Olivia Ball
ECE 516
4 May 2020

Milestone 3

Introduction

The final portion of this project required that the unmanned vehicle moved from an initial position to a desired position by pole placement. The use of pole placement allows us to design a state feedback loop and then design an observer. The poles define the systems eigenvalues, but they may not always appear as poles as pole cancelation sometimes occurs. In order to model the system moving from one position to the next, the model will be converted into polar coordinates. By using the polar coordinates, we are able to better track the movement of the UV over time.

The reason for adding feedback to a system is to achieve better characteristics for how the system performs. A state feedback loop requires that a state feedback gain matrix, *K,* is designed. The system controllability and observability will not be affected by the feedback gain that is used. Due to this, the system needs to be completely controllable and completely observable before applying the state feedback gain matrix. The location of the poles will determine the stability of the overall system. The goal is to design the K matrix so that each state is driven to zero as time goes to infinity. When the system is zero for all states the system will be stable and achieve the desired location.

After the state feedback gain matrix is designed to produce a stable system, the observer can be designed. The goal of the observer is to produce an estimation of all state outputs even when they are not directly measured in the system. The actual state outputs will be used to test how good the estimations of the system are. The output values should still trend to zero as time goes to infinity as this will lead to a stable system.

The overall design of a system that moves from one location to another will require a lot of trial and error to design these poles along with the use of previous knowledge about things such as observability, controllability and continuous state-space formulation. The model shown below in figure 1 is taken from the book *Introduction to Autonomous Mobile Robots* and provides a visual representation of the robot coordinate frame in terms of the goal coordinate frame. These relationships were used when converting the system between its rectangular coordinates to the polar coordinates and vice versa.

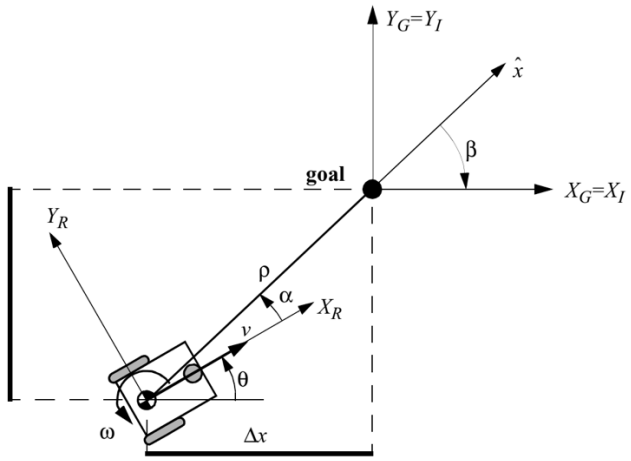

Figure 1: Robot coordinate frame in terms of the goal/inertial coordinate frame

Problem Formulation

The first step was to define the system in new polar coordinates in terms of $\rho, \alpha, \beta, \omega_R$ and ω_L. The three equations below were used to convert the five old nonlinear equations into five new modified nonlinear equations. The equations for ω_R and ω_L remain the same and only the first three equations will require modification. The old five-state space Simulink model can be used and only the robot dynamics subsystem will need to be changed to implement these new equations. The nonlinear model was used in Simulink so that noise and disturbances could be added throughout the system and their effects could be easily measured and accessed. In the three equations below ρ represents the distance between the center of the robot and the desired location, α representats the angle between the center of the robot and the desired goal and β represents the angle from the x axis to the goal as .

$$\boldsymbol{\rho = \sqrt{\Delta x^2 + \Delta y^2}}$$
$$\boldsymbol{\alpha = -\theta + atan2(x, y) \;\rightarrow\; -\theta + \arctan\left(\frac{\Delta y}{\Delta x}\right)}$$
$$\boldsymbol{\beta = -\theta - \alpha}$$

The system used the equations above to define the initial conditions for ρ, α and β. It is clear by the way the system is defined in figure 1 that coordinate transformations needed to be performed to obtain Δx and Δy. The calculations done to achieve this are shown in the mathematical formulation section. The value for θ was defined as the initial angle subtracted from the desired angle. The angular velocity for the system was defined as zero for both the right and left angular velocities. These were both set to zero so that the system is not moving initially.

The linearized model will need to be obtained and evaluated at the nominal point which will be defined as $x_n = [\rho \quad \alpha \quad \beta \quad \omega_R \quad \omega_L]^T$. The use of the nominal point will provide us with the A,B,C and D matrices for our system. In order to achieve complete controllability of the system, it was previously determined that ω_R and ω_L cannot both be zero. In order to ensure the system was moving forward with no additional orientation changes due to the nominal point, these two values were set as the same velocity. The nominal point was set as the max angular velocity which was 3.3724 rad/sec. By evaluating the system to be stable at its max angular velocity, we can see similar results when these nominal points are decreased. The β was set to zero which means that the UV is only going forward in the x-direction. By setting the nominal point at zero we once again avoid assuming additional orientation changes. Due to the definition of beta seen in figure 1 this meant that α at the nominal point is also zero. The ρ value was tested over a range and it was determined that this value greatly affected the input to the system. When the value was too high or too low, the input to the system became very large. It was determined that 1 provided the most realistic input to our system when the max angular velocity was used. The final nominal point is $x_n = [1 \quad 0 \quad 0 \quad 3.3724 \quad 3.3724]^T$.

After the nominal point was defined, the new A and B matrices were achieved. The system needs to achieve full controllability, or a rank of five, which was confirmed before moving forward with designing the feedback gain matrix. In the Simulink model the feedback gain matrix is placed within a gain block which connects from the output and is added back with the input. The gain matrix is defined by the *five* poles; since the A matrix had multiple zero eigenvalues determining our system poles was a process that required a lot of trial and error. In order to test many different poles at one time, the Monte Carlo method was performed to change the poles used to obtain the gain matrix. When it was noticed that all states trended to zero much smaller variations on each number were made until the best location for the pole was found. When all poles were determined the "place" command used the A matrix, the B matrix and the poles to obtain the gain matrix.

After the gain matrix, *K*, was determined the next step was to place the observer in front of the gain matrix and determine the constants within the matrix. There are two inputs into the observer, the output from the

system (dy) and the output from the gain matrix that becomes part of the new input (du). The system requires a new gain matrix B_C to be defined which requires a new set of poles to be defined. These poles need to be significantly larger than the poles for the state feedback gain matrix. These poles were also determined through the use of the Monte Carlo method which only changed the poles used to create the B_C matrix. The "place" command was used once again to create the B_C matrix using the observer poles, A matrix and C matrix. Within the Simulink model for the observer the B_C matrix was used as the gain for du input to the system which was added to the new A_C matrix along with a K_Z matrix. The K_Z matrix was used as the gain for dy.

$$A_C^T = A^T - B_C C^T$$
$$K_Z = B^T - B_C * D^T$$

After both poles were placed determined using all of the states (C matrix was identity matrix), the real C matrix was determined. The output matrix shown below was originally the goal, however poles could not be determined to drive the real alpha output to zero. The matrix achieved full rank and would only have used two sensors which made it a desirable output.

$$\begin{bmatrix} rho \\ beta \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \end{bmatrix} \begin{bmatrix} \rho \\ \alpha \\ \beta \\ \omega_R \\ \omega_L \end{bmatrix}$$

After much trial and error, it was decided that it was better to have three sensors that provided very little error between the real state values and the estimated state values vs. having one less sensor. The output below eliminates two state sensors while still providing that all error trends to zero. This was the output used in the model.

$$\begin{bmatrix} \boldsymbol{rho} \\ \boldsymbol{alpha} \\ \boldsymbol{beta} \end{bmatrix} = \begin{bmatrix} \mathbf{1} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{1} & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{1} & \mathbf{0} & \mathbf{0} \end{bmatrix} \begin{bmatrix} \boldsymbol{\rho} \\ \boldsymbol{\alpha} \\ \boldsymbol{\beta} \\ \boldsymbol{\omega_R} \\ \boldsymbol{\omega_L} \end{bmatrix}$$

The same poles were used for driving the UV from each initial location to the desired location. The outputs were used to confirm the movement of the UV and that the estimation of each state was very close to the real state value. Since the simulation within Simulink was in polar coordinates, the system outputs needed to be converted back to rectangular coordinates for analysis. These equations are seen below, and it should be stated that two different thetas were used due to the way the model was defined. After it was confirmed that the system could move from the initial condition to the desired location a variety of different disturbances and noise were applied to the system to see the effects on the system.

$$theta = desired\ angle * beta$$
$$\boldsymbol{x = rho * \cos(theta)}$$
$$\boldsymbol{y = rho * \sin(theta)}$$
$$\boldsymbol{angle\ uv = desires\ angle - (alpha + beta)}$$

Mathematical Modeling

The translation of the goal location back to the origin can be done by using desired location values as follows where d_a is the desired angle, x_d and y_d are desired locations. The matrix then needs to be multiplied by the initial values and desired values and those will be used to define Δx and Δy.

$$\begin{bmatrix} \cos(d_a) & \sin(d_a) & x_d * \cos(d_a) - y_d * \sin(d_a) \\ -\sin(d_a) & \cos(d_a) & -y_d * \cos(d_a) - x_d * \sin(d_a) \\ 0 & 0 & 1 \end{bmatrix}$$

Using the relationships above the following differential equations were made:

(1) $f1 = \dot{\rho} = \frac{-r}{2}(\omega_l + \omega_r) * cos\alpha$

(2) $f2 = \dot{\alpha} = \left(\frac{rsin(\alpha)}{2\rho} - \frac{r}{W_r}\right)\omega_r + (\frac{rsin(\alpha)}{2\rho} + \frac{r}{W_r})\omega_l$

(3) $f3 = \dot{\beta} = \frac{-rsin(\alpha)}{2\rho}(\omega_l + \omega_r)$

(4) $f4 = \dot{\omega}_r = \frac{K_{eq}}{J_{eq}R_a}v_{ar} - (\frac{B_{eq}+\frac{K_{eq}K_b}{R_a}}{J_{eq}})\omega_r$

(5) $f5 = \dot{\omega}_l = \frac{K_{eq}}{J_{eq}R_a}v_{al} - (\frac{B_{eq}+\frac{K_{eq}K_b}{R_a}}{J_{eq}})\omega_l$

Which was translated into the following five-state representation:

$$\begin{bmatrix} \dot{\rho} \\ \dot{\alpha} \\ \dot{\beta} \\ \dot{\omega}_R \\ \dot{\omega}_L \end{bmatrix} = \begin{bmatrix} 0 & 0 & 0 & -\frac{r}{2}\cos(\alpha) & -\frac{r}{2}\cos(\alpha) \\ 0 & 0 & 0 & \frac{r\sin(\alpha)}{2\rho} - \frac{r}{W_r} & -\frac{r\sin(\alpha)}{2\rho} - \frac{r}{W_r} \\ 0 & 0 & 0 & \frac{-r\sin(\alpha)}{2\rho} & \frac{-r\sin(\alpha)}{2\rho} \\ 0 & 0 & 0 & \frac{B_{eq} + \frac{K_{eq}K_b}{R_a}}{J_{eq}} & 0 \\ 0 & 0 & 0 & 0 & \frac{B_{eq} + \frac{K_{eq}K_b}{R_a}}{J_{eq}} \end{bmatrix} \begin{bmatrix} \rho \\ \alpha \\ \beta \\ \omega_R \\ \omega_L \end{bmatrix} + \begin{bmatrix} \frac{K_{eq}}{J_{eq}R_a} \\ \frac{K_{eq}}{J_{eq}R_a} \end{bmatrix} \begin{bmatrix} v_{ar} \\ v_{al} \end{bmatrix}$$

Use the generalized equation below to come up with the 5 corresponding linearized equations below:

$$\delta\dot{x_n} = \frac{\partial f_n}{\partial \rho}|_{x_n,u_n}\delta\rho + \frac{\partial f_n}{\partial \alpha}|_{x_n,u_n}\delta\alpha + \frac{\partial f_n}{\partial \beta}|_{x_n,u_n}\delta\beta + \frac{\partial f_n}{\partial \omega_r}|_{x_n,u_n}\delta\omega_r + \frac{\partial f_n}{\partial \omega_l}|_{x_n,u_n}\delta\omega_l + \frac{\partial f_n}{\partial v_{ar}}|_{x_n,u_n}\delta v_{ar} + \frac{\partial f_n}{\partial v_{al}}|_{x_n,u_n}\delta v_{al}$$

(1) $\boldsymbol{\delta\dot{\rho}} = 0\delta\rho + \frac{r*\sin(\alpha)*(\omega_R+\omega_L)}{2}|_{x_n,u_n}\delta\alpha + 0\delta\beta + \frac{-r*\cos(\alpha)}{2}|_{x_n,u_n}\delta\omega_r + \frac{-r*\cos(\alpha)}{2}|_{x_n,u_n}\delta\omega_l + 0\delta v_{ar} + 0\delta v_{al}$

(2) $\boldsymbol{\delta\dot{\alpha}} = \frac{-r*\sin(\alpha)*(\omega_R+\omega_L)}{2\rho^2}|_{x_n,u_n}\delta\rho + \frac{r*\cos(\alpha)*(\omega_R+\omega_L)}{2\rho}|_{x_n,u_n}\delta\alpha + 0\delta\beta + \frac{r*\sin(\alpha)}{2\rho} - \frac{r}{W_r}|_{x_n,u_n}\delta\omega_r + \frac{r*\sin(\alpha)}{2\rho} + \frac{r}{W_r}|_{x_n,u_n}\delta\omega_l + 0\delta v_{ar} + 0\delta v_{al}$

(3) $\boldsymbol{\delta\dot{\beta}} = \frac{r*\sin(\alpha)*(\omega_R+\omega_L)}{2\rho^2}|_{x_n,u_n}\delta\rho + \frac{-r*\cos(\alpha)*(\omega_R+\omega_L)}{2\rho}|_{x_n,u_n}\delta\alpha + 0\delta\beta + \frac{-rsin(\alpha)}{2\rho}|_{x_n,u_n}\delta\omega_r - \frac{-rsin(\alpha)}{2\rho}|_{x_n,u_n}\delta\omega_l + 0\delta v_{ar} + 0\delta v_{al}$

(4) $\boldsymbol{\delta\dot{\omega}_r} = 0\delta\rho + 0\delta\alpha + 0\delta\beta + \frac{-(B_{eq}+\frac{K_e*K_b}{R_a})}{J_{eq}}|_{x_n,u_n}\delta\omega_r + 0\delta\omega_l + \frac{K_{eq}}{R_a*J_{eq}}|_{x_n,u_n}\delta v_{ar} + 0\delta v_{al}$

(5) $\boldsymbol{\delta\dot{\omega}_l} = 0\delta\rho + 0\delta\alpha + 0\delta\beta + 0\delta\omega_r + \frac{-(B_{eq}+\frac{K_e*K_b}{R_a})}{J_{eq}}|_{x_n,u_n}\delta\omega_l + 0\delta v_{ar} + \frac{K_{eq}}{R_a*J_{eq}}|_{x_n,u_n}\delta v_{al}$

Simulation- Changes to Implement Observer and Feedback from Previous Model

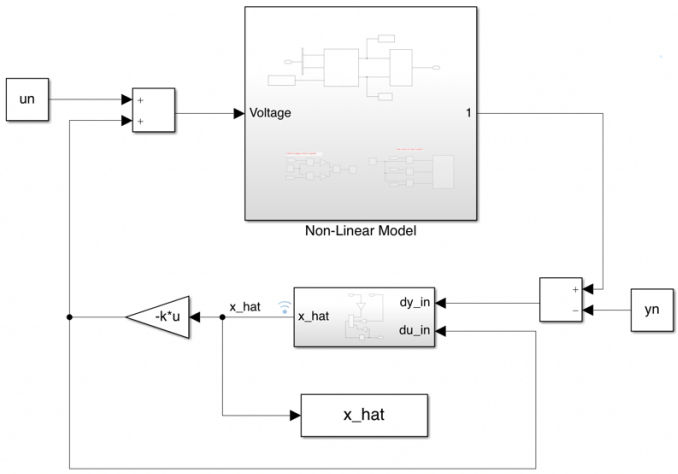

Figure 2: Implementation of the feedback gain matrix, k, and the observer where both of the external inputs are set to zero

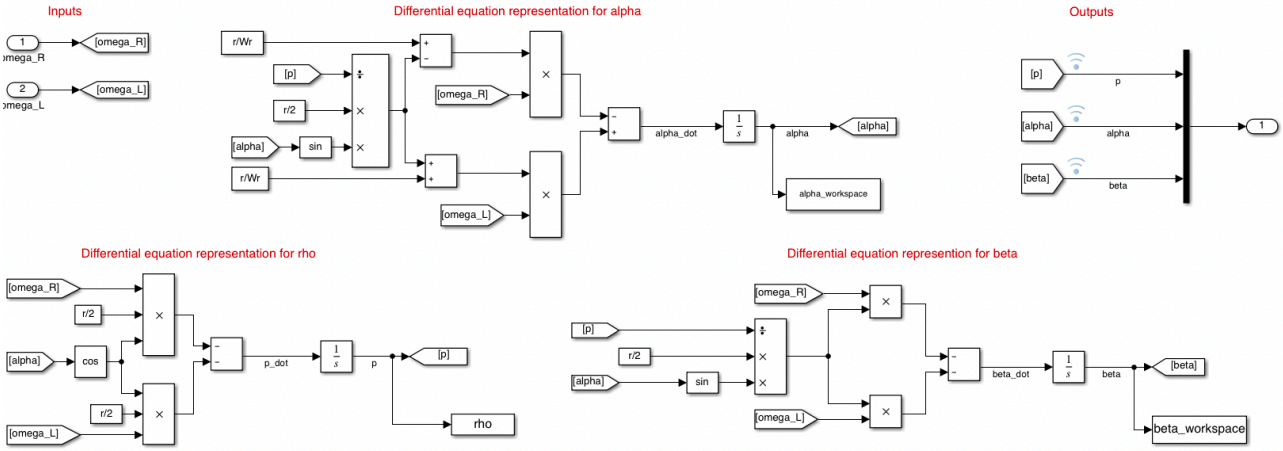

Figure 3: Implementation of new polar coordinates within the Robot Dynamic subsystem and displays that the only three outputs are rho, alpha and beta

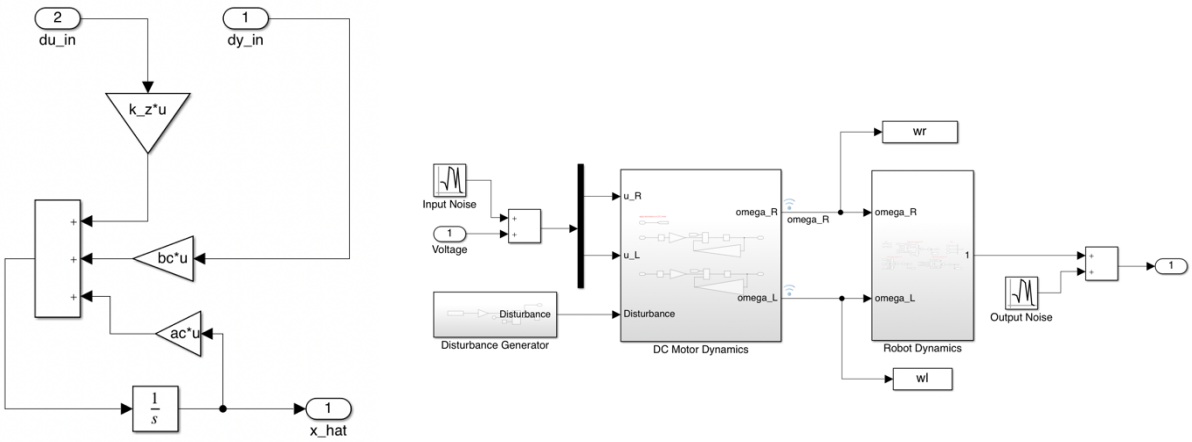

Figure 4: (Left) Implementation of the observer which will estimate the angular velocity for the right and left wheels (Right) Where each of the noises and disturbances were applied within the model

Results

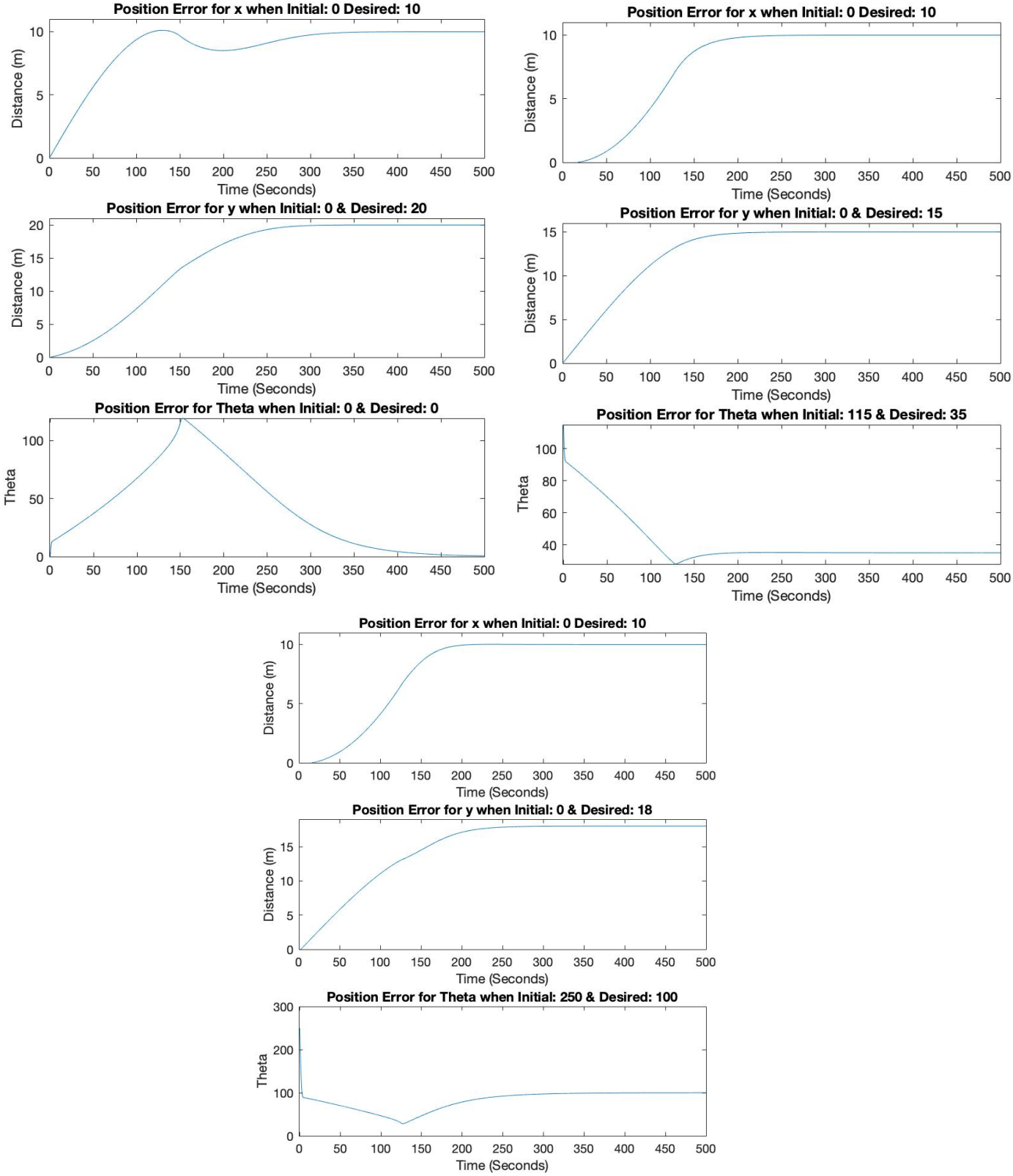

Figure 5: The position error for the trial described where the position error is defined as the difference between the output of the system to the desired location

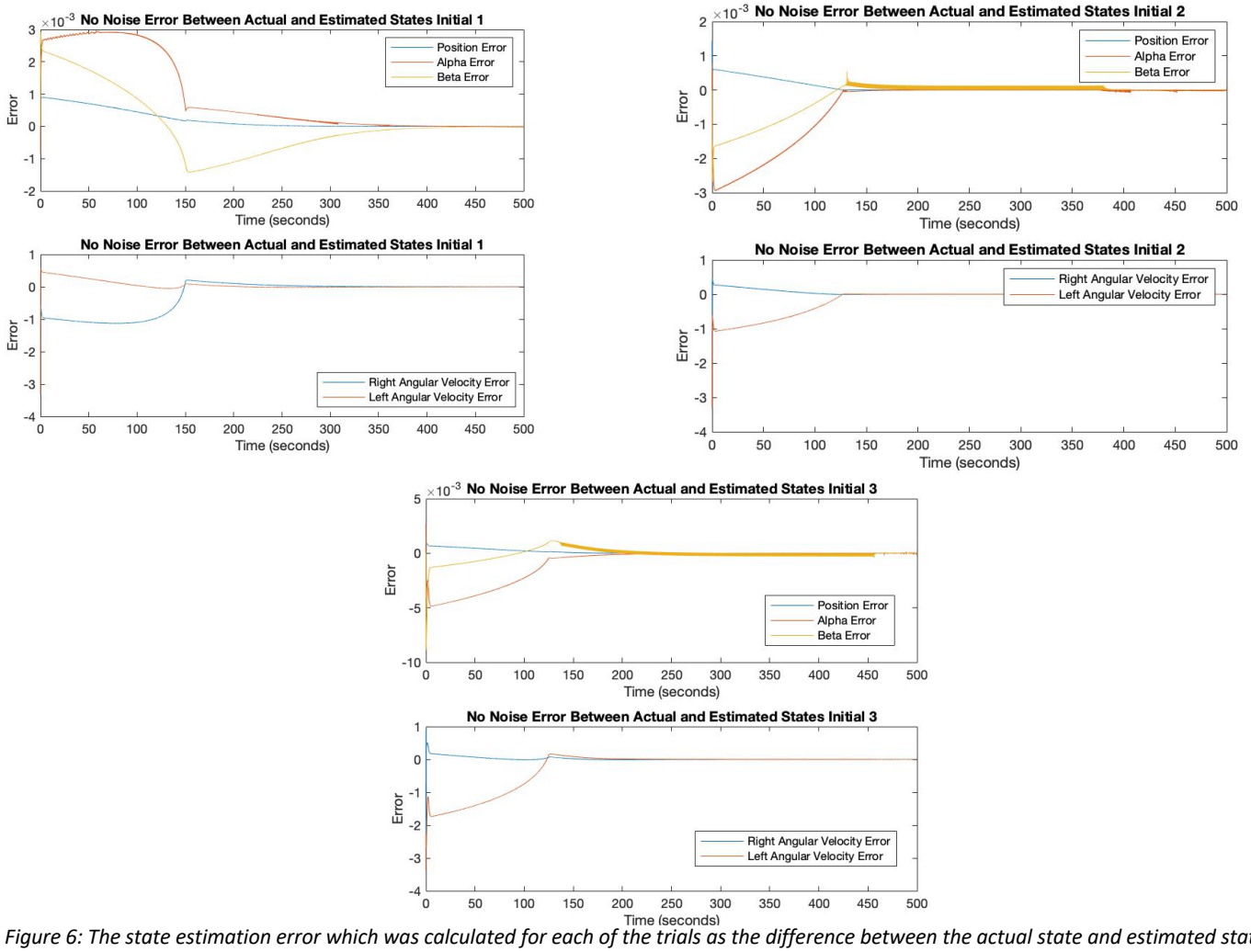

Figure 6: The state estimation error which was calculated for each of the trials as the difference between the actual state and estimated state

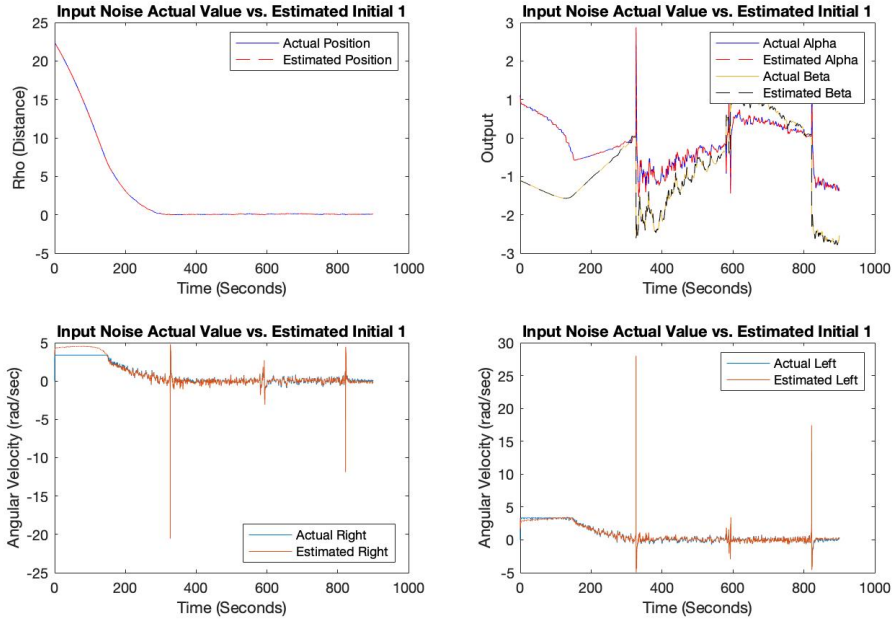

Figure 7: Each actual state and estimated state when gaussian noise added every 10 seconds with a variance of 0.01 to input

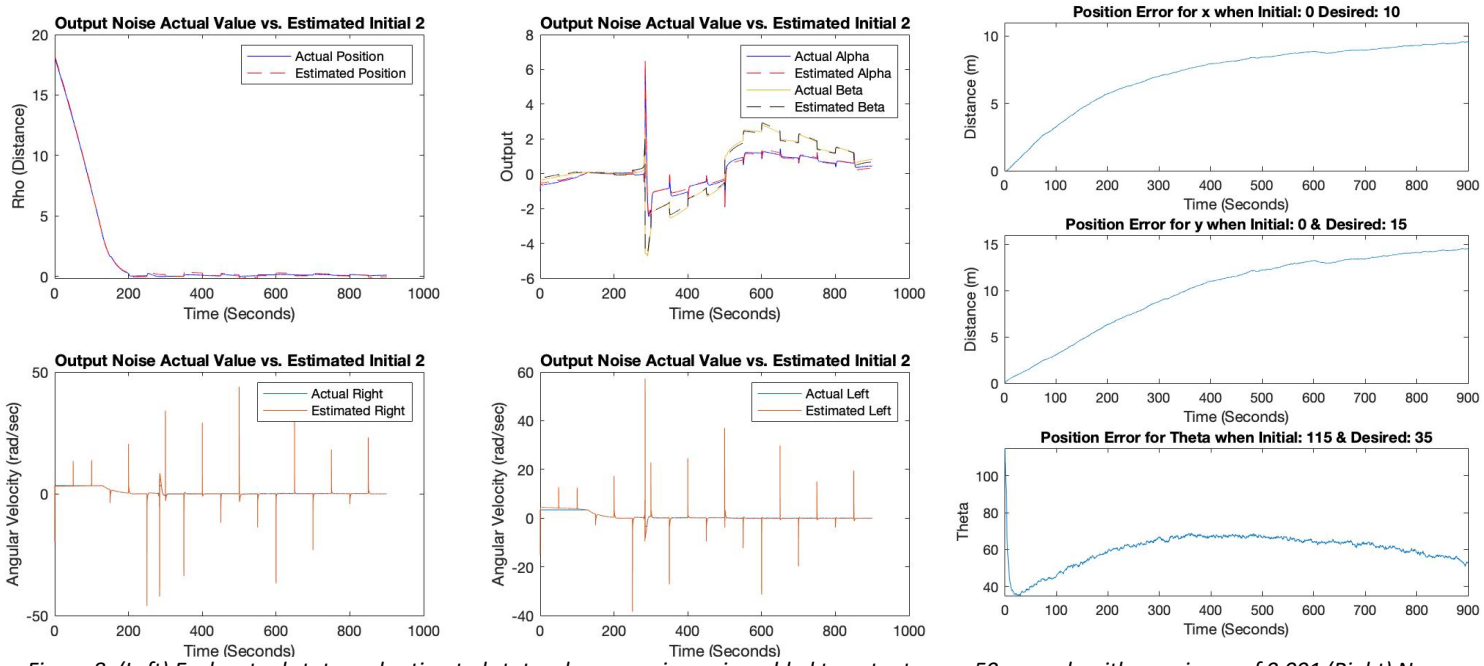

Figure 8: (Left) Each actual state and estimated state when gaussian noise added to output every 50 seconds with a variance of 0.001 (Right) New x,y and theta values after noise

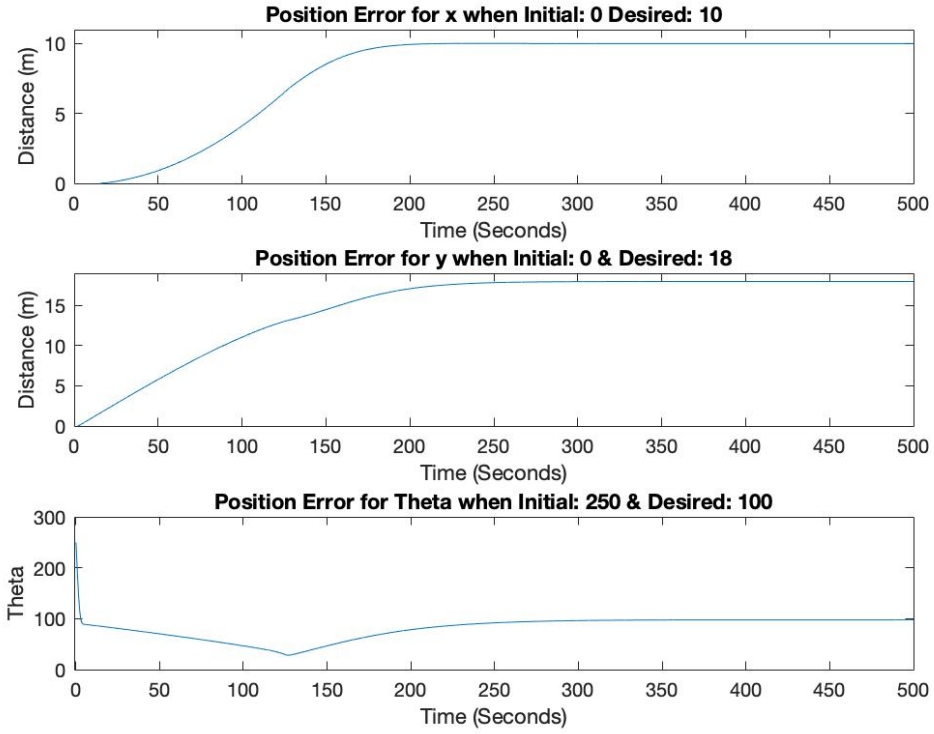

Figure 9: Position error when disturbance added the model is very similar to the original trial three slightly lower theta

Discussion

The final poles that were selected in the model for the controller were [-7.5 -0.4 -0.1 -0.0375 -0.65] and the final poles that were used for the observer were [-41 -36 -43 -44 -45]. The point of the poles is that they are set and allow the UV to make it from the initial location to the final location no what movement is described. The implementation of these poles provided us with the following gain matrices where k was for the controller and B_C was for the observer. Since the poles along with the A, B and C matrices do not change, these matrices will not change when moving between different initial and final positions.

$$k = \begin{bmatrix} -0.0603 & -1.9996 & 1.01409 & -2.2168 & -0.6557 \\ -0.1826 & 2.2106 & -1.1328 & -0.6813 & -2.1171 \end{bmatrix}$$

$$B_c = \begin{bmatrix} 62.6049 & -0.04598 & 0 \\ -0.4599 & 68.6571 & 0 \\ 0 & -0.1231 & -45.0000 \\ -14048.5042 & -1319.2908 & 0 \\ -14107.2114 & -1971.4819 & 0 \end{bmatrix}$$

The number of sensors used for the model was three: one for rho, one for alpha and one for beta. When one of the states does not trend to zero it means that the UV will still be in motion about that state. The use of these three sensors provided the same exact information as if the system used all five state sensors. In order to successfully drive the model from one location to the next, due to the way the model was defined, the angles each have to be defined within [-pi, pi] or [0°, 360°]. The model will not be able to operate in reverse and the x and y positions will not be limited unless there are time constraints placed on the model.

There were three desired movements that needed to be modeled to transport the UV from a set of initial coordinates to a set of desired coordinates. These coordinates were defined in terms of [x_i y_i θ °] and were all initially evaluated without noise to observe the system dynamics. In order to test the performance of the model, each one was evaluated in terms of the position error, the time it took for the UV to arrive at the final destination and the state estimation error. The position error was calculated by subtracting position over time from the desired position and the state estimation error was found by taking the difference between the actual state and the observer state values.

The first trial was to move the UV from the point [0 0 0°] to the desired location [10 20 0°]. When using the Simulink data inspector, it appears that all states reach a final value of zero at 486.2 seconds. When evaluating the model within MatLAB, it is noticed that the UV is at an angle of 0.773° at that time. However, if the system is run until 599.96 seconds the UV remains at 0.0742° for an extended period of time. The extra time allows for us to find a final location that is much closer to the desired location of 0°. In order to obtain the best accuracy, it is determined that for this trial the time from initial location to the final location is 599.96 seconds and the final position is [10 20 0.07°]. In figure 6 above you are able to see the position error. It shows that the robot starts at [0 0 0] and is steady state for each of the values by the end where the x position is 10, y position is 20 and theta appears to be 0. The state estimation error is seen above in figure 6. The state estimation error was extremely good for rho, beta and alpha as it should be noted the error is to a value of 10^{-3}. The state estimation error is higher for the right and left angular velocity. These errors are due to the fact that the system is not considering the input to the system to be a maximum of 10 volts. The actual angular velocities are operating at the max angular velocity however the estimation is higher than the max angular velocity. These variations do not appear to have an effect on the other states.

In the second trial the goal was to move the UV from the initial position [0 0 115°] to a desired location of [10 15 35°]. The overall system responds much quicker when moving between these two points. When looking at the Simulink data inspector, it appears that all states reach zero at a time of 245.1 seconds. Although the system appears to be zero for all states, running the simulation for longer will provide much even more desired location values. When the system is run for a total time of 383.3 seconds, the UV reaches

a desired location of [10 15 35.0210°] which is very close to the desired location. It can be seen that movement from this initial location to the desired location has very similar results to trial 1 for both the position error and the state estimation error. There is very little position error overall in the system and very little state estimation error for rho, beta and alpha. When looking at figure 5, the theta position error will not trend to zero. It will trend to the desired theta value as time goes to infinity, but this will only be achieved when the states for α and β go to zero as time trends to infinity.

The final trial was to test the system for movement from the initial position [0 0 250°] to the final position [10 18 100°]. At 396.2 seconds all states of the system appear to reach the equilibrium point of zero. At this point the system is exactly at the desired x and y position. At this point there is a theta value of 99.5792° which means that there is a variation of 0.4208°. If we assume that an error less than 0.5% is acceptable, we use the time as 396.2 seconds and the location therefore is [10 18 99.5792°]. The overall state estimation error is smaller for the left angular velocity and larger for the right angular velocity. The remaining states still have an estimation error of essentially zero and the position error is very low.

There were two different types of noise added to the system both of which had devastating effects on the system. Since the selected observer poles are very far away from zero, it means that the system is going to be more affected by noise. These diagrams all had very similar results between the actual and estimated values however neither one of them was stable.

When noise was applied directly to the output of the system it made the system almost unrecognizable. The noise applied to the output could be defined as tiny variations in the surface properties of the road like cracks or pieces of loose asphalt. When the amount of contact between the tire and road is altered, it can slight alter the friction coefficient. In order to implement this noise, a gaussian noise was applied to the system with a variance of 0.01 at a sample time of 50 seconds. The sample time of 50 seconds assumes that the system is not constantly exposed road imperfections. In figure 8 above, you can see that the model no longer trends to zero for α and β. The system will never reach the desired orientation due to this fact. After the system was run for 900 seconds, it starts to trend toward the final locations for x and y but still are not desirable outcomes. There are a few types of noise that can be added to the input, specifically if a loose connection starts to happen within the system. If the wires are starting to come loose this could cause large variations in the amount of voltage being placed into the system. In the model, we assumed that the UV is going through a really bumpy section of road that causes a gaussian variance of 0.05 to occur every second. The system was still able to achieve almost the same desired y output and x output; however, still failed to achieve the desired theta location since α and β never achieved steady state which is shown in figure 7. The system was overall less sensitive to noise that was applied input versus noise applied at the output.

In order to demonstrate when a disturbance was applied to the model the UV was designed as a delivery truck. At 100 seconds, the UV picked up the delivery which had a total weight of 0.5 kg. When the disturbance was added it had little effect on the overall system dynamics. As shown in figure 9, the system appears to act almost exactly as it does with no load since ρ and β do trend to zero but α reaches a steady state value that is no longer zero. This does not appear to have much effect on the x and y destination coordinates as they are [10 17.98] after 400 seconds. However, since alpha did not have a setting back to zero, the max theta that was achieved was only 97.6628°. After the investigation of many other loads, it does not seem that this disturbance will cause much effect. When the system was subject to a load over half its weight (1 kg) there is still a small variation to the desired location - [10 17.98 95.56°].

References

[1] Siegwart, R. Y., & Nourbakhsh, I. R. (2004). Introduction to autonomous mobile robots. Cambridge, Mass: MIT.
[2] Brogan, W. L. (1974). Modern Control Theory, Third Edition. Quantum Publishers.
[3] Kelley. (n.d.). Coordinate Transformation of Vector Components. Solid Mechanics Part III