Extended Kalman Filter Based Linear Quadratic Regulator Control for Optical Wireless Communication Alignment

November 26, 2024

Tim Sweeney
tsweene1@umd.edu

Grayson Gilbert
ggilbert@umd.edu

Abstract

This paper aims recreate and verify the results put forth by Alalwan et al. in the paper *Extended Kalman Filter Based Linear Quadratic Regulator Control for Optical Wireless Communication Alignment* [1]. In the original paper Alalwan et al. proposes a new state space model for the control of communication between two underwater robots within an Underwater Wireless Optical Communication System. The new state model uses three state variables compared to the usual two state model [8], allowing for a more precise and robust method of control. The model contains an Extended Kalman Filter (EKF) for optimal state estimation paired with a Linear Quadratic Regulator (LQR) controller to stabilize the communication links between robots. Throughout this paper, the entire system model will be re-derived from the dynamic model, to the EKF, and also the LQR controller. The results in the original paper will then attempt to be recreated using MATLAB. The simulation verification will be broken down into multiple steps, first verifying the EKF state estimation, then verifying the LQR's ability to control the desired state variables, and finally combining the two to form the full system model.

1 Introduction

In today's day and age where people and devices are more connected than ever, establishing fast, secure, and reliable methods of data transfer has become an essential requirement within society. One method that has gained popularity over the last decade is Optical Wireless Communication (OWC). Compared to traditional radio frequency systems, OWC can provide high security and high data rate communication links, for a low cost to manufacture [3]. Visible Light Communication (VLC), which falls under the umbrella of OWC, is frequently used in underwater applications. VLC is also able to provide the high security and data rate communication links found in OWC networks [8] [6]. It's a reliable, adaptable, and cost effective solution for the harsh underwater environment where it is often not easy to access systems for repair. The main challenge plagued by VLC systems is the high directionality of the communication signal which directly relates to the quality of the received communication link between Transmitter and Receiver. Additionally, environmental factors can further degrade the link quality by distorting, or otherwise degrading the signal intensity as it travels through the water [4]. Pointing, Acquisition, and Tracking (PAT) techniques were developed as a solution to maximize the received signal in OWC systems [5]. These techniques involve a closed-loop control system with sensing devices, controllers, and actuators, working together to maintain line of sight communication with the transmitters and receivers. Numerous research papers have been published in recent years diving deeper into the PAT techniques and use a variety of methods from Proportional Integral Derivative (PID) controllers, to Kalman Filtering based prediction systems. However, up to this point there have been no proposed solutions that provide a closed loop control system that is able to guarantee stability in the presence of external disturbances.

With that in mind, the selected paper will extend upon the control model given in [8] which outlines the alignment and control of optical sensors in an underwater communication network using the EKF. Compared against a Proportional (P), a Proportional Integral (PI), and a PID controller, the EKF-LQR controller developed in [1] demonstrated superior performance in stability, and robustness over the traditional methods. The proposed control solution therefore has shown to be a viable approach to controlling VLC networks in the highly dynamic underwater environment.

2 System Description And Dynamical Models

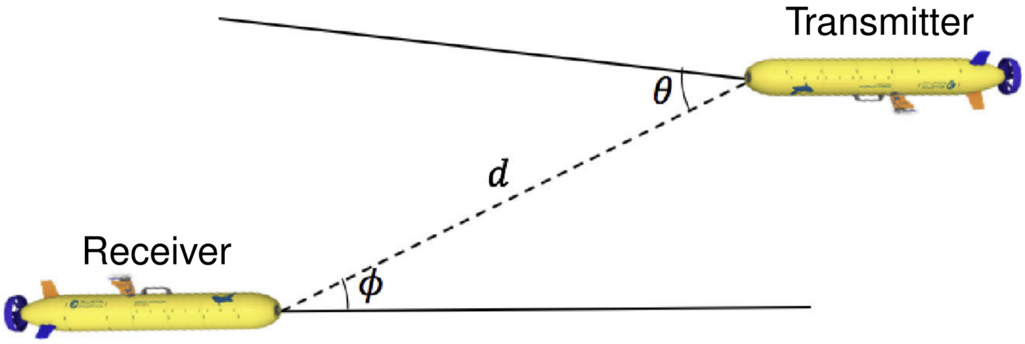

Figure 1: Diagram of orientation between transmitter and receiver.

Optical communication involves a transmitter and receiver that communicate over free space through the transmission and reception of photons. Signal is transmitted from the source, passed through the channel, and received at the receiver by converting the light signal to an electrical signal. This electrical signal is then sampled and digitized to recover the transmitted signal. Underwater Wireless Optical Communication (UWOC) is presented with different challenges than terrestrial Free Space Optical (FSO) communication because of the differences in physical properties of the channels. FSO channel models are not adequate representations of underwater channels, so new models need to be considered.

The light intensity model for this paper has been adapted from [2]. The model accounts for all of the components in the transmitter and receiver, such as lenses, LEDs, photo-diodes, amplifiers, and receiver circuitry. Figure 1 gives a visual of the orientation between the transmitter and receiver. θ represents the transmission angle, d represents the length of the channel, and ϕ represents the incident angle.

The intensity of the light from the transmitter I_θ is symmetric about the normal $\theta = 0$ and decreases as the angle θ increases. $I + \theta$ can be either measured empirically or taken from the transmitter data sheet. The spreading of the light is assumed to be spherical, and the resulting decay in intensity is the result of the inverse-square law.

$$I \propto \frac{1}{r^2} \tag{1}$$

where I is intensity and r is the radius of the sphere, or in this case, d. The clear ocean water assumption prevents light scattering from playing a major role in attenuation, so μ becomes solely based on absorption. The attenuation of the signal caused by absorption can be described by Beer's

Law as

$$\tilde{A} = exp(-cd) \tag{2}$$

Where $\tilde{A}$ is the loss of light caused by absorption, c is the attenuation coefficient of the medium the light moves through, and d is the distance the light travels in that medium. The intensity of light in model environment can then be described as

$$I = I_0 exp(-cd) \tag{3}$$

where the light intensity I, is equal to the initial intensity I_0, times the attenuation caused by absorption in 2. Taking 3 and multiplying by the inverse square law defined in 1 where $r = d$, will yield the irradiance reaching the receiver for a given distance d.

$$E_\theta(d) = I_\theta \ exp(-cd)/d^2 \tag{4}$$

The power of the signal reaching the receiver will be a function of the irradiance, $E_0(d)$, area of the receiver, A_0, and the angle of the receiver normal to the transmitter ϕ.

$$P_{in} = E_\theta(d) A_0 \ cos(\phi) \tag{5}$$

After the light reaches the receiver, it gets filtered and amplified within the digital signal processing circuitry of the system. The final signal model representing the real voltage of signal received is the following equation.

$$V_d = C_p I_\theta \ exp(-cd) cos(\phi)/d^2 \tag{6}$$

V_d is defined as voltage of signal received from the transmitter and C_p is a constant that depends on parameters within the system. Now that the system model is fully defined, we can move on to the state space representation and analysis.

2.1 State-Space Formulation and Analysis

The state space analysis proposed in this paper extends on the model that was put forth in [8]. This model had only two state variables, power at receiver and sensor angle and is defined as

$$x = \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = \begin{bmatrix} C_p I_\theta \ e^{-cd}/d^2 \\ \phi \end{bmatrix} \tag{7}$$

where x_1 is the power at the receiver and ϕ is the angle of the normal of the receiver to the light from the transmitter. This model is then extended upon to include the angular velocity as a third state variable to develop a force-velocity based controller. The advantage of including velocity in the state design is that the physical properties of the receiver have not changed, yet we gain a second dimension with which we can tune the response of the controller. This augmented state vector is defined thusly

$$x_k = \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} = \begin{bmatrix} C_p I_\theta \ e^{-cd}/d^2 \\ \phi \\ \dot{\phi} \end{bmatrix} \tag{8}$$

The variables of interest in this model are d, θ, ϕ and $\dot{\phi}$ which represent the distance between the two robots, the angle of the transmitter, the angle of the receiver, and the angular velocity of the receiver actuator. In this model we assume that the dynamics of the system are slow and x_1 is a static value which can be taken with a Gaussian process. This assumption is based on the fact that changing the distance d is a process that requires rotating and translating the entire robot, while changing the receiver angle ϕ is a localized process that can change rapidly. Furthermore, the communication in this process is half-duplex and the transmitter makes no adjustments based on the power received at the receiver.

With these assumptions the equations for continuous time can then be discretized to produce the following state equations.

$$x_k = \begin{bmatrix} x_{1,k} \\ x_{2,k} \\ x_{3,k} \end{bmatrix} = \begin{bmatrix} x_{1,k-1} + w_{1,k-1} \\ x_{2,k-1} + x_{3,k-1} + w_{2,k-1} \\ x_{3,k-1} + u_{k-1} + w_{3,k-1} \end{bmatrix} \tag{9}$$

w_k represents the Gaussian noise described as "process noise", or noise that is introduced by unmodeled changes in the system (robot position drift, changes in water particulate density, etc). x_1 will be the same at $k = n$ as it was at $k = n - 1$, plus Gaussian noise. x_2 will only be changed by the receiver velocity, plus process noise. And x_3 will be changed by the input to the system u_{k-1}, plus process noise.

We now define the output of our system to be $V_{d,k}$ the voltage at our receiver.

$$V_{d,k} = x_{1,k} g(x_{2,k}) + v_k \tag{10}$$

where $V_{d,k}$ is the voltage received at the output, $g(x_{2,k})$ is a cosine function, and v_k is Gaussian noise described as "measurement noise", or noise that is introduced by inaccuracies in output measurement.

3 Estimation and Control Design

Real world systems are not perfect. They are often noisy, and are sometimes unmeasurable with sensors. In these situations, a Kalman Filter can be applied to provide an optimal estimate of the system state. The major limitation of the Kalman Filter is that it requires the system to be linear. Many real world applications, including the one in this paper, are nonlinear, so a regular Kalman Filter cannot be used. However, the original Kalman Filter has been adapted in various ways to handle nonlinear systems. We will focus on the Extended Kalman Filter (EKF) due to its ability to yield high performance with relatively low computational cost compared to other Kalman Filter variants [7].

Before we are able to apply the EKF to the system we first must check to see if the system is Observable. This is a fundamental requirement for the system because the EKF relies on measurements to estimate the system state. If the system is not observable, then the system states cannot be accurately estimated based on the given measurements. In order to verify the system Observability, the Observation matrix must have a rank that is equal to the number of state variables for the system. The Observation matrix can be defined as

$$O = \begin{bmatrix} C \\ CA \\ CA^2 \end{bmatrix} \tag{11}$$

where C is the Jacobian matrix. The Jacobian is one of the main tools used by the EKF to linearize a nonlinear system. C is calculated by taking the first order partial derivative of the measurement with respect to the state variables. Utilizing equation 6, A and B can be defined as follows

$$A = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 1 \\ 0 & 0 & 1 \end{bmatrix}, B = \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix} \tag{12}$$

Putting it all together

$$C = \begin{bmatrix} cos(x_2) & -x_1 sin(x_2) & 0 \end{bmatrix} \tag{13}$$

$$CA = \begin{bmatrix} cos(x_2) & -x_1 sin(x_2) & -x_1 sin(x_2) \end{bmatrix} \tag{14}$$

$$CA^2 = \begin{bmatrix} cos(x_2) & -x_1 sin(x_2) & -2x_1 sin(x_2) \end{bmatrix} \tag{15}$$

$$O = \begin{bmatrix} cos(x_2) & -x_1 sin(x_2) & 0 \\ cos(x_2) & -x_1 sin(x_2) & -x_1 sin(x_2) \\ cos(x_2) & -x_1 sin(x_2) & -2x_1 sin(x_2) \end{bmatrix} \tag{16}$$

Since O is a square matrix, we can use the determinant to check if it is full rank.

$$det(O) = 0 \tag{17}$$

As it turns out, the determinant is equal to 0, which means O is not full rank. Therefore, the system is not observable with only one measurement because the rank of O is less than 3, which is the number of state variables for the system.

A second measurement will be needed to properly observe the system, which can be defined by the following

$$y_k = \begin{bmatrix} x_{1,k}cos(x_{2,k}) + v_k \\ x_{1,k-1}cos(x_{2,k-1}) + v_{k-1} \end{bmatrix} \tag{18}$$

Where the Jacobian, C, is

$$C = \begin{bmatrix} cos(x_{2,k}) & -x_{1,k}sin(x_{2,k}) & 0 \\ cos(x_{2,k-1}) & -x_{1,k-1}sin(x_{2,k-1}) & 0 \end{bmatrix} \tag{19}$$

and the rest of the Observation matrix is defined as

$$CA = \begin{bmatrix} cos(x_{2,k}) & -x_{1,k}sin(x_{2,k}) & -x_{1,k}sin(x_{2,k}) \\ cos(x_{2,k-1}) & -x_{1,k-1}sin(x_{2,k-1}) & -x_{1,k-1}sin(x_{2,k-1}) \end{bmatrix} \tag{20}$$

$$CA^2 = \begin{bmatrix} cos(x_{2,k}) & -x_{1,k}sin(x_{2,k}) & -2x_{1,k}sin(x_{2,k}) \\ cos(x_{2,k-1}) & -x_{1,k-1}sin(x_{2,k-1}) & -2x_{1,k-1}sin(x_{2,k-1}) \end{bmatrix} \tag{21}$$

Combining the previous three equations will give the Observation matrix O.

$$O = \begin{bmatrix} cos(x_{2,k}) & -x_{1,k}sin(x_{2,k}) & 0 \\ cos(x_{2,k-1}) & -x_{1,k-1}sin(x_{2,k-1}) & 0 \\ cos(x_{2,k}) & -x_{1,k}sin(x_{2,k}) & -x_{1,k}sin(x_{2,k}) \\ cos(x_{2,k-1}) & -x_{1,k-1}sin(x_{2,k-1}) & -x_{1,k-1}sin(x_{2,k-1}) \\ cos(x_{2,k}) & -x_{1,k}sin(x_{2,k}) & -2x_{1,k}sin(x_{2,k}) \\ cos(x_{2,k-1}) & -x_{1,k-1}sin(x_{2,k-1}) & -2x_{1,k-1}sin(x_{2,k-1}) \end{bmatrix} \tag{22}$$

This is a 6x3 matrix, so in order to calculate the rank of O, matrix row operations are performed to get O in Row Echelon Form. The resulting matrix will look like the following:

$$O = \begin{bmatrix} cos(x_{2,k}) & -x_{1,k}sin(x_{2,k}) & 0 \\ cos(x_{2,k-1}) & -x_{1,k-1}sin(x_{2,k-1}) & 0 \\ 0 & 0 & -x_{1,k}sin(x_{2,k}) \\ 0 & 0 & -x_{1,k-1}sin(x_{2,k-1}) \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix} \tag{23}$$

The rank of this matrix is 3, which is equal to the number of state variables, n, so the system is Observable.

In order to collect multiple measurements in the physical system the paper offers two different solutions to solve this problem. The first method adds a second receiver into the system and the second method uses a single receiver, slowly rotating around a center position. The second solution was selected to reduce system complexity and overall cost. This is modeled in the system as an array of predefined angles changing by 2° with each step.

$$\psi_k = -2, -4, -6, -8, -10, -8, -6, -4, -2, 0, 2, 4, 6, 8, 10, 8, 6, 4, 2, 0$$

For each iteration of the algorithm, one value is chosen sequentially, and two total measurements are required for the y_k vector. For the EKF equations, P is the conditional error covariance matrix, which is initialized as positive definite. Q_{EKF} is the process noise covariance, which is a positive definite and diagonal matrix. R_{EKF} is the measurement covariance, which is also a positive definite and diagonal matrix

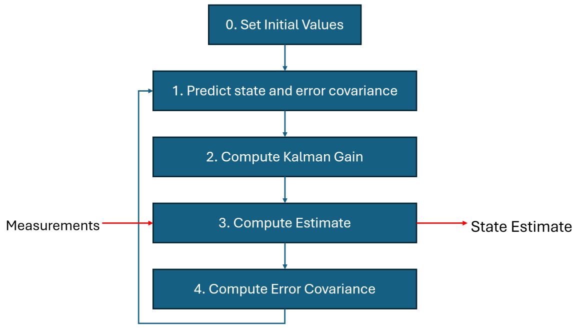

Figure 2: General Algorithm for the Extended Kalman Filter.

3.1 Extended Kalman Filter Algorithm

0. Initial values

$$\hat{x}_0, P_0$$

1. Predict the state and error covariance

Using equation 9, the predicted state, $\hat{x}_k$, can be estimated

$$\begin{aligned}\hat{x}_{1,k} &= \hat{x}_{1,k-1},\\ \hat{x}_{2,k} &= \hat{x}_{2,k-1} + \hat{x}_{3,k-1},\\ \hat{x}_{3,k} &= \hat{x}_{3,k-1} + u_{k-1},\end{aligned} \tag{24}$$

The predicted error covariance is calculated using the following

$$P_k = AP_{k-1}A^T + Q_{\text{EKF}} \tag{25}$$

In the EKF implementation for this system the error covariance P_k can be computed directly with A because the state update equation is already linear. The nonlinearity requiring EKF is only present in the output equation. In other EKF applications the F matrix must be calculated on every iteration of the algorithm and replaces A in equation 25. The equation for F is

$$\begin{aligned}\hat{x}_k &= f(\hat{x}_{k-1}, u_{k-1})\\ F_k &= \frac{\partial f}{\partial x}\end{aligned} \tag{26}$$

2. Compute the Kalman Gain

Using equation 18, the output measurement of the system, y_k, is defined as

$$y_k = \begin{bmatrix} V_{d,k} \\ V_{d,k-1}\end{bmatrix} \tag{27}$$

Similarly, the predicted measurement of the system, $\hat{y}_k$, is defined as

$$\hat{y}_k = \begin{bmatrix} x_{1,k} cos(x_{2,k} + \psi_k) \\ x_{1,k-1} cos(x_{2,k-1} + \psi_{k-1})\end{bmatrix} \tag{28}$$

This can be rearranged to eliminate the $x_{1,k-1}$, and $x_{2,k-1}$ values from the function and results in the following equation.

$$\hat{y}_k = h(\hat{x}_{1,k}, \hat{x}_{2,k}, u_{k-1}) = \begin{bmatrix} \hat{x}_{1,k}\cos(\hat{x}_{2,k} + \psi_k) \\ \hat{x}_{1,k}\cos(\hat{x}_{2,k} - \hat{x}_{3,k} + u_{k-1} + \psi_{k-1})\end{bmatrix} \tag{29}$$

The Jacobian is computed using the previous $\hat{y}_k$ by taking the partial derivatives with respect to each state variable.

$$C_k = \frac{\partial h(\hat{x}_{1,k}, \hat{x}_{2,k}, \hat{x}_{3,k}, u_{k-1})}{\partial x_k} = \begin{bmatrix} C_{k,1,1} & C_{k,1,2} & 0 \\ C_{k,2,1} & C_{k,2,2} & C_{k,2,3}\end{bmatrix} \tag{30}$$

$$\begin{aligned}C_{k,1,1} &= \cos(\hat{x}_{2,k}+\psi_k),\\ C_{k,1,2} &= -\hat{x}_{1,k}\sin(\hat{x}_{2,k}+\psi_k),\\ C_{k,2,1} &= \cos(\hat{x}_{2,k}-\hat{x}_{3,k}+u_{k-1}+\psi_{k-1}),\\ C_{k,2,2} &= -\hat{x}_{1,k}\sin(\hat{x}_{2,k}-\hat{x}_{3,k}+u_{k-1}+\psi_{k-1}),\\ C_{k,2,3} &= \hat{x}_{1,k}\sin(\hat{x}_{2,k}-\hat{x}_{3,k}+u_{k-1}+\psi_{k-1})\end{aligned} \tag{31}$$

Now, we can compute the Kalman Gain using

$$K_k = P_k C_k^T (C_k P_k C_k^T + R_{\text{EKF}})^{-1} \tag{32}$$

3. Compute the estimate
The estimate of the system, $\hat{x_k}$, can be calculated using the following equation. Here, the current state estimate is added with difference between the measurement and predicted measurement multiplied by the Kalman Gain.

$$\hat{x}_k = \hat{x}_k + K_k(y_k - \hat{y}_k) \tag{33}$$

4. Compute the error covariance
Finally, the error covariance is updated for the next iteration of the estimation loop.

$$P_k = (I - K_k C_k) P_k \tag{34}$$

3.2 LQR Control Design

The LQR controller is a control algorithm that produces the optimal control response for a given cost function. The cost function provided creates weighted costs for each of the systems states, as well as weighted costs for each of the systems inputs, to determine the optimal plant input to drive the states to the desired values. In our application the LQR controller is designed to minimize the pointing error of the receiver. The cost function can be described as

$$J_{LQR} = \frac{1}{2}\sum_{k=0}^{\infty}(x_k^T \ Q_{LQR} \ X_k + u_k^T \ R_{LQR} \ u_k) \tag{35}$$

where Q_{LQR} and R_{LQR} are the weighting matrices. Q_{LQR} is an $n*n$ matrix where n is the number of states and R_{LQR} is an $m*m$ matrix where m is the number of outputs. Q_{LQR} and R_{LQR} are also necessarily positive semi-definite, with the most simple way of guaranteeing this being that they are also diagonal.
Given the system: $x_{k+1} = Ax_k + Bu_k$ the input to the system u_k that takes the x_2 and x_3 states to zero in the optimal way is given by the equation

$$u_k = -K_{LQR} x_k \tag{36}$$

where

$$K_{LQR} = (B^T X B + R_{LQR})^{-1}(B^T X A) + Q_{LQR} = 0 \tag{37}$$

where X is a positive semi-definite solution of the Riccati equation

$$A^T X A - X - (A^T X B)(B^T X B R_{LQR})^{-1}(B^T X A) + Q_{LQR} = 0 \tag{38}$$

After computing the estimated states for x_2 and x_3 using the EKF, equation 36 becomes

$$u_k = -K_{LQR}\begin{bmatrix}\hat{x}_{2,k}\\ \hat{x}_{3,k}\end{bmatrix} \tag{39}$$

4 Numerical Simulations

In the following section we will describe the setup of our simulation, the parameters that we used for calculation, and the results of the analysis.

Our analysis was done using MATLAB and Simulink. The components of our Simulink simulation were modeled using custom MATLAB function blocks, as opposed to the prebuilt Simulink blocks provided by MathWorks. Code for the function blocks can be found in the appendix.

We built our model using the Principle of Separation, which states that the problem of state estimation and state control can be separated into independent analytical problems. We first designed and tested the Extended Kalman Filter to see that we could accurately predict state variables given system outputs. We then designed the LQR controller using the known true states from the system. We then combined the two systems so that the LQR controller used the estimated states from the Extended Kalman Filter to complete the control design.

Parmeters	**Values**
Q_{sys}	$\begin{bmatrix} 0.0025 & 0 & 0 \\ 0 & 0.0025 & 0 \\ 0 & 0 & 0.0025 \end{bmatrix}$
R_{sys}	0.001
P_0	$\begin{bmatrix} 100 & 0 & 0 \\ 0 & 100 & 0 \\ 0 & 0 & 100 \end{bmatrix}$
Q_{EKF}	$\begin{bmatrix} 0.0025 & 0 & 0 \\ 0 & 0.0025 & 0 \\ 0 & 0 & 0.0025 \end{bmatrix}$
R_{EKF}	$\begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$
Q_{LQR}	$\begin{bmatrix} 30 & 0 \\ 0 & 1 \end{bmatrix}$
R_{LQR}	1000
$\hat{x}_0$	$[5, 2, 3]^T$
K_{LQR}	$[0.1287, 0.5764]$

Table 1: System Parameters

4.1 Kalman Filter Simulation

To simulate the Kalman filter, we first modeled the system using a MATLAB function block and the equations outlined in section 2.1. The Kalman filter was also modeled using a MATLAB function block and the equations outlined in section 3.1. The code for the function blocks can be found in the Appendix. Simulink does provide a prebuilt Extended Kalman Filter block, however it was not robust enough to be used in our application. The block diagram for the plant can be seen in Figure 3. The block diagram of the connection between the plant and the EKF can be seen in 4

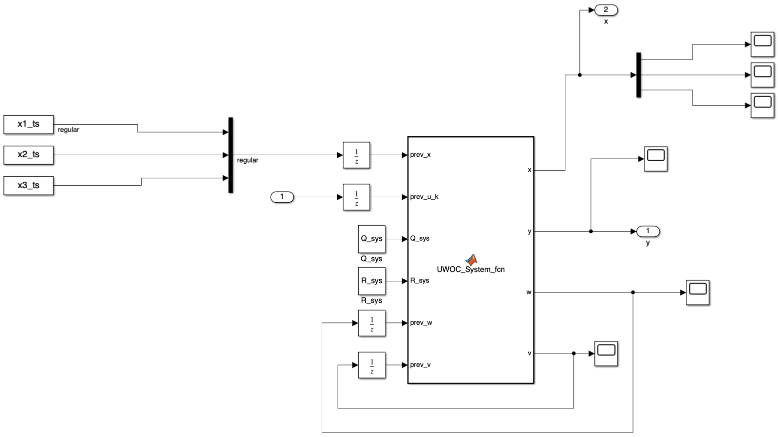

Figure 3: Block diagram of the plant

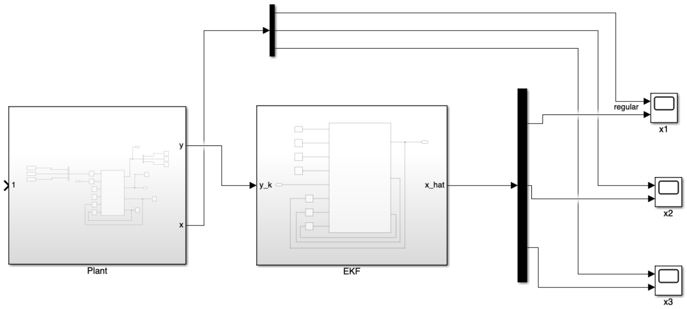

Figure 4: Connection between system and EKF

Because the system was an open loop we had no input control to the plant. To account for this we left the input disconnected but passed predefined data to the plant to determine the next system step. The system would then produce the corresponding output that is then passed to the EKF. The EKF would then produce an estimate of what the system states are.

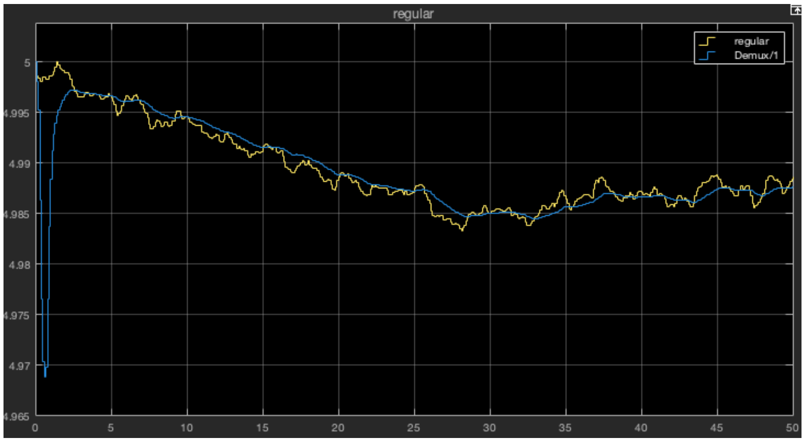

Figure 5: EKF Tracking Results for x1

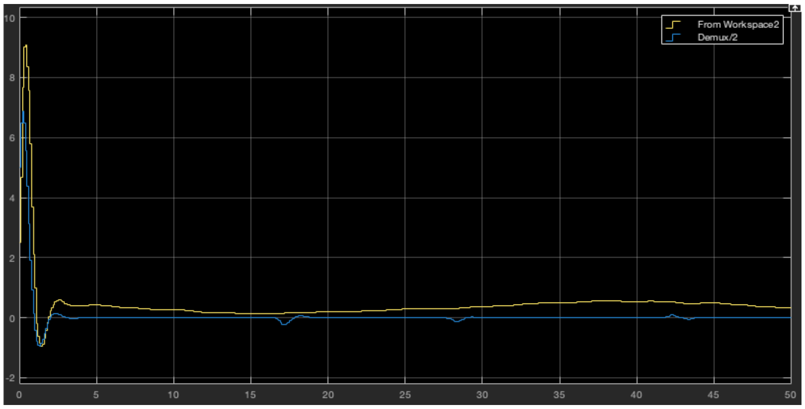

Figure 6: EKF Tracking Results for x2

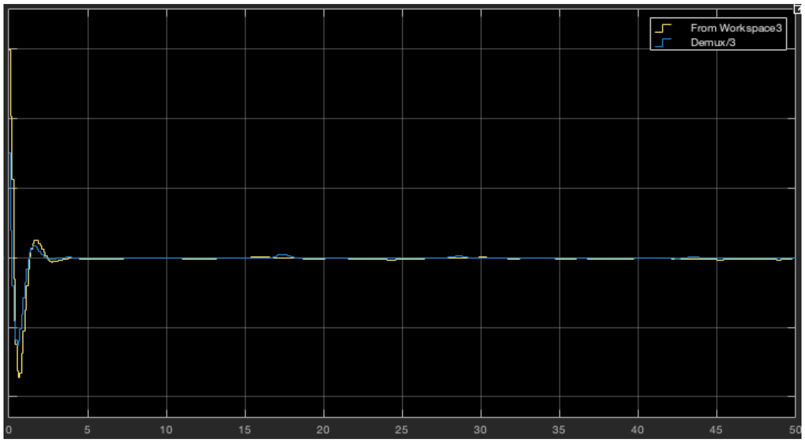

Figure 7: EKF Tracking Results for x3

In each of the tracking results images the yellow line represents the true system values while the blue line represents the EKF's expected values. As can be seen in the images, the EKF closely tracks the true values of the system after about 2 seconds.

4.2 LQR Controller Simulation

To simulate the LQR control, we used a modified version of the plant model that was used to develop the Kalman filter. For the LQR controller design, we output the state variables from the plant directly. These state variables were then multiplied with the LQR gain to determine the optimal control input to the system. The block diagram of the state controller can be seen in Figure 8.

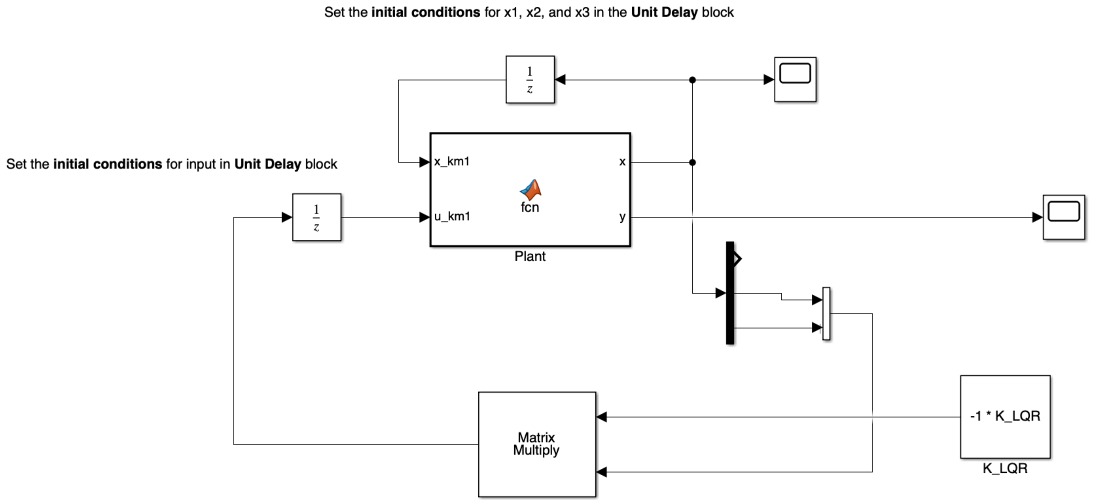
Set the **initial conditions** for x1, x2, and x3 in the **Unit Delay** block
Set the **initial conditions** for input in **Unit Delay** block
1/z
1/z
x_km1
u_km1
fcn
x
y
Plant
Matrix Multiply
-1 * K_LQR
K_LQR

Figure 8: Block diagram of LQR controller

In Figure 9, with the initial conditions $x = [5, 2, 3]^T$ we see that the state variables x_2 and x_3 are brought to the desired state (0 deg, 0 deg/s) in around 2 seconds, or 20 time steps.

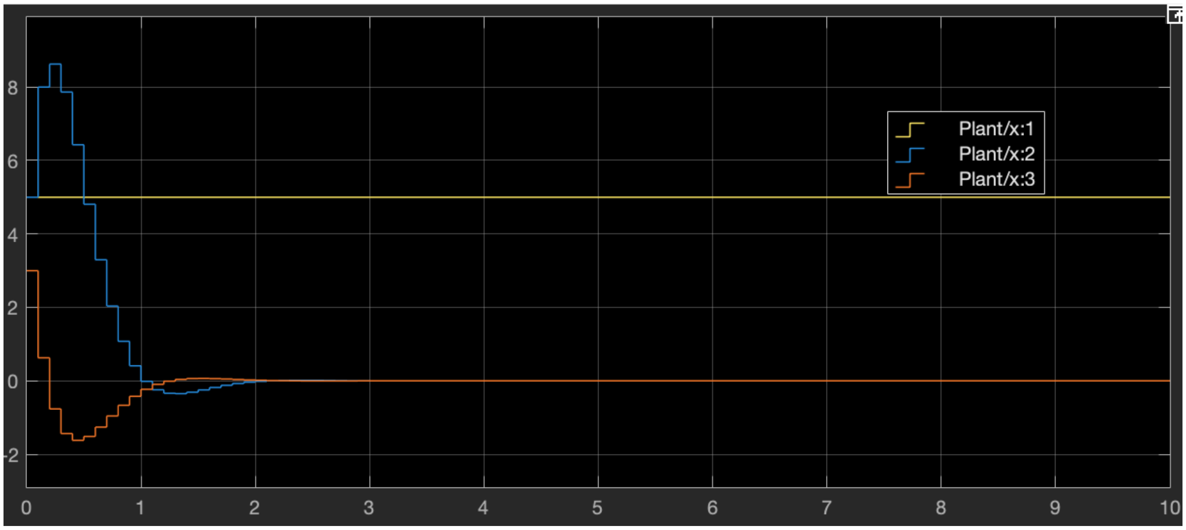

Figure 9: Block diagram of LQR controller

4.3 Stability and Trajectory Analysis

After modeling the EKF and LQR controller as separate systems, we closed the loop on the system by passing the estimated states from the EKF to the LQR controller, and the control output from the LQR to the plant and the EKF. The block diagram for the closed loop system can be seen in 10.

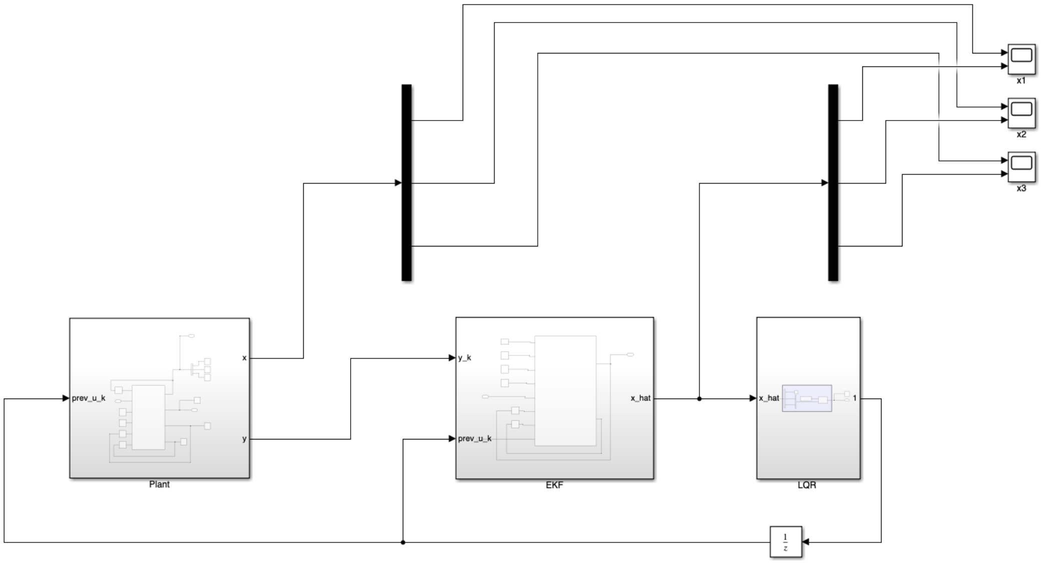

Figure 10: Block diagram of LQR controller

With the initial conditions of $x = [5, 2.5, 3]^T$ and $\hat{x} = [5, 2.5, 3]^T$, we get the following results. As can be seen the estimated states do not track the actual states, and the x_2 and x_3 state variables are not controlled to $[0, 0]$

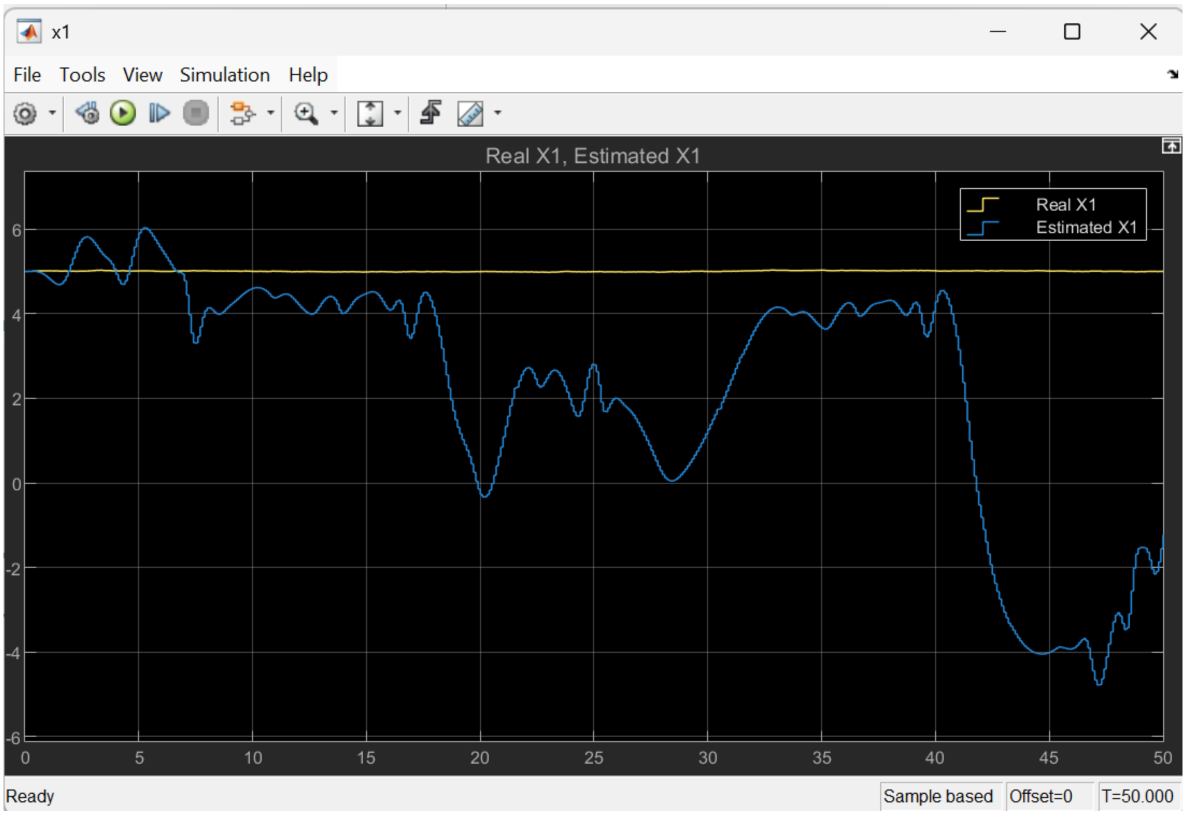

Figure 11: State and State Estimate Comparison: x1

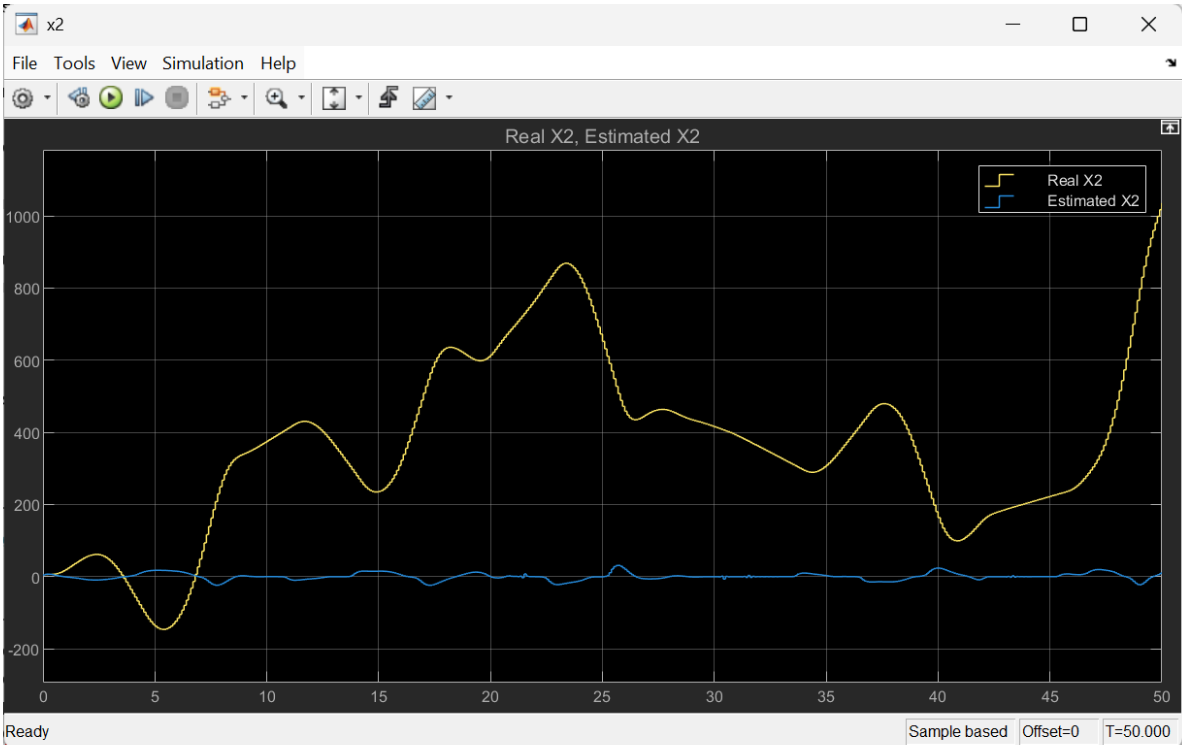

Figure 12: State and State Estimate Comparison: x2

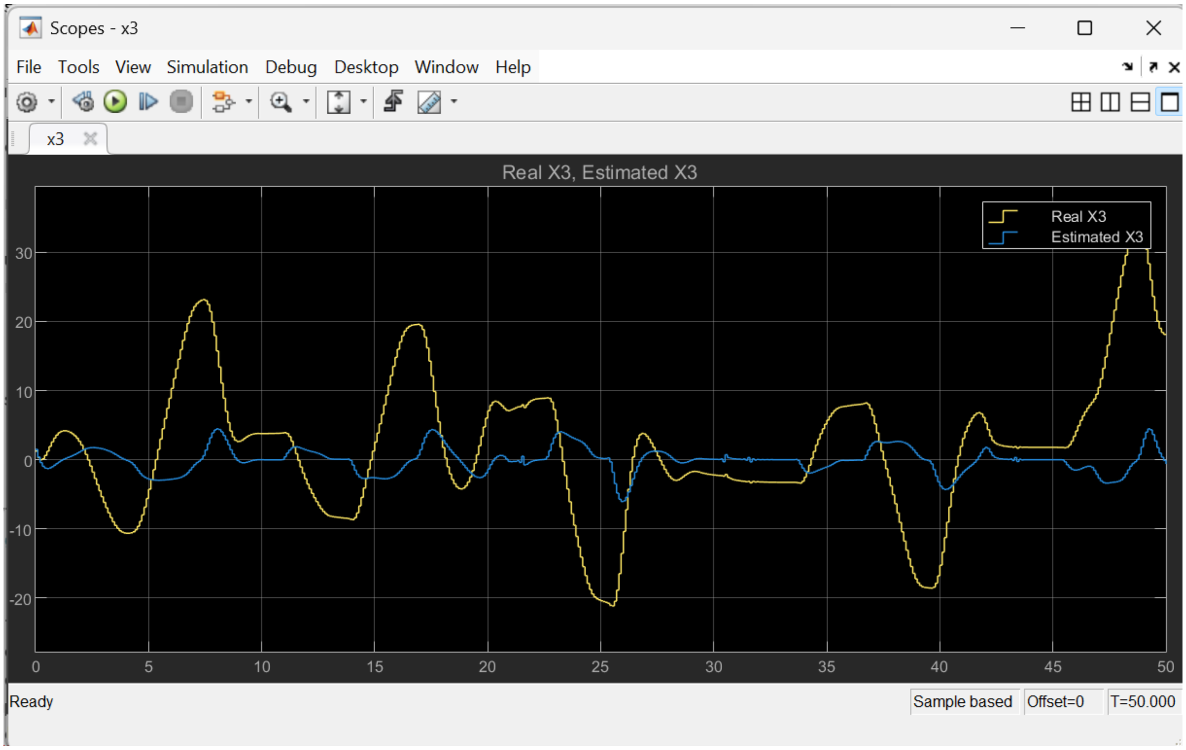

Figure 13: State and State Estimate Comparison: x3

5 Conclusion

We are not sure why the control loop is not operating correctly and the states are not being controlled and driven to the $[0, 0]$ position. According to the principle of separation the state estimation problem and the LQR optimal control problem can be solved independently and then combined to produce a controlled system. Our suspicion is that there is a modeling issue somewhere in the plant because we showed that the Kalman filter and LQR were operating independently.

One difference between our system and the system described in [1] is that they use an additional ψ_k term in their state estimation equations that is not represented in our system. The ψ_k term functions by sequentially selecting a new angle in a list of predefined angles from -10 to 10. This change in angle is not actually present in the physical system, only in the state estimation. The idea behind adding the ψ_k to the estimate is that by introducing some error it prevents instability in the system.

Several steps were taken to fix the closed loop system. Some of these steps are:

- Including the ψ_k term in the state estimation equations. This worsened the results and introduced additional oscillations.
- Tuning the Q_{EKF}, R_{EKF}, and Q_{LQR} parameters. The behavior of the filter would change but no demonstrable improvement was made.
- Changing the initial conditions for x and $\hat{x}$. No demonstrable improvement.
- Changing the timestep of the simulation. This only elongated or shortened the filter/control response.
- Introducing state/control limits. The state and control values are unbounded by the equations for the plant. We introduced limits so that the x_2 state could not go beyond -15 or +15 degrees. We limited x_2 to 0.2 degrees/timestep. This resulted in a slower response from the filter/control but would eventually end in oscillation/instability.

6 Appendix

6.1 Plant Model MATLAB function block

```
function [x, y, w, v] = UWOC_System_fcn(prev_x, prev_u_k, Q_sys, R_sys, prev_w, prev_v)

% Noise Parameters
mu = 0;
w = Q_sys * randn(3,1) + mu;
v = R_sys * randn(1,1) + mu;


w = [0; 0; 0];
v = 0;


x1 = prev_x(1) + prev_w(1);
x2 = prev_x(2) + prev_x(3) + prev_w(2);
x3 = prev_x(3) + prev_u_k + prev_w(3);

y1 = x1*cosd(x2) + v;
y2 = prev_x(1) * cosd(prev_x(2)) + prev_v;

y = [y1; y2];
x = [x1; x2; x3];
```

6.2 EKF Calculations

```
    function [x_hat, P_k, u_k]= ekf(A, Q_ekf, R_ekf, psi_vals, y_k, prev_x_hat, prev_P_k, prev_u_k)

% Initial state prediction
x_hat = [prev_x_hat(1); prev_x_hat(2) + prev_x_hat(3); prev_x_hat(3) + prev_u_k];

% Initial error covariance
P_k = A * prev_P_k* A’ + Q_ekf;

% Output Meaurements
h = [x_hat(1) * cosd(x_hat(2) + psi_vals(1));
    x_hat(1) * cosd(x_hat(2) - x_hat(3) + prev_u_k + psi_vals(1))];

y_hat_k = h;

% Jacobian
C = [cosd(x_hat(2) + psi_vals(1)), -x_hat(1) * sind(x_hat(2) + psi_vals(1)), 0;
    cosd(x_hat(2)-x_hat(3) + prev_u_k + psi_vals(1)), -x_hat(1) * sind(x_hat(2)-x_hat(3)+ prev_u_k +

% Compute Kalman Gain
K_k = P_k * C’ * ((C * P_k * C’) + R_ekf)^-1;

% Compute the Estimate

x_hat = x_hat + K_k*(y_k - y_hat_k);

% Compute the Error Covariance

P_k = P_k - (K_k * C * P_k);
```

```
u_k = (-(0.1) * x_hat(2)) - ((0.4) * x_hat(3));

end
```

6.3 LQR Gain K calculation

```
A = [1, 0, 0;
     0, 1, 1;
     0, 0, 1];
B = [0;
     0;
     1];
% A_p stands for A_partial
% This is needed because the LQR controller is only dependent on x2, x3
A_p = [1,1;
       0,1];
B_p = [0;
       1];

% We don't use C or D for LQR control
% C = [cos(x2k), -x1k*sin(x2k), 0;
     % cos(x2k-x3k+uk), -x1k*sin(x2k-x3k+uk), x1k*sin(x2k-x3k+uk)];
% D = [0, 0;
     % 0, 0];

Q_LQR = [30, 0;
          0,  1];
R_LQR = 1000;

[K_LQR, S, P] = dlqr(A_p, B_p, Q_LQR, R_LQR);
K_LQR
```

References

[1] Asem Alalwan, Tadjine Mohamed, Messaoud Chakir, and Taous Meriem Laleg. Extended kalman filter based linear quadratic regulator control for optical wireless communication alignment. *IEEE Photonics Journal*, 12(6):1–12, 2020.

[2] Marek Doniec, Michael Angermann, and Daniela Rus. An end-to-end signal strength model for underwater optical communications. *Oceanic Engineering, IEEE Journal of*, 38:743–757, 10 2013.

[3] T.-H. Ho. *Pointing Acquisition and Tracking Systems for Free-Space Optical Communication Links*. Ph.d. dissertation, University of Maryland, College Park, College Park, MD, USA, 2007.

[4] A. G. Nasser and M. A. A. Ali. Performance of led for line-of-sight (los) underwater wireless optical communication system. *Journal of Optical Communications*, 2020. To be published.

[5] T. Kane P. Deng and O. Alharbi. Reconfigurable free space optical data center network using gimbal-less mems retroreflective acquisition and tracking. In *Proc. Free-Space Laser Communication and Atmospheric Propagation XXX*, volume 10524, page 1052403. SPIE, 2018.

[6] N. Anous A. Boyaci M. Abdallah S. Hessien, S.C. Tokgoz and K.A. Qaraqe. Experimental evaluation of ofdmbased underwater visible light communication system. *Photnoics, IEEE Journal of*, 10(5), 11 2018.

[7] J. Simanek. *Data Fusion for Localization Using State Estimation and Machine Learning*. Ph.d. dissertation, Czech Technical University, Prague, Czechia, 2015.

[8] Pratap Bhanu Solanki, Mohammed Al-Rubaiai, and Xiaobo Tan. Extended kalman filter-based active alignment control for led optical communication. *IEEE/ASME Transactions on Mechatronics*, 23(4):1501–1511, 2018.