Delft University of Technology

INTEGRATION PROJECT SYSTEMS AND CONTROL
(COURSE CODE: SC42035)

Flexlink
Group 55

Authors:
Olav Jacob (4305213)
Shreyas Nikte (4716132)

June 28, 2018

Delft
University of
Technology

Summary

In this report, we will explain the steps we took to control the flexlink setup. The flexlink setup consists of to DC motors, with a small elastic band connecting them. Than, one of the DC motors is used to control the position of the second motor. The difficulty in this setup comes from the slip of the elastic band, which causes non-linear behaviour.

In this report, we will describe how we control the setup. The first step is to make a model of the system. The next step is to identify the system, using subspace identification. This is done for two separate system. The first system for the position of the first motor, and the second system for the second motor. The reason we did this is to decouple the two motors, making the slip less of an issue for designing observers.

The next step of the control is to design state observers. This is because we want to use a linear quadratic regulator, which is a full information state feedback control method. Therefore we will need to reconstruct all the state of the system.

The last step is to design the two controllers for the two systems. The first controller will control the velocity of the driving DC motor. The second motor will control the position of the second motor, by returning a reference velocity for the first motor to track.

Contents

1 Introduction to the setup

The setup flexlink consists of two DC motors, with a small elastic band to connect them. The goal of the project is to design a controller to control the position of the second motor (the second motor acting as a passive load and the first motor acting as a driving motor). The second motor will only be excited to simulate small disturbances. For this report, α will be used to denote the position of the first (driving) motor and β will be used to denote the position of the second motor.

What makes controlling the position of the second motor difficult, is the fact that the elastic band has a lot of slip. This causes the nonlinear behaviour in the system. In this report we will assume a linear model of the system. We then need to find a way to overcome the non-linearity's. This is done by decoupling the two motors into two separate systems. That way, we can use state observers to see the system states, without the non-linear slip causing the observers to give an inaccurate representation of the states.

The next section will cover the details about black box identification of our system, results and limitation of the identification method.

2 Black Box Controller

In order to find out the system behavior, initially black box identification method was implemented. For this implementation, the system is assumed to be linear for its frequency bandwidth. The method assumes no knowledge of the system and outputs the transfer function of the system based on input-output response and user-determined order. Using the frequency response of the system, the Proportional-Integral (PI) controller was tuned.

2.1 System identification

For gathering the data for system identification, we selected two different inputs:

- Square wave
- Sum of two sine waves

Here, the use of sum of two sine waves of slightly different frequencies provide the response of the system for large number of frequencies. The data gathered by introducing these inputs to the system is used to identify a system. Since until now, we do not have any insights about the order of the system (as we are working on a black box model), we use n4sid with order 5. The bode plot of the identified system is plotted in Figure 1. From the bode plot we can see a very large phase drop around 10 rad/s. Therefore we use a proportional controller to push the bandwidth to about 5 rad/s. To increase the gain for low frequencies, we also add an integral term. Because of the large phase drop, adding an derivative term will not allow us to increase the bandwidth of the system. Therefore our initial controller is just a PI controller.
The black-box modelling is easy to implement, but it not a good choice for the system which is highly non linear. The PI controller for black box model is not robust, as, the system response still contains high overshoots for sudden input changes or disturbances. Hence, it leads us to understand the effect of system dynamics on its response. The next section will cover the derivation of the system model for the setup.
The response of PI controller is shown in Figure 2

3 System Model

The first step in controlling the system, is to make a model of it. To do this, we will try to describe all of the dynamics of this system, and write them down in a state space. This state space consists of 2 parts. The first one is for the DC motor with states α, $\dot{\alpha}$ and i, which are the motors position, angular velocity and armature current. The equations of motion of a DC motor are a combination of newtons second law and Kirchhoffs voltage law [2]:

$$J\ddot{\alpha} + b\dot{\alpha} = Ki$$

$$L\frac{di}{dt} + Ri = V - K\dot{\alpha}$$

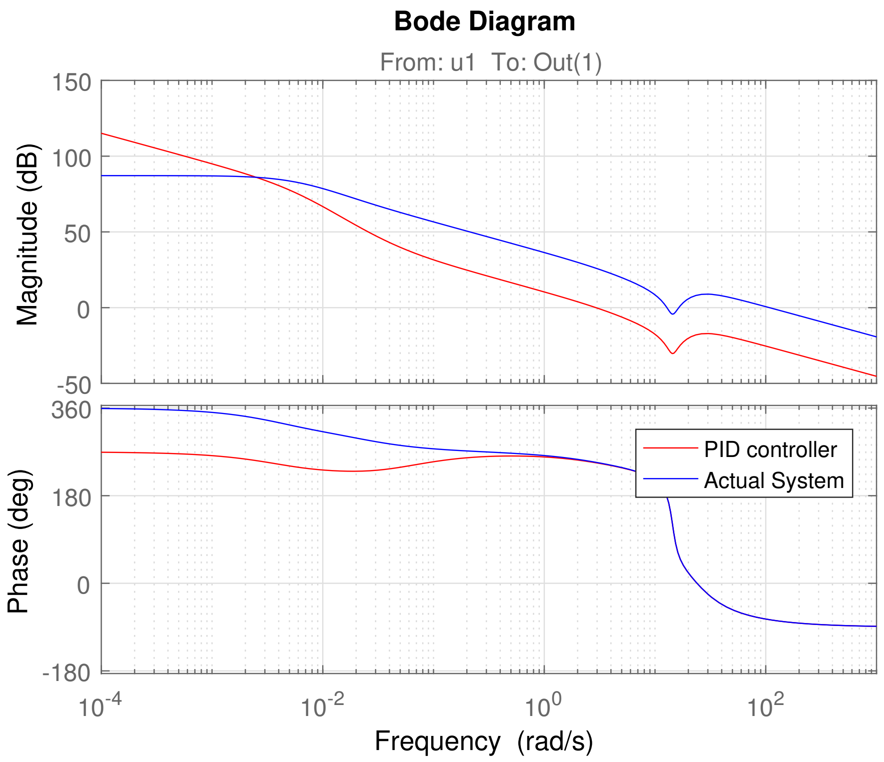

Figure 1: Bode plot of black-box system identification of order 5

Where α represents the angle of the DC motor, i is the electric current in the motor. J is the inertia of the motor and the load, K is the electromotive force constant, L is the motor inductance, R is the motors electric resistance and V is the motors input voltage. To write this as a state space, we will rewrite these equations as:

$$\ddot{\alpha} = -\frac{b}{J}\dot{\alpha} + \frac{K}{J}$$

$$\frac{di}{dt} = -\frac{K}{L}\dot{\alpha} + \frac{R}{L}i + \frac{1}{L}V$$

Using, that the input is equal to V $(u = V)$, we can write down the following state space equation for the speed of the DC motor:

$$\frac{d}{dt}\begin{pmatrix}\dot{\alpha}\\ i\end{pmatrix} = \begin{pmatrix}-\frac{b}{J} & \frac{K}{J}\\ -\frac{K}{L} & -\frac{R}{L}\end{pmatrix}\begin{pmatrix}\dot{\alpha}\\ i\end{pmatrix} + \begin{pmatrix}0\\ \frac{1}{L}\end{pmatrix}u$$

The system has the position of α as its output instead of the velocity. We can easily add this to the system as the position is just the integral of the velocity. In the state space, we can just add an extra state. Then the derivative of the state is equal to the second state.

$$\frac{d}{dt}\begin{pmatrix}\alpha\\ \dot{\alpha}\\ i\end{pmatrix} = \begin{pmatrix}0 & 1 & 0\\ 0 & -\frac{b}{J} & \frac{K}{J}\\ 0 & -\frac{K}{L} & -\frac{R}{L}\end{pmatrix}\begin{pmatrix}\alpha\\ \dot{\alpha}\\ i\end{pmatrix} + \begin{pmatrix}0\\ 0\\ \frac{1}{L}\end{pmatrix}u$$

The second part is the interaction of two spinning disks, with a spring between them. We will model the rubber band as a spring for now, to make sure system mechanics are linear. The equations of motion for two spinning disks with a spring are:

$$J\ddot{\alpha} + b\dot{\alpha} = k(\beta - \alpha)$$

$$J\ddot{\beta} + b\dot{\beta} = k(\alpha - \beta)$$

Where k is the spring coefficient, and β is the angular position of the second wheel. This formula can easily be validated, as we know that the force that the spring exerts on the disks is equal to the difference in the wheels angle times the spring coefficient. The equations result in the state space:

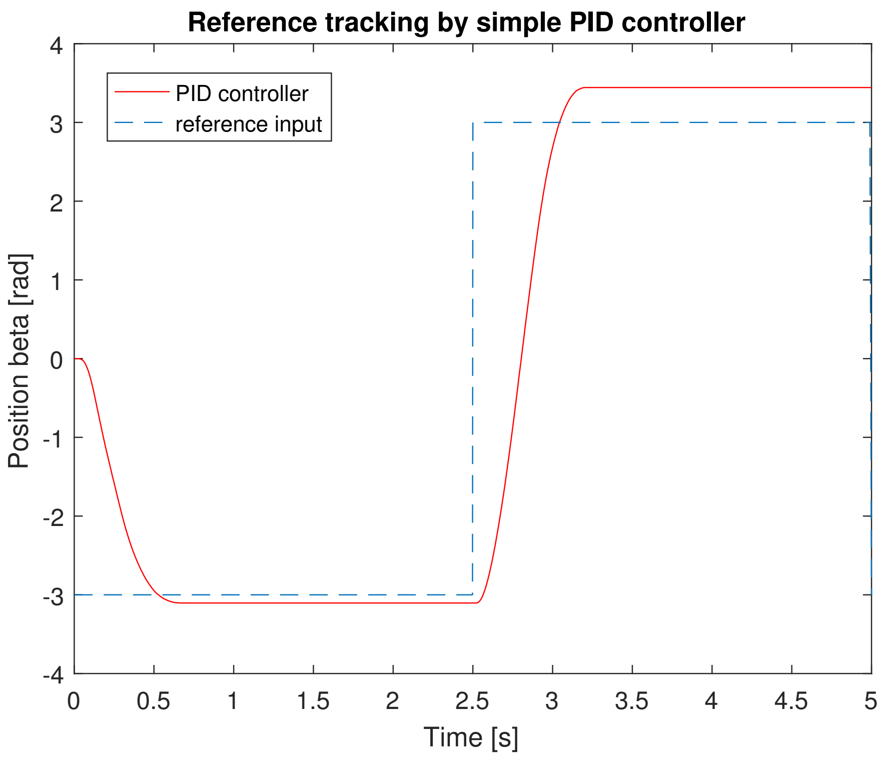

Figure 2: Rensponse of simple PID controller for the blackbox system

$$\frac{d}{dt}\begin{pmatrix}\alpha\\ \dot{\alpha}\\ \beta\\ \dot{\beta}\end{pmatrix} = \begin{pmatrix}0 & 1 & 0 & 0\\ -\frac{k}{J} & -\frac{b}{J} & \frac{k}{J} & 0\\ 0 & 0 & 0 & 1\\ \frac{k}{J} & 0 & -\frac{k}{J} & -\frac{b}{J}\end{pmatrix}\begin{pmatrix}\alpha\\ \dot{\alpha}\\ \beta\\ \dot{\beta}\end{pmatrix}$$

Then, we can combine these two state spaces to give us a linear model for our system, of which we can estimate the parameters:

$$\frac{d}{dt}\begin{pmatrix}\alpha\\ \dot{\alpha}\\ i\\ \beta\\ \dot{\beta}\end{pmatrix} = \begin{pmatrix}0 & 1 & 0 & 0 & 0\\ -\frac{k}{J} & -\frac{b}{J} & \frac{K}{J} & \frac{k}{J} & 0\\ 0 & -\frac{K}{L} & -\frac{R}{L} & 0 & 0\\ 0 & 0 & 0 & 0 & 1\\ \frac{k}{J} & 0 & 0 & -\frac{k}{J} & -\frac{b}{J}\end{pmatrix}\begin{pmatrix}\alpha\\ \dot{\alpha}\\ i\\ \beta\\ \dot{\beta}\end{pmatrix} + \begin{pmatrix}0\\ 0\\ \frac{1}{L}\\ 0\\ 0\end{pmatrix} v$$

We can see that to describe the system, we will need to find the following parameters:

- k: the spring coefficient [N/rad]
- b: the damping of the motor [Nms]
- J: the motors moment of inertia [kgm^{2}s^{-2}]
- K: the electromotive force constant [Nm/Amp]
- R: the motors electric resistance [Ω]
- L: the electric inductance [H]

For now, the moment of inertia J and the damping b is assumed to be equal for both disks, for simplicity.

4 System Identification

For the identification of the system, instead of using the model from the last section, we will go with different approach. There are two main reasons to avoid the use of aforementioned model of the system:

- The first problem with the model is that the state space does not account for non linearity's in the model. Since the flexible belt has a lot of slip when moving at higher speeds, the two wheels will not always have the same relative angle. Since the model calculates the tension of the belt by taking the difference in angles between the two wheels, the tension in the model and in the real setup is going to be different once the belt slips.
- Secondly, the model depends on a lot of parameters. There are actually to many parameters to estimate using system identification. Therefore it will be hard to find actual parameter values.

Therefore we decide to identify a grey box model instead. First we use a black box identification, but by manipulating the resulting state space, we will be able to tell some of the states physical meaning.

Secondly, we will split the system in two subsystems. The first subsystem is a model for the first wheel (or first motor). We create a velocity controller for this subsystem. Then, we use the velocity of the first subsystem as an input for the second system. Here, the use of velocity of the first wheel instead of position is an important leap, since, using the position of both wheels will result in an inaccurate error signal once the slip takes place in the system. The second system has the velocity of the first wheel as its input, and the position of the second wheel as output. Since we are identifying two different systems, our end result will be two cascading controllers.

4.1 Collecting the data for the first subsystem

For the identification to be accurate, it is important that the data holds enough information about the system. There are a few different methods for this. For instance a step response, or a square wave. These signals hold a wide range of frequencies, and therefore, are suitable for identification. However, we decided to use a sine wave, which is increasing in frequency over time.
The input signal we use is:

$$y(t) = \sin 0.1t^2$$

This signal is run for 60 seconds. We will run the system without the belt, so we will only identify the first DC motors mechanics.

4.2 Identifying the system for the first DC motor

Now that we have collected our data, we will use the MATLAB function n4sid to make a state space model of the first DC motor. This function will use the subspace model to derive the state space. We can see from Section 3 that a single DC motor should yield a state space system of order 3. Therefore we will estimate a model of order 3. The resulting system is very much a blackbox model of the system, since we do not know what any of the state mean. Therefore we will put the system in the observable canonical form. The canonical form will look as:

$$\dot{x} = \begin{pmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \\ a_1 & a_2 & a_3 \end{pmatrix} x + \begin{pmatrix} b_1 \\ b_2 \\ b_3 \end{pmatrix} u$$

$$y = \begin{pmatrix} 1 & 0 & 0 \end{pmatrix} x$$

From this form, we can see that the output is equal to the first state. Since we know that the output of the system is the position of the first DC motor, we can conclude that the first state is the position of the motor. From the system matrix we can also see that the second state is the derivative of the first state, thus we can also conclude that the second state of the system is equal to the angular velocity of the first motor.

4.3 Validation

Now that we have identified the system, we will need to validate it. For the validation it is important that we use different data than the identification data. Therefore we will now use a square wave input for both the physical system. We will use the same input to simulate the identified system using the MATLAB function lsim. The results are plotted in Figure 3. From this figure we can see that the identified system and the actual system are a very close match. We conclude that the identified system is good enough for our control purposes.

Since we want to use the identified system to control the velocity of first DC motor, we also need to verify that the second state of the system indeed gives us a good representation of the velocity. The reason we use the second state of the identified system, instead of just using a derivative block in Simulink, is that the derivative

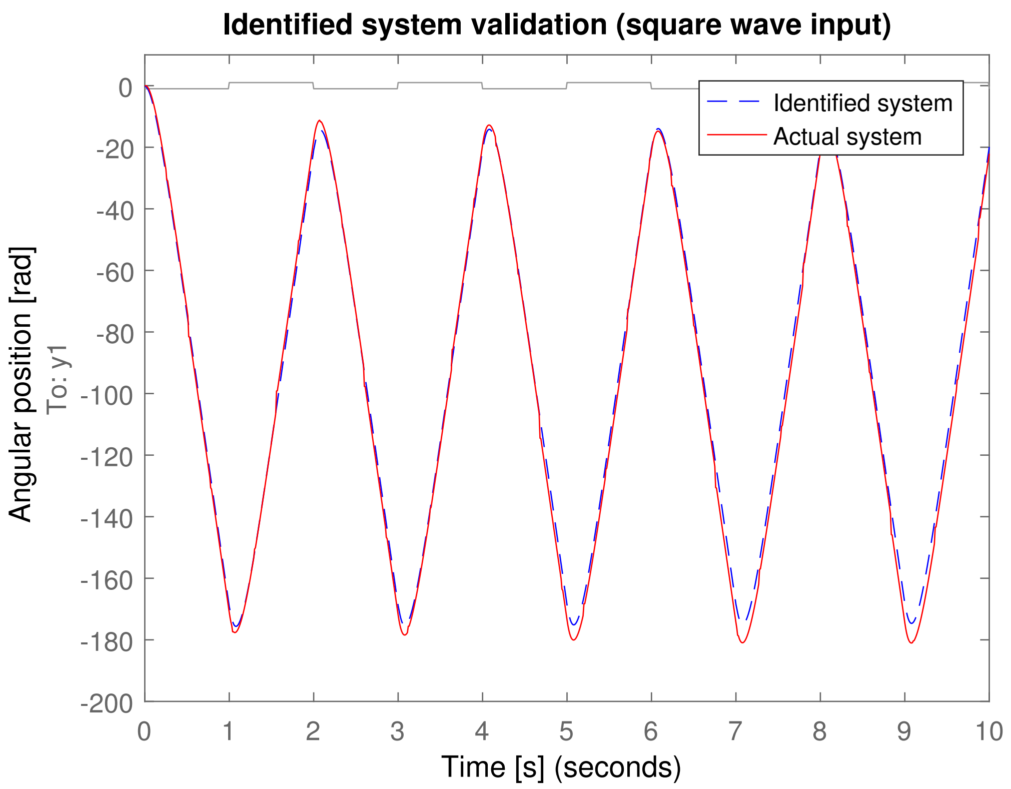

Figure 3: Simulation for a square wave input for the identified system and the physical system

block is very sensitive to measurement noise, and will give very large spikes in the velocity. Therefore, using the second state of the observed system should give a better indication of the velocity of the first wheel. To validate this, we will change the output of the system to be the second state, so $C = \begin{pmatrix} 0 & 1 & 0 \end{pmatrix}$. Then we will output the differential of the position of the physical system. Since the differential of the physical system is very sensitive to measurement errors, we will use the function `smooth`, to smooth out the large peaks in the differential a bit. The results of this comparison are plotted in Figure 4.
Knowing that the peaks in the plot are caused by small measurement errors, we can conclude that the second state of the system indeed gives us a fairly good representation of the velocity of the first DC motor.

4.4 Collecting data for the second system

Now that we have a good identified model for the first DC motor, we will need to identify the second system, which has the velocity of the first DC motor as its input, and the position of the second wheel as its output. To collect the data for this identification we will use that same method as before (using a sine wave of increasing frequencies), this time with the belt attached to the system. Then, we will calculate the velocity of the first motor by calculating the derivative of the first motors position.

4.5 System identification for second system

For the second system, we will also use `n4sid`, to derive a observable canonical form state space. This time, the state space will have order 2, such that the two systems combined will have 5 states, which is identical to the system derived in Section 3.

4.6 System validation

For the system validation, we will use a square wave input for the complete system. We will then compare the response of the position of the second wheel to that of the identified system, where we will use the derivative of the position of the first wheel as input. The results are plotted in Figure 5

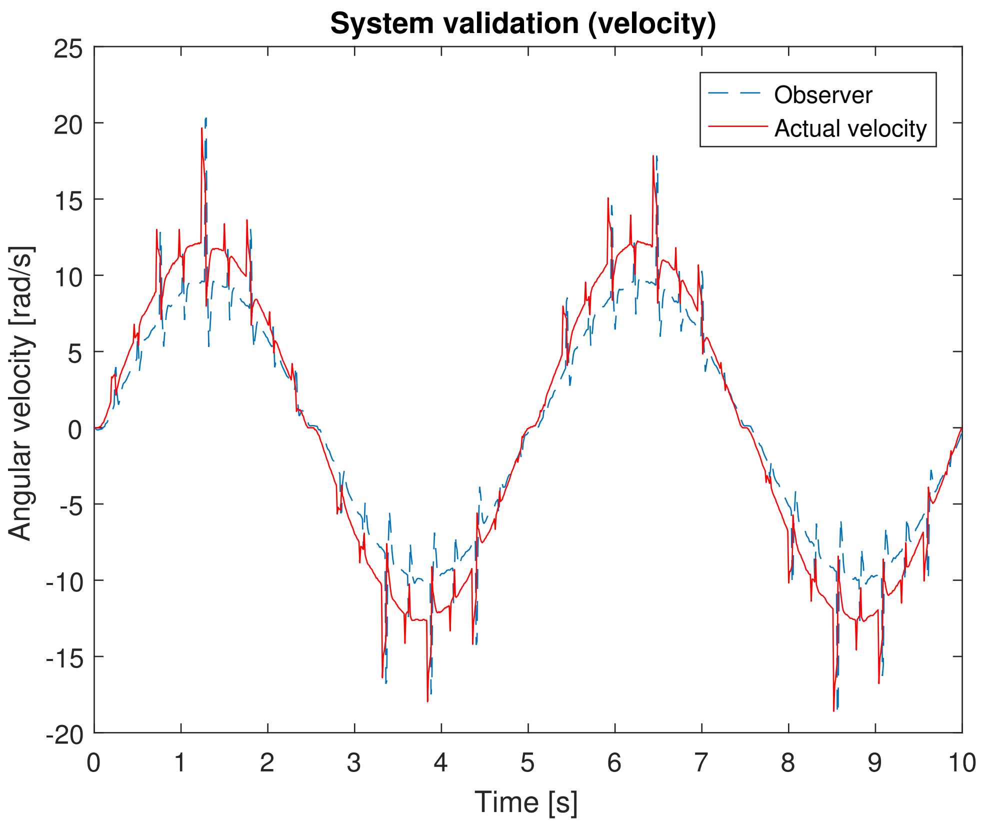

Figure 4: Velocity of the identified system and

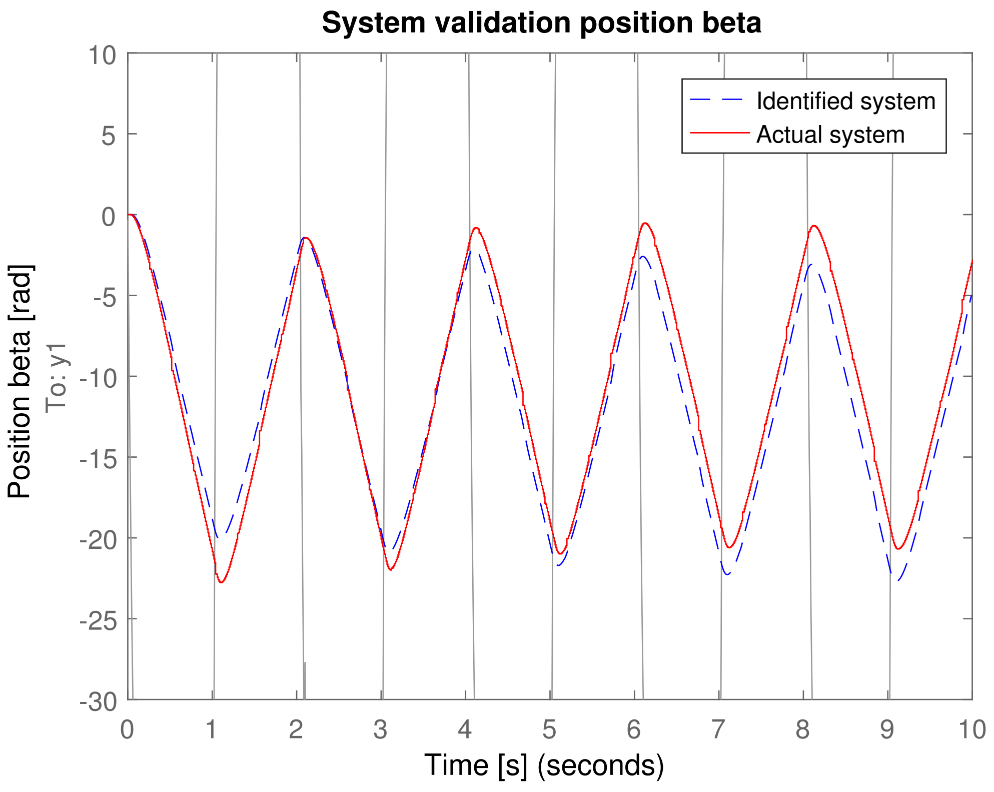

Figure 5: Position of beta simulated and actual position

It has to be noted, that while the system is very accurate for the shown plot, the identified system can also deviate a lot from the actual system for very high, or very low speeds. This is because the slip of the elastic belt is very high for high velocities, and very low for lower velocities, therefor causing non-linearity's in the system. Therefore a linear state space is not the most accurate representation of the system. For now, we will assume that the identified system is good enough for our control purposes.

5 Observer Design

5.1 First system

Our state space system considers two directly measurable states (position & velocity) as well as one unknown inaccessible hidden state. Hence, it becomes necessary to estimate the unknown state from measurements of input and output. Hence, for this purpose a full state observer is built. The system is said to be observable only if the observability matrix has full rank. As our system is already in observable canonical form, we can see that the observability matrix will have full rank. The poles of the observer are tuned by comparing observer's estimated output with actual output of the system in open-loop. As we have discretized the system earlier, we also discretize the poles by taking exponential of product of sampling rate and continuous system poles.
At this time, we will pick a sampling frequency of 100Hz. To prevent aliasing the poles of the observer should be less than Nyquist frequency (50Hz). The selection criteria for observer poles might vary depending on the kind of behaviour that we are expecting from the controller. The main expected behaviour of the system was fast and stable response. Hence, the poles were selected far away from the origin. The poles were placed at -100, -102 & -104 (in rad/sec).

5.2 Second system

As we have identified the second system in observable canonical form, we have an observable system. Hence, now we proceed to built an observer for the second system. As mentioned in the observer design of the first system, the selection criteria for the poles selection is fast and stable response. Hence, the observer poles for second system are places at -100 & -102 (rad/sec).

6 Velocity control of driving motor

The system is considered in two parts for simplicity of control. The first part consists of velocity control of driving motor using reference signal as input. And the second part consists of position control of load considering velocity of first motor as input. Hence this cascading controller will provide stable control of load angle β. In this section let us consider the first part i.e. velocity control of driving motor.

6.1 LQR controller

The basic idea behind LQR control is to drive the states x of a linear system to origin by minimizing the following quadratic cost function [1]:

$$J = \int_{t=0}^{\infty} \frac{1}{2}(x_k^T Q x_k + u_k^T R u_k) dt$$

where, $Q \geq 0$ and $R > 0$.

As we are considering the velocity control, the Q vector will only impose higher weight on velocity state. To remove the influence of other states on the cost function, Q has only zeros for those states. A very small value of constant $R (R = 10^{-8})$ is chosen to reduce the weight on the input signal as we want a very fast response, and the size of the input signal is not important. The final Q and R matrices used are:

$$Q = \begin{pmatrix} 0 & 0 & 0 \\ 0 & 500 & 0 \\ 0 & 0 & 0 \end{pmatrix}$$

$$R = 10^{-8}$$

Then, minimizing the cost function will give us the linear quadratic regulator. The state feedback law will look as follows:

$$u(k) = -K_d x(k) + G_d r(k)$$

where, $G_d = (C(I_n - A_d + B_d K_d)^{-1} B_d)^{-1}$ is unitary steady state gain. Here $C = [0\ 1\ 0]$ which represents the velocity as output.

The layout of the system, controlling the velocity of α is shown in Figure 6. In the figure can be seen, that the input is the reference velocity. The reference velocity is multiplied by the unitary steady state gain G. Then it is summed with the output of the full information state feedback controller K, yielding the system input u. Since, the DC motor can only receive inputs between -1 and 1, the input has to go trough a saturation block, which limits the signal between -1 and 1, such that the observer receives the same input signal as the actual system. The output of the plant (DC motor), is then used together with the input u, to reconstruct the states in observer 1. The observer has 2 outputs, the reconstructed states $\hat{x}$ and the velocity of the first DC motor, or $\dot{\alpha}$. The states are then multiplied with the state feedback controller K, closing the loop. The outputs of the system are the measure angle of the DC motor α, and the angular velocity of the DC motor $\dot{\alpha}$, as reconstructed by the observer.

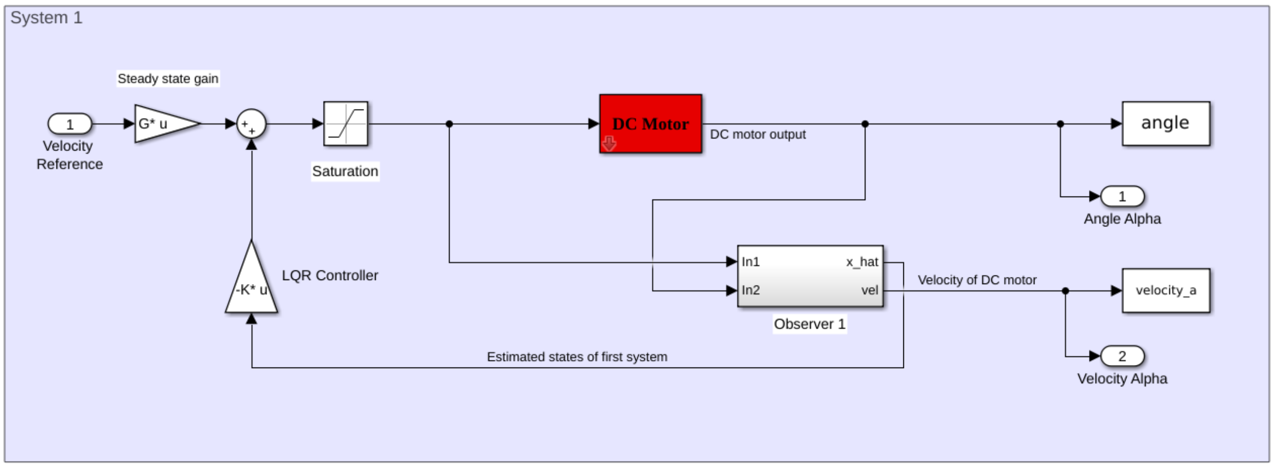

Figure 6: Simulink layout for the velocity control of the first motor using LQR

A sine wave reference tracking for the system with LQR controller is shown in figure 7.

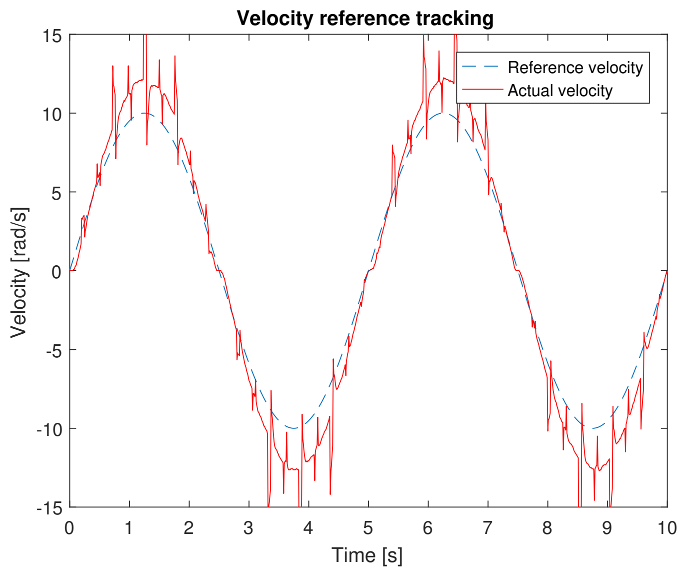

Figure 7: Velocity reference tracking

We can see that the LQR controller tracks the reference velocity. Therefore we will be able to use the velocity as the controlled input for the second system.

7 Position control (β)

Now that we have the controller for the velocity of the driving motor, we continue to the position control of load disk. As discussed earlier, we will be using cascading controllers (i.e. there will be two nested controller loops, inner loop for velocity control of driving motor and outer loop for the control of the load angle β). So our goal is so design a controller, which controls the position of the second wheel β, and does so by giving a reference velocity for the first wheel.

7.1 LQR controller

We will once again use an LQR controller. This time, we do not want the input signals to be to high, as our driving motor can not achieve very high speeds. Therefore we start by setting the R matrix to 1. For the Q matrix, we only want to put weight on the first state, which represents the position of the second wheel. Our final Q and R matrices are:

$$Q = \begin{pmatrix} 500 & 0 \\ 0 & 0 \end{pmatrix}$$

$$R = 1$$

Minimizing the cost function will give us a full information state feedback controller K_d for the state feedback law:

$$u(k) = -K_d x(k) + G_d r(k)$$

where, $G_d = (C(I_n - A_d + B_d K_d)^{-1} B_d)^{-1}$ is unitary steady state gain.

With this controller, and the controller designed in Section 6, we can complete our cascading controller. The Simulink schematic for the two cascading controller can be seen in the Figure 8. Both cascading controllers consist of a observer, which estimates the states. The states then go to a state feedback gain K, which is calculated using the LQR method described. Then, the sum between the state feedback gain, and the reference multiplied with the unitary steady state gain G is added, to give the controller output. For system 2, the controller output becomes the reference for system 1. For system 1, the controller output is the input for the driving motor.
Now we will plot the step response and a disturbance rejection scheme of the complete system, to see how the controllers preform. The step response simply changes the reference from 0 to 3 rad at $t = 1$. The disturbance rejection is plotted by keeping the reference constant at 0, and giving a small impulse to the second motor. The step response and disturbance rejection are plotted in Figure 9 and Figure 10 respectively.

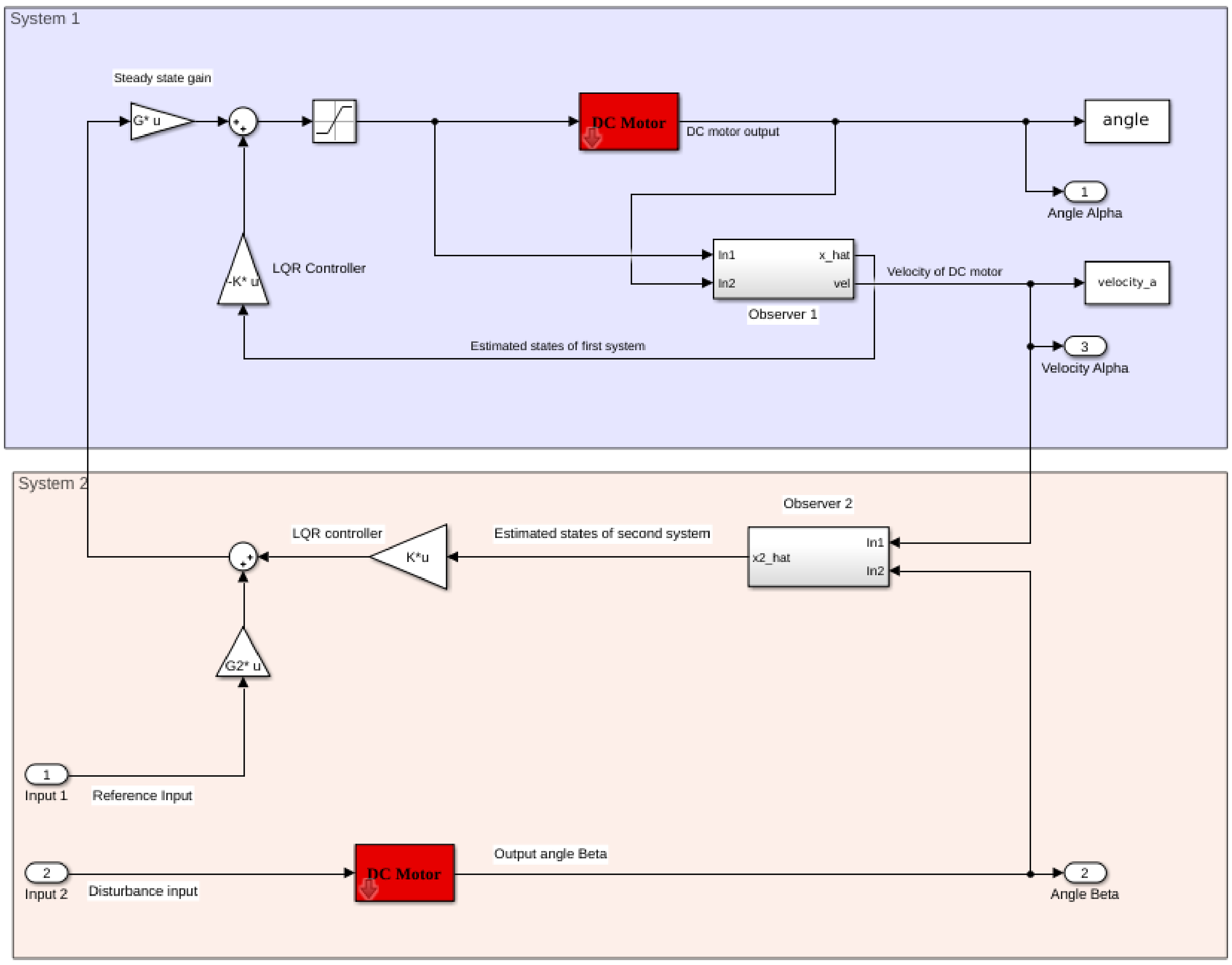

Figure 8: Simulink layout for the two cascading LQR controllers

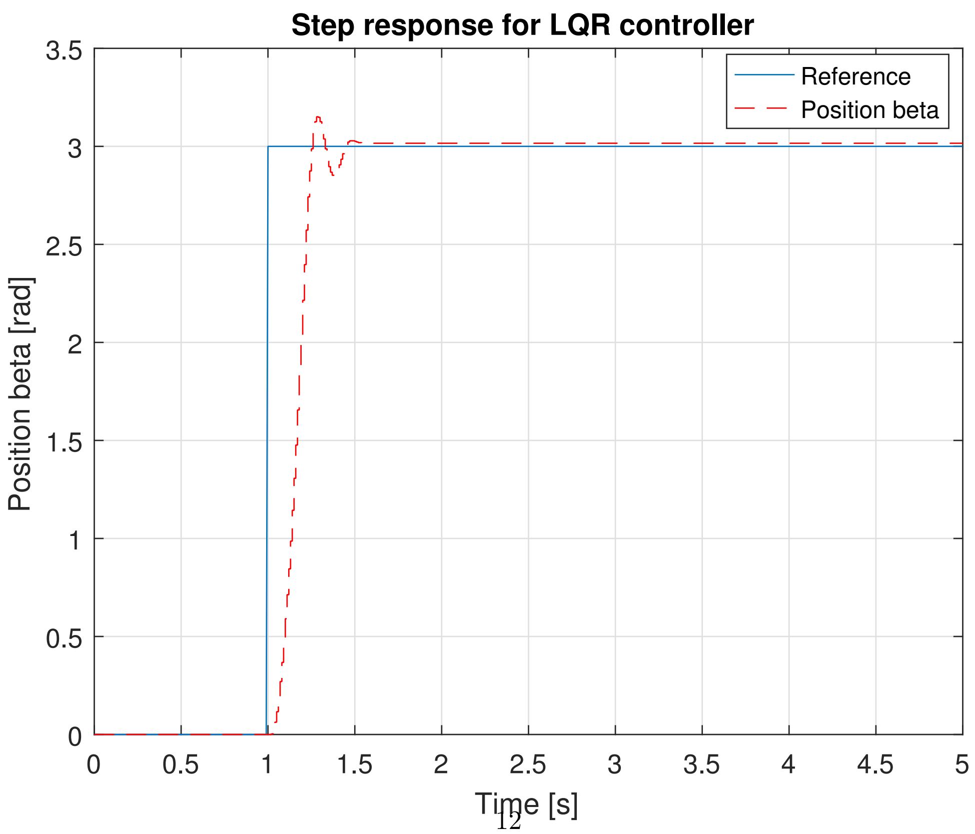

Figure 9: Step response for the 2 cascading LQR controllers

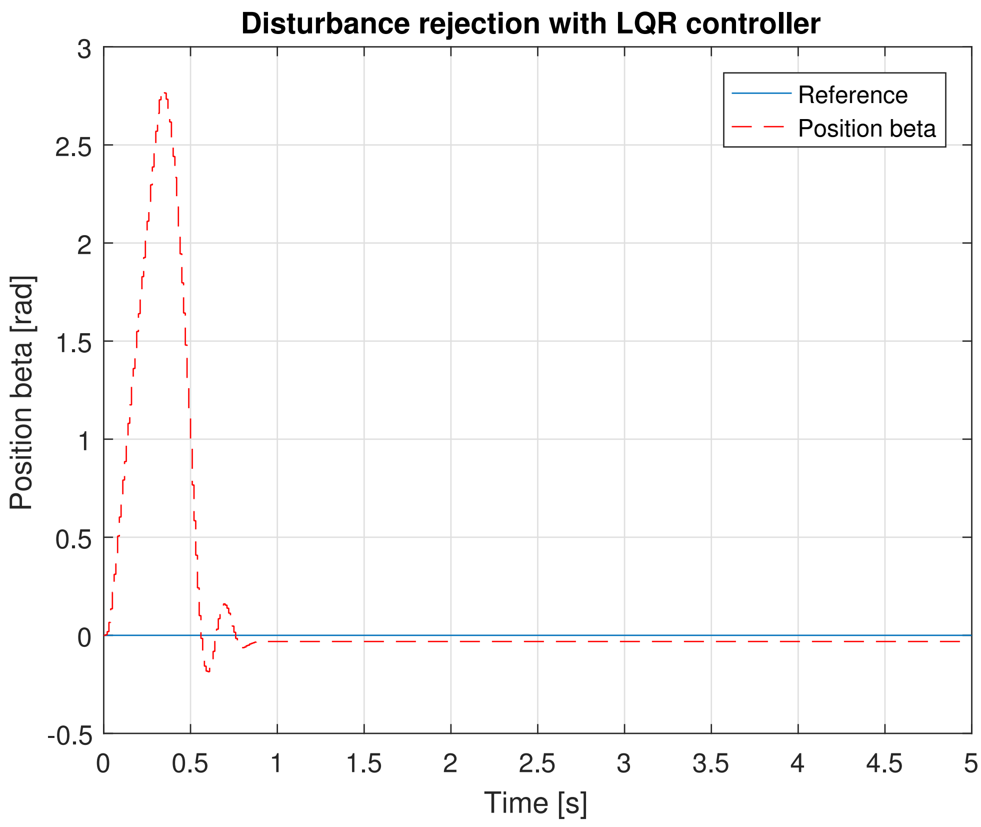

Figure 10: Disturbance rejection for the 2 cascading LQR controllers

We can see that the system has a fast response, and small overshoot ($< 10\%$). The system also has fast disturbance rejection.

We do however also see that the wheel does not move to 0% steady state error. This is most likely due to the friction of the motors being to high to be overcome for a very small input signal. Therefore we will implement an LQI controller. This is similar to the LQR controller, however, an extra state is added which is equal to the integral of the error signal, to make sure that the final controller will not have any steady state error.

7.2 LQI controller

Since the LQR controller results in a small steady state error, we will need to add an integral term to the controller, to make sure it move to 0% steady state error. The LQI controller consists of a similar full information state feedback controller K, however there will no also be an extra state added to the system. This state is equal to the integral of the error signal (the reference signal minus the output).

We will use the same Q and R matrices a for the LQR controller, however, we will need to add an extra row to Q for the extra state. We use a fairly high number for the last state, since we want the output to move back to 0% steady state error rather quickly. The Q and R matrices used are:

$$Q = \begin{pmatrix} 500 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 100 \end{pmatrix}$$

$$R = 1$$

Using the MATLAB function `lqi`, we get a LQI controller for our system. The new system, with the intergral term added to the controller is plotted in Figure 11. In this figure we can see that only the second system changed. Now, the difference between the reference and the output is calculated which results in the error signal. This is then send trough an integrator block. Then the integral of the error signal is added to the states observed states of the system, as an extra state. The state feedback gain K will therefore have one extra number, compared to the previous layout.

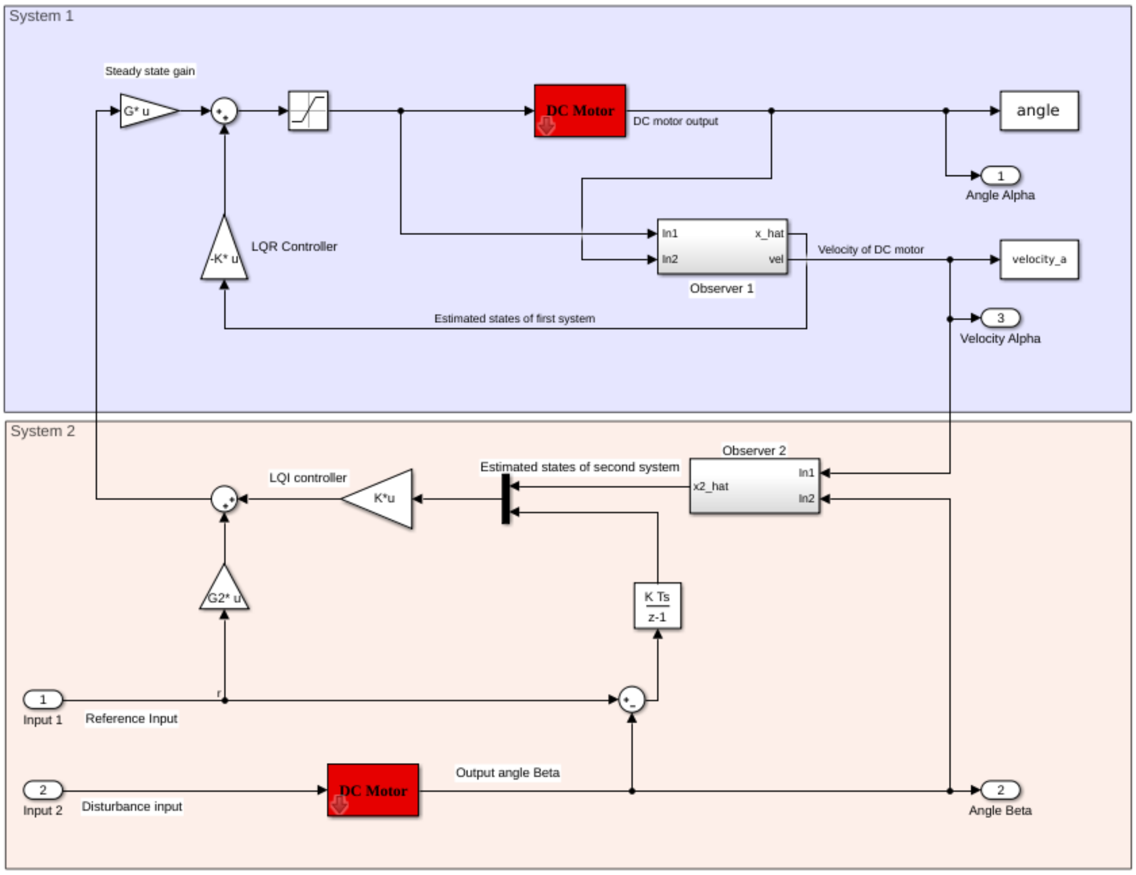

Figure 11: Simulink layout for the LQI controller

The step response and disturbance rejection schemes of the system, with LQI controller are plotted in Figure 12 and Figure 13 respectively.

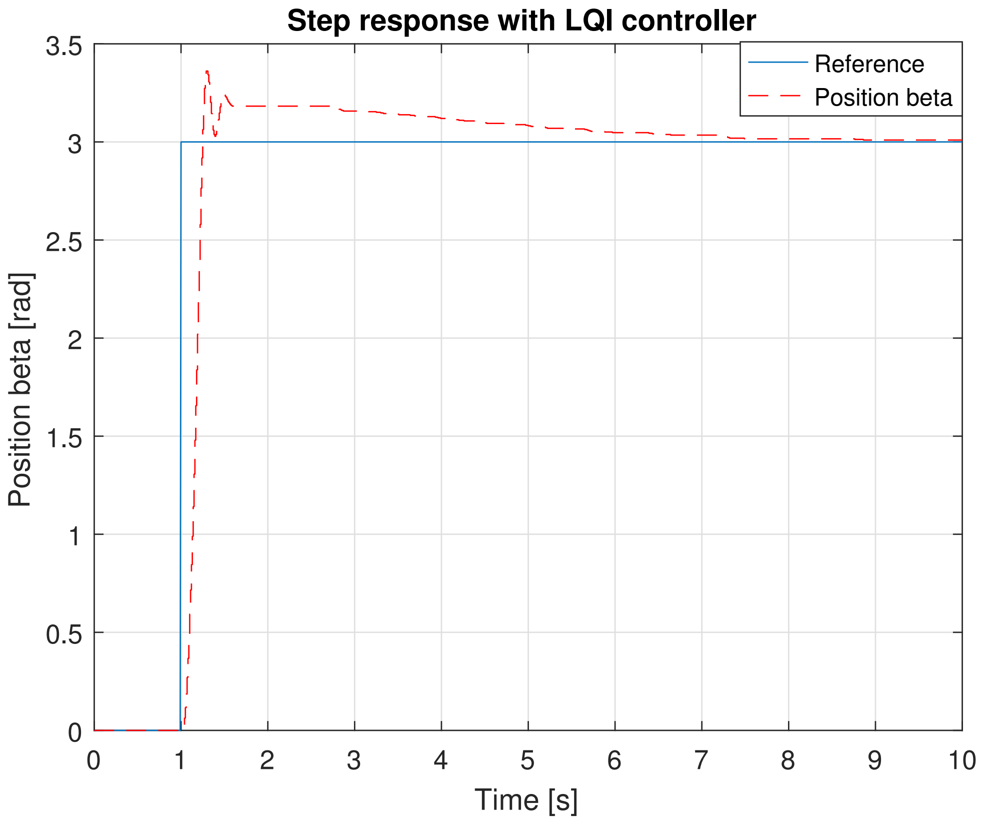

Figure 12: Step response for the 2 cascading controllers, with an integral term for the second controller

From the step response in Figure 12, we can see that the initial overshoot is actually a bit more than for the original LQR controller. However, the system does move back to 0 % steady state error as time goes to infinity. It is possible to make the system converge to the reference quicker, by increasing the integral term in the Q matrix, however, this will also increase the initial overshoot.

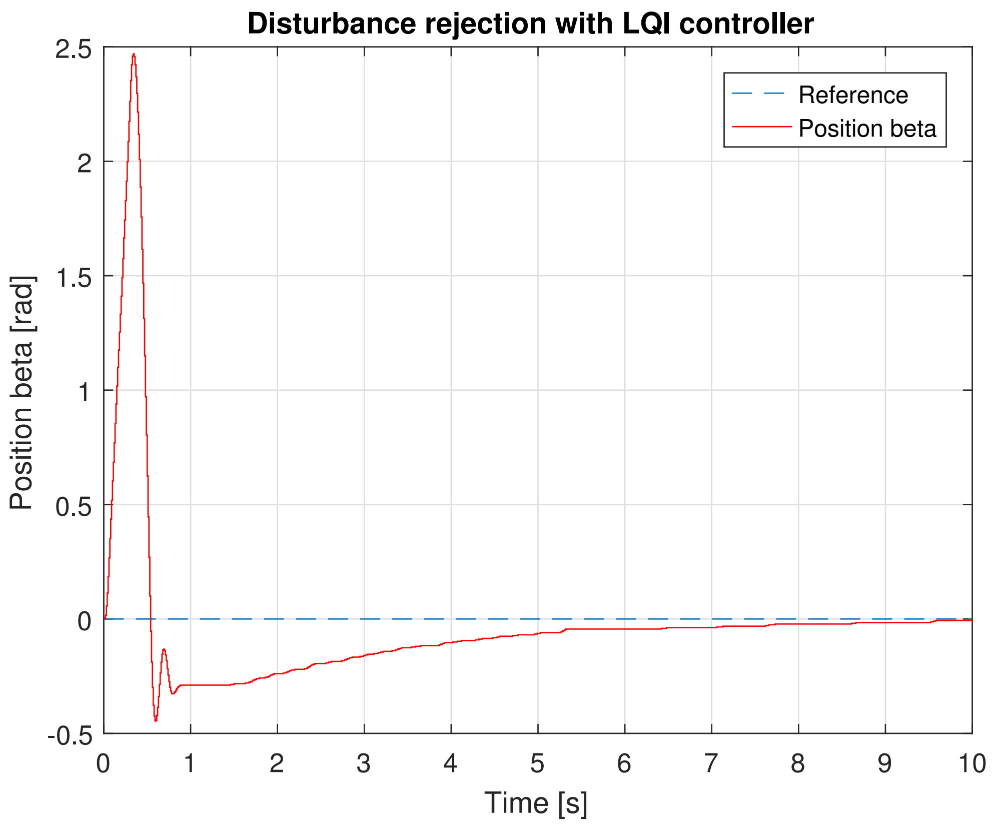

Figure 13: Disturbance rejection for the 2 cascading controllers, with an integral term for the second controller

From the disturbance rejection, plotted in Figure 13, we can also see that the initial overshoot is quite a bit larger than for the original controller, however, the system does move to 0% steady state error.

7.3 PID Controller

In the initial stage of the project, we tried using a PI controller to control the load angle β directly from the input of reference signal. The purpose was to understand the effect on nonlinearity on the system response (for a linear controller). The high nonlinearity due to the slip results in high overshoots in the response during fast accelerations. The response of the PI controller was discussed in the Section 2 of the report.

As a second controller, we selected the PID controller due to its response characteristics. The good response of the PID controller can be justified by the fact that second subsystem is decoupled from the first subsystem. Hence, effect of nonlinear slip does not hold anymore. We now tune the PID controller using bode plot.

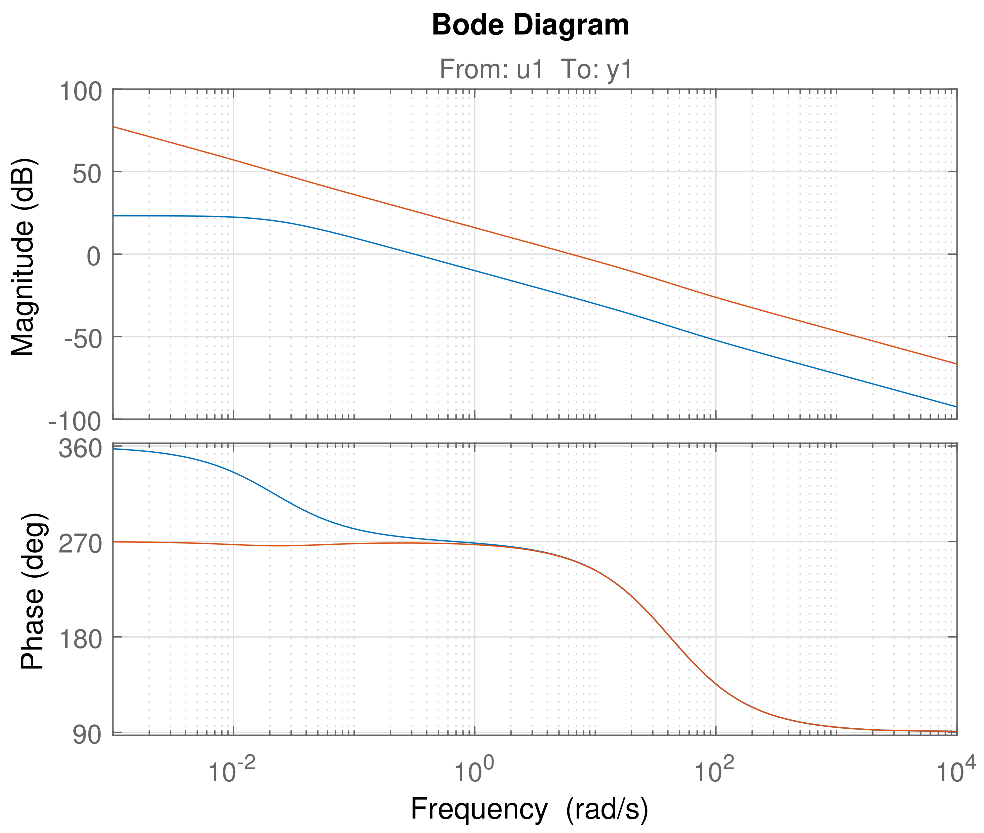

Figure 14: Bode plot for original system (shown in blue), and bode plot for tuned PID controller (shown in red)

Figure 14 shows the bode plot for the second system. It can be seen that it is fairly easy to tune the PID controller for this subsystem. To increase the bandwidth we use the proportional controller. We added integral term to increase the gain at lower frequencies to remove steady state errors. We also added the derivative term to increase the phase for higher frequencies, resulting in a better damped system. Figure 15 shows the system response of the system.The overshoot of the system is very small and controller has fairly fast response.

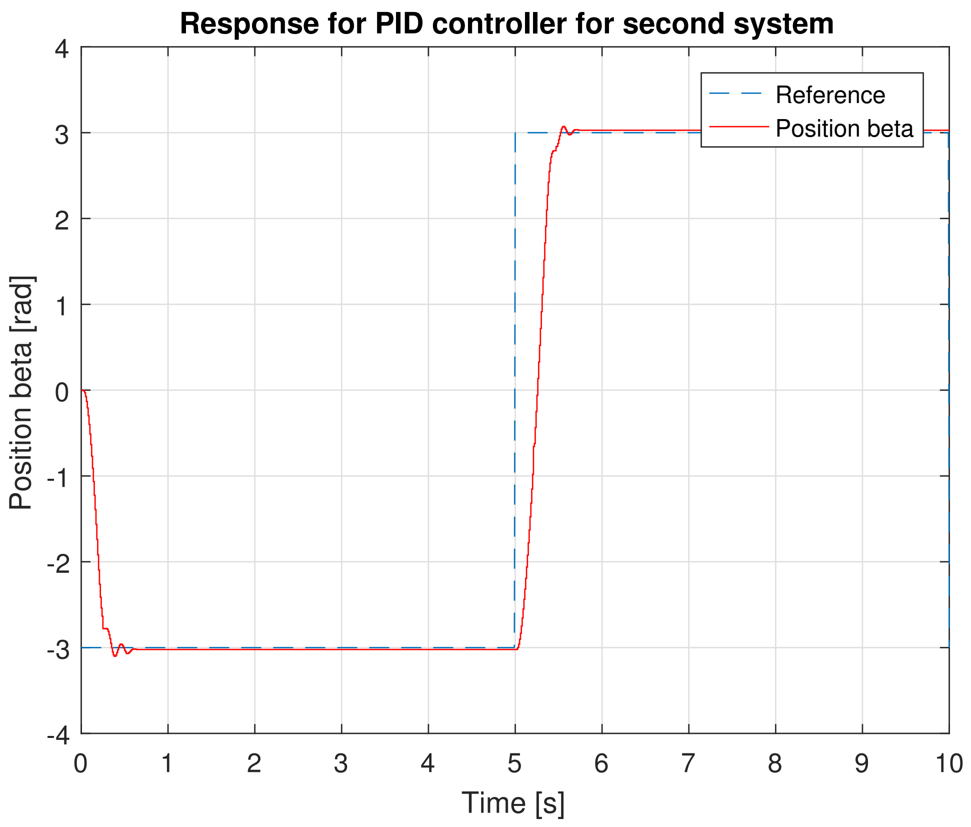

Figure 15: System response for cascading LQR controller (inner loop) and PID controller (Outer loop)

8 Conclusion

For designing a suitable controller for the flexlink setup, we identified and validated a linear model for both motors separately, to make sure slip does not cause the linear models to be inaccurate. The first system give good results for all inputs, however, the slip of the elastic belts makes it so the linear model is only accurate at certain speeds.

We then used the identified systems, to construct two observers, one for each motor, such that we could use state feedback control.

Then we used the identified systems, and the state observers, to make two state feedback controller using the LQR method. From the plots can be seen that these controllers give a fast response, and small overshoot and settling times, for both a step response an disturbance rejection.

The LQR controller do however have a small steady state error. This motivates us to change the controller for the second system to an LQI controller, which adds an integral term to the system, to make sure that the position of the second wheel does go to the reference, when time goes to infinity. The LQI controller does cause the overshoot and the settling time to be significantly larger.

Finally, we tried the PID controller to control the second system. Although initial PID controller for the black box system (Figure 2) failed to perform well, when we decouple two systems, PID for second subsystem performs faster and more accurate than before. The bode plot of the second system shows that bandwidth of the second subsystem (Figure 14) is higher when compared to frequency response of full system (Figure 1). This can prove that decoupling the systems has helped to minimize the non-linearity in the system.

References

Friedland, B. (2005). *Control system design: An introduction to state-space methods.* Dover Publications Inc.

University of Michigan. (n.d.). *DC motor position: System modeling.* Retrieved 2018-06-21, from `http://ctms.engin.umich.edu/CTMS/index.php?example=MotorPosition&section=SystemModeling`