Cooper Union for the Advancement of Science and Art

Implementing Guidance, Navigation, and Control System for Missile Interception

ME-451: Modern Control

Authors:
Khushant Khurana

Professor:
Professor Luchtenburg

December 16, 2023

Contents

1 Motivation

Guidance, Navigation, and Control (GNC) System allows complicated systems to estimate its state, determine its next trajectory, and steps to achieve that next trajectory. Air crafts, satellites, cube sats, missiles and other automated vehicles use GNC to guide themselves to the target. In this paper, a GNC model is implemented for a missile to successfully intercept a target projectile.

2 Model Assumptions and Simplifications

To simplify the model, only longitudinal dynamics of the missile are analyzed. Accordingly, the height of the missile and distance along the ground are the only elements of interest. This is referred to as "short period approximation." Through this approximation, the longitudinal dynamics are determined at the equilibrium where thrust effects are considered to be zero. Accordingly, the linear velocity is assumed to be constant. Moreover, yaw and pitch velocities also assumed to be zero to avoid the coupling between lateral and longitudinal dynamics. The relative position of the missile with respect to the target from *MAVSIM Public Repository* is shown in Figure 2.1. Furthermore, the air mass relative flight path angle for longitudinal dynamics is neglected as seen in Figure 2.2. Accordingly, the flight path angle is calculated as using Equation 1.

$$\gamma = \theta - \alpha \tag{1}$$

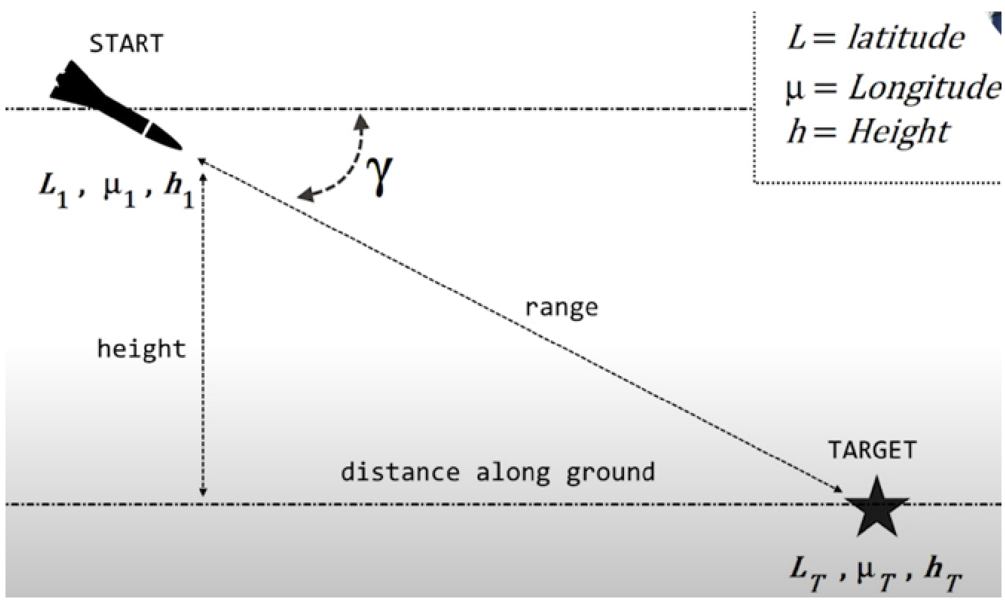


Figure 2.1: 2D diagram of the missile with respect to the target

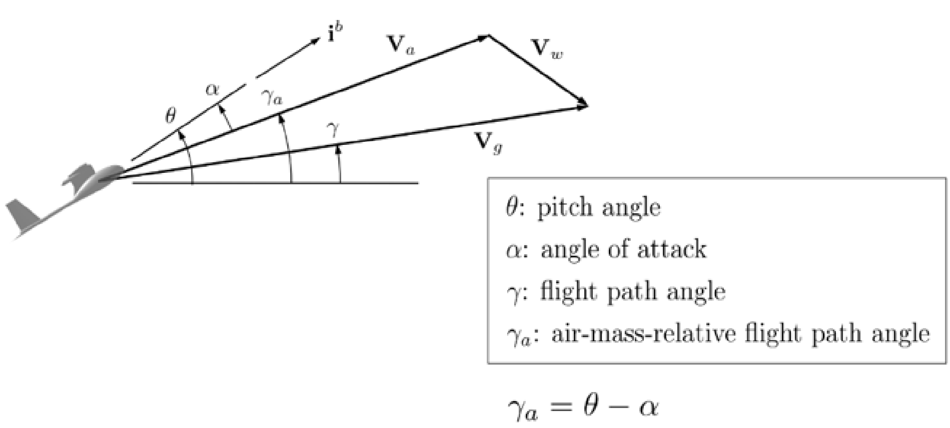


Figure 2.2: Elements of interest in the longitudinal dynamics.

3 Missile Dynamics

The corresponding state space matrices are taken from *Missile Longitudinal Autopilots: Connections Between Optimal Control and Classical Topologies* published by Raytheon Technologies. The symbolic form is shown in 3.1a and the magnitudes for the corresponding symbols are shown in Figure 3.1. The actual model with numbers are shown in Equation 2 and the corresponding poles of the plant are shown in Equation 3. Clearly, the system is unstable and hence needs a controller to stabilize.

$$A = \begin{bmatrix} \frac{1}{V_{m_0}}\left[\frac{\bar{Q}SC_{z\alpha_0}}{m} - A_{X_0}\right] & 1 \\ \frac{\bar{Q}SdC_{m\alpha_0}}{I_{YY}} & 0 \end{bmatrix} \qquad B = \begin{bmatrix} \frac{\bar{Q}SC_{z\delta_{p_0}}}{mV_{m_0}} \\ \frac{\bar{Q}SdC_{m\delta_{p_0}}}{I_{YY}} \end{bmatrix}$$

$$C = \begin{bmatrix} \frac{\bar{Q}SC_{z\alpha_0}}{mg} - \frac{\bar{Q}SdC_{m\alpha_0}\bar{x}}{gI_{YY}} & 0 \\ 0 & 1 \end{bmatrix} \qquad D = \begin{bmatrix} \frac{\bar{Q}SC_{z\delta_{p_0}}}{mg} - \frac{\bar{Q}SdC_{m\delta_{p_0}}\bar{x}}{gI_{YY}} \\ 0 \end{bmatrix}$$

(a) Missile state space dynamics in symbolic form

Variable	Value	Units	Description
V_{m_0}	3350	ft/sec	Total Missile Velocity
m	11.1	slug	Total Missile Mass
I_{YY}	137.8	slug-ft^2	Pitch Moment of Inertia
$\bar{x}$	1.2	ft	Distance from CG to IMU Positive Forward
A_{X_0}	-60	ft/sec^2	Axial Acceleration Positive Forward
$C_{z\alpha_0}$	-5.5313	-	Pitch Force Coefficient due to Angle of Attack
$C_{m\alpha_0}$	±6.6013	-	Pitch Moment Coefficient due to Angle of Attack
$C_{z\delta_{p_0}}$	-1.2713	-	Pitch Force Coefficient due to Fin Deflection
$C_{m\delta_{p_0}}$	-7.5368	-	Pitch Moment Coefficient due to Fin Deflection
$\bar{Q}$	13332	lb/ft^2	Dynamic Pressure
S	0.5454	ft^2	Reference Area
d	0.8333	ft	Reference Length
g	32.174	ft/sec^2	Gravity Constant

(b) Magnitude of the missile parameters.

Figure 3.1: Aspect ratio for the generated mesh.

$$A = \begin{bmatrix} -1.064 & 1 \\ 290.26 & 0 \end{bmatrix} \qquad B = \begin{bmatrix} -0.25 \\ -331.40 \end{bmatrix}$$
$$C = \begin{bmatrix} -123.34 & 0 \\ 0 & 1 \end{bmatrix} \qquad D = \begin{bmatrix} -13.51 \\ 0 \end{bmatrix} \tag{2}$$

$$\text{Open Loop Poles} = [-17.5, 16.5] \tag{3}$$

The state vector and the input are shown in Equation 4

$$x = \begin{bmatrix} \alpha & \text{angle of attack} \\ p & \text{pitch velocity} \end{bmatrix}, \quad u \quad = \begin{bmatrix} \sigma_p & \text{torque around the pitch axis} \end{bmatrix}. \tag{4}$$

4 Designing the Navigation System

4.1 Implementing An Observer

The navigation system is designed by implementing an observer and linear quadratic regulator on the estimated states to control the actual states. Since the plant model is unknown, noise is added to the state space model to make the plant model. The corresponding plant model us shown in Figure 4.1 and the noise added to the system is shown in Figure 4.2.

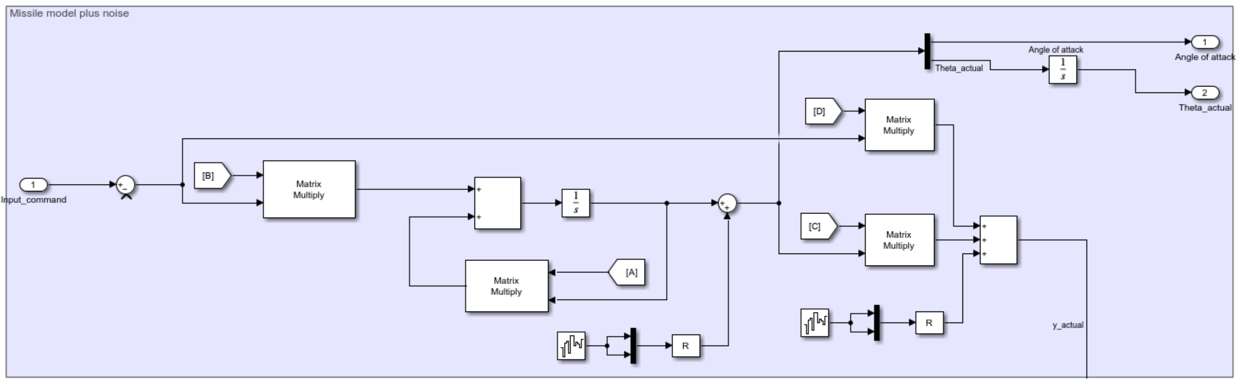


Figure 4.1: Implementing plant model by adding noise to the estimated model.

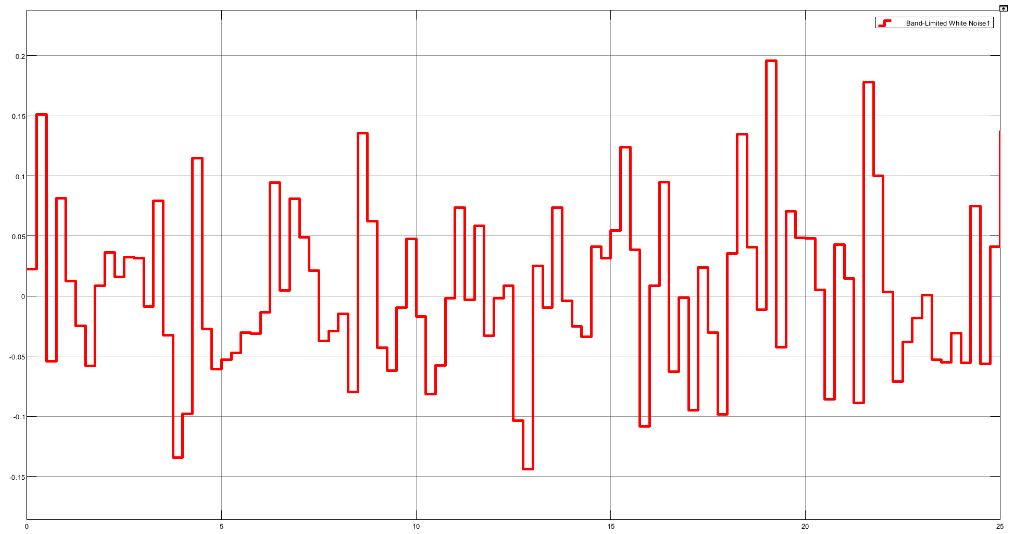


Figure 4.2: Band limited white noise added to the plant model as sensor and process noise.

The synthesis of the observer and the the model is shown in Figure 4.3. The dynamics of the observer is shown in Equation 5

$$\dot{x} = Ax + Bu + L(y_{\text{actual}} - y_{\text{estimated}}) \tag{5}$$

The Kalman gain, also referred to as observer gain, is modelled to allow the error to go to infinity. Using the gain L, the derivative of the error between actual and expected output of the system can be modelled using Equation 6. Accordingly, pole placement can be used to determine the gain L. The rule of thumb is to have the poles of the error matrix to be 10 times those of the plant and they should be in left half plane to send the error to zero.

$$\dot{e} = Ax - CLx \tag{6}$$

As mentioned below, the estimated states are controlled to adjust the true states using LQR. The corresponding Q and R matrices are shown in Equation 7. The reason behind choosing a smaller weight for the second state element - pitch velocity - is to allow for faster changes. A higher weight would have tried to conserve the pitch velocity hence not allowing for big changes in the flight path angle. Since that would constrain the missile's ability to follow extremely curved trajectories, a lower weight would be optimum. Finally the LQR gains are determined using LQR command from Python's control module.

$$Q = \begin{bmatrix} 1 & 0 \\ 0 & 0.1 \end{bmatrix}, \quad R \quad = [1] \tag{7}$$

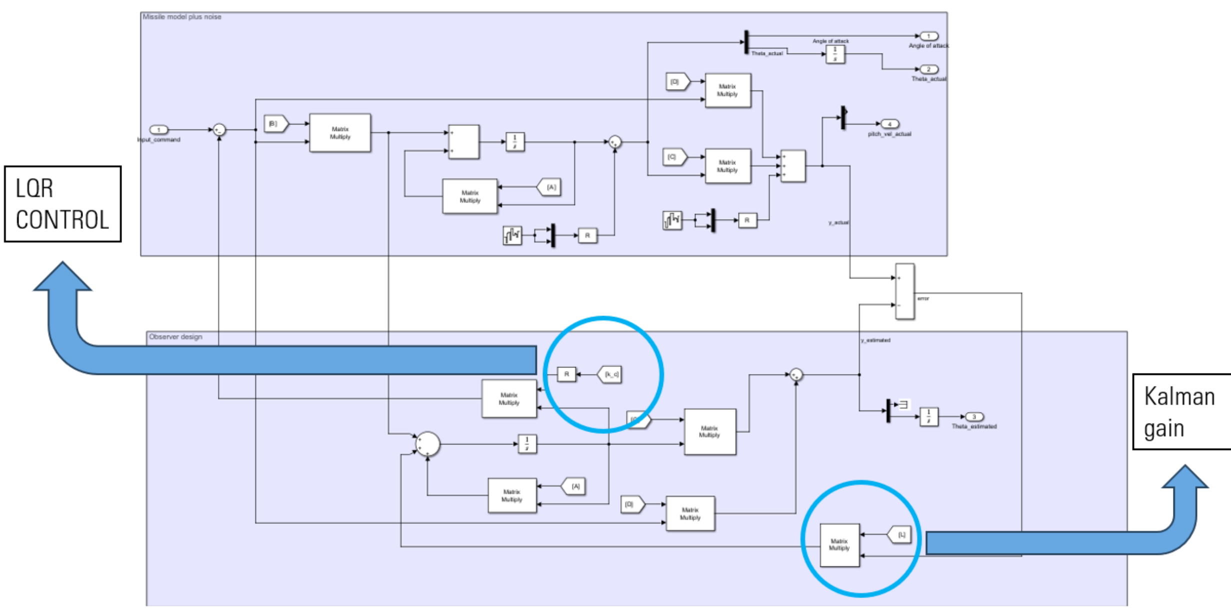


Figure 4.3: Implementing the complete navigation system. The Kalman gain and LQR controller are highlighted in the figure.

4.2 State Estimation Unit Test

After implementing the navigation system, a unit test was performed by analyzing a step response of the system as shown in Figure 4.4 and Figure 4.5 demonstrates the tracking abilities of the navigation system despite the noise. A PID controller with gains K = -1 is used to act as the outer loop control and the response reaches a steady state value of approximately 1. The steady state error can be reasoned to come from the process and sensor noise added to the system.

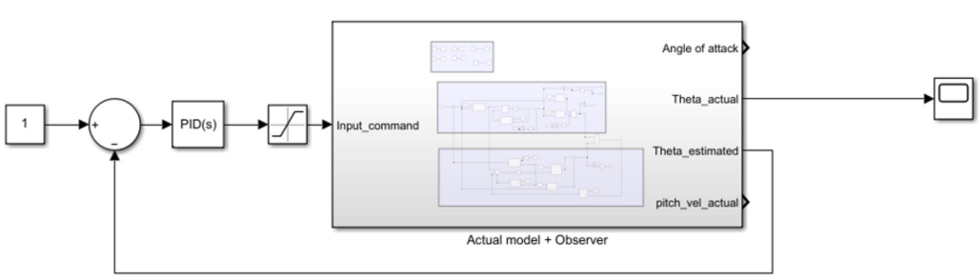


Figure 4.4: Implementing a step response on the navigation system.

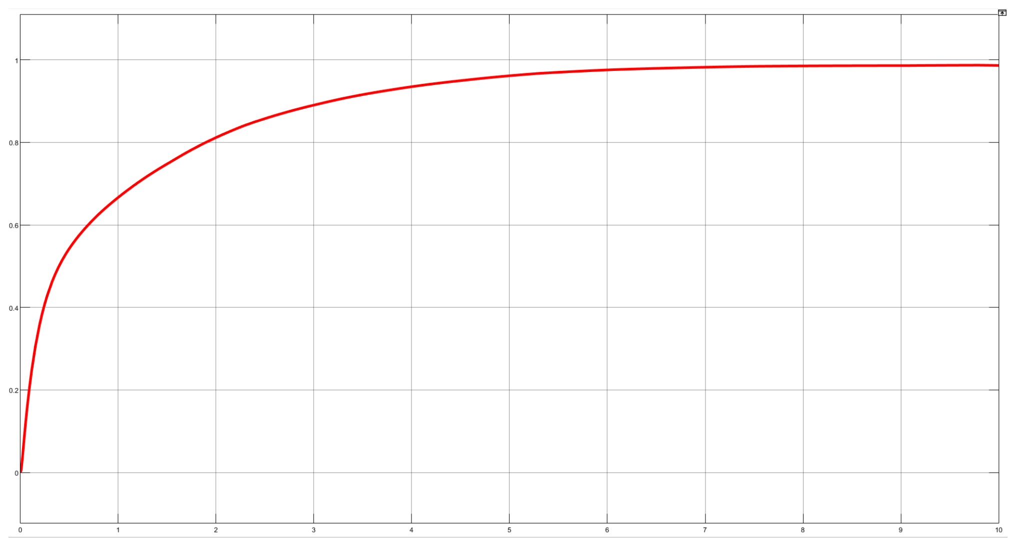


Figure 4.5: Result of the step response. It can be seen that navigation system reaches steady state value of approximately 1.

5 Designing the Guidance System

5.1 Determining the Position in 2D Plane

In order to develop the guidance model, it is crucial to know the current position of the missile. Using Equation 1, the pitch angle of the missile is determined and is multiplied by the velocity components of the missile. A relatively large number, 200 ft/sec, is chosen to allow for a large lead angle. It is to be remembered that the vertical velocity is considered to be negative to match the NED (north-east-down) frame. Finally, to get the position, integration is performed on the vertical and horizontal velocity components. An implementation of this logic is shown in Figure 5.1

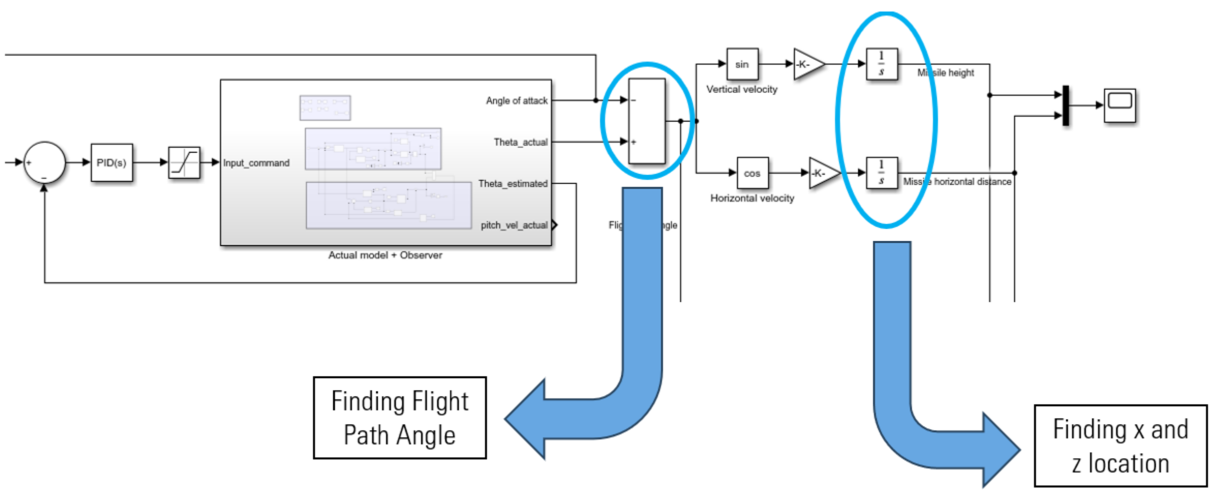
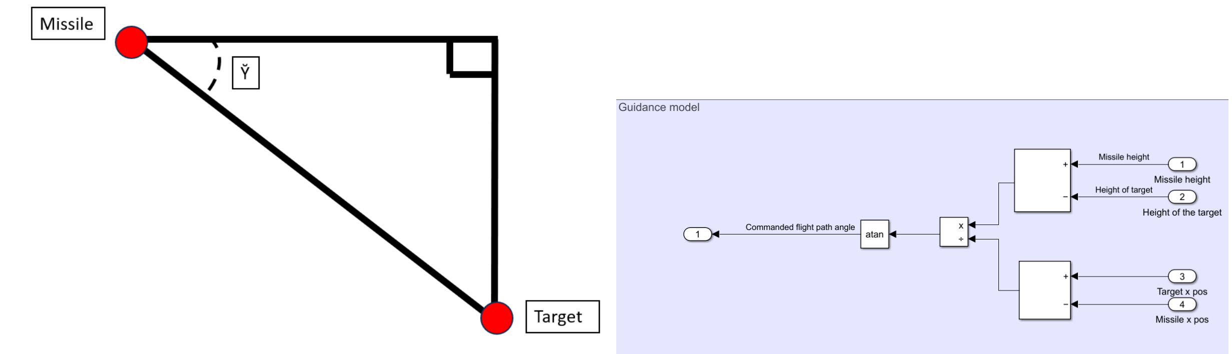


Figure 5.1: Getting missile position from the flight path angle and velocity.

5.2 Implementing Mathematics Used For Guidance

The flight angle is calculated using basic trigonometry on the right triangle formed by the origin, target position, and missile position.. The mathematical model to do so is shown in Equation 8 and its implementation in Simulink is shown in Figure 5.2

$$\gamma = \frac{\text{Missile height - Target height}}{TargetXlocation - MissileXlocation} \tag{8}$$



(a) Trigonometry Used to Determine Desired Flight Path Angle

(b) Implementation of the Guidance Math in Simulink

Figure 5.2: Implementing the Guidance System

6 Designing the High Level Controller

A standard PID controller is used as the outer loop control. The constants, shown in Equation 9 are tuned through trial and error.

$$K_P = -3, \quad K_I = 0.009, \quad K_D = 0.4 \tag{9}$$

7 Implementing Target Projectile Dynamics

A simple projectile with both vertical and horizontal velocity equalling 100 m/sec is implemented to be the target. An acceleration vector of [0, -9.81] is integrated twice to get the position. It is to be remembered that since the vertical projectile velocity is implemented to be positive, the acceleration due to gravity needs to be in the opposite direction and hence a negative sign is added before 9.81. An implementation of the target dynamics is done in Figure 7.1

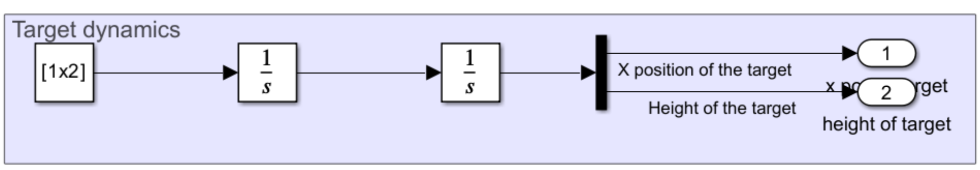


Figure 7.1: Implementation of the target projectile dynamics in Simulink

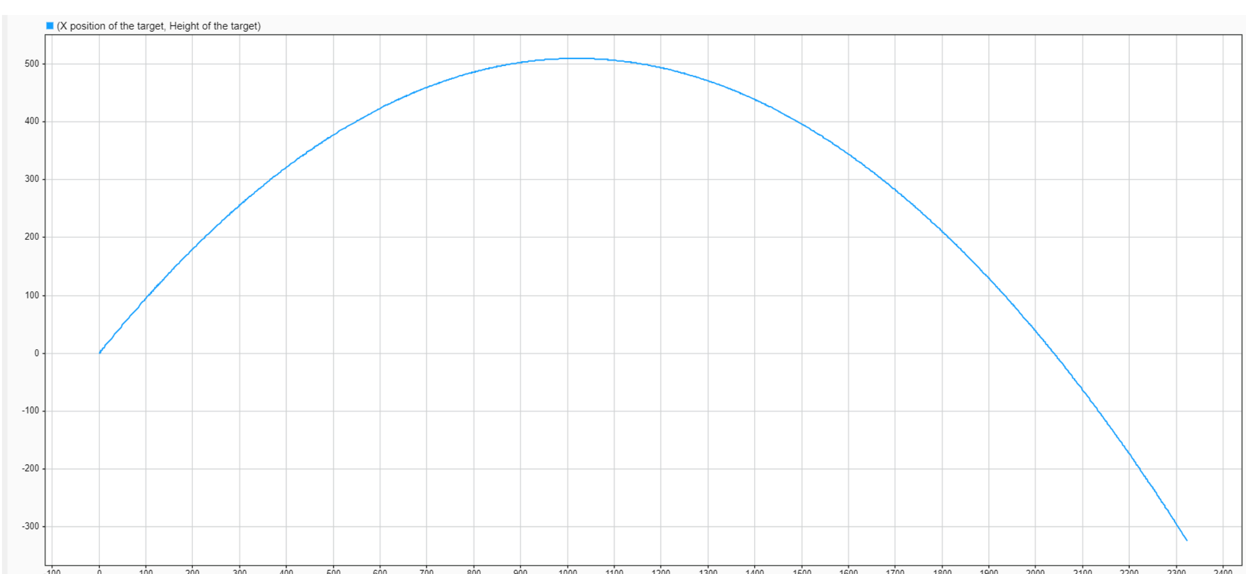


Figure 7.2: Height of the target as a function of time.

8 Integrating The Three Models and Results

A synthesis of the the guidance, navigation, and control subsystems is shown in Figure 8.1. Using the initial conditions of [0 0] for both the missile and the target, the results are determined. As seen in Figure 8.2, the missile intercepts the target projectile 6 times. However, there is a constant steady state error after the target projectile is on its descent. Initially, it is assumed that since the missile and the target both start at 0 m, the denominator of the guidance model, Target x location - Missile x location, becomes zero and hence the model goes to infinity. This assumption is tried by giving different initial conditions however it didn't solve the problem.

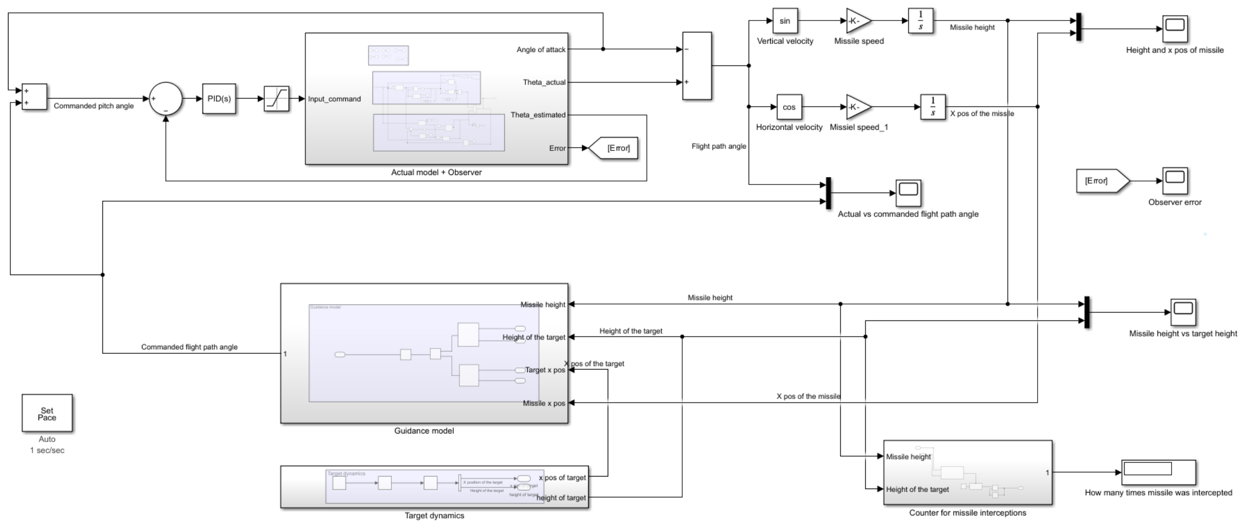


Figure 8.1: Integrating the Guidance, Navigation, and Control Models

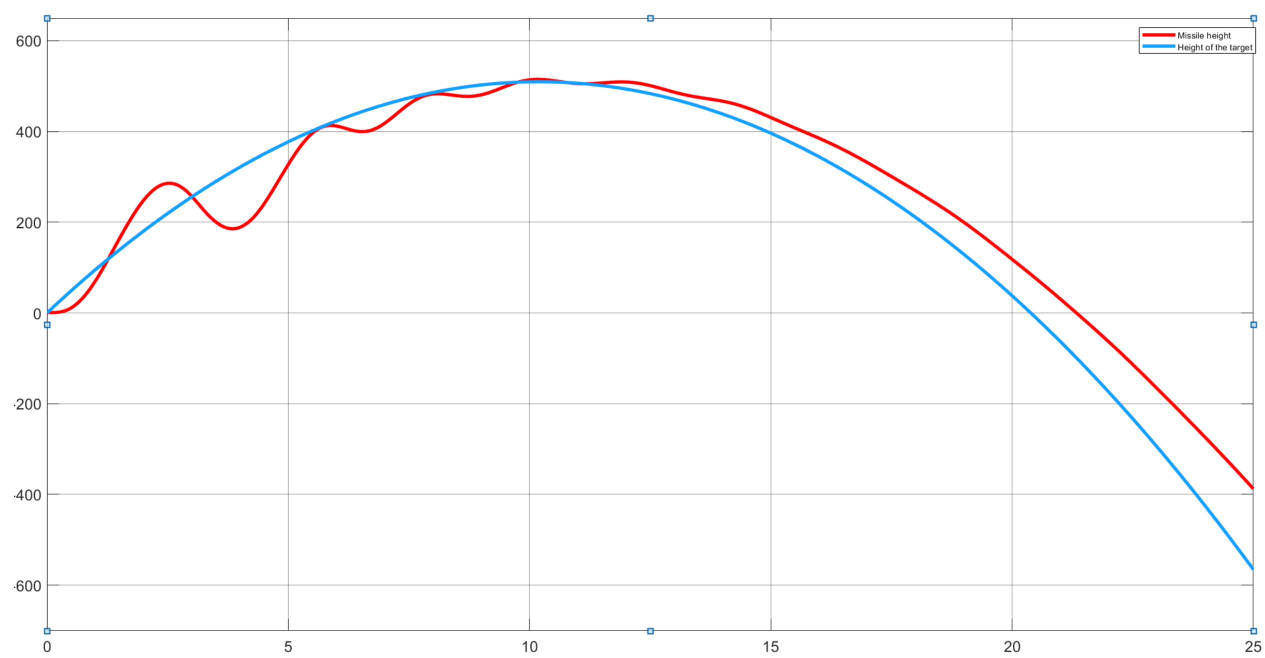


Figure 8.2: Missile height vs target height as a function of time. The graph shows the missile interceptions of the target.

The commanded and flight path angle is also plotted versus time in Figure 8.3. As expected, the commanded flight path angle goes to infinity because the missile and the target start at the same x location. The flight path angle of the missile starts at 0 rad because the model starts at the equilibrium position which is horizontal. This creates a phase delay between the actual and commanded flight path angle but the steady state error goes to zero as the model progresses. Finally, the error between the actual and estimated output of the navigation system is shown in Figure 8.4. The first signal in blue shows the error between the change in angle of attack and second signal shows the error between the pitch velocity.

The PID controller is tuned with different constants to allow for the missile to perfectly track the projectile in its descent however, it didn't help. The next steps for the model might be to implement a higher fidelity model for the navigation system and not use the short period approximation. Regardless, the GNC model works well and the missile intercepts the target six times.

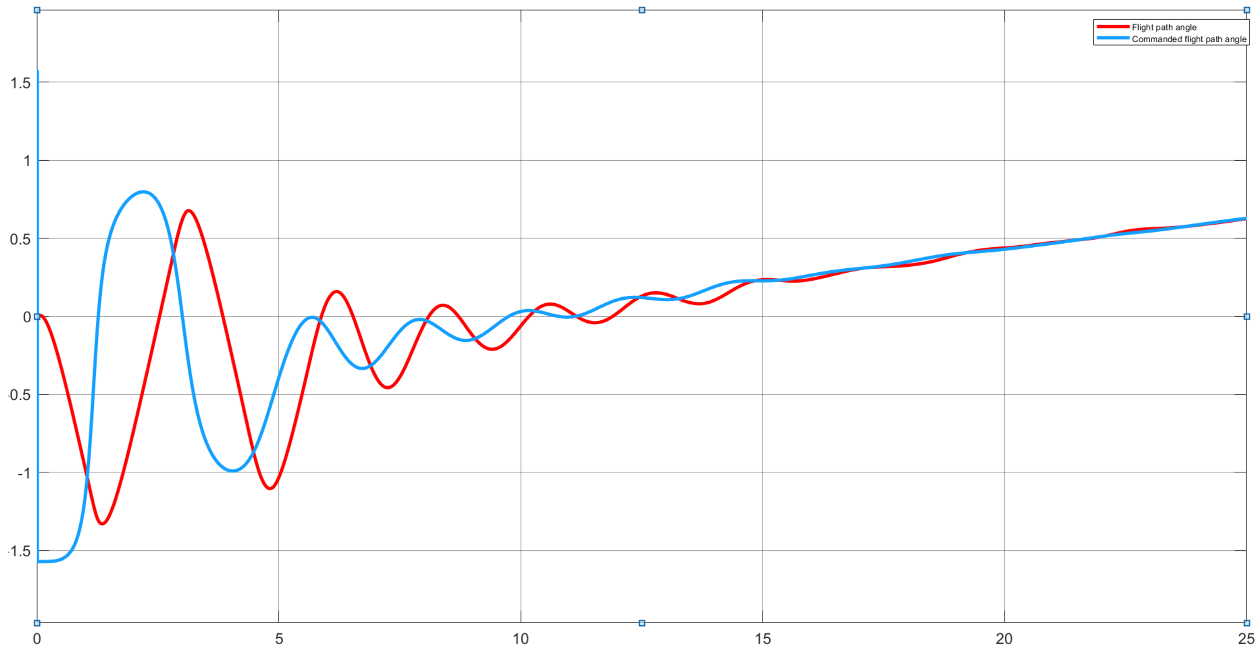


Figure 8.3: The commanded and actual flight path angle over time.

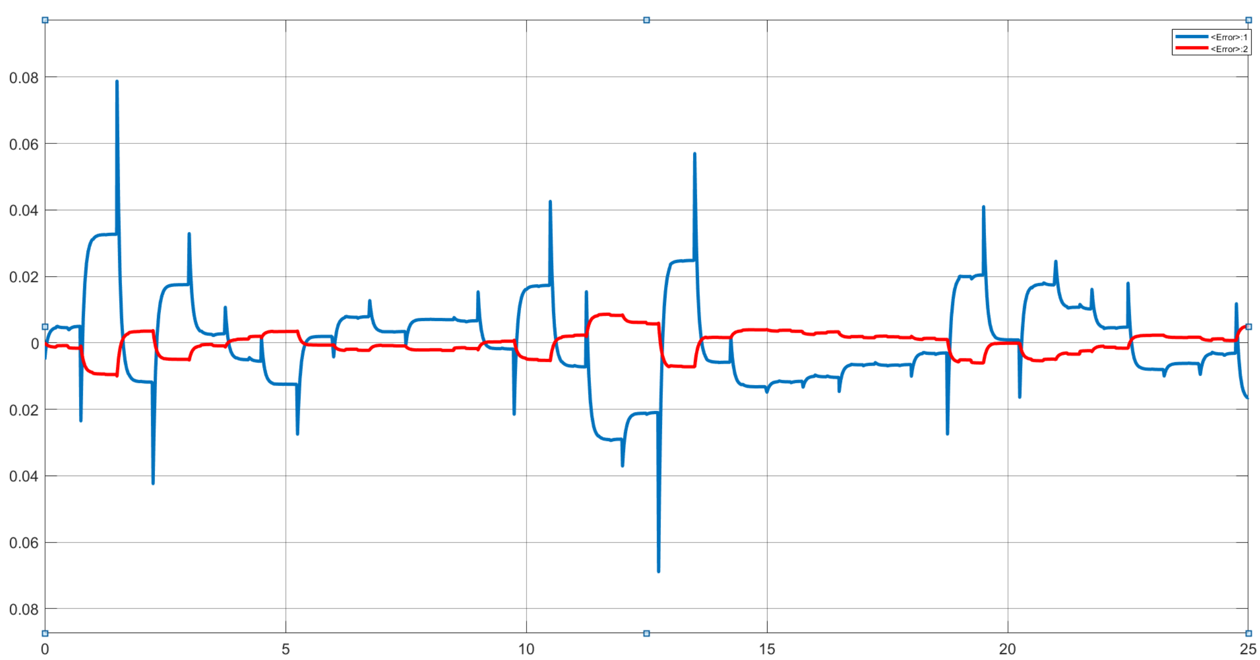


Figure 8.4: Error between the actual states and estimated states. 1st signal is the error for angle of attack and the second signal is the error for pitch velocity.

References

[1] R. Beard, *Mavsim public repository*, GitHub, https://github.com/randybeard/mavsim_public.

[2] Raytheon Technologies, *Missile Longitudinal Autopilots: Connections Between Optimal Control and Classical Topologies*. AIAA, 2005, ASHRAE Handbook: Refrigeration. [Online]. Available: https://www.researchgate.net/profile/Curtis-Mracek/publication/303256153.