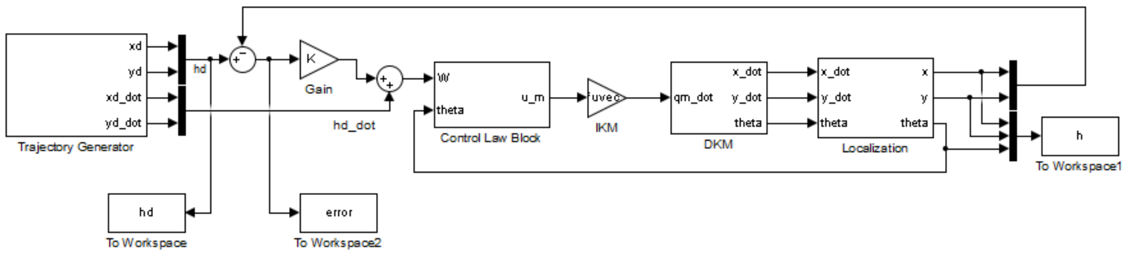

Figure I.2.1: Global Simulink Model for Static Decoupling Control of Robot Type (2,0)

Figure I.2.1 above shows the implementation of static decoupling control law for root type (2,0):

- `Trajectory Generator` block outputs the desired positions to track using equations (I.4) and (I.5).
- `Control Law` block employs equation (I.8) to calculate the input signal u_m.
- `IKM` block uses the Inverse Kinematic Model equation (I.2) to find $\dot{q}_m$, which are $\dot{\varphi}_1$ and $\dot{\varphi}_2$. The values of wheel radius (r_e) and track gauge ($2L_e$) are estimated.
- `DKM` block is the same as the one developed in the previous section. The wheel radius and track gauge used in this block are r and $2L$, respecitvely.
- `Localization` block calculates the actual position of tracking point P' using equation (I.6).

The detail of the subsystems are shown below:

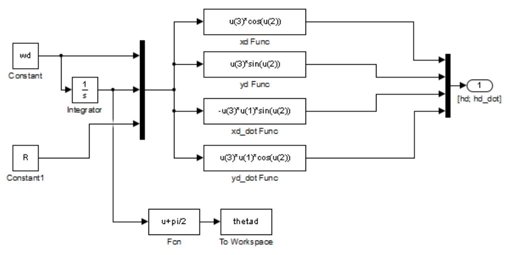

Figure I.2.2: Localization Subsystem for Robot type (2,0) using Static Decoupling Control Law

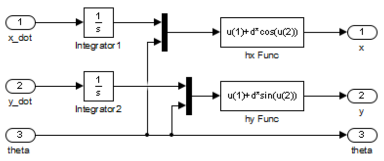

Figure I.2.3: Trajectory Generator Subsystem for Static Decoupling Control Law

I.2.3 Validation and Simulation

Remark: The simulations are carried out using Runge-Kutta (ode4) fixed-step solver with 0.01s step size. The Static decoupling control linearizes the non-linear system to a first-order one. Therefore, the steps to tune its proportional gain are:

1. Set K_p at 1 to evaluate the default performance of the system.
2. Increase the value of K_p until the system becomes marginally unstable then slightly reduce it.
3. In practice, other aspects of the system must be taken into account. For instance,the value of K_p might need to be reduced so that the inputs do not exceed the motor saturation.

Case 1: $\mathbf{L}_{\text{est}} = \mathbf{L} \mid \mathbf{r}_{\text{est}} = \mathbf{r}$

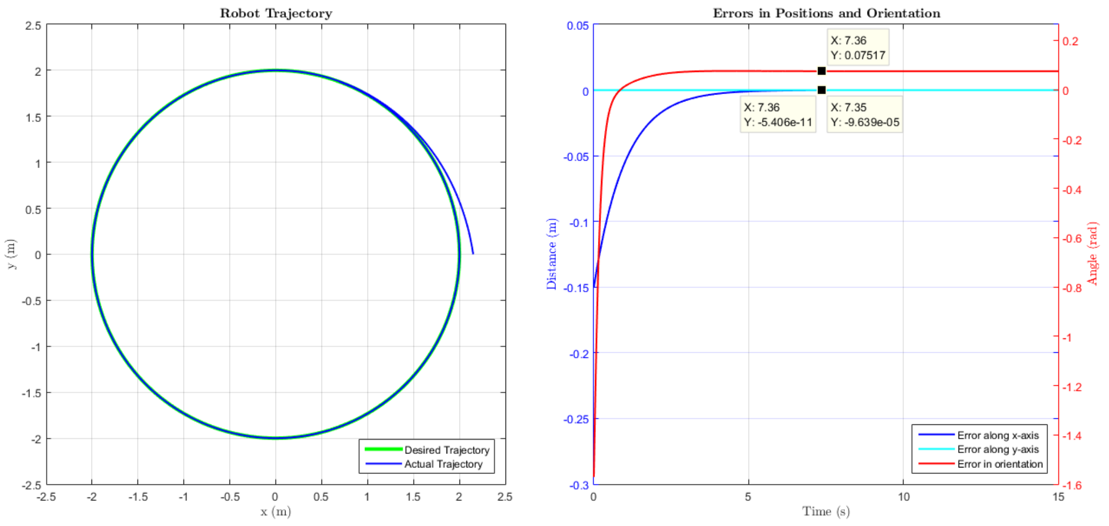

Figure I.2.4: Trajectory and Tracking Errors of Type (2,0) using static decoupling control law, with $K_p = 1$

Remark: The performance of each model settings is plotted on 2 graphs:

• The left subplot shows the desired and actual trajectory of the robot on (x,y) plane over 15 seconds.

• The right subplot, which has 2 y-axes (Left y-axis: Distance/Right y-axis: Angle), shows the error in position and orientation of the robot. All performance plots of the robot onward will follow this fashion.

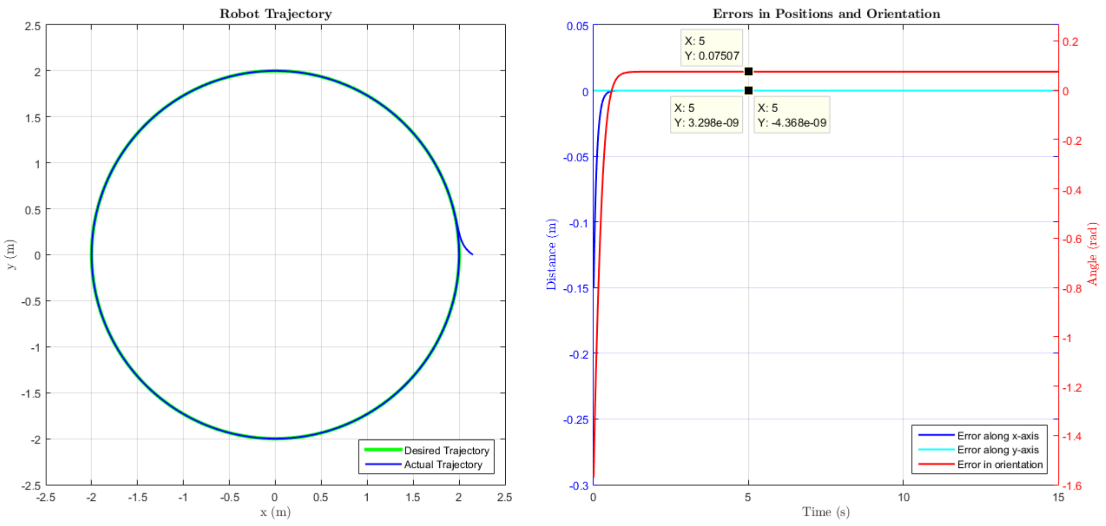

Figure I.2.5: Trajectory and Tracking Errors of Type (2,0) using static decoupling control law, with $K_p = 10$

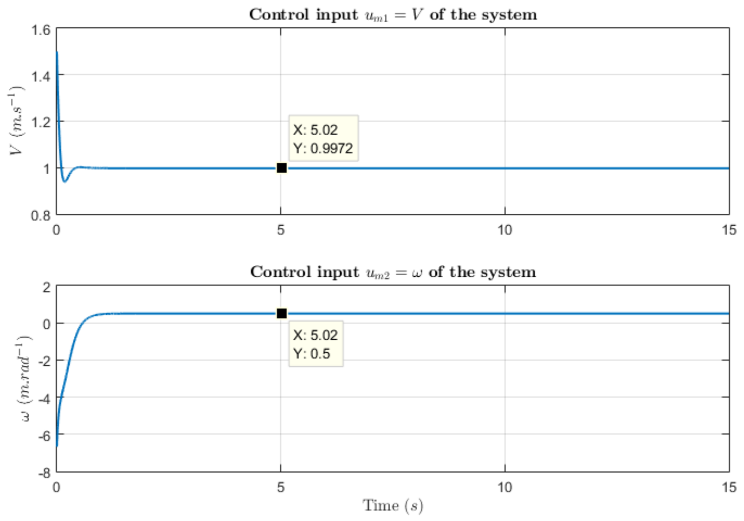

Figure I.2.6: Control Input of the system

From Figure I.2.4, it can be seen that the position tracking error converges to zero while orientation error converges to a constant value (0.075rad), as discussed in the previous Theoretical Study section. The graph also shows that the response time (rise time) and settling time of the robot is relative slow, at around 2.3sec and 7.35sec, respectively. Thus, the proportional gain K_p is increased to a reasonable value to ensure a shorter response time while keeping the system stable. The final value of K_p is chosen to be 10, so that the response time will be reduced to 0.23sec and settling time to to about 1.5sec, which is shown in Figure I.2.5. Even though K_p can be inscreased further to have an even shorter response, doing so might not be realistic if motor saturation is taken to account, as mentioned before.

Case 2: With 10% change in the estimation of half track gauge

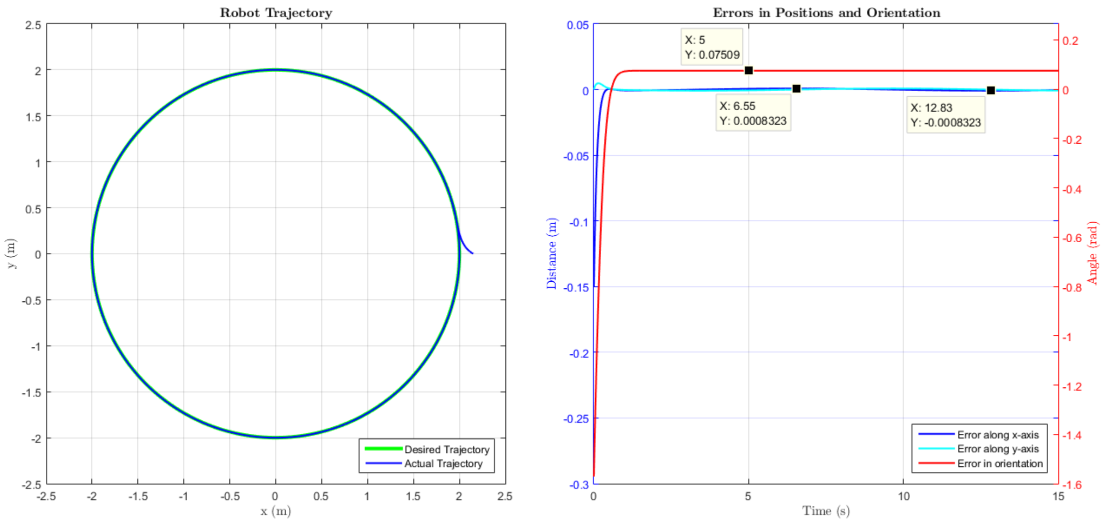

Figure I.2.7: Trajectory and Tracking Errors of Type (2,0) using static decoupling control law, with $K_p = 10$ and $L_{\text{est}} = 90\%L$

Figure I.2.7 above shows that a 10% reduction in the estimation of L has caused some changes in the tracking error: instead of converging to zero, now the tracking error exhibit the characteristics of a sine wave with magnitude in 10^{-4} order and the period of 12.56s, which is the time it takes to completely finish 1 revolution on the desired circle. The plots of error in x and y have the same magnitude and period but are different in phase. For the case when L is increased 10%, the errors show the same characteristics but with different phases.On the other hand, the error in orientation still converges to a constant value, which slightly different from that of the previous case, and can be neglected.
In conclusion, the change in the estimation of track gauge is not a severe error, and the error magnitude can be easily reduced by slightly increase the proportional gain K_p.

Case 3: With 10% change in the estimation of fixed wheel radius

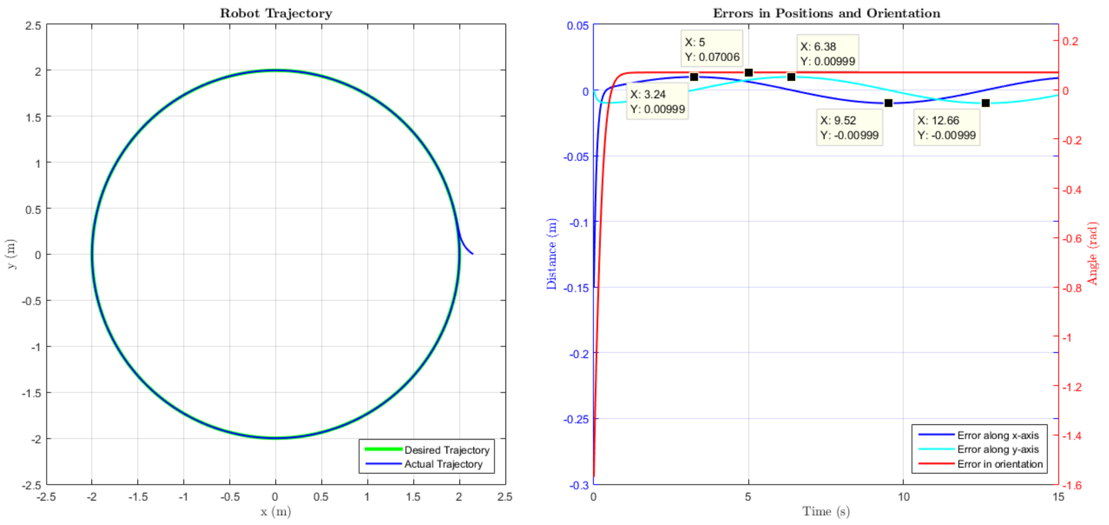

Figure I.2.8: Trajectory and Tracking Errors of Type (2,0) using static decoupling control law, with $K_p = 10$ and $r_{\text{est}} = 90\%r$

The change in estimation of fixed wheel radius caused severe problem in position tracking. While the period of the sinusoidals remains the same as previous case, the error magnitude is in 10^{-2} order and cannot be neglected. The error magnitude can be reduced by significantly increase the K_p gain, but cannot be completely removed. Compare to the previous case, the orientation error changed ± 0.05(rad), which corresponds to $\pm 10\%$ change in radius.

I.2.4 Fixed Point Tracking

For this section, a `Switch Block` is added to the simulink model to select either circular trajectory tracking or fixed point tracking. For this case, $\mathbf{h}^d$ is a constant, therefore $\dot{\mathbf{h}}^d = [0\ 0]^T$, as shown in Figure I.2.9.

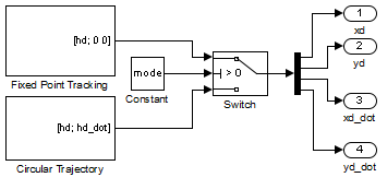

Figure I.2.9: Modified Trajectory Generator Block with Mode Switch added

The figure bellow shows the trajectory and error when the robot tracks the fixed point $[-1\ \ -1]^T$ from it initial position $[2.3\ 0]^T$. The desired orientation is also specified to show that this variable cannot be controlled by this model, which is indicated by the orientation error in the plot. As shown, the position can be successfully tracked while the orientation is not.

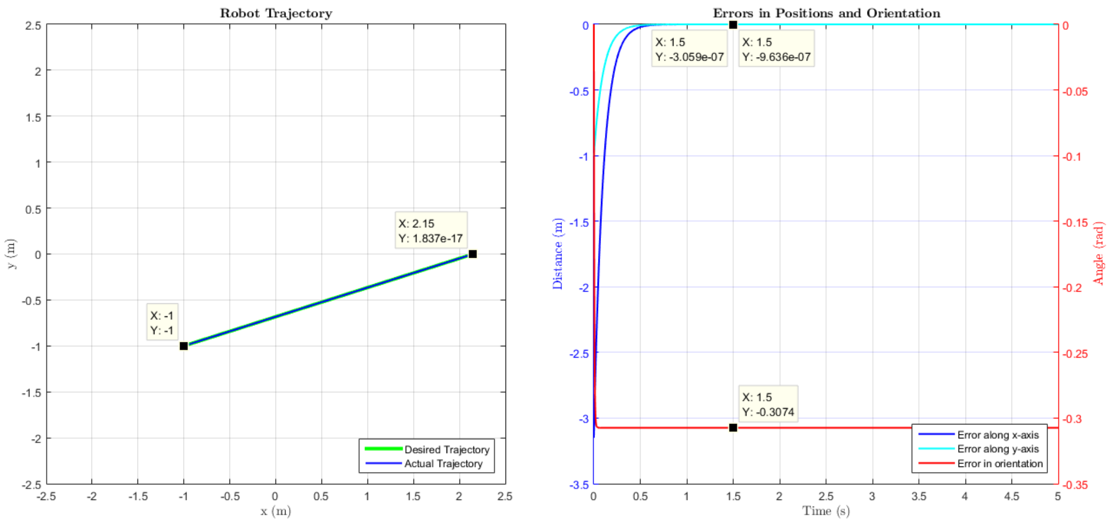

Figure I.2.10: Trajectory and Errors of Type (2,0) when tracking a fixed point using static decoupling control law

Remark: In order to track both position and orientation, one need 2 simulink models: the position tracking model will be run until the robot reaches its destination, then another model which tracks orientation is run to rotate the robot to the desired orientation. In other words, for the robot to move from one posture to another, first it has to rotate toward the destination, then translate to goal, and rotate again to achieve desire orientation, by using 2 different control system.

I.3 Dynamic Decoupling Control

In this section, instead of using point P located on the x-axis of robot frame $\mathcal{R}_m$, we will use the origin O_m of $\mathcal{R}_m$.

I.3.1 Theoretical Study

From Equation (I.4) and (I.5), we have the second derivative of desired position:

$$\ddot{\mathbf{h}}^d = \begin{bmatrix} -R(\omega^d)^2\cos(\omega^d t) \\ -R(\omega^d)^2\sin(\omega^d t) \end{bmatrix} \tag{I.10}$$

In order to use the dynamic decoupling control law, we consider V as a new state variable. The new input of the system becomes $\mathbf{u}' = [\dot{V}\ \omega]^T$.
Since $\mathbf{h} = [x\ y]^T$ in this case, we have:

$$\dot{\mathbf{h}} = \begin{bmatrix} \dot{x} \\ \dot{y} \end{bmatrix} = \begin{bmatrix} V\cos\theta \\ V\sin\theta \end{bmatrix} \tag{I.11}$$

$$\begin{aligned} \Rightarrow \quad \ddot{\mathbf{h}} &= \begin{bmatrix} \ddot{x} \\ \ddot{y} \end{bmatrix} = \begin{bmatrix} \dot{V}\cos\theta - V\sin(\theta)\dot{\theta} \\ \dot{V}\sin\theta + V\cos(\theta)\dot{\theta} \end{bmatrix} \\ &= \begin{bmatrix} \cos\theta & -V\sin\theta \\ \sin\theta & V\cos\theta \end{bmatrix} \begin{bmatrix} \dot{V} \\ \dot{\theta} \end{bmatrix} \\ &= \mathbf{F}(\theta)\mathbf{u}' \end{aligned} \tag{I.12}$$

Introducing an auxiliary control $\mathbf{W}$ such that:

$$\begin{aligned} \mathbf{W} &= \ddot{\mathbf{h}} = \mathbf{F}(\theta)\mathbf{u}' \\ \Rightarrow \quad \mathbf{u}' &= \mathbf{F}^{-1}(\theta)\mathbf{W} \quad (V \neq 0) \end{aligned} \tag{I.13}$$

Define $\mathbf{e}(t) = \mathbf{h}^d - \mathbf{h}$, which is the tracking error. By using the control law:

$$\mathbf{W} = \ddot{\mathbf{h}}^d + K_v(\dot{\mathbf{h}}^d - \dot{\mathbf{h}}) + K_p(\mathbf{h}^d - \mathbf{h}) \tag{I.14}$$

$$\begin{aligned} \Rightarrow \quad \ddot{\mathbf{h}} &= \ddot{\mathbf{h}}^d + K_v(\dot{\mathbf{h}}^d - \dot{\mathbf{h}}) + K_p(\mathbf{h}^d - \mathbf{h}) \\ \Rightarrow \quad 0 &= (\ddot{\mathbf{h}}^d - \ddot{\mathbf{h}}) + K_v(\dot{\mathbf{h}}^d - \dot{\mathbf{h}}) + K_p(\mathbf{h}^d - \mathbf{h}) \\ \Rightarrow \quad 0 &= \ddot{\mathbf{e}}(t) + K_v\dot{\mathbf{e}}(t) + K_p\mathbf{e}(t) \end{aligned} \tag{I.15}$$

The characteristic equation of the second order differential equation of $\mathbf{e}(t)$ is: $s^2 + K_v s + K_p = 0$.
Since K_v and K_p are postive, the roots of the characteristic equation always have negative real part:

- If $\Delta = \sqrt{K_v^2 - 4K_p} \geq 0$, the roots are $s_{1,2} = \frac{-K_v \pm \Delta}{2} < 0$. Thus:

$$\begin{aligned} \mathbf{e}(t) &= c_1 \mathrm{e}^{s_1 t} + c_2 \mathrm{e}^{s_2 t} \\ \Rightarrow \quad \lim_{t\to\infty} \mathbf{e}(t) &= 0 \end{aligned}$$

- If $\Delta = \sqrt{K_v^2 - 4K_p} < 0$, the roots are $s_{1,2} = \frac{-K_v \pm i\Delta}{2}$. Thus:

$$\begin{aligned} \mathbf{e}(t) &= \mathrm{e}^{\frac{-K_v}{2}t}(c_1\cos\Delta t + c_2\sin\Delta t) \\ \Rightarrow \quad \lim_{t\to\infty} \mathbf{e}(t) &= 0 \end{aligned}$$

Therefore, it can be concluded that the system is globally asymptotically stable.
It can also be proven that the error in orientation e_θ of the robot will converges to 0 when $t \to \infty$. As shown above, we already have:

$$\lim_{t\to\infty} \mathbf{e}(t) = \lim_{t\to\infty} \dot{\mathbf{e}}(t) = 0$$

From Equation (I.14) and (I.12):

$$\lim_{t\to\infty} \mathbf{W} = \ddot{\mathbf{h}}^d = \begin{bmatrix} -R(\omega^d)^2 \cos(\omega^d t) \\ -R(\omega^d)^2 \sin(\omega^d t) \end{bmatrix}$$

$$\text{From Eqn. (I.13)} \ \Rightarrow \lim_{t\to\infty} = \lim_{t\to\infty} \mathbf{F}^{-1}(\theta)\mathbf{W}$$

$$\Rightarrow \quad \begin{bmatrix} \dot{V}_\infty \\ \omega_\infty \end{bmatrix} = \begin{bmatrix} \cos\theta & \sin\theta \\ -\frac{\sin\theta}{V_\infty} & \frac{\cos\theta}{V_\infty} \end{bmatrix} \begin{bmatrix} -R(\omega^d)^2 \cos(\omega^d t) \\ -R(\omega^d)^2 \sin(\omega^d t) \end{bmatrix}$$

$$\Rightarrow \quad \omega_\infty = \frac{R(\omega^d)^2}{V_\infty} \sin(\theta - \omega^d t)$$

$$\text{With } \omega_\infty = \omega^d \text{ and } V_\infty = R\omega^d \quad \Rightarrow \quad \sin(\theta - \omega^d t) = 1$$

$$\theta - \omega^d t = \frac{\pi}{2}$$

$$(\frac{\pi}{2} + \omega^d t) - \theta = 0$$

$$\Rightarrow \quad e_\theta = \theta^d - \theta = 0$$

I.3.2 Modeling

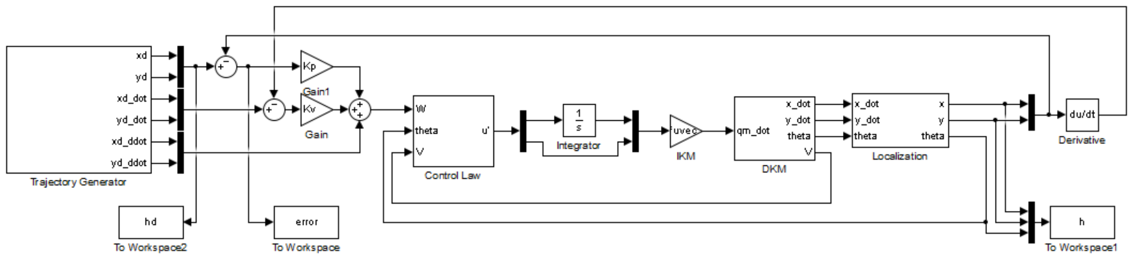

Figure I.3.1: Global Simulink model for Robot type (2,0) using Dynamic Decoupling control law

Figure I.3.1 above shows the Simulink model of the dynamic decoupling control loop for robot type (2,0), which consists of:

- `Trajectory Generator` block provides the values of $\mathbf{h}^d$, $\dot{\mathbf{h}}^d$ and $\ddot{\mathbf{h}}^d$ for every sampling step.
- `Control Law` block gets the value of $\mathbf{W}$, V and θ to calculate $\mathbf{F}^{-1}(\theta)$ and input $\mathbf{u}'$.
- Element $\dot{V}$ of $\mathbf{u}'$ is integrated to find V. Note that the initial value of V must be specified in the integrator block with a value different from 0 to avoid singularity.
- `IKM` block is the Inverse Kinematic model of the mobile robot, which will output $\dot{\varphi}_1$ and $\dot{\varphi}_2$ using the estimated values L_e and r_e.
- `DKM` block is the Direct Kinematic model computing ${}^0\dot{\xi}$ from $\dot{q}_m = [\dot{\varphi}_1 \ \dot{\varphi}_2]^T$.
- `Localization` block can compute the posture ${}^0\xi$ of the robot, as well as $\mathbf{h}$ and $\dot{\mathbf{h}}$.

I.3.3 Validation and Simulation

Remark: The simulations are carried out using Dormand-Prince(ode5) fixed-step solver with 0.001s step size for 15s.
The values of K_v and K_p are tuned so that the response time is 1s and the errors do not exhibit any oscillation. The Equation (I.15): $\ddot{\mathbf{e}}(t) + K_v\dot{\mathbf{e}}(t) + K_p\mathbf{e}(t) = 0$ shows that this is a second order system. Consequently, the equation can be rewritten in the form of: $\ddot{\mathbf{e}}(t) + 2\zeta\omega_n\dot{\mathbf{e}}(t) + \omega_n^2\mathbf{e}(t) = 0$. Where ζ and ω_n are the damping ratio and natural frequency of the system, respectively.

- To remove the oscillations in the response of $\mathbf{e}(t)$, damping ratio ζ must be greater or equal to 1.
- In case $\zeta = 1$, it is proven that in order to achieve the response time equal to 1sec, $\zeta\omega_n = 5$, thus $\omega_n = 5$.
- Therefore, we can tune the gains $K_v = 2\zeta\omega_n = 10$ and $K_p = \omega_n^2 = 25$.

Case 1: $\mathbf{L}_{\text{est}} = \mathbf{L} \mid \mathbf{r}_{\text{est}} = \mathbf{r}$

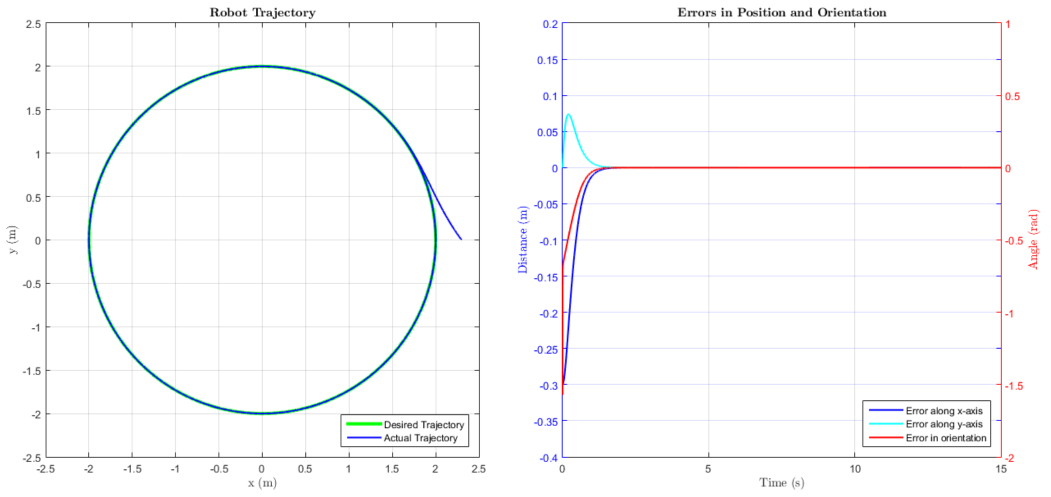

Figure I.3.2: Trajectory and Errors of robot type (2,0) using Dynamic decoupling control law with $K_v = 10$ and $K_p = 25$

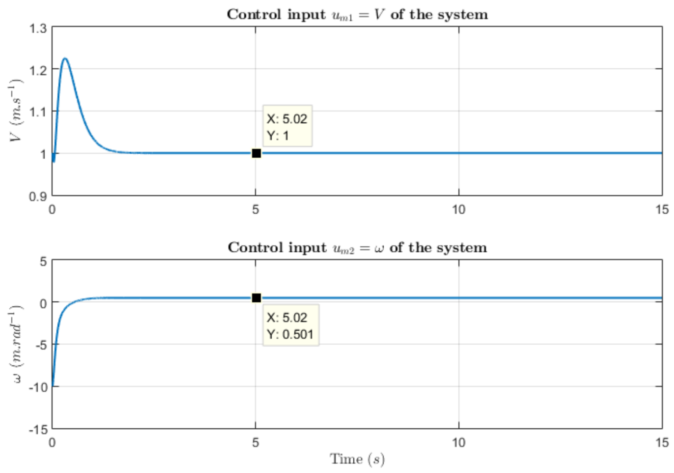

Figure I.3.3: Control input of the system

Figure I.3.2 above plot the trajectory and error of the robot type (2,0). As shown, the response time (the time it takes for the response to rise to 90%) is 1s while the settling time is about 2s. It is also validate that the error in x, y and θ all converge to 0 over time.

Case 2: There is 10% change in the estimated value of L

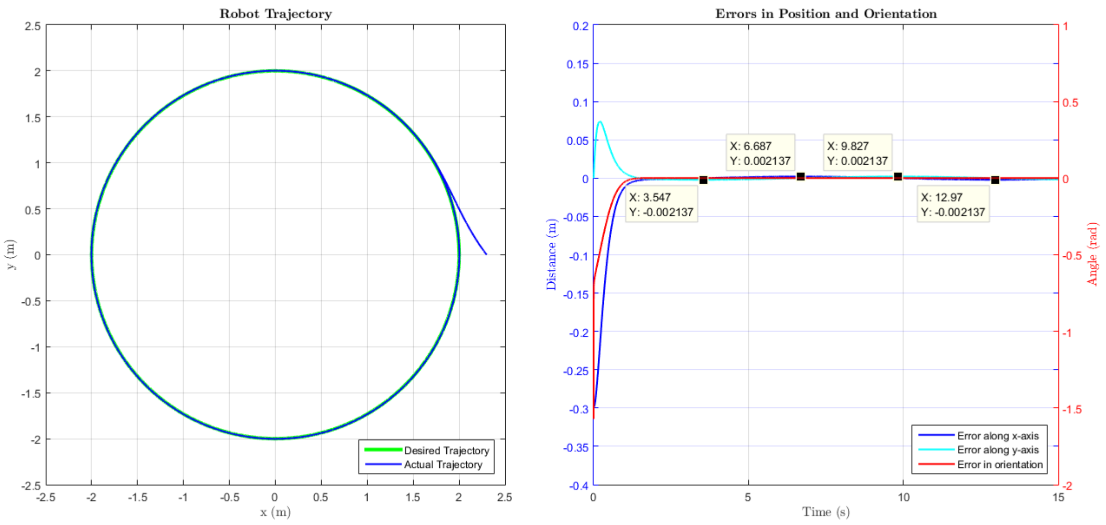

Figure I.3.4: Trajectory and Errors of robot type (2,0) using Dynamic decoupling control law with same gains and $L_{\text{est}} = 90\%L$

When the changes in the estimated value of half track gauge is ±10%, the errors e_x and e_y exhibit the characteristics similar to those of a sinusoidal, where the magnitude is 0.00185m and the period is about 12.56S (time to complete one circular trajectory), with different phases.
On the other hand, the change in r cause the error in orientation e_θ to converge to constant value instead of 0: $L_\theta = 0.00018$rad when $L_{\text{est}} = 110\%r$ and $e_\theta = -0.00018$rad when $L_{\text{est}} = 90\%L$.

Case 2: There is 10% change in the estimated value of r

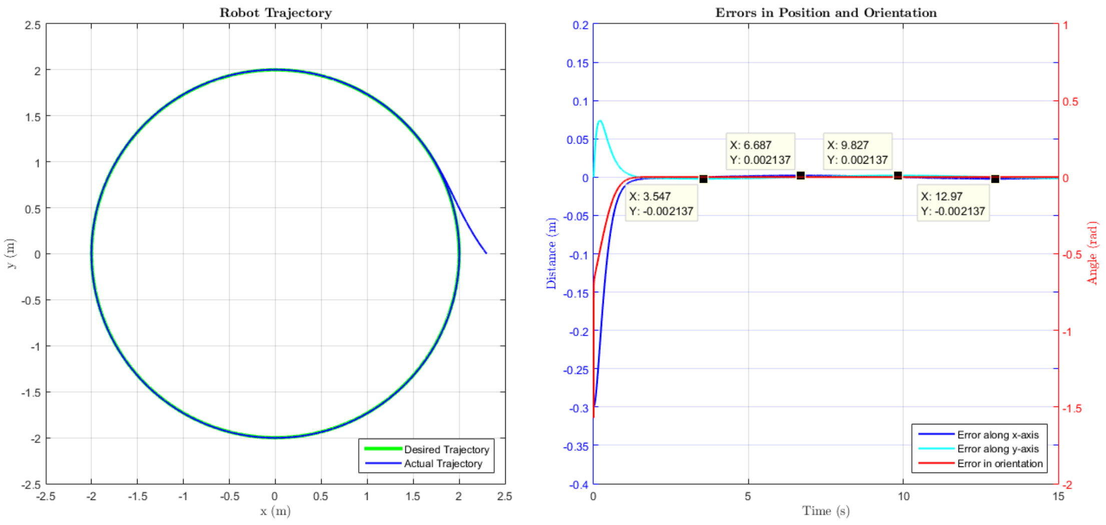

Figure I.3.5: Trajectory and Errors of robot type (2,0) using Dynamic decoupling control law with same gains and $r_{\text{est}} = 90\%r$

When the changes in the estimated value of fixed wheel radius is $\pm 10\%$, the errors e_x and e_y exhibit the characteristics similar to those of a sinusoidal, where the magnitude is 0.002m and the period is about 12.56S, with different phases. It can be seen that the magnitude of the error is 1 order smaller than the error in static decoupling control case.
On the other hand, the change in r cause the error in orientation e_θ to converge to constant value instead of 0: $e_\theta = 0.0002\text{rad}$ when $r_{\text{est}} = 110\% r$ and $e_\theta = -0.0002\text{rad}$ when $r_{\text{est}} = 90\% r$.

I.4 Lyapunov Control Law

I.4.1 Theoretical Study

The following Lyapunov function candidate to be used:

$$W = \frac{1}{2}(x_e^2 + y_e^2 + \frac{\theta_e^2}{K_y}) \tag{I.16}$$

$$\text{We have to make sure that: } \dot{W} = x_e \dot{x_e} + y_e \dot{y_e} + \frac{\theta_e \dot{\theta_e}}{K_y} \leq 0 \tag{I.17}$$

In the mobile robot frame $\mathcal{R}_m$, let ${}^m\xi_d$ be the desired posture to be tracked, and ${}^m\xi_e$ is the error between the reference and actual posture of the robot. The desired and error input are also defined as $\mathbf{u}_d$ and $\mathbf{u}_e$, respectively.

We will find the derivative ${}^m\dot{\xi}_e$ as a function of ${}^m\xi_e$, $\mathbf{u}_d$, $\mathbf{u}_e$ as followed: ${}^m\xi_e = \begin{bmatrix} x_e \\ y_e \\ \theta_e \end{bmatrix} = {}^m\mathbf{R}_0({}^0\xi_d - {}^0\xi)$

$$\Rightarrow \quad {}^m\dot{\xi}_e = \begin{bmatrix} \dot{x}_e \\ \dot{y}_e \\ \dot{\theta}_e \end{bmatrix} = {}^m\dot{\mathbf{R}}_0({}^0\xi_d - {}^0\xi) + {}^m\mathbf{R}_0({}^0\dot{\xi}_d - {}^0\dot{\xi}) = {}^m\dot{\mathbf{R}}_0 {}^0\mathbf{R}_m {}^m\xi_e + {}^m\mathbf{R}_0(\mathbf{B}_d\mathbf{u}_d - \mathbf{B}\mathbf{u})$$

$$\text{Where} \quad {}^0\mathbf{R}_m = \begin{bmatrix} \cos\theta & -\sin\theta & 0 \\ \sin\theta & \cos\theta & 0 \\ 0 & 0 & 1 \end{bmatrix} \qquad {}^m\mathbf{R}_0 = \begin{bmatrix} \cos\theta & \sin\theta & 0 \\ -\sin\theta & \cos\theta & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

$${}^m\dot{\mathbf{R}}_0 = \begin{bmatrix} -\sin\theta & \cos\theta & 0 \\ -\cos\theta & -\sin\theta & 0 \\ 0 & 0 & 0 \end{bmatrix} \dot{\theta} = \begin{bmatrix} -\sin\theta & \cos\theta & 0 \\ -\cos\theta & -\sin\theta & 0 \\ 0 & 0 & 0 \end{bmatrix} \omega$$

$$\mathbf{B}_d\mathbf{u}_d = \begin{bmatrix} \cos\theta_d & 0 \\ \sin\theta_d & 0 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} V_d \\ \omega_d \end{bmatrix} = \begin{bmatrix} V_d\cos\theta_d \\ V_d\sin\theta_d \\ \omega_d \end{bmatrix}$$

$$\mathbf{B}\mathbf{u} = \begin{bmatrix} \cos\theta & 0 \\ \sin\theta & 0 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} V \\ \omega \end{bmatrix} = \begin{bmatrix} V\cos\theta \\ V\sin\theta \\ \omega \end{bmatrix}$$

$$\Rightarrow \quad {}^m\dot{\xi}_e = \begin{bmatrix} \omega y_e + Vr(\text{C}\theta\text{C}\theta_d + \text{S}\theta\text{S}\theta_d) - V \\ -\omega x_e + V_d(\text{S}\theta_d\text{C}\theta - \text{C}\theta_d\text{S}\theta) \\ \omega_d - \omega \end{bmatrix}$$

$$= \begin{bmatrix} \omega y_e + V_d\text{C}\theta_e + V_e - V_d \\ -\omega x_e + V_d\text{S}\theta_e \\ \omega_e \end{bmatrix}$$

$$= \begin{bmatrix} 0 & \omega_d & \frac{V_d(\text{C}\theta_e - 1)}{\theta_e} \\ -\omega_d & 0 & \frac{V_d\text{S}\theta_e}{\theta_e} \\ 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} x_e \\ y_e \\ \theta_e \end{bmatrix} + \begin{bmatrix} 1 & -y_e \\ 0 & x_e \\ 0 & 1 \end{bmatrix} \begin{bmatrix} V_e \\ \omega_e \end{bmatrix} \tag{I.18}$$

Substituting the value of ${}^{m}\dot{\xi}_e = [\dot{x}_e \ \dot{y}_e \ \dot{\theta}_e]^T$ from Equation (I.18) to Equation (I.17), we have:

$$\dot{W} = (V_d(\cos\theta_e - 1) + V_e)\, x_e + \left(V_d y_e \frac{\sin\theta_e}{\theta_e} + \frac{\omega_e}{K_y}\right)\theta_e \leq 0 \tag{I.19}$$

In order to make sure that the condition in Equation (I.19) is always satisfied, we define K_x and K_θ such that:

$$\begin{aligned} K_x x_e &= -\left(V_d(\cos\theta_e - 1) + V_e\right) \\ \frac{K_\theta}{K_y}\theta_e &= -\left(V_d y_e \frac{\sin\theta_e}{\theta_e} + \frac{\omega_e}{K_y}\right) \\ \Rightarrow \quad \dot{W} &= -K_x x_e^2 - \frac{K_\theta}{K_y}\theta_e^2 \leq 0 \quad (\forall K_x, K_y, K_\theta > 0) \end{aligned} \tag{I.20}$$

The equilibrium point of the Lyapunov function is $[0\ 0\ 0]^T$ which means the system is globally asymptotically stable, the error in robot posture will always coverge to 0 given any initial value.
Furthermore, from Equation (I.20), the error in input $\mathbf{u}_e$ is:

$$\mathbf{u}_e = \begin{bmatrix} -V_d(\cos\theta_e - 1) - K_x x_e \\ -K_y V_d y_e \frac{\sin\theta_e}{\theta_e} - K_\theta\theta_e \end{bmatrix} \tag{I.21}$$

Thus the control input is:

$$\begin{aligned} \mathbf{u} = \mathbf{u}_d - \mathbf{u}_e &= \begin{bmatrix} V_d \\ \omega_d \end{bmatrix} - \begin{bmatrix} -V_d(\cos\theta_e - 1) - K_x x_e \\ -K_y V_d y_e \frac{\sin\theta_e}{\theta_e} - K_\theta\theta_e \end{bmatrix} \\ &= \begin{bmatrix} V_d\cos\theta_e + K_x x_e \\ \omega_d + K_y V_d y_e \frac{\sin\theta_e}{\theta_e} + K_\theta\theta_e \end{bmatrix} \end{aligned} \tag{I.22}$$

I.4.2 Modeling

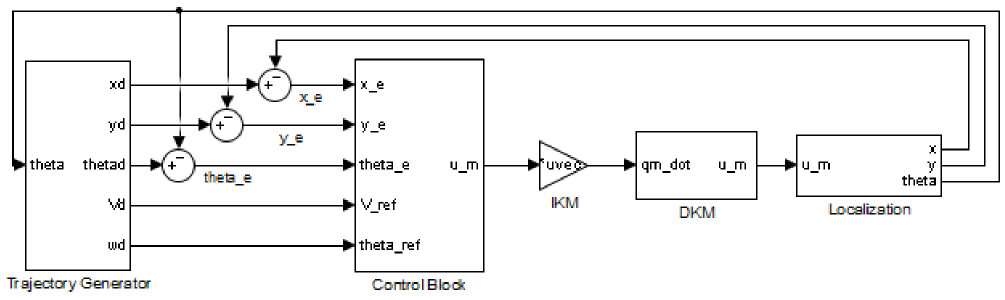

Figure I.4.1: Global Simulink model for robot type (2,0) using Lyapunov control law

Figure I.4.1 above shows the global Simulink model for robot type (2,0) using Lyapunov control law:

- `Trajectory Generator` block ouputs the desired posture of the robot in robot ${}^{m}\xi_d$ in robot frame $\mathcal{R}_m$ as well as the desired control input $\mathbf{u}_d$, as shown in Figure I.4.2.
- `Control` block calculates the control input using Equation I.22, as shown in Figure I.4.3.
- `IKM` and `DKM` blocks remain the same as previous cases.
- `Localization` block calculate the actual posture of the robot ${}^{m}\xi$ in robot frame, as shown in Figure I.4.4.

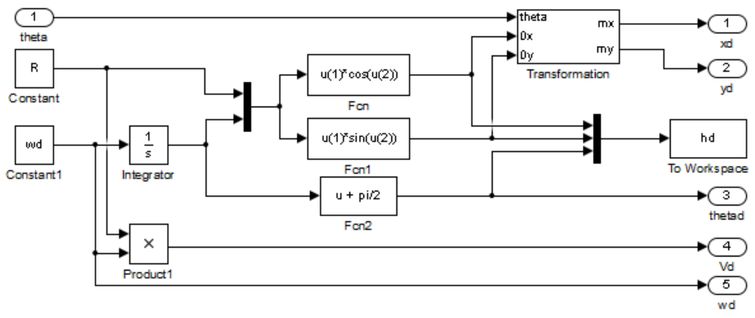

Figure I.4.2: Trajectory Generator block for Lyapunov Control law model

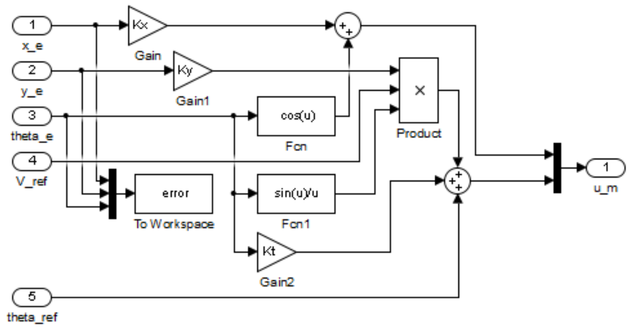

Figure I.4.3: Control block for Lyapunov Control law model

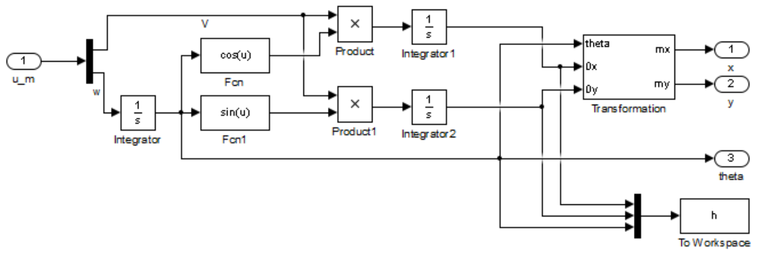

Figure I.4.4: Localization block for Lyapunov Control law model

I.4.3 Validation and Simulation

Remark: The simulations are carried out using Dormand-Prince(ode5) fixed-step solver with 0.001s step size for 15s.

Tuning the Gains $\mathbf{K_x}$, $\mathbf{K_y}$, $\mathbf{K_\theta}$

In order to tune the gains for Lyapunov control, we will use the following procedure:

1. Set $K_x = K_y = K_\theta = 1$ to observe the response of the robot.
2. Change K_x, K_y, K_θ individually to observe the impact of each gain to the system.
3. Adjust the gains according to their impacts to achieve the desirable results such as response time and settling time.

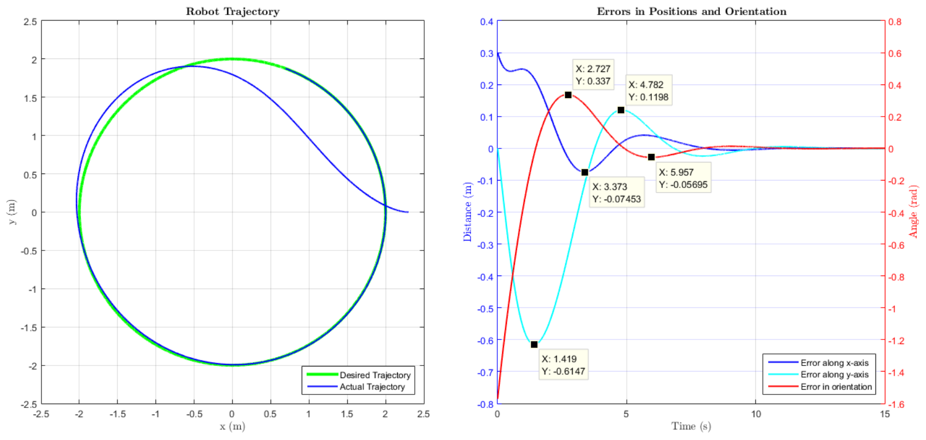

Figure I.4.5: Trajectory and Error response in $\mathcal{R}_m$ frame of robot type (2,0) using Lyapunov control with $K_x = K_y = K_\theta = 1$

Figure I.4.5 shows that with $K_x = K_y = K_\theta = 1$, even though the posture error of the robot converges to 0 eventually, the response and settling time of the system are very slow. Therefore, we increase the gains to 10 individually to observe the impact of each gain to the system:

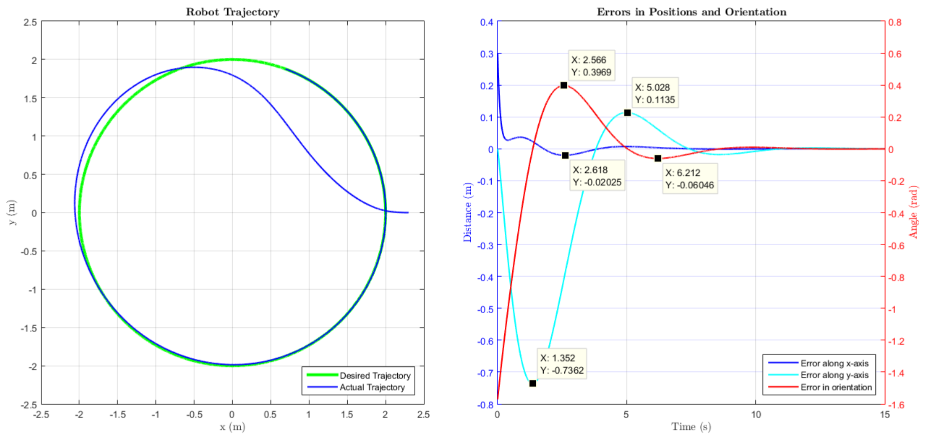

Figure I.4.6: Trajectory and Error response in $\mathcal{R}_m$ frame of robot type (2,0) using Lyapunov control with $K_x = 10$ and $K_y = K_\theta = 1$

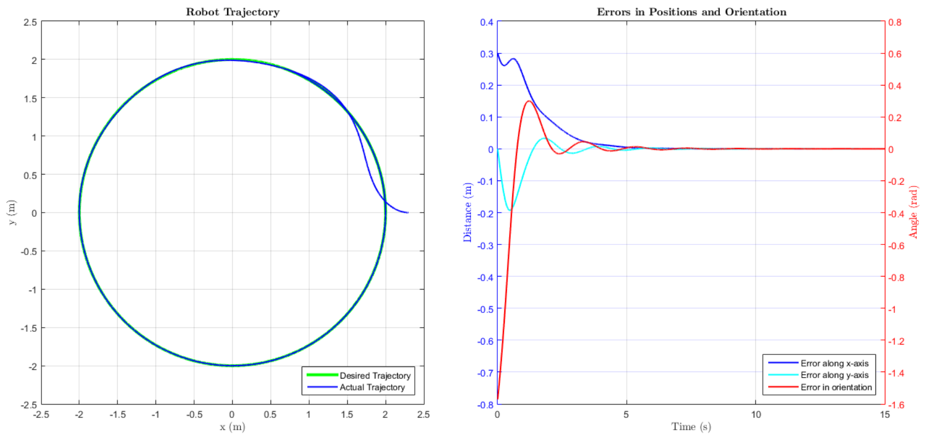

Figure I.4.7: Trajectory and Error response in $\mathcal{R}_m$ frame of robot type (2,0) using Lyapunov control with $K_x = 1$, $K_y = 10$ and $K_\theta = 1$

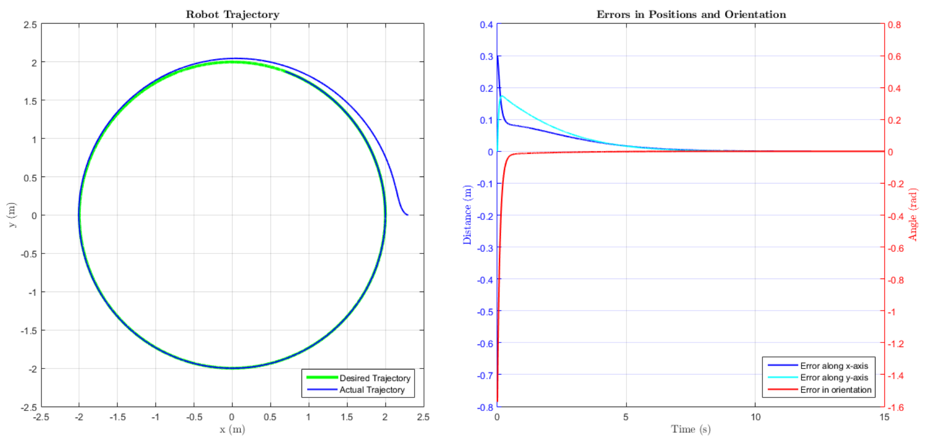

Figure I.4.8: Trajectory and Error response in $\mathcal{R}_m$ frame of robot type (2,0) using Lyapunov control with $K_x = 1$, $K_y = 1$ and $K_\theta = 10$

- With $K_x = 10$, the magnitudes of the peaks of x_e are significantly reduced, the response time of x_e becomes faster. However, y_e and θ_e remain almost the same. The settling time of the system also increase slightly.
- With $K_y = 10$, the magnitudes of the peaks of x_y are significantly reduced. The response time of the whole system becomes faster, the settling time is also significantly reduced. However, the number of oscillations are increased.
- With $K_\theta = 10$, the response time of θ_e becomes very short, response times of x_e and y_e are also decreased but still not enough to achieve the desirable performance. It is also noted that the oscillation in the system is removed by increaseing K_θ.

Therefore, it can be concluded that each gain impacts the response of its corresponding error the most. However, K_y plays an important role in reducing the settling time while K_θ helps removing the oscillation of the system. In the end, the gains are tuned such that $K_x = 10$, $K_y = 15$ and $K_\theta = 10$ in order to achieve the desirable performance, as shown in Figure I.4.9 below, where the response time is about 1s and settling time is about 1.5s:

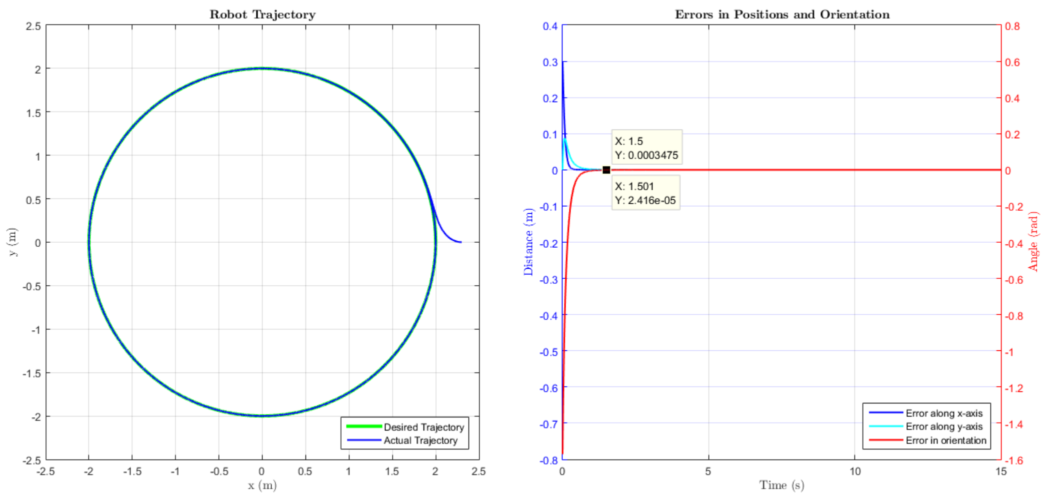

Figure I.4.9: Trajectory and Error response of robot type (2,0) using Lyapunov control with $K_x = 10$, $K_y = 15$ and $K_\theta = 10$

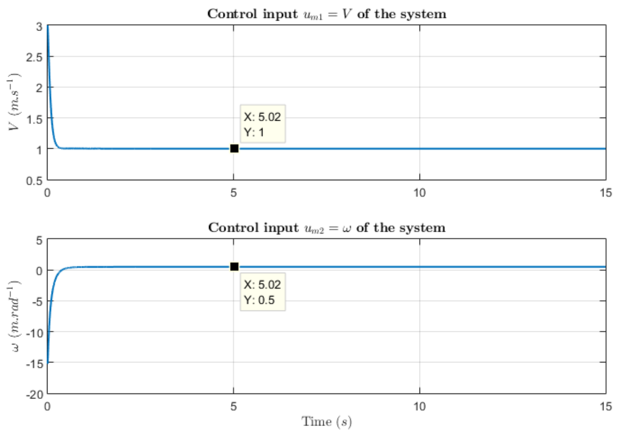

Figure I.4.10: Control input of the system

Case 1: There is $\pm 10\%$ change in the estimated value of L

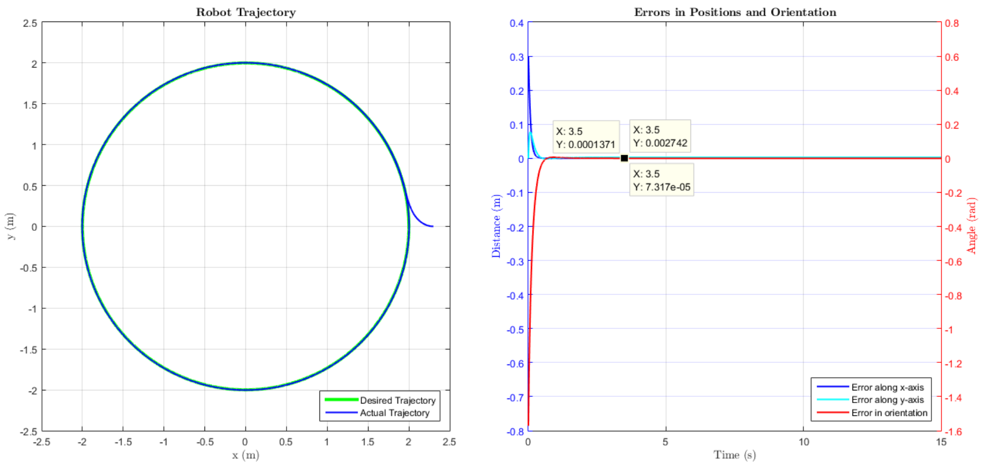

Figure I.4.11: Trajectory and Error response in $\mathcal{R}_m$ frame of robot type (2,0) using Lyapunov Control law when $L_{\text{est}} = 90\% L$

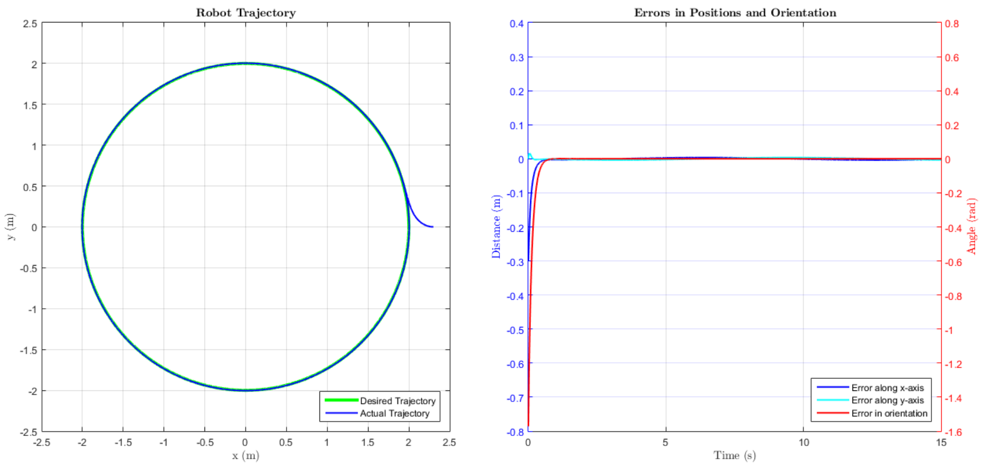

Figure I.4.12: Trajectory and Error response in $\mathcal{R}_0$ frame of robot type (2,0) using Lyapunov Control law when $L_{\text{est}} = 90\% L$

Figure I.4.11 and Figure I.4.12 show the tracking error in the robot frame $\mathcal{R}_m$ and global frame $\mathcal{R}_0$, respectively. As shown, the errors in $\mathcal{R}_m$ converges to constant values close to 0, which means the system stablize around a new equilibrium point where ${}^m\xi_e = [1.4\text{e}{-04} \;\; 2.7\text{e}{-03} \;\; 7.3\text{e}{-05}]^T$. Since there is a steady state error in θ, when converted to global frame $\mathcal{R}_0$, the error along x and y will oscillate with a small magnitude due to the rotation matrix.

Case 2: There is $\pm 10\%$ change in the estimated value of r

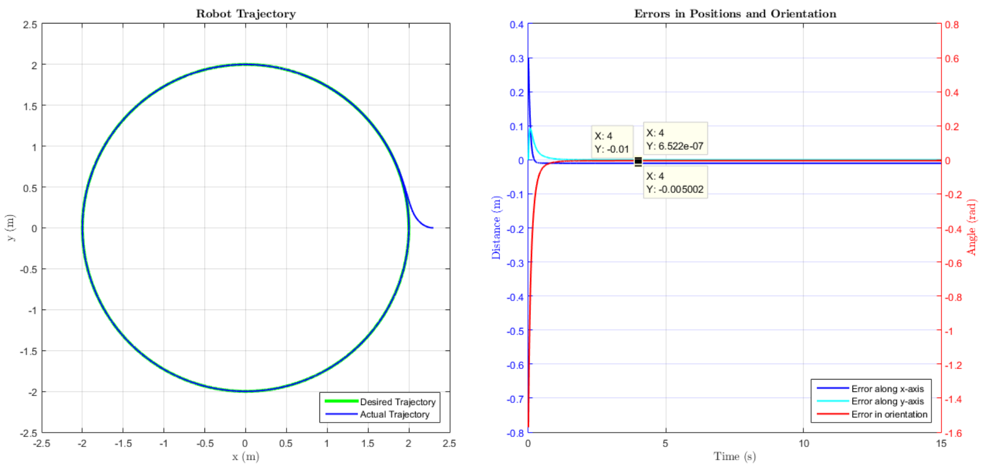

Figure I.4.13: Trajectory and Error response in $\mathcal{R}_m$ of robot type (2,0) using Lyapunov Control law when $r_{\text{est}} = 90\% r$

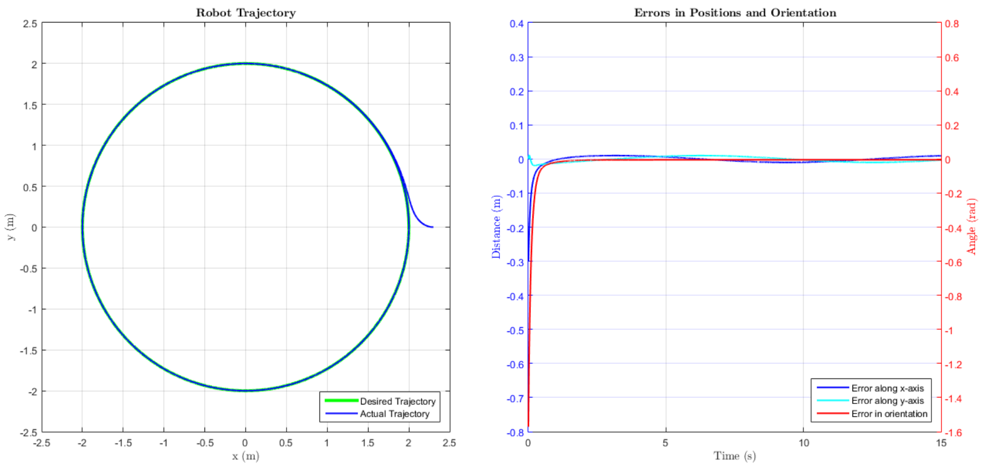

Figure I.4.14: Trajectory and Error response in $\mathcal{R}_0$ of robot type (2,0) using Lyapunov Control law when $r_{\text{est}} = 90\% r$

Similar to the previous case, a $\pm 10\%$ change in the estimated value of fixed wheel radius r causes steady state error in robot frame $\mathcal{R}_m$ and oscillating response in global frame $\mathcal{R}_0$. The new equilibrium point in $\mathcal{R}_m$ is ${}^0\xi_e = [-1.0\text{e}{-02} \;\; 6.5\text{e}{-07} \;\; -5\text{e}{-03}]^T$. Due to the fact that the error in θ is more severe, the magnitude of error along x and y axes in $\mathcal{R}_0$ are greater. It can be concluded that the change in r_{est} is severe and should be avoid by better identification.

Part II: Mobile Robot Type (1,1)

II.1 Complete Kinematic Study

II.1.1 Parameterization

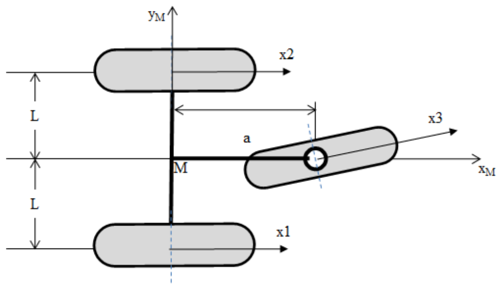

Figure II.1.1: Schematic of mobile robot type (1,1)

Table II.1: Parameters of type (1,1) mobile robot

W	L	α	d	β	γ	φ	ψ
1_f	L	$-\pi/2$	0	0	$\pi/2$	φ_1	0
2_f	L	$\pi/2$	0	0	$-\pi/2$	φ_2	0
3_s	a	0	0	β_3	0	φ_3	β_3

The configuration vector is: $q = \begin{bmatrix} x & y & \theta & \beta_3 & \varphi_1 & \varphi_2 & \varphi_3 \end{bmatrix}^T$

II.1.2 Posture Kinematic Model

We have:

$$\mathbf{J_1} = \begin{bmatrix} 1 & 0 & L \\ 1 & 0 & -L \\ \cos\beta_3 & \sin\beta_3 & a\sin\beta_3 \end{bmatrix} \qquad \mathbf{J_2} = -r\mathbf{I}_3$$

$$\mathbf{C_1} = \begin{bmatrix} 0 & 1 & 0 \\ 0 & 1 & 0 \\ -\sin\beta_3 & \cos\beta_3 & a\cos\beta_3 \end{bmatrix} \qquad \mathbf{C_2} = \text{null}$$

Since:

$$\mathbf{C}_1^*(\beta_s) = \begin{bmatrix} \mathbf{C}_{1f} \\ \mathbf{C}_{1s} \end{bmatrix} = \begin{bmatrix} 0 & 1 & 0 \\ 0 & 1 & 0 \\ -\sin\beta_3 & \cos\beta_3 & a\cos\beta_3 \end{bmatrix} \qquad \Rightarrow \quad \text{rank}(\mathbf{C}_1^*) = 2$$

Degree of mobility of the robot: $\delta_m = 3 - \text{rank}(\mathbf{C}_1^*) = 1$
Degree of steerability of the robot: $\delta_s = 1$
We have: $\dot{\xi} = [\dot{x} \;\; \dot{y} \;\; \dot{\theta}]^T \in \text{Ker}(\mathbf{C}_1^*)$ iff: $\mathbf{C}_1^*.{}^m\mathbf{\Omega}_0(\theta)\dot{\xi} = \mathbf{C}_1^*.{}^m\dot{\xi} = 0$
Hence:

$$\begin{cases} {}^m\dot{y} = 0 \\ {}^m\dot{\theta} = \frac{\sin\beta_3}{a\cos\beta_3}{}^m\dot{x} \end{cases}$$

Choose ${}^{m}\dot{x} = a\cos\beta_3$ so that the kernel of $\mathbf{C}_1^*$ is:

$$\text{Ker}(\mathbf{C}_1^*) = \mathbf{\Sigma} = \begin{bmatrix} a\cos\beta_3 \\ 0 \\ \sin\beta_3 \end{bmatrix}$$

The use of $\tan\beta_3$ in $\mathbf{\Sigma}$ is not recommended because it might cause numerical singularity when $\beta_3 = \frac{\pi}{2} + k\pi$, which is not an actual physical singularity, and produce uneccesary and avoidable complication and problem for the system.
Based on the value of $\delta_m = 1$ and $\delta_s = 1$, it can be concluded that $\dim(u_m) = \delta_m = 1$ and $\dim(u_s) = \delta_s = 1$.
Thus the control input of the system is: $\mathbf{u} = [u_m \; u_s]^T$
The state-space representation of the system is: $\dot{\mathbf{z}} = \mathbf{B}(\mathbf{z})\mathbf{u}$
Where: $\mathbf{B}(z) = \begin{bmatrix} {}^0\mathbf{\Omega}_m\mathbf{\Sigma} & 0 \\ 0 & I \end{bmatrix}$ and $z = [\xi \; \beta_s]^T = [x \; y \; \theta \; \beta_3]^T$

$$\mathbf{B}(\mathbf{z}) = \begin{bmatrix} {}^0\mathbf{\Omega}_m\mathbf{\Sigma} & 0 \\ 0 & I \end{bmatrix} = \begin{bmatrix} \begin{bmatrix} \cos\theta & -\sin\theta & 0 \\ \sin\theta & \cos\theta & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} a\cos\beta_3 \\ 0 \\ sin\beta_e \end{bmatrix} & \mathbf{0}_{3\times1} \\ 0 & 1 \end{bmatrix} = \begin{bmatrix} a\cos\beta_3\cos\theta & 0 \\ a\cos\beta_3\sin\theta & 0 \\ \sin\beta_3 & 0 \\ 0 & 1 \end{bmatrix}$$

$$\Rightarrow \quad \dot{\mathbf{z}} = \begin{bmatrix} \dot{x} \\ \dot{y} \\ \dot{\theta} \\ \dot{\beta}_3 \end{bmatrix} = \mathbf{B}(\mathbf{z})\mathbf{u} = \begin{bmatrix} a\cos\beta_3\cos\theta & 0 \\ a\cos\beta_3\sin\theta & 0 \\ \sin\beta_3 & 0 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} u_m \\ u_s \end{bmatrix} = \begin{bmatrix} a\cos\beta_3\cos\theta u_m \\ a\cos\beta_3\sin\theta u_m \\ \sin\beta_3 u_m \\ u_s \end{bmatrix} \tag{II.1}$$

II.1.3 Configuration Kinematic Model

We have:

$$\mathbf{D}(\beta_c) = \text{null}$$

$$\mathbf{E}(\beta_s, \beta_c) = -\mathbf{J}_2^{-}1\mathbf{J}_1(\beta_s, \beta_c) = \begin{bmatrix} 1/r & 0 & L/r \\ 1/r & 0 & -L/r \\ \cos\beta_3/r & \sin\beta_3/r & a\sin\beta_3/r \end{bmatrix}$$

$$\Rightarrow \quad \mathbf{S}(q) = \begin{bmatrix} a\cos\beta_3\cos\theta & 0 \\ a\cos\beta_3\sin\theta & 0 \\ \sin\beta_3 & 0 \\ 0 & 1 \\ (L\sin\beta_3 + a\cos\beta_3)/r & 0 \\ -(L\sin\beta_3 - a\cos\beta_3)/r & 0 \\ a/r & 0 \end{bmatrix}$$

II.1.4 Wheel Motorization

From the configuration kinematic model, we have:

$$\Rightarrow \quad \dot{q}_m = \begin{bmatrix} \dot{\varphi}_1 \\ \dot{\varphi}_2 \\ \dot{\varphi}_3 \end{bmatrix} = \mathbf{F}u_m = \begin{bmatrix} (L\sin\beta_3 + a\cos\beta_3)/r \\ -(L\sin\beta_3 - a\cos\beta_3)/r \\ a/r \end{bmatrix} um$$

From the equation above, it can be seen that $\dot{\varphi}_3$ should be motorized due to its simplicity in term of control, which means the steerable wheel would be fully motorized, as in both of its orientation and spining. From this onward, only φ_3 and β_3 are considered, so they will be denoted as φ and β hereafter for the sake of simplicity. In conclusion, we have the formula:

$$\text{DKM Equation:} \quad \mathbf{u} = \begin{bmatrix} u_m \\ u_s \end{bmatrix} = \begin{bmatrix} r/a & 0 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} \dot{\varphi} \\ \dot{\beta} \end{bmatrix} \tag{II.2}$$

$$\text{IKM Equation:} \quad \dot{\mathbf{q}} = \begin{bmatrix} \dot{\varphi} \\ \dot{\beta} \end{bmatrix} = \begin{bmatrix} a/r & 0 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} u_m \\ u_s \end{bmatrix} \tag{II.3}$$

II.1.5 Modeling - Validation and Simulation

In this section, a Simulink model for the Direct Kinematic and Localization of the robot is developed. It would be used to validate Equation (II.1), (II.2) discussed above.

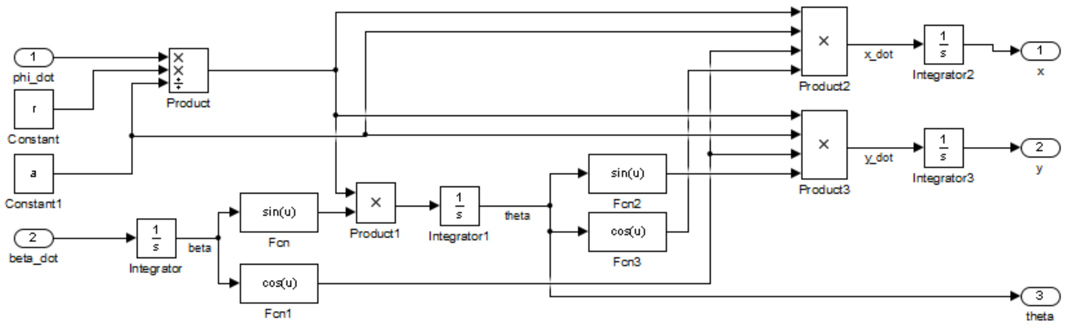

Figure II.1.2: Direct Kinematic Model and Localization for robot type (1,1)

The input of the model is $\dot{\varphi} = \text{const}$, $\dot{\beta} = 0$ and $\beta(0) = \text{const}$. The expected outputs are the trajectory of the robot, which is a straight line when $\beta(0) = 0$, or circles with different radii when $\beta(0) \neq 0$. They are validated in Figure below:

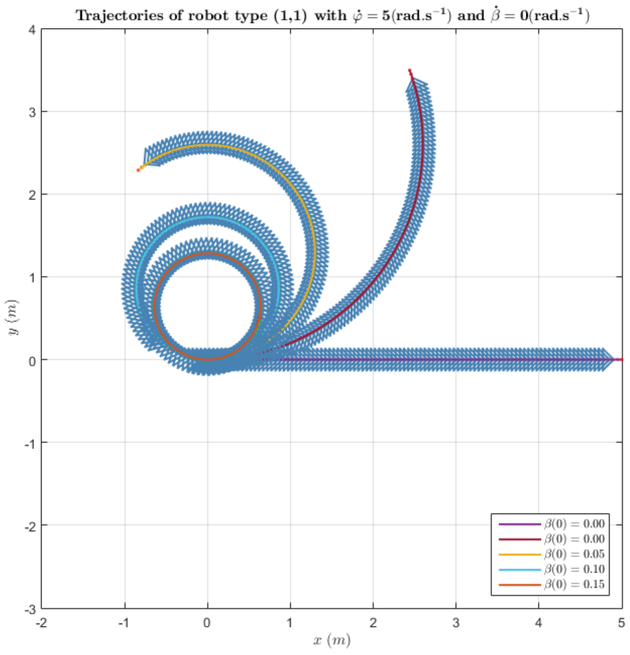

Figure II.1.3: Different trajectories of robot type (1,1) when $\dot{\beta} = 0$ and $\beta(0) = \text{const}$

II.2 Static Decoupling Control

II.2.1 Theoretical Study

Similar to type (2,0), mobile robot type (1,1) is not fully linearizable by static decoupling control law either. Furthermore, in order to avoid singularity, the tracking point P is not be located on the chassis of the robot, but on plane (x_3,z_3) of the steerable wheel, with $d \neq 0$ is the distance between P and O_3 along x_3 axis.
The expression of $\mathbf{h}^d$ and $\dot{\mathbf{h}}^d$ will remain the same as in Equation (I.4) and (I.5) since the robot also move on a circle with $R = 2m$ and $\omega^d = 0.5rad.s^{-1}$.
However, the expression of $\mathbf{h}$ becomes:

$$\mathbf{h} = \begin{bmatrix} x + a\cos\theta + d\cos(\theta+\beta) \\ y + a\sin\theta + \sin(\theta+\beta) \end{bmatrix} \tag{II.4}$$

$$\Rightarrow \quad \dot{\mathbf{h}} = \begin{bmatrix} \dot{x} - a\sin(\theta)\dot{\theta} - d\sin(\theta+\beta)(\dot{\theta}+\dot{\beta}) \\ \dot{y} + a\cos(\theta)\dot{\theta} + d\cos(\theta+\beta)(\dot{\theta}+\dot{\beta}) \end{bmatrix}$$

$$\text{From Eqn. (II.1):} \quad \begin{bmatrix} \dot{x} \\ \dot{y} \\ \dot{\theta} \\ \dot{\beta}_3 \end{bmatrix} = \begin{bmatrix} a\cos\beta_3\cos\theta u_m \\ a\cos\beta_3\sin\theta u_m \\ \sin\beta_3 u_m \\ u_s \end{bmatrix}$$

$$\Rightarrow \quad \dot{\mathbf{h}} = \begin{bmatrix} a\mathrm{C}\theta\mathrm{C}\beta u_m - a\mathrm{S}\theta\mathrm{S}\beta u_m - d\mathrm{S}(\theta+\beta)(\mathrm{S}\beta u_m + u_s) \\ a\mathrm{S}\theta\mathrm{C}\beta u_m + a\mathrm{C}\theta\mathrm{S}\beta u_m + d\mathrm{C}(\theta+\beta)(\mathrm{S}\beta u_m + u_s) \end{bmatrix}$$

$$= \begin{bmatrix} a\mathrm{C}(\theta+\beta) - d\mathrm{S}\beta\mathrm{S}(\theta+\beta) & -d\mathrm{S}(\theta+\beta) \\ a\mathrm{S}(\theta+\beta) + d\mathrm{S}\beta\mathrm{C}(\theta+\beta) & d\mathrm{C}(\theta+\beta) \end{bmatrix} \begin{bmatrix} u_m \\ u_s \end{bmatrix} \tag{II.5}$$

$$= \mathbf{K}(\theta,\ \beta)\mathbf{u}$$

By defining an auxiliary control $\mathbf{W}$ such that:

$$\mathbf{W} = \dot{\mathbf{h}} = \mathbf{K}(\theta,\ \beta)\mathbf{u}$$

$$\Rightarrow \quad \mathbf{u} = \mathbf{K}^{-1}(\theta,\ \beta)\mathbf{W} \tag{II.6}$$

We have:

$$\mathbf{W} = \dot{\mathbf{h}} = \dot{\mathbf{h}}^d + K_p(\mathbf{h}^d - \mathbf{h}) \tag{II.7}$$

$$\Rightarrow \quad 0 = (\dot{\mathbf{h}}^d - \dot{\mathbf{h}}) + K_p(\mathbf{h}^d - \mathbf{h})$$

Let $\mathbf{e}(t) = \mathbf{h}^d - \mathbf{h}$ be the error between the desired and actual positions, equation (I.9) becomes:

$$\frac{d\mathbf{e}}{dt} + K_p\mathbf{e}(t) = 0$$

$$\Rightarrow \quad \frac{d\mathbf{e}}{\mathbf{e}} = -K_p dt$$

$$\Rightarrow \quad \int_0^t \frac{d\mathbf{e}}{\mathbf{e}} = \int_0^t -K_p dt$$

$$\Rightarrow \quad \ln\left(\frac{\mathbf{e}(t)}{\mathbf{e}(0)}\right) = -K_p t$$

$$\Rightarrow \quad \mathbf{e}(t) = \mathbf{e}(0)\mathrm{e}^{-K_p t}$$

$$\Rightarrow \quad \lim_{t\to\infty} \mathbf{e}(t) = 0$$

Therefore the system is asymptotically exponentially stable. Which means the tracking errors $\mathbf{e}(t) = [e_x(t)\ e_y(t)]^T$ will converge to 0 over time.

As for the orientation of the robot, it can be proven that even though it is uncontrollable, the error will also converge to a constant value.
Firstly, the following shows that when robot type (1,1) tracks point P located on a circle, β will converge to a constant value and $\dot{\beta}$ converges to 0.

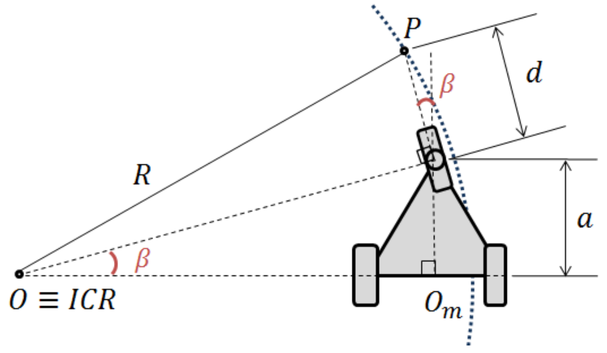

Figure II.2.1: Illustration of robot type (1,1) tracking point P located on a circle of radius R

Figure II.2.1 above illustrate the robot posture at a point of time when the system has stabilized and is tracking point P located on a circle of radius R. In such case, it can be seen that the instantaneous center of rotation (ICR) of the robot coincides with the center of the circle. Based on the figure:

$$\begin{aligned} \tan\beta_\infty &= \frac{a}{\sqrt{R^2-d^2}} \\ \Rightarrow \beta_\infty &= \arcsin\left(\frac{a}{\sqrt{R^2-d^2}}\right) \\ \Rightarrow \dot{\beta}_\infty &= 0 \end{aligned} \tag{II.8}$$

When $t \to \infty$, we have:

$$\begin{aligned} \lim_{t\to\infty}\mathbf{e}(t) &= \lim_{t\to\infty}(\mathbf{h}^d-\mathbf{h}) = 0 \\ \Rightarrow \lim_{t\to\infty}\mathbf{W} &= \dot{\mathbf{h}}^d + K_p(\mathbf{h}^d-\mathbf{h}) = \dot{\mathbf{h}}^d \end{aligned}$$

From Eqn. (II.7) $\Rightarrow \lim_{t\to\infty}\mathbf{W} = \dot{\mathbf{h}}^d = \begin{bmatrix} -R\omega^d\sin(\omega^d t) \\ R\omega^d\cos(\omega^d t) \end{bmatrix}$

From Eqn. (II.6) $\Rightarrow \mathbf{u}_\infty = \mathbf{K}^{-1}(\theta,\beta_\infty)\mathbf{W}_\infty$

$$= \begin{bmatrix} \frac{C(\theta+\beta_\infty)}{a} & \frac{S(\theta+\beta_\infty)}{a} \\ -\frac{S(\theta+\beta_\infty)}{d} - \frac{S\beta_\infty S(\theta+\beta_\infty)}{a} & \frac{C(\theta+\beta_\infty)}{d} - \frac{S\beta_\infty S(\theta+\beta_\infty)}{a} \end{bmatrix} \begin{bmatrix} -R\omega^d\sin(\omega^d t) \\ R\omega^d\cos(\omega^d t) \end{bmatrix}$$

$$\Rightarrow \begin{bmatrix} u_{m\infty} \\ u_{s\infty} \end{bmatrix} = \begin{bmatrix} \frac{R\omega^d S(\omega^d t-\theta-\beta_\infty)}{a} \\ \frac{R\omega^d C(\omega^d t-\theta-\beta_\infty)}{d} + \frac{R\omega^d S\beta_\infty S(\omega^d t-\theta-\beta_\infty)}{a} \end{bmatrix}$$

When the system has stabilized, we have $u_{s\infty} = \dot{\beta}_\infty = 0$, therefore:

$$\begin{aligned} \frac{R\omega^d C(\omega^d t-\theta-\beta_\infty)}{d} + \frac{R\omega^d S\beta_\infty S(\omega^d t-\theta-\beta_\infty)}{a} &= 0 \\ \Rightarrow \quad \tan(\omega^d t-\theta-\beta_\infty) &= -\frac{a}{d\sin\beta_\infty} \\ \Rightarrow \quad \frac{\pi}{2}+\omega^d t-\theta &= \frac{\pi}{2}+\beta_\infty+\operatorname{atan}\left(-\frac{a}{d\sin\beta_\infty}\right) \\ \Rightarrow \quad e_\theta &= \text{const} \end{aligned}$$

Thus, it is proven that when the system become stable, the error in orientation e_θ is also stable and converges to a constant value. Substituting the constants values $R = 2m$, $a = 0.15m$, $d = 0.05m$, we get the results: $\beta_\infty = 0.0751rad$ and $e_\theta = 0.1rad$.

II.2.2 Modeling

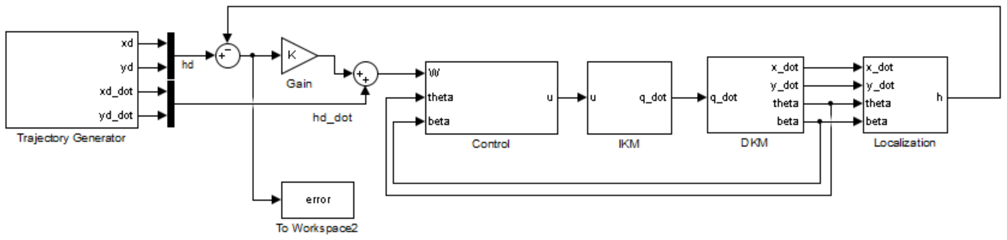

Figure II.2.2: Simulink model for control of robot type (1,1) using static decoupling law

Figure II.2.2 above shows the global Simulink model to control robot type (1,1) using static decoupling control law:

- `Trajectory Generator` block remains the same as in Type (2,0) case.
- `Control` block take the value of $\mathbf{W}$, β and θ to find the control input $\mathbf{u}$ using Equation (II.6).
- `IKM` block is the inverse kinematic model of the robot, which will compute $\dot{q}$ using Equation (II.3) and the estimated values of a and r.
- `DKM` block is the direct kinematic model of the robot, which uses $\dot{q}$, and actual values of a and r to compute $\mathbf{u}$ and then ${}^0\dot{\xi}$.
- `Localization` block calculate the position of the tracking point $\mathbf{h}$ using Equation (II.4).

II.2.3 Validation and Simulation

Remark: The simulations are carried out using Dormand-Prince(ode5) fixed-step solver with 0.001s step time in 15s. The values of the variables are:

- $R = 2.00m$: Radius of the circular trajectory
- $a = 0.15m$: Distance between O_m (origin of robot frame) to center of steerable wheel.
- $r = 0.10m$: Radii of the wheels
- $d = 0.05m$: Distance between O_3 (center of the steerable wheel) to the tracking point along x_m axis.

Similar to Type (2,0) case, we also tune the value of proportional gain K_p until the desirable response time and settling time are achieved. In this case we also use $K_p = 5$, which produces the trajectory and response shown below:

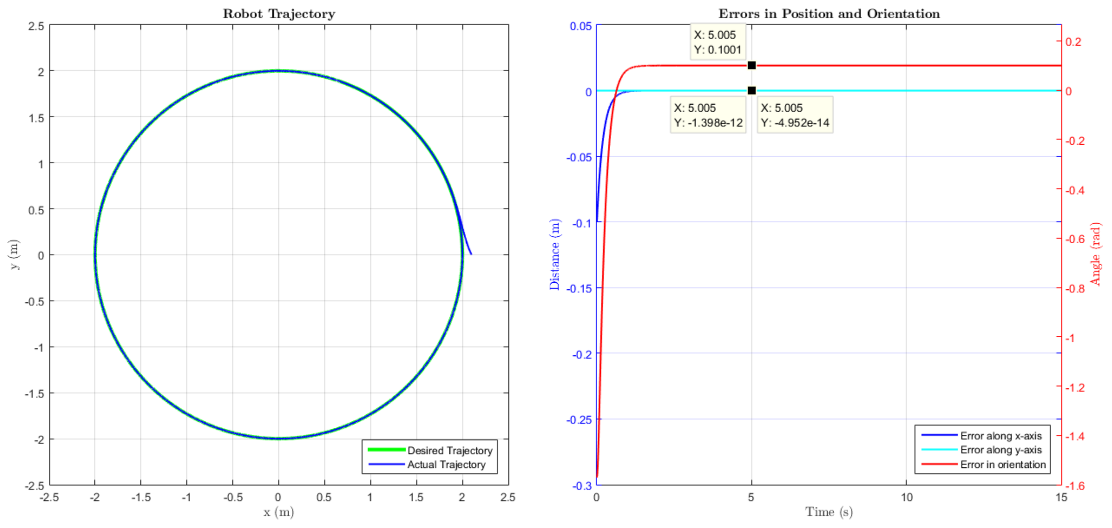

Figure II.2.3: Trajectory and Error response of robot type (1,1) using static decoupling control law with $K_p = 5$

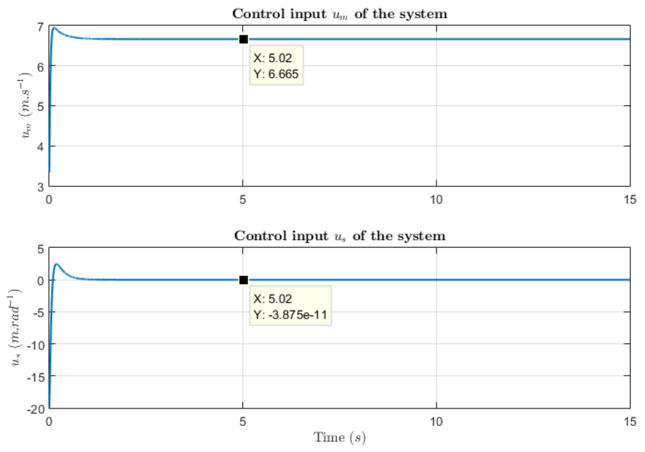

Figure II.2.4: Control input of the system

From the figure, the error in orientation can be seen to converge to a constant value of $0.1rad$ as proven in the theoretical study section. If inspected closely, the error in x and y axes oscillate around 0 with a very small magnitude in the order of 10^{-12}. The phenomenon can be explained by inspecting the value of $\dot{\beta}$ overtime. In theory $\dot{\beta}$ will converge to 0 eventually, but in practice, it will never become exactly 0. Therefore, as long as there is a infinitely small value of $\dot{\beta}$, there will be corresponding error in x and y. However, since the error is insignificant, they can be be ignored.

There are 10% error in the estimated value of a and r

Similar to (2,0) type robot, while the error in orientation coverges to a constant value, errors in x and y axis exhibit characteristics of a sinusoidal. The magnitude of the errors might be consider significant. Therefore, in order to reduce the amount of errors, either the proportional gain K_p has to be increased or the system requires a better the identification of a and r.

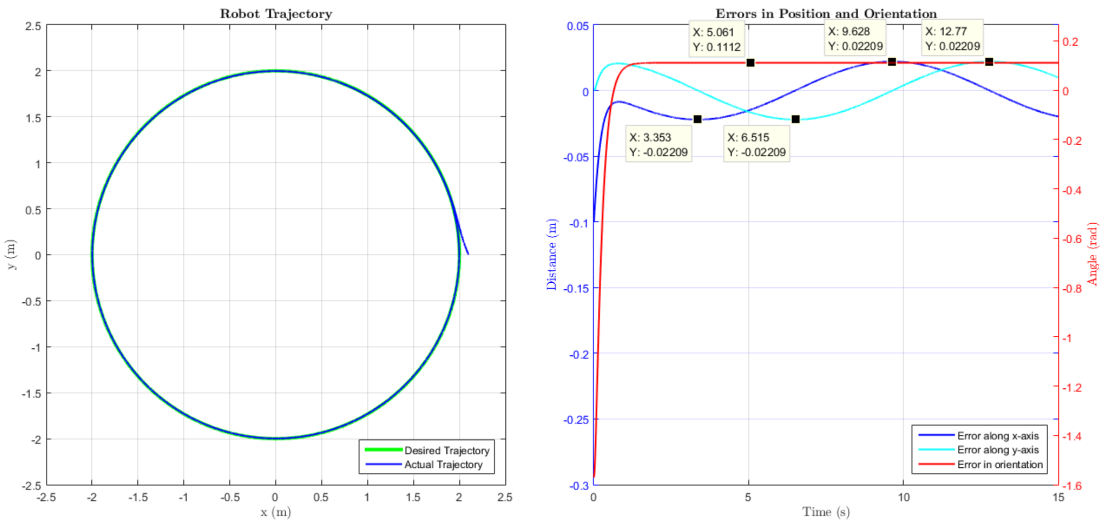

Figure II.2.5: Trajectory and Error response of robot type (1,1) using static decoupling law with $a_{\text{est}} = 90\% a$

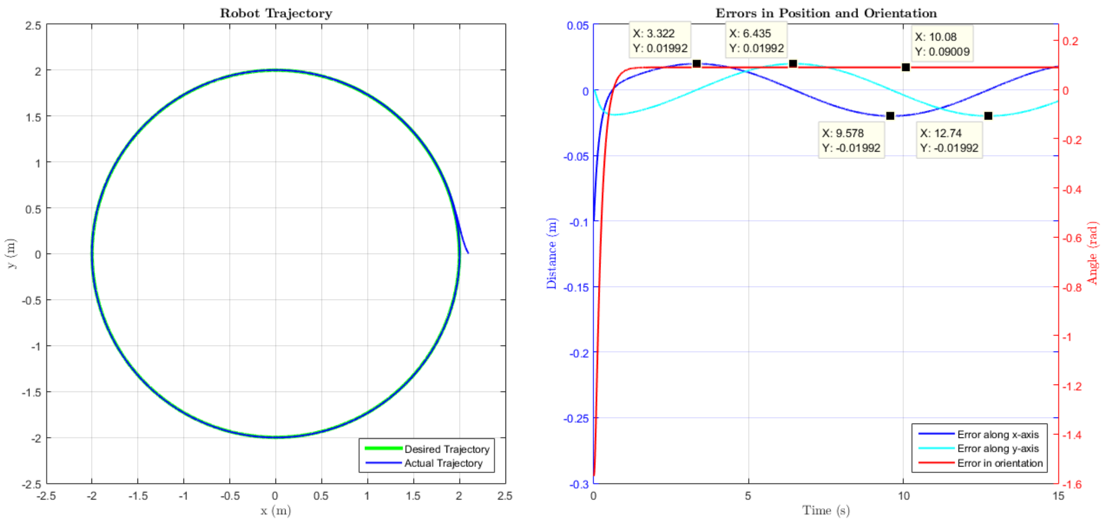

Figure II.2.6: Trajectory and Error response of robot type (1,1) using static decoupling law with $r_{\text{est}} = 90\% r$

II.3 Dynamic Decoupling Control

II.3.1 Theoretical Study

Since the trajectory remains the same, the expressions of $\mathbf{h}^d$, $\dot{\mathbf{h}}^d$ and $\ddot{\mathbf{h}}^d$ are unchanged, which shown in Equation (I.4), (I.5) and (I.10), respectively.
In order to use the dynamic decoupling control law, we consider the previous vector $\mathbf{u}$ as new state variables, and the new control input of the system is $\dot{\mathbf{u}} = [\dot{u}_m \ \dot{u}_s]^T$. From Equation (II.5) we have:

$$\begin{aligned} & \dot{\mathbf{h}} = \mathbf{K}(\theta,\ \beta)\mathbf{u} \\ \Rightarrow \quad & \ddot{\mathbf{h}} = \dot{\mathbf{K}}(\theta,\ \beta)\mathbf{u} + \mathbf{K}(\theta,\ \beta)\dot{\mathbf{u}} \\ & \quad = \mathbf{F}(\theta,\ \beta) + \mathbf{K}(\theta,\ \beta)\dot{\mathbf{u}} \\ \text{Where } & \mathbf{F}(\theta,\ \beta) = \dot{\mathbf{K}}(\theta,\ \beta) \end{aligned} \tag{II.9}$$

$$= \begin{bmatrix} -u_m^2 \mathrm{S}\beta[a\mathrm{S}(\theta+\beta)+d\mathrm{S}\beta\mathrm{C}(\theta+\beta)] - u_m u_s[a\mathrm{S}(\theta+\beta)+2d\mathrm{S}\beta\mathrm{C}(\theta+\beta)+d\mathrm{C}\beta\mathrm{S}(\theta+\beta)] - u_s^2 d\mathrm{C}(\theta+\beta) \\ u_m^2 \mathrm{S}\beta[a\mathrm{C}(\theta+\beta)-d\mathrm{S}\beta\mathrm{S}(\theta+\beta)] + u_m u_s[a\mathrm{C}(\theta+\beta)-2d\mathrm{S}\beta\mathrm{S}(\theta+\beta)+d\mathrm{C}\beta\mathrm{C}(\theta+\beta)] - u_s^2 d\mathrm{S}(\theta+\beta) \end{bmatrix}$$

Introducing an auxiliary control $\mathbf{W}$ such that:

$$\begin{aligned} \mathbf{W} &= \ddot{\mathbf{h}} = \mathbf{F}(\theta,\ \beta) + \mathbf{K}(\theta,\ \beta)\dot{\mathbf{u}} \\ \Rightarrow \quad \dot{\mathbf{u}} &= \mathbf{K}^{-1}(\mathbf{W}-\mathbf{F}) \end{aligned} \tag{II.10}$$

Define $\mathbf{e}(t) = \mathbf{h}^d - \mathbf{h}$, which is the tracking error. By using the control law:

$$\mathbf{W} = \ddot{\mathbf{h}}^d + K_v(\dot{\mathbf{h}}^d - \dot{\mathbf{h}}) + K_p(\mathbf{h}^d - \mathbf{h}) \tag{II.11}$$

$$\begin{aligned} \Rightarrow \quad \ddot{\mathbf{h}} &= \ddot{\mathbf{h}}^d + K_v(\dot{\mathbf{h}}^d - \dot{\mathbf{h}}) + K_p(\mathbf{h}^d - \mathbf{h}) \\ \Rightarrow \quad 0 &= (\ddot{\mathbf{h}}^d - \ddot{\mathbf{h}}) + K_v(\dot{\mathbf{h}}^d - \dot{\mathbf{h}}) + K_p(\mathbf{h}^d - \mathbf{h}) \\ \Rightarrow \quad 0 &= \ddot{\mathbf{e}}(t) + K_v\dot{\mathbf{e}}(t) + K_p\mathbf{e}(t) \end{aligned} \tag{II.12}$$

The characteristic equation of the second order differential equation of $\mathbf{e}(t)$ is: $s^2 + K_v s + K_p = 0$.
Since K_v and K_p are postive, the roots of the characteristic equation always have negative real part:

- If $\Delta = \sqrt{K_v^2 - 4K_p} \geq 0$, the roots are $s_{1,2} = \frac{-K_v \pm \Delta}{2} < 0$. Thus:

$$\begin{aligned} \mathbf{e}(t) &= c_1 \mathrm{e}^{s_1 t} + c_2 \mathrm{e}^{s_2 t} \\ \Rightarrow \quad \lim_{t\to\infty} \mathbf{e}(t) &= 0 \end{aligned}$$

- If $\Delta = \sqrt{K_v^2 - 4K_p} < 0$, the roots are $s_{1,2} = \frac{-K_v \pm i\Delta}{2}$. Thus:

$$\begin{aligned} \mathbf{e}(t) &= \mathrm{e}^{\frac{-K_v}{2}t}(c_1\cos\Delta t + c_2 \sin\Delta t) \\ \Rightarrow \quad \lim_{t\to\infty} \mathbf{e}(t) &= 0 \end{aligned}$$

Therefore, it can be concluded that the system is globally asymptotically stable. Also, the error in orientation of the robot can be proven to convert to a constant value overtime:
When $t \to \infty$, the system has stabilized, $\mathbf{e}(t) = \dot{\mathbf{e}}(t) = 0$,

$$\text{From Eqn. (II.6)} \quad \mathbf{K}_\infty^{-1} = \begin{bmatrix} \frac{\mathrm{C}(\theta+\beta_\infty)}{a} & \frac{\mathrm{S}(\theta+\beta_\infty)}{a} \\ -\frac{\mathrm{S}(\theta+\beta_\infty)}{d} - \frac{\mathrm{S}\beta_\infty\mathrm{C}(\theta+\beta_\infty)}{a} & \frac{\mathrm{C}(\theta+\beta_\infty)}{d} - \frac{\mathrm{S}\beta_\infty\mathrm{S}(\theta+\beta_\infty)}{a} \end{bmatrix}$$

$$\text{From Eqn. (II.11)} \quad \mathbf{W}_\infty = \begin{bmatrix} -R(\omega^d)^2\mathrm{C}(\omega^d t) \\ -R(\omega^d)^2\mathrm{S}(\omega^d t) \end{bmatrix}$$

$$\text{From Eqn. (II.9)} \quad \mathbf{F}_\infty = \begin{bmatrix} -u_m^2 a\mathrm{S}\beta_\infty\mathrm{S}(\theta+\beta_\infty) - u_m^2 d\mathrm{S}^2\beta_\infty\mathrm{C}(\theta+\beta_\infty) \\ u_m^2 a\mathrm{S}\beta_\infty\mathrm{C}(\theta+\beta_\infty) - u_m^2 d\mathrm{S}^2\beta_\infty\mathrm{S}(\theta+\beta_\infty) \end{bmatrix}$$

$$\text{From Eqn. (II.10)} \quad \dot{\mathbf{u}}_\infty = \mathbf{K}^{-1}(\mathbf{W}_\infty - \mathbf{F}_\infty) = \begin{bmatrix} \dot{u}_{m\infty} \\ \dot{u}_{s\infty} \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \end{bmatrix}$$

$$\Rightarrow \quad \dot{u}_{m\infty} = -\frac{R(\omega^d)^2}{a}\mathrm{C}(\theta+\beta_\infty-\omega^d t) + \frac{d}{a}u_m^2\mathrm{S}^2\beta_\infty = 0$$

Let V be the linear velocity of the robot when it has become stable, so:

$$\begin{aligned} V &= R\omega^d \\ \dot{x} &= R\omega^d\mathrm{C}\theta = a\mathrm{C}\beta_\infty\mathrm{C}\theta u_m \\ \Rightarrow \quad u_m &= \frac{R\omega^d}{a\mathrm{C}\beta_\infty} \end{aligned}$$

Substituting the expression of u_m into $\dot{u}_{m\infty}$, we have:

$$
\begin{aligned}
-\frac{R(\omega^d)^2}{a}\mathrm{C}(\theta+\beta_\infty-\omega^d t)+\frac{d}{a}\left(\frac{R\omega^d}{a\mathrm{C}\beta_\infty}\right)^2\mathrm{S}^2\beta_\infty &= 0\\
-\frac{R(\omega^d)^2}{a}\mathrm{C}(\theta+\beta_\infty-\omega^d t)+\frac{R^2 d(\omega^d)^2}{a^3\mathrm{C}^2\beta_\infty} &= 0\\
\Rightarrow\quad \mathrm{C}(\theta+\beta_\infty-\omega^d t) &= \frac{Rd}{a^2}\tan^2\beta_\infty\\
\Rightarrow\quad \theta+\beta_\infty-\omega^d t &= \operatorname{acos}\left(\frac{Rd}{a^2}\tan^2\beta_\infty\right)\\
\Rightarrow\quad \frac{\pi}{2}+\omega^d t-\theta &= \frac{\pi}{2}+\beta_\infty-\operatorname{acos}\left(\frac{Rd}{a^2}\tan^2\beta_\infty\right)\\
e_\theta &= \frac{\pi}{2}+\beta_\infty-\operatorname{acos}\left(\frac{Rd}{a^2}\tan^2\beta_\infty\right)
\end{aligned}
$$

Since β_∞ is a constant value, the error in orientation of the robot e_θ is also a constant value. Substituting the values $R = 2m$, $d = 0.05m$, $a = 0.15m$ and $\beta_\infty = 0.0751rad$, we have $e_\theta = 0.1rad$

II.3.2 Modeling

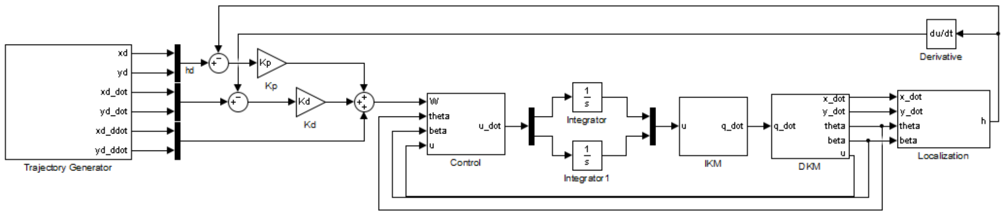

Figure II.3.1: Simulink model of the control of robot type (1,1) using dynamic decoupling law

Components of the model:

- `Trajectory Generator` block remains the same as the dynamic control for type (2,0), which output $\mathbf{h}^d$, $\dot{\mathbf{h}}^d$ and $\ddot{\mathbf{h}}^d$.
- `Control` block uses Equation (II.11) to compute $\dot{\mathbf{u}}$ from $\mathbf{W}$, $\mathbf{K}(\theta,\beta)$ and $\mathbf{F}(\theta,\beta)$.
- `IKM`, `DKM`, and `Localization` blocks are exactly the same as those of robot type (1,1) using static decoupling control law.

II.3.3 Validation and Simulation

Remark: The simulations are carried out using Dormand-Prince(ode5) fixed-step solver with step size 0.001s in 15s.
Similar to the dynamic decoupling control of robot type (2,0), the system in this case is also a second order system, thus the gain K_p and K_v can be tune the same way with same value of damping factor $\zeta = 1$ and $\omega_n = 5$, which means $K_p = {\omega_n}^2 = 25$ and $K_v = 2\zeta\omega_n = 10$. The trajectory and error response of the system are shown in Figure below:

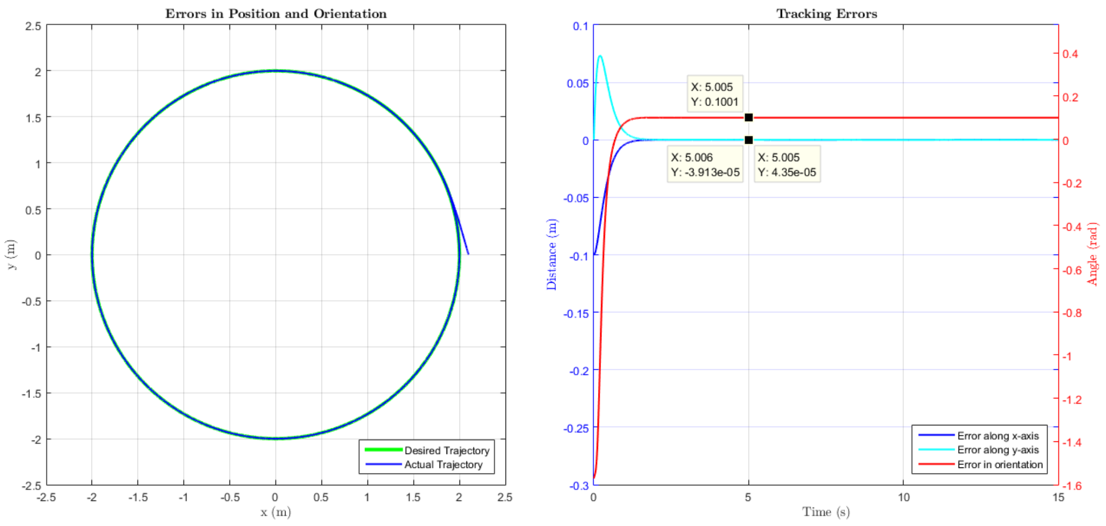

Figure II.3.2: Trajectory and errors response of robot type (1,1) using dynamic decoupling control law

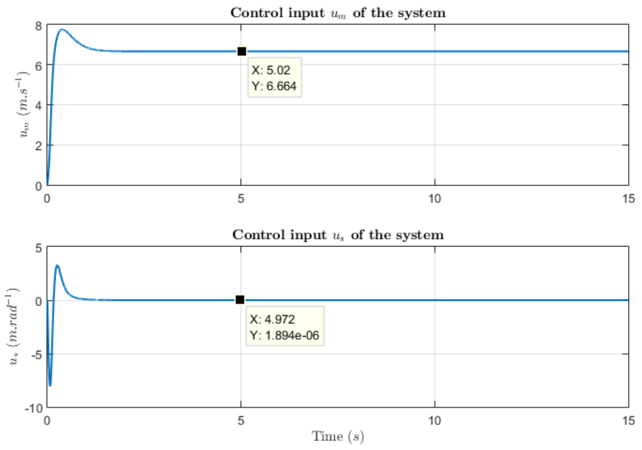

Figure II.3.3: Input **u** of the system

When inspect closely, the error in x and y are oscillate with the magnitude 5.85×10^{-5} due to the same reason it happened in static decoupling case, which because $\dot{\beta}$ never actually converge to exact 0. A very small value of $\dot{\beta}$ will cause the corresponding error in x and y. Since the errors are insignificant, they can be ignored. When the estimated valuse of r and a are changed by $\pm 10\%$, the magnitude of error oscillation is also in 10^{-5} order, which can be ignored as well.

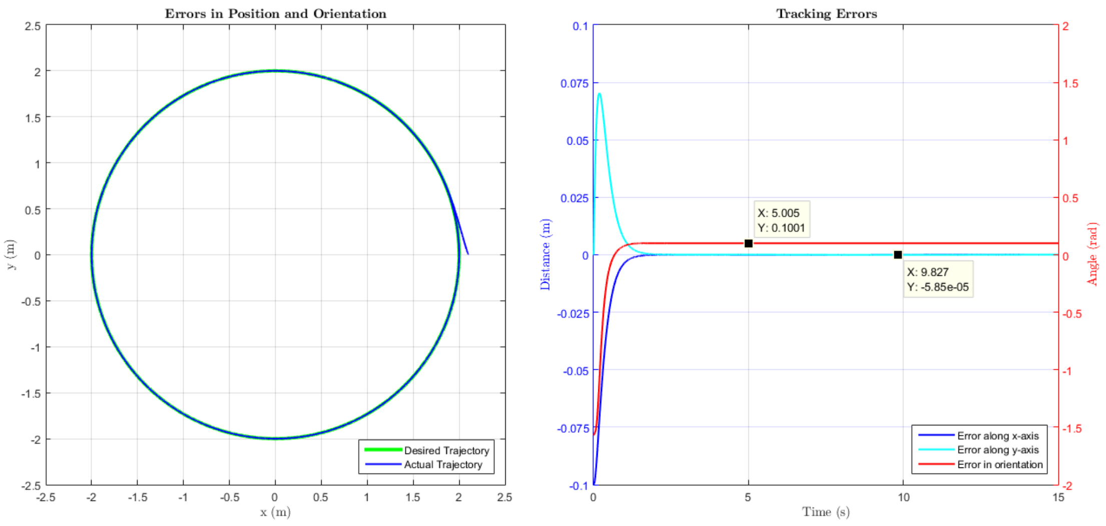

Figure II.3.4: Trajectory and errors response of robot type (1,1) using dynamic decoupling control law, with $r_{\text{est}} = r$

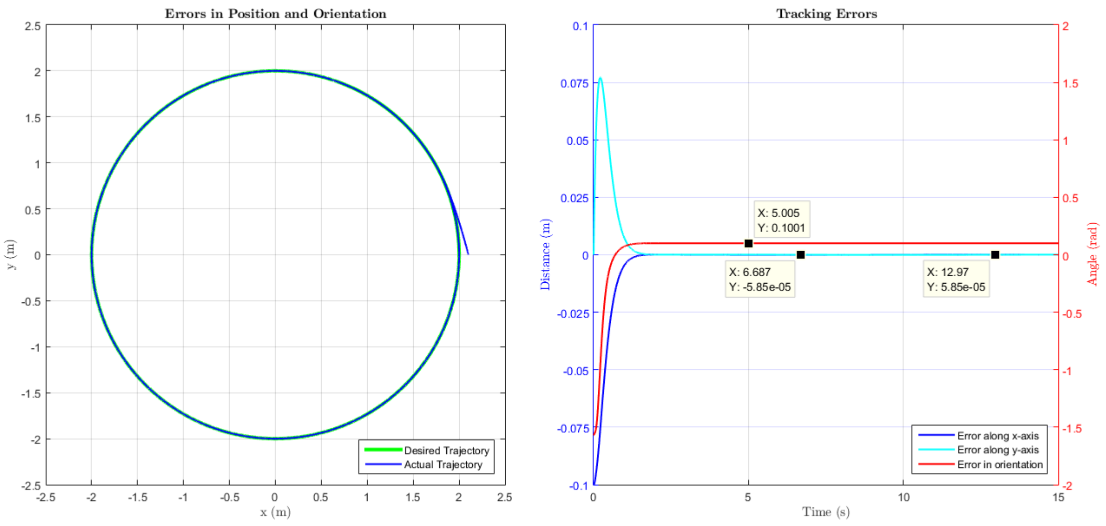

Figure II.3.5: Trajectory and errors response of robot type (1,1) using dynamic decoupling control law, with $a_{\text{est}} = a$

II.4 Lyapunov Control Law

II.4.1 Theoritical study

For tracking, point P' coincided with origin O_s of steerable wheel is used $(d = 0)$.

$$x_s = x + a\cos\theta$$
$$y_s = y + a\sin\theta$$
$$\psi = \theta + \beta$$

Here, (x,y) is the coordinate of the point m in the R_0 frame. On taking the time derivative of the equations above, the following equation are obtained:

$$\begin{aligned}\dot{x}_s &= \dot{x} - a\dot{\theta}\sin\theta \\ \dot{y}_s &= \dot{y} + a\dot{\theta}\cos\theta \\ \dot{\psi} &= \dot{\theta} + \dot{\beta}\end{aligned}$$

Using the the posture kinematic model of section 2, the above equations can be written as:

$$\begin{aligned}\dot{x}_s &= a\cos\theta\cos\beta u_m - a\sin\theta\sin\beta u_m = a\cos(\theta+\beta)u_m = a\cos\psi u_m \\ \dot{y}_s &= a\sin\theta\cos\beta u_m + a\cos\theta\sin\beta u_m = a\sin(\theta+\beta)u_m = a\sin\psi u_m \\ \dot{\psi} &= \sin\beta u_m + u_s\end{aligned}$$

$$\begin{bmatrix}{}^0\dot{x}_s \\ {}^0\dot{y}_s \\ \dot{\psi}\end{bmatrix} = \begin{bmatrix}a\cos\psi & 0 \\ a\sin\psi & 0 \\ \sin\beta & 1\end{bmatrix}\begin{bmatrix}u_m \\ u_s\end{bmatrix}$$

This is the rate of change of state variables of the actual robot. Similarly, the virtual robot (the reference), which needs to be tracked (or followed) can be described by the following equations:

$$\begin{bmatrix}{}^0\dot{x}_{sr} \\ {}^0\dot{y}_{sr} \\ \dot{\psi}_r\end{bmatrix} = \begin{bmatrix}a\cos\psi_r & 0 \\ a\sin\psi_r & 0 \\ \sin\beta_r & 1\end{bmatrix}\begin{bmatrix}u_{mr} \\ u_{sr}\end{bmatrix}$$

All the state space equations developed above are with respect to the frame R_0.
Let error be defined as: $error = reference - actual$. In the R_s frame, the errors in rate of change of x,y and ψ is:

$$\begin{bmatrix}{}^s\dot{x}_e \\ {}^s\dot{y}_e \\ \dot{\psi}_e\end{bmatrix} = \begin{bmatrix}0 \\ 0 \\ -\dot{\psi}\end{bmatrix} \times \begin{bmatrix}{}^s x_e \\ {}^s y_e \\ \psi\end{bmatrix} +^s \Omega_0(\psi)\left[\begin{bmatrix}{}^0\dot{x}_{sr} \\ {}^0\dot{y}_{sr} \\ \dot{\psi}_r\end{bmatrix} - \begin{bmatrix}{}^0\dot{x}_s \\ {}^0\dot{y}_s \\ \dot{\psi}\end{bmatrix}\right]$$

Here,

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

Using the equations developed in this section, the error model is simplified as:

$$\begin{bmatrix}{}^s\dot{x}_e \\ {}^s\dot{y}_e \\ \dot{\psi}_e\end{bmatrix} = \begin{bmatrix}au_{mr}\cos\psi_e \\ au_{mr}\sin\psi_e \\ u_{mr}\sin\beta_r + u_{sr}\end{bmatrix} + \begin{bmatrix}{}^s y_e\sin\beta - a & {}^s y_e \\ -{}^s x_e\sin\beta & -{}^s x_e \\ -\sin\beta & -1\end{bmatrix}\begin{bmatrix}u_m \\ u_s\end{bmatrix}$$

In short, the error model can be written as: $\dot{X} = f(X, \mathbf{u})$ Here, X is the state variable vector and $\mathbf{u}$ is the control input vector. Consider a Lyapunov function $V(X)$ for the system $\dot{X} = f(X, \mathbf{u})$. Designing a Lyapunov controller means finding a control input vector $\mathbf{u}$ to satisfy:

$$\dot{V}(X) = \frac{\partial V}{\partial X}\dot{X} = \frac{\partial V}{\partial X}f(X, \mathbf{u}) < 0$$

The Lyapunov function chosen must be positive definite. For this system, the Lyapunov function chosen is:

$$V(X) = \frac{1}{2}({}^s x_e^2 +^s y_e^2 + \frac{{}^s\psi_e^2}{K_y})$$

It can be guaranteed that this is a Lyapunov function because it is positive definite and clearly has continous partial derivatives.

$$\dot{V}(X) = \frac{\partial V}{\partial X}\dot{X} = \frac{\partial V}{\partial X} f(X, \mathbf{u}) < 0$$

$$\Rightarrow \qquad \begin{bmatrix} {}^s x_e & {}^s y_e & \frac{\psi_e}{K_y} \end{bmatrix} \begin{bmatrix} {}^s\dot{x}_e \\ {}^s\dot{y}_e \\ \dot{\psi}_e \end{bmatrix} < 0$$

On simplifying:

$$(au_{mr}\cos\psi_e - au_{mr})^s x_e + (\frac{au_{mr}\sin\psi_e^s y_e}{\psi_e} + \frac{u_{mr}\sin\beta_r + u_{sr}}{K_y} - \frac{u_{mr}\sin\beta + u_{sr}}{K_y})\psi_e < 0$$

This inequality is satisfied if:

$$\begin{aligned} au_{mr}\cos\psi_e - au_{mr} &= -K_x^s x_e \\ \frac{au_{mr}\sin\psi_e^s y_e}{\psi_e} + \frac{u_{mr}\sin\beta_r + u_{sr}}{K_y} - \frac{u_{mr}\sin\beta + u_{sr}}{K_y} &= -\frac{K_\psi}{K_y}^s \psi_e \end{aligned}$$

$$\begin{aligned} \Rightarrow \qquad u_m &= \frac{au_{mr}\cos\psi_e + K_x^s x_e}{a} \\ u_s &= K_y \frac{au_{mr}\sin\psi_e^s y_e}{\psi_e} + K_\psi\psi_e + u_{mr}\sin\beta_r + u_{sr} - u_{mr}\sin\beta \end{aligned} \tag{II.13}$$

The above equations yield the value of the control inputs.
Hence,

$$\mathbf{u} = \begin{bmatrix} \frac{au_{mr}\cos\psi_e + K_x^s x_e}{a} \\ \frac{K_y au_{mr}\sin\psi_e^s y_e}{\psi_e} + K_\psi\psi_e + u_{mr}\sin\beta_r + u_{sr} - u_{mr}\sin\beta \end{bmatrix}$$

The use of this control law makes the system globally asymptotically stable because the substitution of these values in $\dot{V}(x)$ yields:

$$\dot{V}(x) = -K_x^s x_e^2 - K_\psi \frac{\psi_e^2}{K_y}$$

Choice of positive gains ensures $\dot{V} < 0$ globally, hence the stability.
As can be seen in the Equation II.13, the knowledge of reference steering angle β_r, the reference steering angular velocity u_s and the reference translational velocity of the steering wheel u_m is required.
The control objective is to force the robot to move along a circular trajectory centred at $(0, 0)$ with radius $R = 2$ m and angular velocity of $\omega_d = 0.5$ rad/s, this implies that the tangential velocity of the robot must be 1 m/s.
Since the robot is expected to follow a circular trajectory, the reference steering angle must be a constant. From Figure 1.1 it can be concluded that for a circular path, the constant reference steering angle is:

$$\beta_r = \sin^{-1}\frac{a}{R}$$

The translational velocity of the type (1,1) robot is $\dot{x}^2 + \dot{y}^2 = au_m cos\beta$ must be equal to the tangential velocity required to satisfy the control objective. If the tangential velocity is $V_{tangential}$ then:

$$\dot{x}^2 + \dot{y}^2 = au_m cos\beta = V_{tangential}$$

$\Longrightarrow$

$$u_m = \frac{V_{tangential}}{a\cos\beta}$$

For the control objective specified, $V_{tangential} = R\omega_d = 1$ m/s.

II.4.2 Modelling

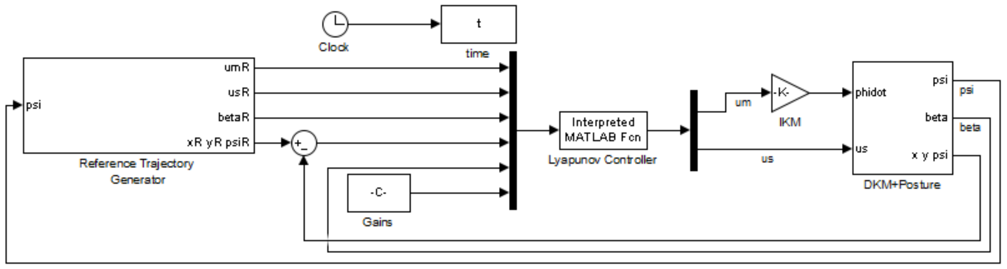

Figure II.4.1: Simulink model of a Lyapunov function controlled Type(1,1) robot

Figure 5.1 presents the **simulink** model of the Lyapunov function controlled Type(1,1) robot. A brief description of all the subsystems is presented below:

- The `Reference Trajectory Generator` subsystem generates the desired trajectory, in the R_s frame, that the robot is expected to follow. Besides these, it also generates the reference steering wheel translational velocity u_{mr}, the reference steering wheel angular velocity u_{sr} and the reference steering angle β_r.
- The `Lyapunov Based Controller` subsystem implements the Lyapunov control law deduced in the previous section. It takes the errors in position (expressed in R_s frame) & orientations , u_{mr}, u_{sr}, β_r, β and the controller gains as inputs and gives $\mathbf{u} = [u_m, u_s]^T$ as the output.
- The `DKM+Posture` block takes as input the steering wheel spin velocity $\dot{\phi}$ and the the steering angular velocity u_s as input. The subsytem implements the direct kinematic model and the posture kinematic model for the type (1,1) robot. There is a `Localisation` block inside this subsytem which shifts the tracking point to point s. Then a frame transformation is performed and the subsytem outputs the posture of the robot in the R_s frame.

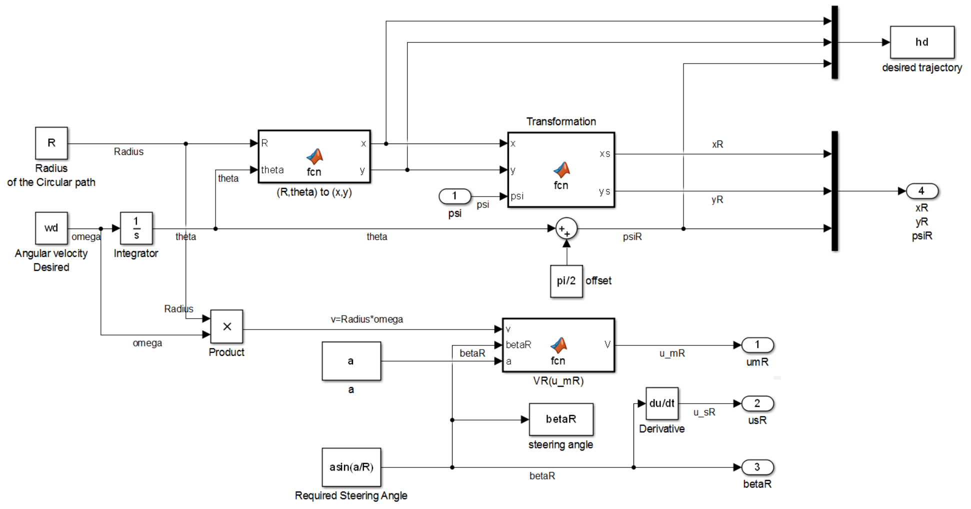

Figure II.4.2: Reference Trajectory Generator subsystem

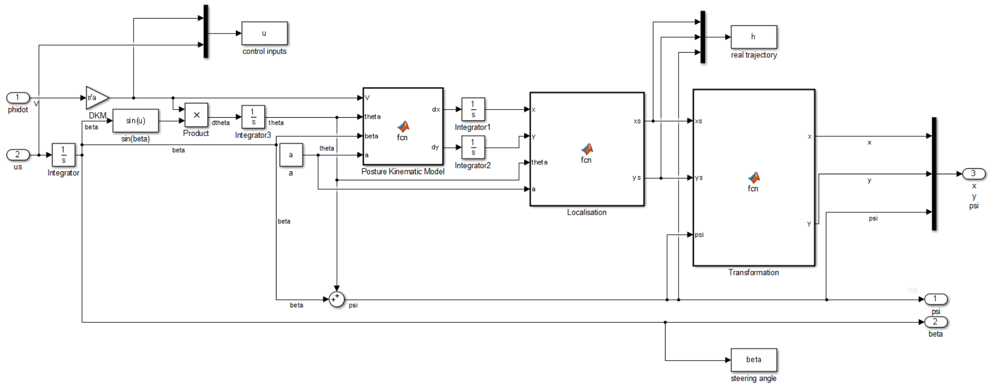

Figure II.4.3: DKM+Posture subsystem

II.4.3 Validation and Simulation

Remark: The `fixed step ode4 Runge Kutta` solver is used with a time step of 0.001 second.
Tuning: K_x,K_y **and** K_θ
To ensure $\dot{V} < 0$, K_x,K_y and K_θ must all be positive.

- Tuning with $K_x = 1$, $K_y = 1$ and $K_\theta = 1$:

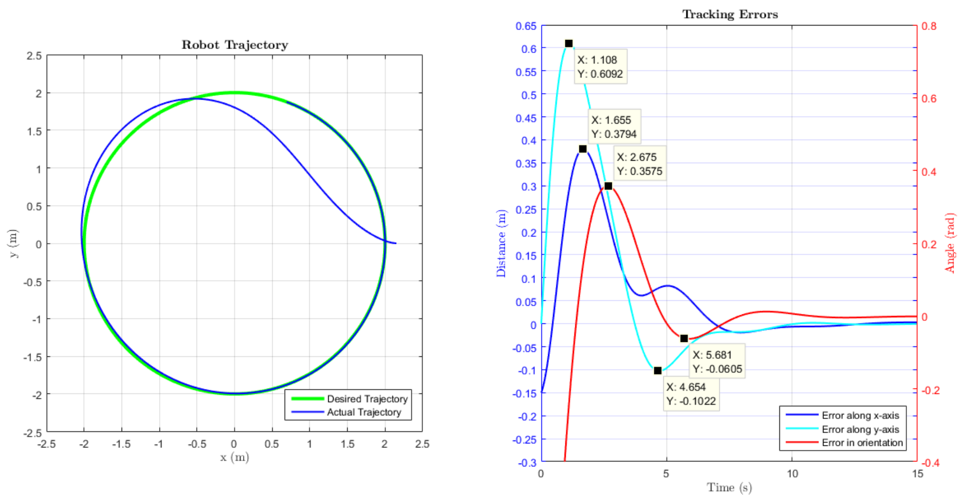

Figure II.4.4: Robot Trajectory and errors in R_0 frame with $K_x = 1$, $K_y = 1$ and $K_\theta = 1$

It can be clearly noted in the Figure 5.5 that the trajectory is stable and converges close to the reference, but the response time is high and the gains need to be re tuned to give better dynamic performance. Increasing the gains may improve the dynamic performance.

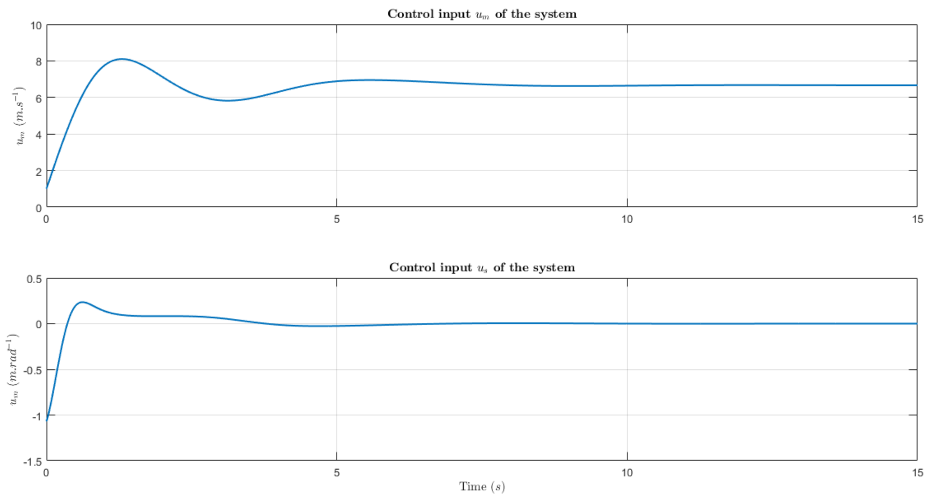

Figure II.4.5: The control inputs with $K_x = 1$, $K_y = 1$ and $K_\theta = 1$

- Tuning with $K_x = 10$, $K_y = 10$ and $K_\theta = 10$:

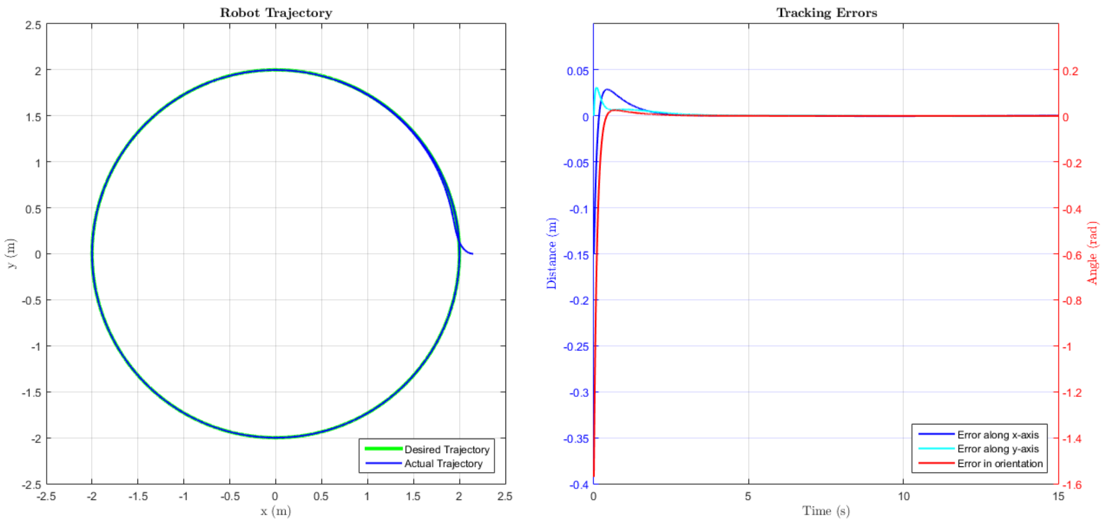

Figure II.4.6: Robot Trajectory and errors in R_0 frame with $K_x = 10$, $K_y = 10$ and $K_\theta = 10$

As can be noted in the Figure 5.7, the dynamic performance has improved significantly and the error in θ converges to a constant value in the order of 10^{-4} radians but the error in x and y oscillate slowly about zero with a significantly low magnitude in the order of 10^{-5} metres. The settling time is approximately 3.5 seconds. Increasing the gains further may give better results in the simulation but practically the gain cannot be increased indefinitely as high gains can make the system unstable and saturate the actuators.

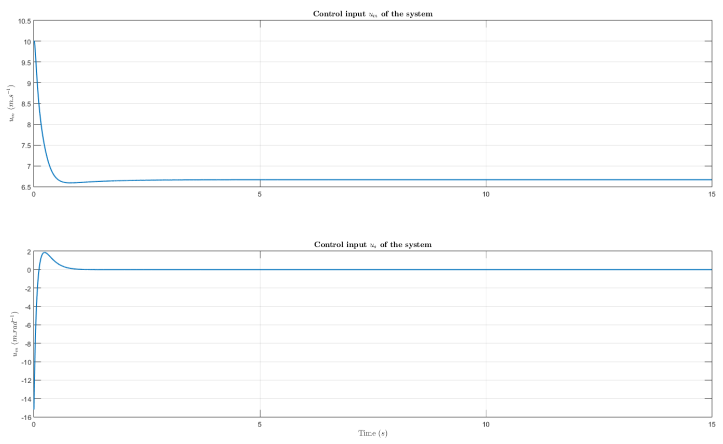

Figure II.4.7: Control inputs with $K_x = 10$, $K_y = 10$ and $K_\theta = 10$

Effect of Parameter Variation:

- **Case 1:** 10% increase in value of **a** only

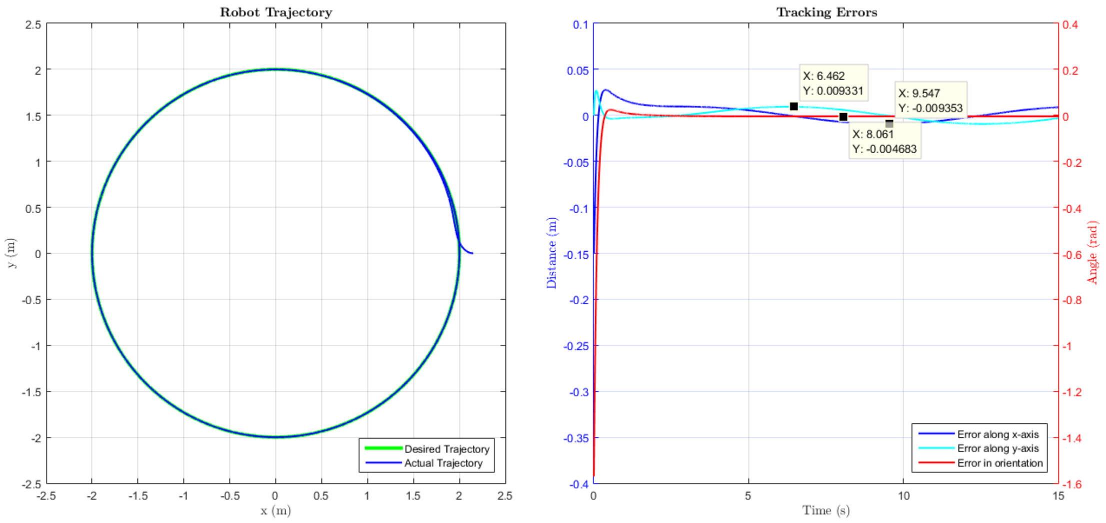

Figure II.4.8: Robot Trajectory and errors in R_0 frame with $K_x = 10$, $K_y = 10$ and $K_\theta = 10$ and 10% increase in **a**

The effect of parameter variation can be observed by comparing Figure 5.7 and Figure 5.9. In this case, error in x and y oscillate with a greater amplitude (0.0094 metres and 0.0093 metres respectively) and the error in orientation θ increases to a new steady state value (-0.004683 radians). The overall performance has deteriorated due to parameter variation.

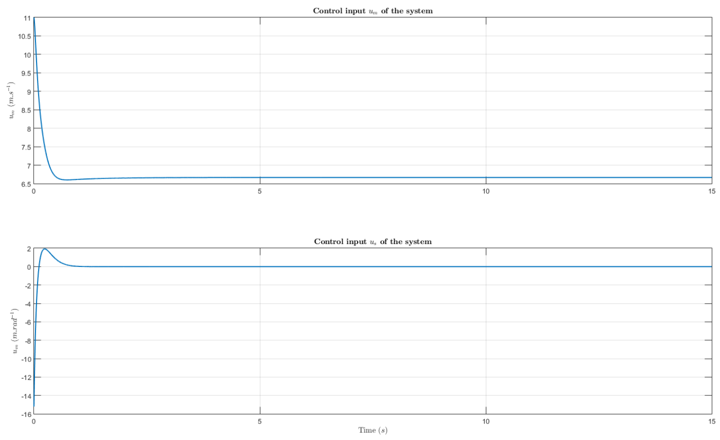

Figure II.4.9: Control inputs with $K_x = 10$, $K_y = 10$ and $K_\theta = 10$ and 10% increase in **a**

A possible solution to reduce the effect of parameter variation could be increasing the gains within reasonable limits. On making $K_x = 25$, $K_y = 25$ and $K_\theta = 15$, a reduction in error magnitudes can be seen (Figure 5.11). The errors in x and x now oscillate slowely with a lower amplitude (0.0037 metres in both axes) and the error in orientation θ has also reduced to a lower steadt state value (-0.001881 radians).

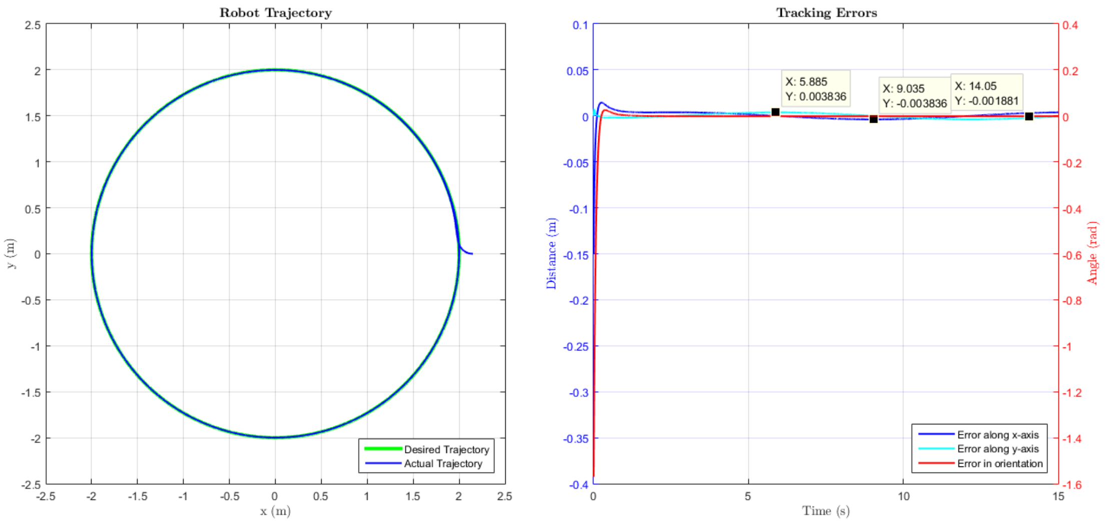

Figure II.4.10: Robot Trajectory and errors in R_0 frame with $K_x = 25$, $K_y = 25$ and $K_\theta = 15$ and 10% increase in **a**

- **Case 2:** 10% increase in value of **r** only
  As in Case 1, the effect of variation in **r** results in oscillatory errors in x and y with higher amplitude(approximately 0.009717 metres in both axes) and increased steady state error in θ (approximately 0.004859 radians) as shown in Figure 5.12. A higher gain tends to mitigate this problem as observed in Figure 5.14. The errors in x and y reduce to a lower amplitude (approximately 0.003978 metres in both axes) and converge to a reduced steady state error in θ (approximately 0.001952 radians).

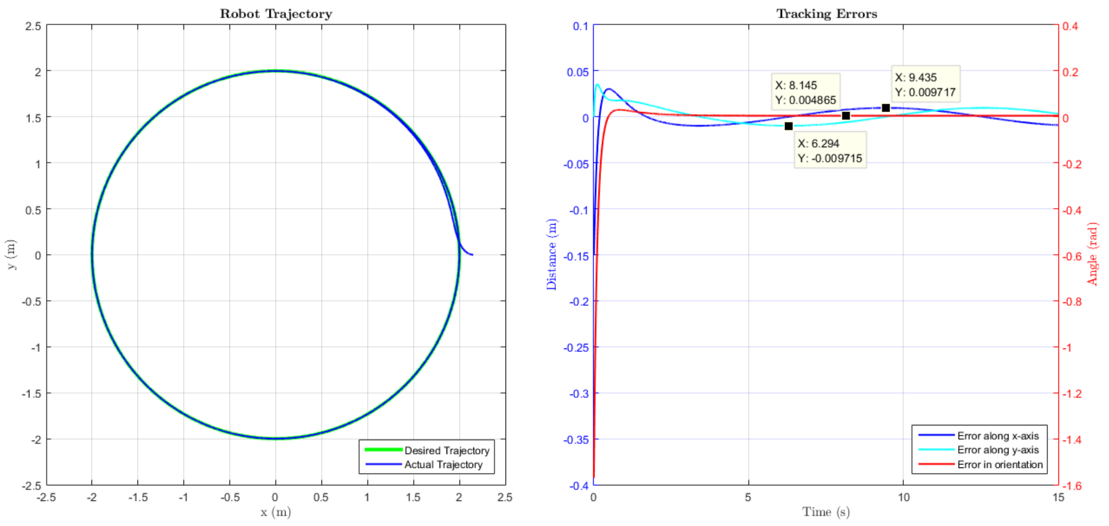

Figure II.4.11: Robot Trajectory and errors in R_0 frame with $K_x = 10$, $K_y = 10$ and $K_\theta = 10$ and 10% increase in $\mathbf{r}$

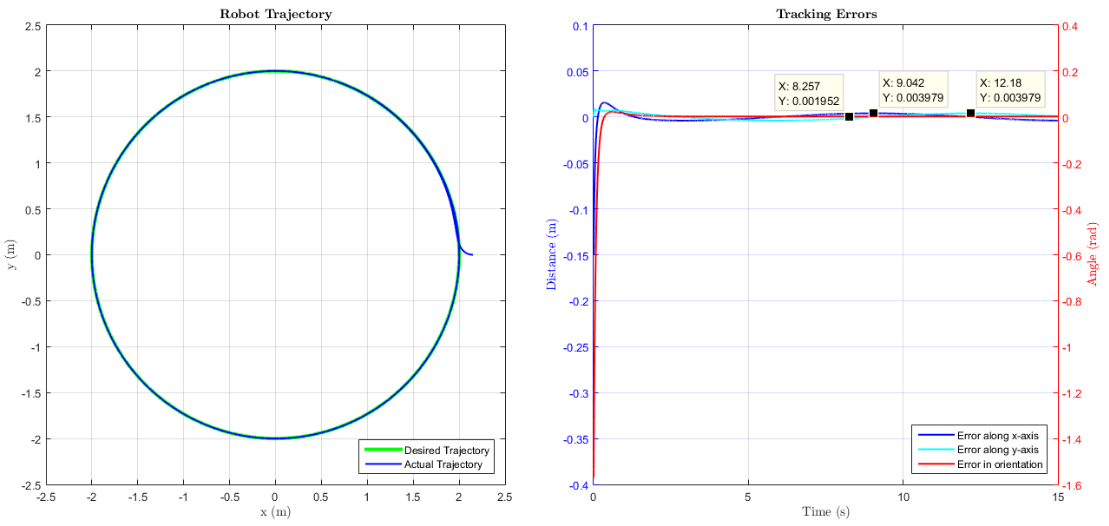

Figure II.4.12: Robot Trajectory and errors in R_0 frame with $K_x = 25$, $K_y = 25$ and $K_\theta = 15$ and 10% increase in $\mathbf{r}$

The effect of 10% decrease in the above parameters gives similar results but with changes in magnitude and phases, nevertheless, the errors reduced when a higher gains are applied.