ASE 479W: Measurement Simulation and State Estimation

Victor Branea

February 14, 2023

1. Introduction

This lab introduced implementations for a GNSS receiver, camera, and inertia measurement unit (IMU). In order to simulate real-world noise, the sensor implementations' output had random zero mean gaussian noise added to them. These outputs where then passed through a state estimator to infer the quad's state, which was then used to control the quadcopter.

2. Theoretical Analysis and Implementation

GNSS Measurement Simulation

The gnssMeasSimulator() function was written to take as inputs the quad's state at time k along with the sensorParams structure and output the measured position of the primary antenna and that of the secondary antenna with respect to the primary antenna. The second output is very interesting, as it was implemented using a special covariance matrix which assured a noise vector perpendicular to the true vector. This matrix looks as follows:

$$R_{bG} = \|\vec{r}_{bG}\|^2 \sigma_b^2 [I_3 - \vec{r}^u_{bG}(\vec{r}^u_{bG})^T] \quad (1)$$

It can be shown that this matrix is of rank 2, regardless of the unit vector $\vec{r}^u_{bG} = \begin{bmatrix} x \\ y \\ z \end{bmatrix}$. As the terms in front of the matrices are scalars, it must be true that:

$$\det(I_3 - \vec{r}^u_{bG}(\vec{r}^u_{bG})^T) = 0 => \begin{bmatrix} 1-x^2 & -xy & -xz \\ -xy & 1-y^2 & -yz \\ -xz & -yz & 1-z^2 \end{bmatrix} = 0 \; => 1 - x^2 - y^2 - z^2 = 0$$

Which is true, as $\vec{r}^u_{bG}$ is a unit vector so the sum of the squares of its components must equal 1. Next, we can form three convenient 2x2 matrices and calculate their determinants:

$$det\begin{bmatrix} 1-x^2 & -xy \\ -xy & 1-y^2 \end{bmatrix} = 1 - x^2 - y^2, \; det\begin{bmatrix} 1-x^2 & -xz \\ -xz & 1-z^2 \end{bmatrix} = 1 - x^2 - z^2,$$

$$det\begin{bmatrix} 1-y^2 & -yz \\ -yz & 1-z^2 \end{bmatrix} = 1 - y^2 - z^2, \text{ and by adding them together we get:}$$

$$det\begin{bmatrix} 1-x^2 & -xy \\ -xy & 1-y^2 \end{bmatrix} + det\begin{bmatrix} 1-x^2 & -xz \\ -xz & 1-z^2 \end{bmatrix} + det\begin{bmatrix} 1-y^2 & -yz \\ -yz & 1-z^2 \end{bmatrix} =$$

$$= 3 - 2(x^2 + y^2 + z^2) = 1$$

Therefore, at least one of the three determinants is nonzero, resulting in the fact that the original matrix is of rank 2.

Camera Measurement Simulation

In this section the location of known features were projected onto the camera's image plane and it was checked whether the camera can see them, meaning if the features were in front of the image plane and would land within the image plane's border.

An interesting problem that arose in the lab handout was to show that the cross product of two line vectors gives the coordinates for their intersection. To prove this, say we have two lines:

$ax + by + c = 0$ and $dx + ey + f = 0$ with vectors $l_1 = \begin{bmatrix} a \\ b \\ c \end{bmatrix}, l_2 = \begin{bmatrix} d \\ e \\ f \end{bmatrix}$, so that

$l_1 \times l_2 = \begin{bmatrix} bf - ce \\ cd - af \\ ae - bd \end{bmatrix} => x = \begin{bmatrix} \frac{bf-ce}{ae-bd} \\ \frac{cd-af}{ae-bd} \end{bmatrix}$. This is the same answer that can be derived by writing y as a function of x in both cases, finding the unique value for x, and finding the value for y with said value of x.

IMU Measurement Simulation

This section was divided into two subsections. The first one was concerned with the acceleration measurement. This depends on the location of the IMU with respect to the center of mass. This can be shown starting from the relation of a vector in the inertial frame to the same vector in the body frame.

$\overrightarrow{l_B} = R_{BI}\overrightarrow{l_I} => \dot{\overrightarrow{l_B}} = R_{BI}\dot{\overrightarrow{l_I}} - \overrightarrow{\omega_B} \times \overrightarrow{l_B} =>$

$\ddot{\overrightarrow{l_B}} = R_{BI}\ddot{\overrightarrow{l_I}} - \dot{\overrightarrow{\omega_B}} \times \overrightarrow{l_B} - 2\left(\overrightarrow{\omega_B} \times \dot{\overrightarrow{l_B}}\right) - \overrightarrow{\omega_B} \times \left(\overrightarrow{\omega_B} \times \overrightarrow{l_B}\right)$ as the point is fixed in B, we get the desired equation:

$$a_I = \ddot{\overrightarrow{R_I}} + R_{BI}^T\left(\overrightarrow{\omega_B} \times \left(\overrightarrow{\omega_B} \times \overrightarrow{l_B}\right) + \dot{\overrightarrow{\omega_B}} \times \overrightarrow{l_B}\right)$$

For the second part, the gyro, it is unnecessary to include the lever arm, as the rate of rotation is the same throughout a rigid body. As the rate of rotation is the change in angle over time, and any point in a rigid body will not change it's position with respect to the axis of rotation, the angle between the vector pointing to a specific point A in the rigid body at two different times will be the same regardless of how far away the point is from the axis of rotation.

Wahba's problem

In order to get a reasonable first estimate of the altitude, a singular value decomposition solution was used to Wahba's problem. The function wahbaSolver() was tested using a Monte Carlo simulation over 1000 iterations in order to graph the errors between the solution provided by this function and the known Euler angles used to compute the inputs that go into the function.

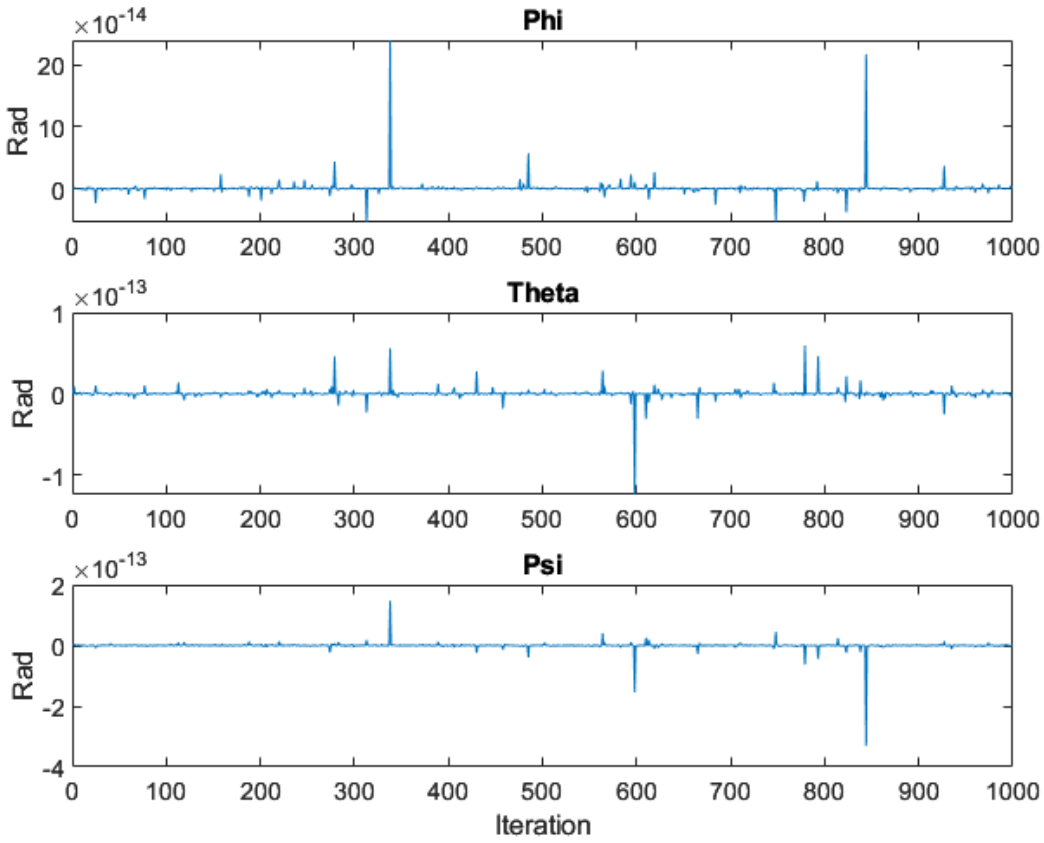

Figure 1 Results from Monte Carlo simulation

As can be seen in figure 1, the errors between the known Euler angles and the Euler angles provided by the wahbaSolver() function are very small.

3. Results

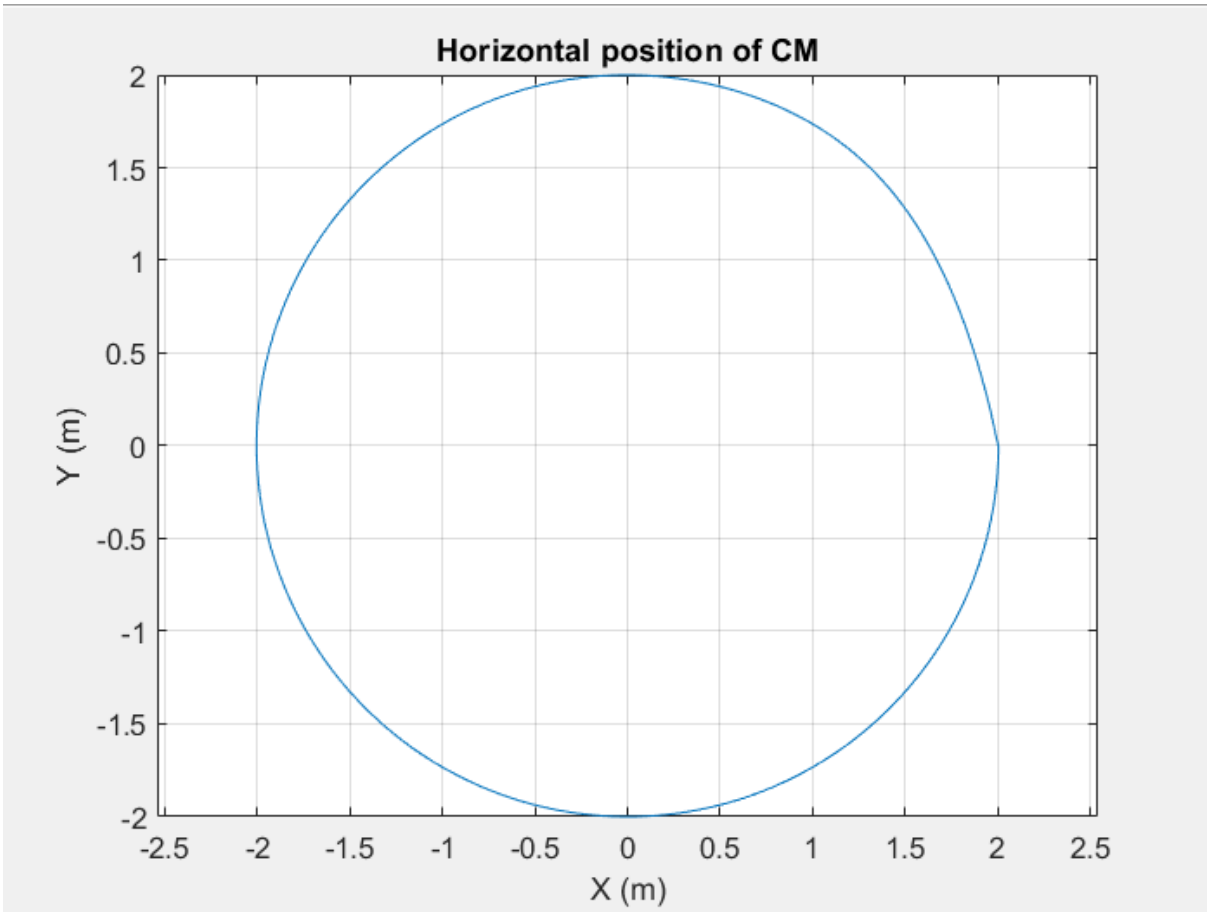

Figure 2 Horizontal position

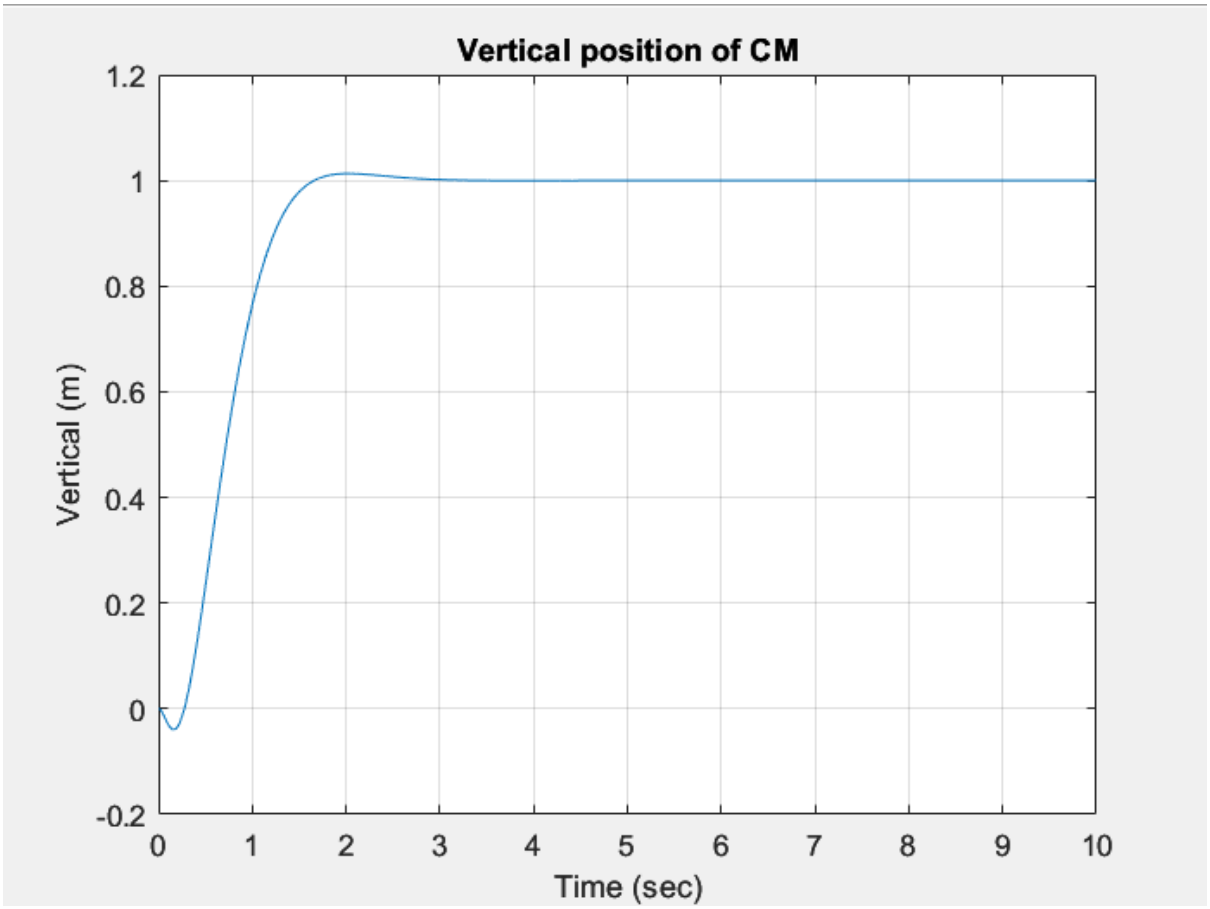

Figure 3 Vertical position

As can be seen in figures 2 and 3, the quad's true position is almost identical to a perfect circle and is level for most of the flight.

When the noisy measurements are used, the quad mirrors the circular path quite closely, within a few centimeters. This can be seen across a multitude of attempts using rng('shuffle'), as can be seen in figures 4, 5, and 6.

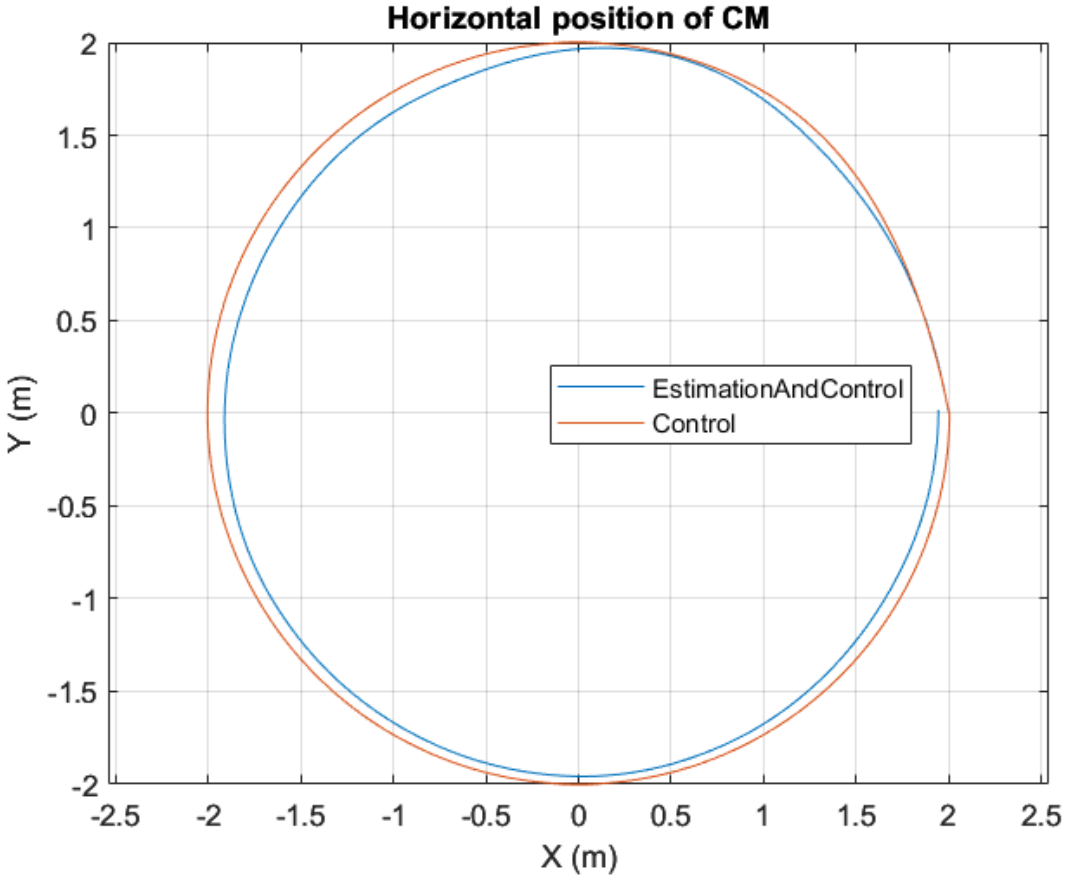

Figure 4 Comparison between EstimationAndControl and Control

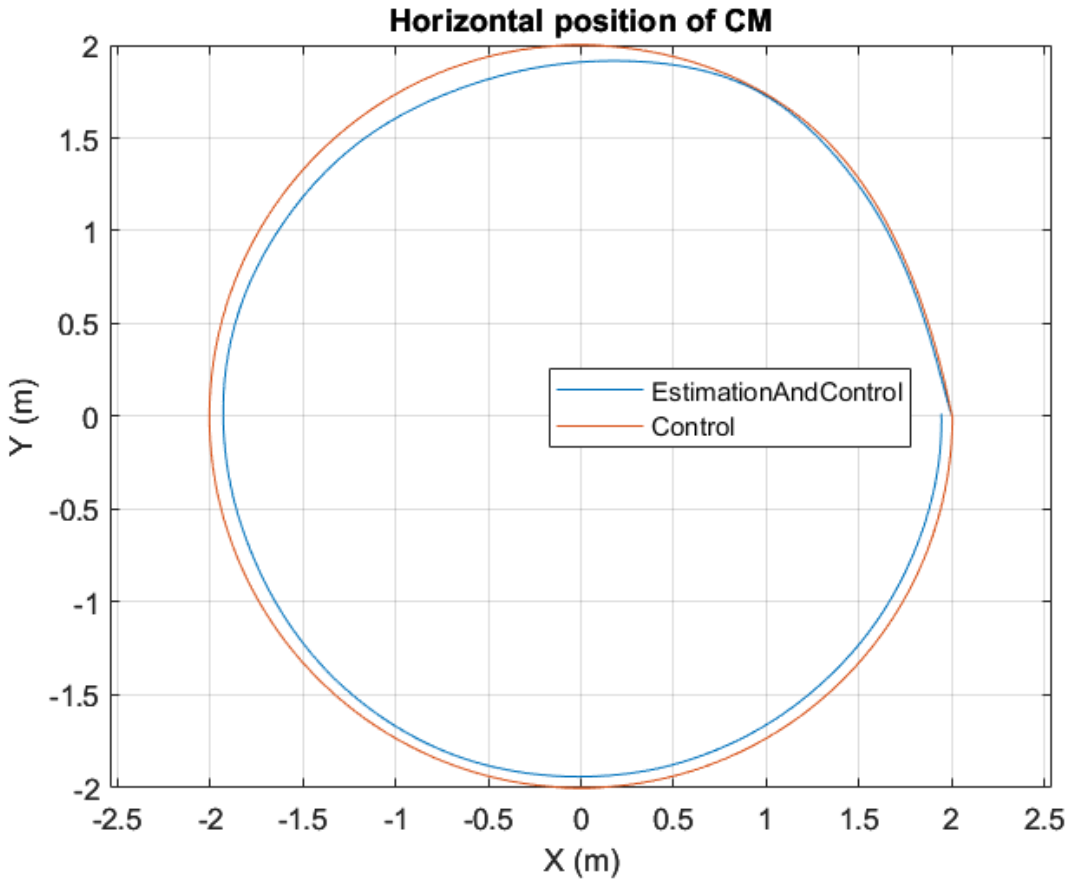

Figure 5 Comparison between EstimationAndControl and Control

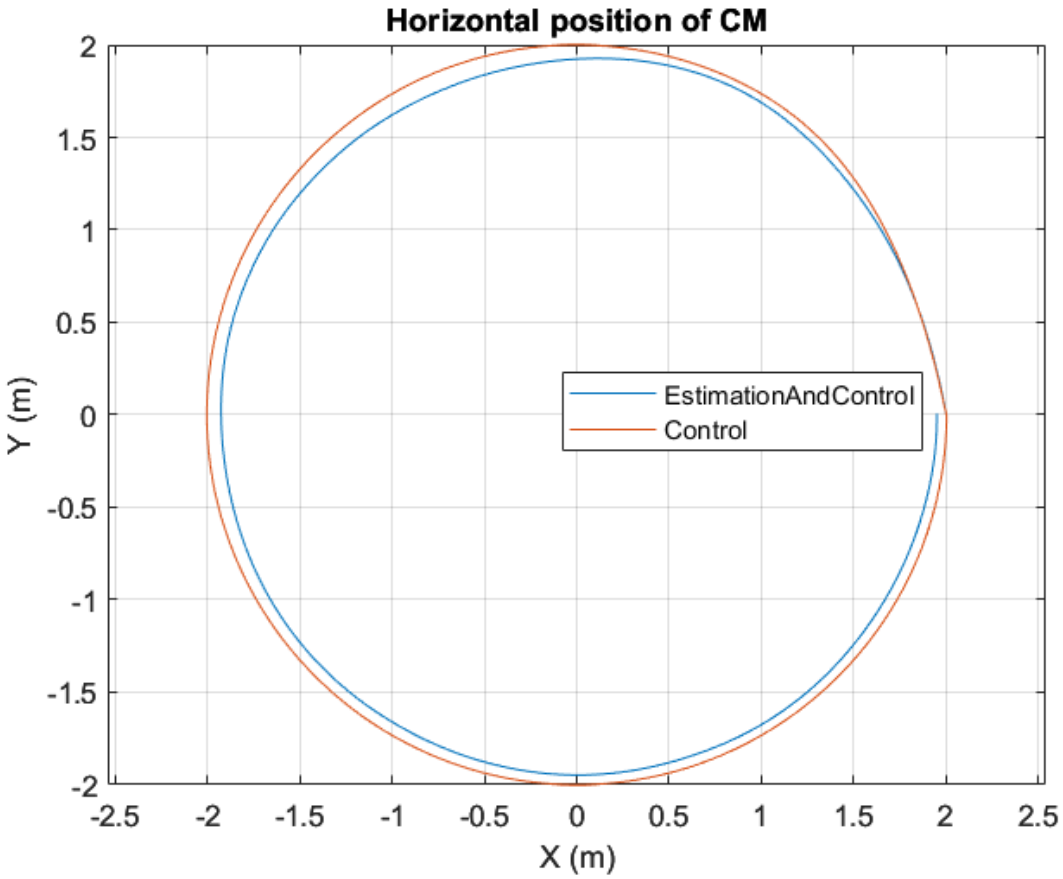

Figure 6 Comparison between EstimationAndControl and Control

Finally, the noise-less simulation was compared to the simulation using the sensor implementations, but with the camera shut off. As can be seen below, in figure 7, the camera being turned off doesn't change the final trajectory by much.

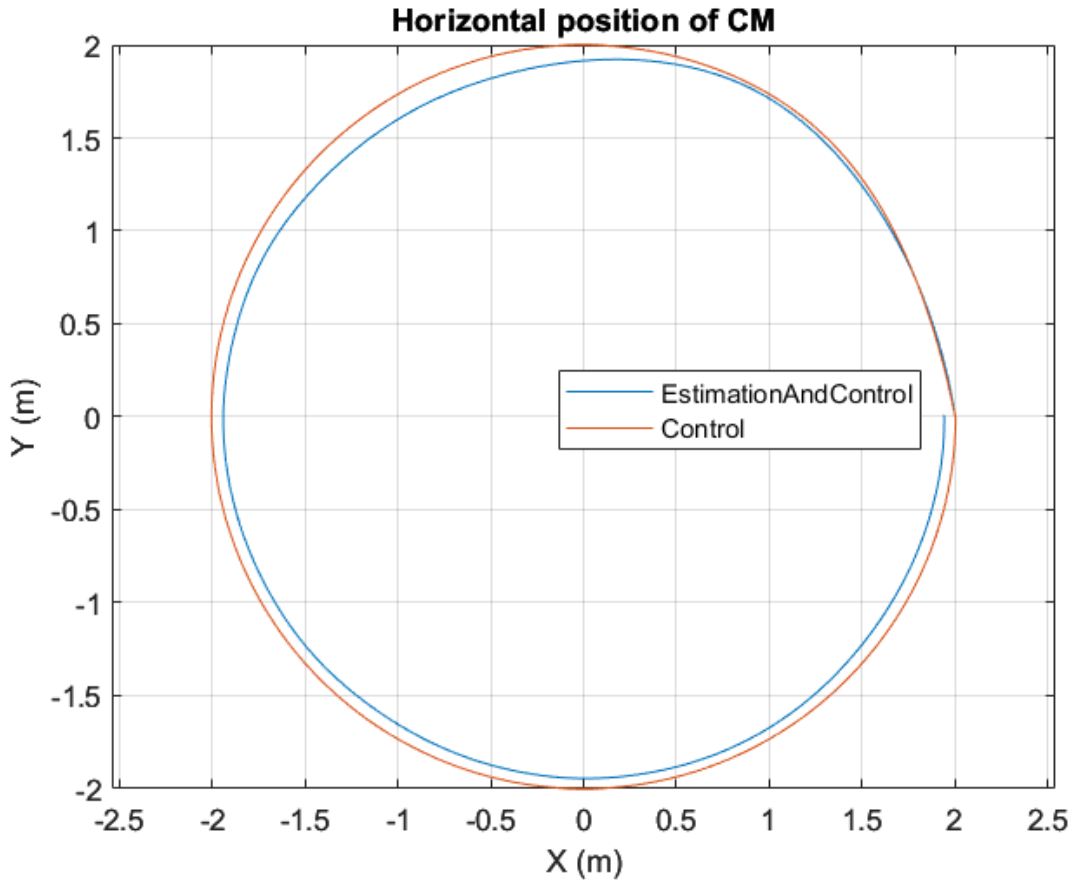

Figure 7 Comparison between true trajectory and trajectory from sensors with camera turned off

4. Conclusion

In conclusion, code was implemented to simulate noisy measurements from sensors and compared to the noise-less simulation. The two simulations were found to be very similar.