EEEN 415: Assignment 2

H. S. Chin; 300493343

2nd of September 2022

Section A - Formative Questions

$$A = \begin{bmatrix} 12.5314 & -91.36 & 28.7129 & 59.9628 \\ 21.316 & -115.6631 & 37.5584 & 75.0519 \\ 13.967 & -53.5817 & 15.4161 & 33.4391 \\ 20.5222 & -123.3107 & 42.267 & 79.7156 \end{bmatrix}$$

$$B = \begin{bmatrix} 1.7649 & 1.7649 \\ 2.8318 & 2.8318 \\ 2.2353 & 2.2353 \\ 2.7294 & 2.72947 \end{bmatrix}$$

$$C = \begin{bmatrix} -0.46166 & -4.4635 & 1.9151 & 3.7274 \\ 1.0853 & -12.82 & 4.5753 & 8.3759 \end{bmatrix}$$

$$D = \begin{bmatrix} 0 & 0 \\ 0 & 0 \end{bmatrix}$$

---

a

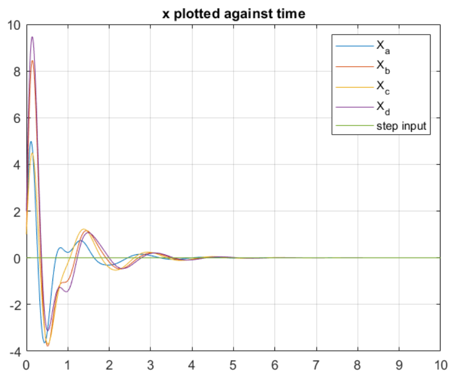
poles of the open loop system are stated below

$$poles = \begin{cases} -3 + 9.9996i \\ -3 - 9.9996i \\ -1 + 4i \\ -1 - 4i \end{cases}$$

---

b

the MIMO system can be plotted against time shown in Figure 1. This was done by creating an array for t $\in [0, 10]$ and having 1001 samples between them (including 0 and 10). As this is a autonomous linear dynamical system, $\mathbf{u} = 0$.



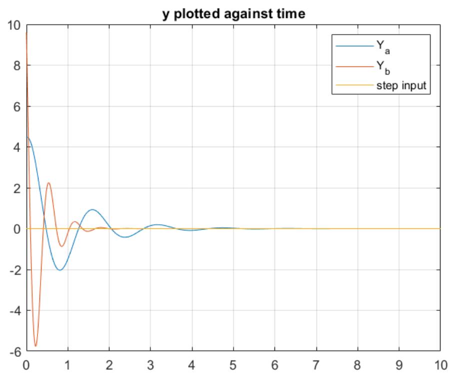


Figure 1: x and y plotted against time and a step input

These plots also required the initial condition below.

$$x(0) = \begin{bmatrix} 1 & 1 & 1 & 2 \end{bmatrix}^T$$

c

$$\phi(t) = e^{At}$$
$$\mathcal{X}_o(t) = \phi(t) * x(0)$$
$$\mathcal{Y}_o(t) = C * \mathcal{X}_o(t)$$

Creating ϕ for the matrix exponential will require a 3d matrix, this 3d matrix will 4 x 4 x length(t) (1001). This is due to the dimensions of A; being a 4 x 4 matrix and the total length of the interval from question b. This 3d matrix will be multiplied with the system's initial state which will make each iteration of time, a vector of X; thus a 4 x 1001 matrix is created and set to $\mathcal{X}_o(t)$. Consequently, $\mathcal{Y}_o(t)$ can be calculated by multiplying $\mathcal{X}_o(t)$ with C as shown in Figure 2.

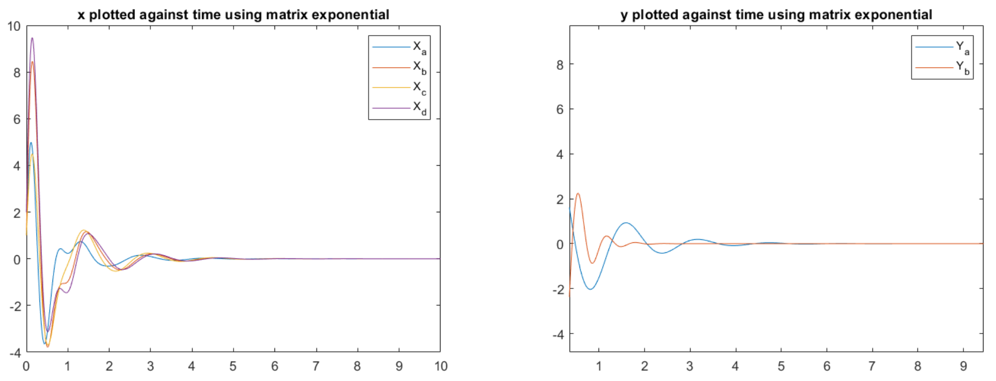


Figure 2: x and y plotted against time using matrix exponential

---

d

$$\mathbf{u} = \begin{bmatrix} u(t-1) & 2u(t-5) \end{bmatrix}^T$$

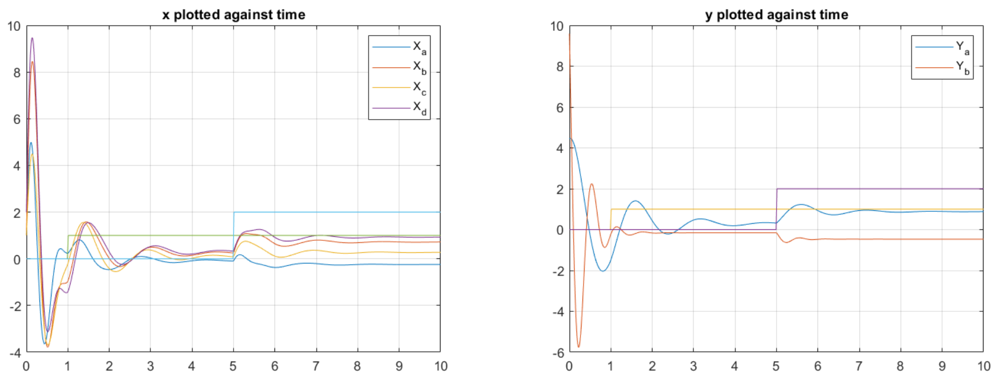


Figure 3: x and y plotted with step inputs of **u**

Using the **u** input stated above, the step response provided by **lsim** can be observed in Figure 3.

e

To calculate the appropriate sampling period of the system, the poles of the system must be observed as shown below. The magnitude of the fastest pole can be doubled to extract the minimum sampling frequency required to accurately reconstruct the discrete system. This resulted in a sampling period of 0.0479 s, hence rounding it to 0.05 s.

$$|poles| = \begin{bmatrix} 10.4399 \\ 10.4399 \\ 4.1231 \\ 4.1231 \end{bmatrix}$$

$$\begin{aligned} t_s &= \frac{1}{2 * 10.4399} \\ &= 0.0479s \end{aligned}$$

Using a sampling period of 0.05 s, the step response of the system will look very similar to the continuous time step response. To convert the continuous time state space model to discrete, **c2d** would be used along with the sampling period. This will generate a discrete state space model as shown below.

$$\hat{\mathbf{x}} = \begin{bmatrix} 1.117 & -0.8878 & 0.2819 & 0.5806 \\ 0.2063 & -1464 & 0.3758 & 0.741 \\ 0.1362 & -0.536 & 1.56 & 0.3334 \\ 0.1978 & -1.216 & 0.4209 & 1.783 \end{bmatrix} x + \begin{bmatrix} 0.01719 & 0.01719 \\ 0.02823 & 0.02823 \\ 0.02227 & 0.02227 \\ 0.02722 & 0.02722 \end{bmatrix} u$$

$$\mathbf{y} = \begin{bmatrix} -0.4617 & -0.8878 & 1.915 & 3.727 \\ 1.085 & -12.82 & 4.575 & 8.376 \end{bmatrix} x + \begin{bmatrix} 0 & 0 \\ 0 & 0 \end{bmatrix} u$$

This discrete state space model can be plotted against time as shown in Figure 4.

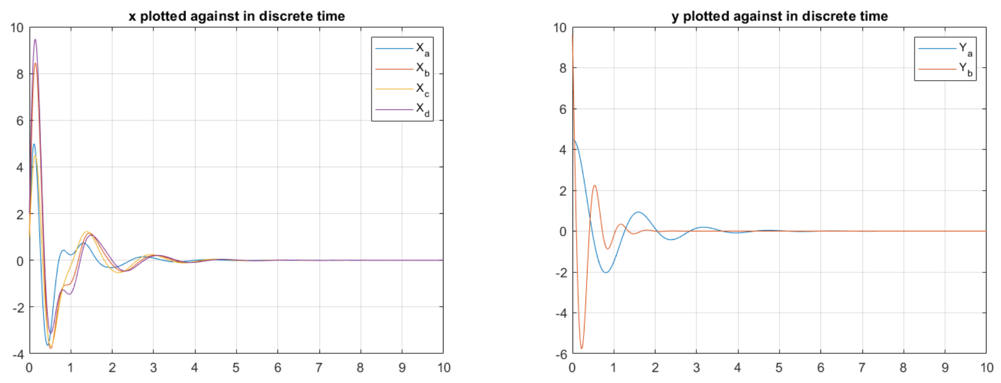


Figure 4: x and y plotted in discrete time

f

Using the previously obtained sample period of 0.05 s for the discrete state space model. The matrix exponential can be created using the equations below. **t** represents each time set. Each iteration of time will produce a matrix multiplication of A, thus modelling the state space model in discrete time.

$$\mathbf{X}[t] = A_{dis}^{t} * x[0]$$
$$\mathbf{Y}[t] = C_{dis} * \mathbf{X}[t]$$

The matrices below can be obtained from the discrete state space model produced in question e.

$$\mathbf{A_{dis}} = \begin{bmatrix} 1.117 & -0.8878 & 0.2819 & 0.5806 \\ 0.2063 & -1464 & 0.3758 & 0.741 \\ 0.1362 & -0.536 & 1.56 & 0.3334 \\ 0.1978 & -1.216 & 0.4209 & 1.783 \end{bmatrix} \quad \mathbf{C_{dis}} = \begin{bmatrix} -0.4617 & -0.8878 & 1.915 & 3.727 \\ 1.085 & -12.82 & 4.575 & 8.376 \end{bmatrix}$$

Finally, the discrete state space model can be determined directly using powers of **A** as shown in Figure 5.

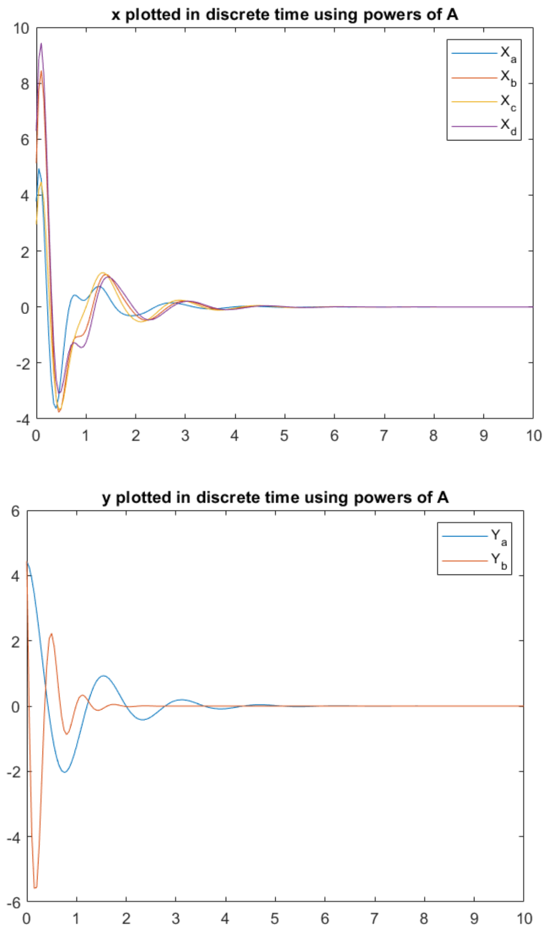


Figure 5: x and y plotted in discrete time using powers of **A**

g

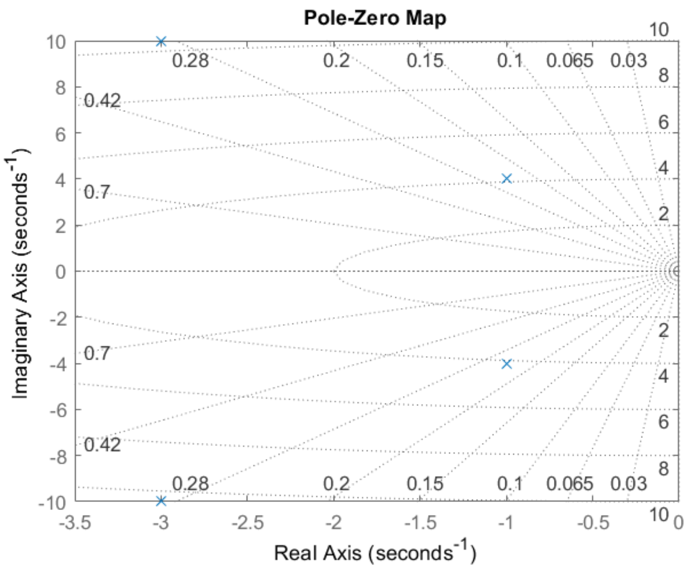


Figure 6: poles of the state space model

To determine the butter-worth filter's cutoff frequency, we must observe the poles of the continuous time, state-space system. By calculating the magnitude of the poles in Figure 6, we can calculate the speed of the open loop poles as shown in the vector below.

$$|poles| = \begin{bmatrix} 10.4399 \\ 10.4399 \\ 4.1231 \\ 4.1231 \end{bmatrix}$$

The speed of the slowest pole (4.1231) can be doubled for Butterworth's filter cut-off frequency to generate the controller (K) using the **butter** and **place** commands. The butter-worth filter generates a new set of poles, these new poles will be placed using the **place**, producing a 2 x 4 matrix as a proportional controller. Furthermore, these new sets of poles can also be calculated by finding the eigenvalues of $(\mathbf{A} - \mathbf{BK})$.

$$poles_{new} = \begin{bmatrix} -7.6185 \pm 3.1557i \\ -3.1557 \pm 7.6185i \end{bmatrix}$$

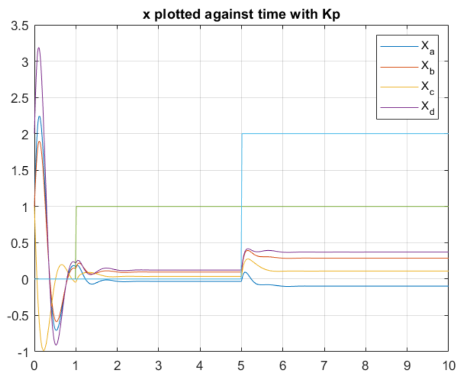
$$K = \begin{bmatrix} 0.4456 & -11.4177 & 5.0543 & 9.9005 \\ 0.4456 & -11.4177 & 5.0543 & 9.9005 \end{bmatrix}$$

The **K** controller can be fitted with **ss** command and the closed loop, state space system would be produced as shown in the command below.

$$system_{closed} = \mathbf{ss}(A - BK, B, C, D)$$

h

Using the same inputs as question d, the outputs of the closed-loop control system can be shown below. As expected, the closed loop, controlled system with a proportional controller has shown to have reduced oscillation. This can be compared to Figure 3, where the X and Y plots are more volatile. However, As shown in Figure 7, the X and Y output plotted against the step input (r) has shown to have a greater margin error.



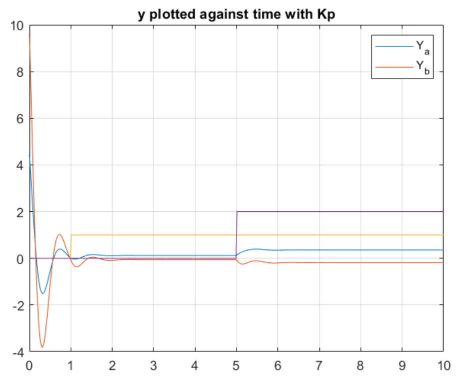


Figure 7: Closed loop configuration of the state space system with K_p plotted with r

$$u = -K\hat{x}$$

The **u** values can be calculated by multiplying K with the **x** output of the system as shown in the equation above. The 2 **u** values (shown in Figure 8) are the same for this MIMO system as they are now the intrinsic value of the system fed into the observer.

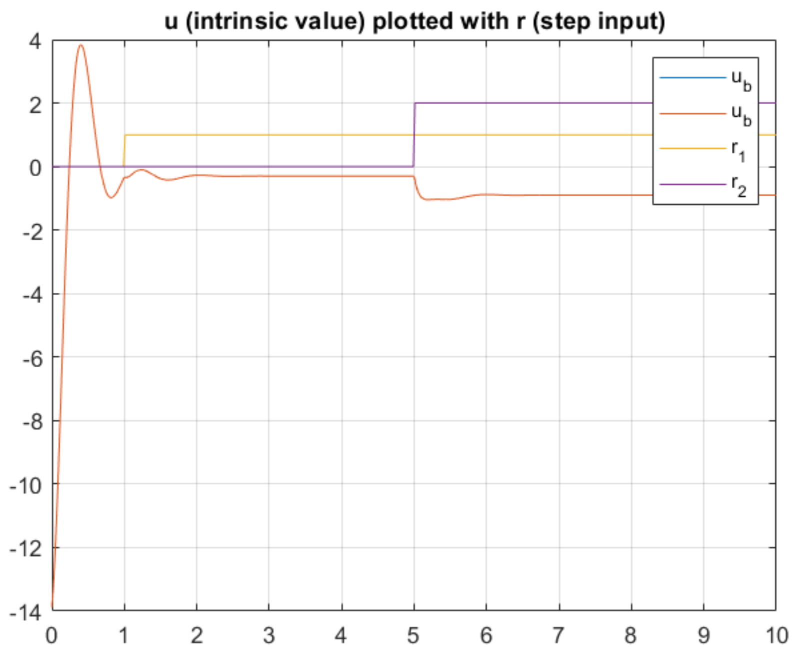


Figure 8: u plotted with r

i

The speed of the butterworth's pole is determined by its cut-off frequency. the cutoff frequency is increased from 2*4.1231 to 3*4.1231. As a result, the **u** can be plotted accordingly. As shown in the plot below, the intrinsic value tends to oscillate at a higher amplitude as the cutoff frequency increase.

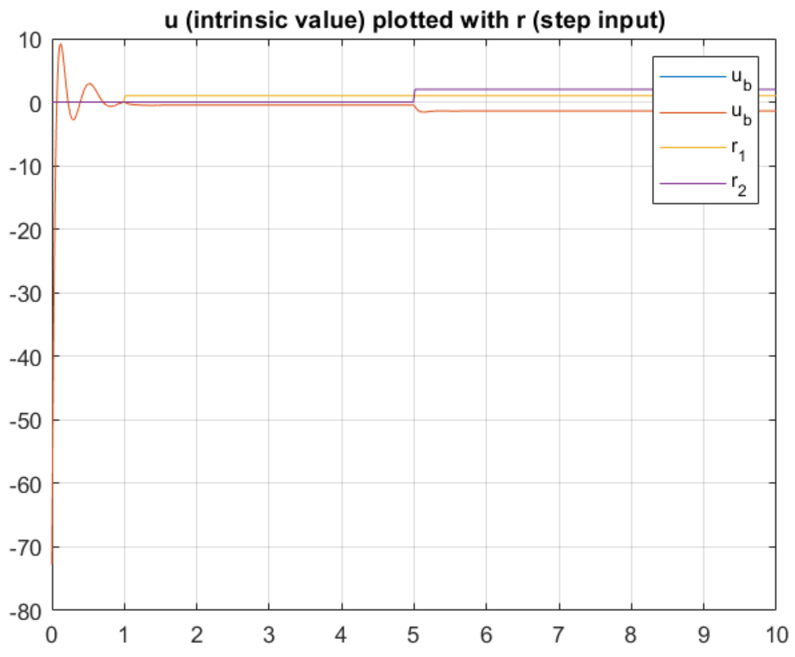


Figure 9: u plotted with r with different Butter-worth poles

---

j

When the discrete-time model was used with K_p we can expect to see similar results. However, factors like sampling rate will affect the transient response of the system as shown in Figure 10. This could be the result of sampling delays.

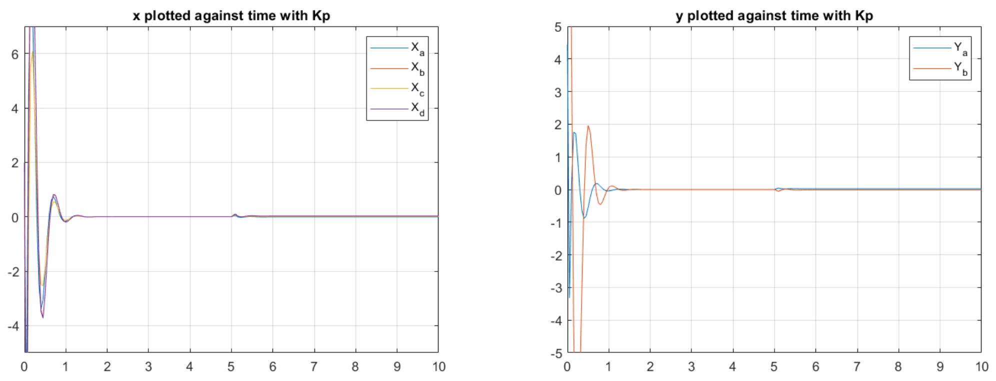


Figure 10: u plotted with r with different Butter-worth poles

Section B - Summative Questions

$$A = \begin{bmatrix} 1 & t_s & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & t_s \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$B = \begin{bmatrix} 0 & 0 & 0 \\ 10 & 0 & -0.6 \\ 0 & 0 & 0 \\ 0 & 10 & 0.4 \end{bmatrix}$$

$$C = \begin{bmatrix} 0 & 0.5 & 0 & 0.5 \\ 0 & -2 & 0 & 2 \end{bmatrix}$$

$$D = \begin{bmatrix} 0 \end{bmatrix}$$
$$t_s = 1s$$

---

a

When the operation of the open-loop grader drives forward at its nominal setting ($u_1 = u_1 = 0$). It encountered some difficult dirt such as the presence of a unit step, at $t = 1$. The open-loop system has shown to be unstable when provided with a step input as observed in Figure 11. This is expected as all poles of this system are on the RHS of the s plane.

$$poles = \begin{cases} 1 \\ 1 \\ 1 \\ 1 \end{cases}$$

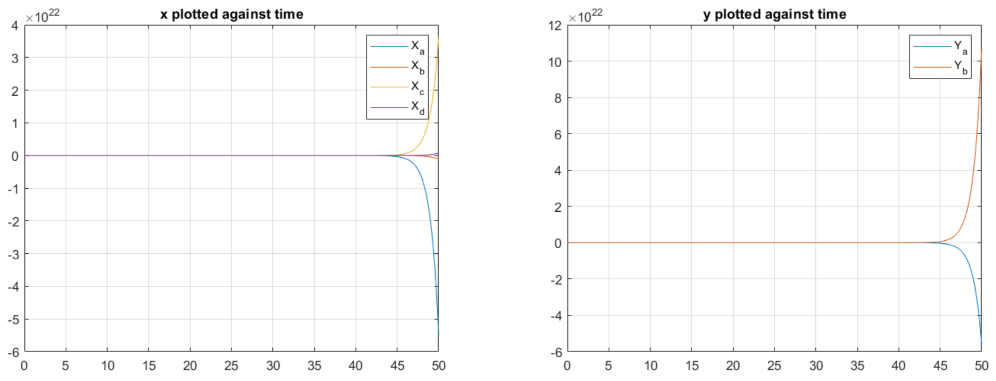


Figure 11: X and Y plotted against time

b

To design a discrete-time compensator that can deal with momentary perturbation in the soil properties a Butterworth's compensator can be used. Setting the cut-off frequency to 0.5, as that is half the speed of the slowest pole. This would achieve a settling time of less than 20 s when given an impulse at 5 s as shown in Figure 12. As it is in discrete time, the Butterworth's configuration will produce new poles of $0.1989i, 0.1989i, 0.6682i, 0.6682i$, resulting in a K_p as shown below.

$$K_p = \begin{bmatrix} 0.1133 & 0.1997 & -0.0465 & -0.0463 \\ 0.0471 & 0.0473 & 0.1127 & 0.1992 \\ -0.0049 & -0.0101 & 0.0073 & 0.0107 \end{bmatrix}$$

Using this controller, the system produces a transient response as shown in Figure 12. As expected, the controller made improvements compared to the open-loop system. Unfortunately, it can be seen in the Y plot that the rightward curve of the grader would oscillate. Furthermore, the oscillation of the track's position may affect road grader performance metrics. This may result in workplace hazards for the driver and could be addressed with an integral controller.

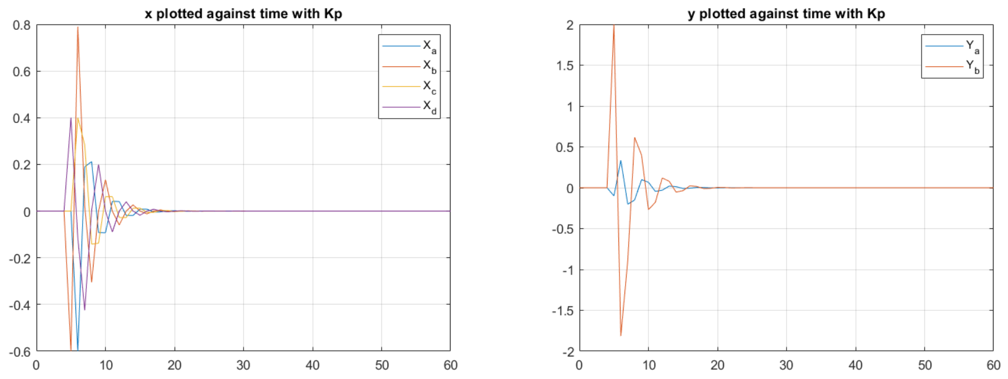


Figure 12: X and Y plotted with K_p against time

c

To design an integrator, another state would variable would need to be added. This will produce an augmented matrix with a size 5 x 5, which can be written as. As the purpose of an integral controller is to integrate the difference between the reference input and the output, this can be represented by the new state x_i where it considers the output's error via the equation y = -Cx.

$$\begin{bmatrix} x \\ x_i \end{bmatrix} = \begin{bmatrix} A & 0 \\ -C & 1 \end{bmatrix} \begin{bmatrix} x \\ x_i \end{bmatrix} + \begin{bmatrix} B \\ 0 \end{bmatrix} u(t) + \begin{bmatrix} 0 \\ 1 \end{bmatrix} r(t)$$

$$y = \begin{bmatrix} -C & 0 \end{bmatrix} \begin{bmatrix} x \\ x_i \end{bmatrix}$$

Only the 2nd row of y will be used in the augmented matrix as we observe the rightward curvature of the road grader. The state space system can be observed below. The extra 1 in the augmented matrix is used for the discrete-time system.

$$\begin{bmatrix} x_1(t+1) \\ x_2(t+1) \\ x_3(t+1) \\ x_4(t+1) \\ x_i(t+1) \end{bmatrix} = \begin{bmatrix} 1 & 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 1 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & -2 & 0 & 2 & 1 \end{bmatrix} \begin{bmatrix} x_1(t) \\ x_2(t) \\ x_3(t) \\ x_4(t) \\ x_i(t) \end{bmatrix} + \begin{bmatrix} 0 & 0 & 0 \\ 10 & 0 & -0.6 \\ 0 & 0 & 0 \\ 0 & 10 & 0.4 \\ 0 & 0 & 0 \end{bmatrix} u(t) + \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 1 \end{bmatrix} r(t)$$

$$\begin{bmatrix} y_1 \\ y_2 \end{bmatrix} = \begin{bmatrix} 0 & -0.5 & 0 & -0.5 & 0 \\ 0 & 2 & 0 & -2 & 0 \end{bmatrix} \begin{bmatrix} x_1(t) \\ x_2(t) \\ x_3(t) \\ x_4(t) \\ x_i(t) \end{bmatrix}$$

After creating this system, Matlab can generate a K_i controller using the augmented A, augmented B and some predefined poles. However, the 3rd column of the augmented B should be removed to avoid considering the dirt as part of the controller. As this is a discrete-time system, the desired poles must be within the unit circle, thus [1,1,0.7,0.7,**0.5**] was chosen. **0.5** was selected as the additional pole due to being half the magnitude of the open loop poles.

$$A_i = \begin{bmatrix} 1 & 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 1 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & -2 & 0 & 2 & 1 \end{bmatrix} \qquad B_i = \begin{bmatrix} 0 & 0 \\ 10 & 0 \\ 0 & 0 \\ 0 & 0 \\ 0 & 10 \end{bmatrix} \qquad poles_{new} = \begin{bmatrix} 1 \\ 1 \\ 0.7 \\ 0.7 \\ 0.5 \end{bmatrix}$$

Using the previously generated augmented matrices as arguments for the **place** command, an integral controller K_i was generated. The additional zeros that were integrated as part of the augmented matrix were there to satisfy the parameters of matrix multiplication.

$$K_i = \begin{bmatrix} 0 & 0.0050 & 0 & 0.0250 & 0.0088 \\ 0 & 0.0250 & 0 & 0.0050 & -0.0087 \end{bmatrix}$$

The Closed loop PI controller can be constructed similarly to **b** through Matlab's **lsim** command, producing a viable transient response as shown in Figure 13. The rightward curve of the grader is expected to increase at 5 seconds, due to the soil resistance, before reaching a steady state of 0 as intended. Presenting a significant improvement compared to just using a proportional controller as shown in Figure 12. The green line represents the new state variable x_i

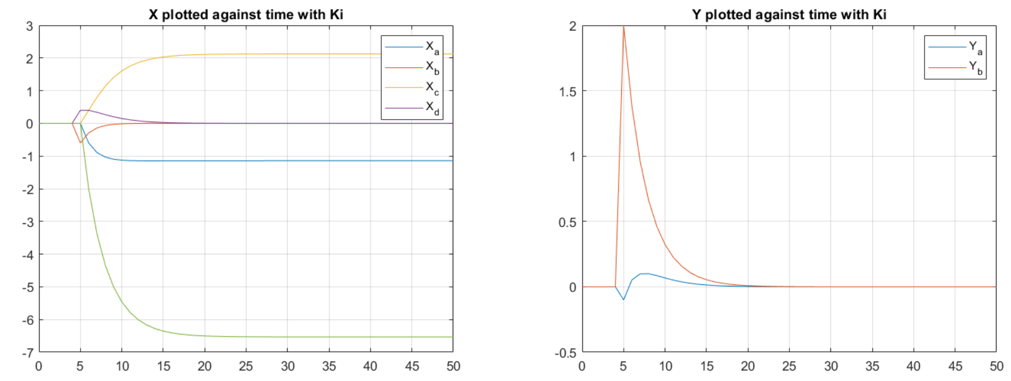


Figure 13: X and Y plotted with K_i against time, with an impulse soil resistance

The limitation of this integral controller is if the reference input is a step input. This will cause the tracks to diverge as the system cannot handle high soil resistance for a long time. Hence, this causes the rightward curve of the grader to reach steady steady state at a high angle as shown in Figure 14.

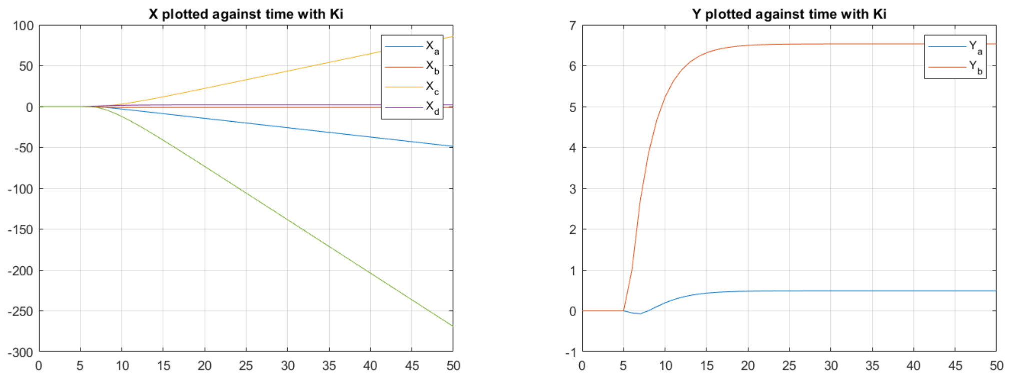


Figure 14: X and Y plotted with K_i against time, with high soil resistance over a long period

d

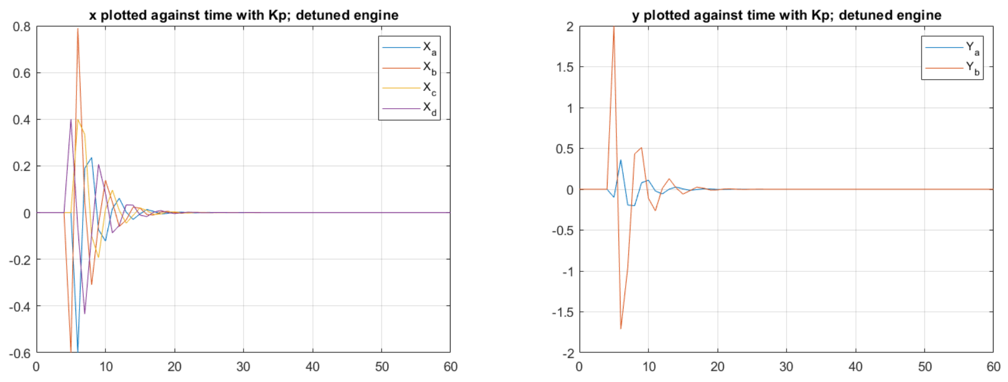


Figure 15: Detuned engine: X and Y plotted with K_p against time

As the controllers were created before the detuning of the right grader engine. The controllers would remain the same as we do not need to recreate both K_p and K_i. It is expected such a minute difference would **not** affect the transient response of both controllers, as illustrated in Figures 15 and 16.

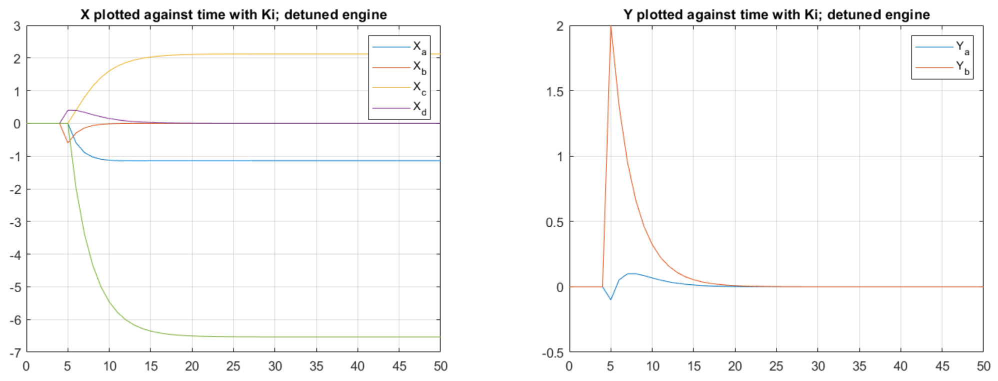


Figure 16: Detuned engine: X and Y plotted with K_i against time