AE353: Design Problem 04

Aaryaman Batra

November 9, 2020

1 Goal

The goal is to make the robot race along a randomly generated road at the fastest possible speed without crashing. The following requirements will make sure our goal is achieved

- The robot must be able to complete at least 95 out of 100 randomly generated roads.
- The robot must be able to complete any road in less than 50 seconds.

Verification ensures that the system meets the established requirements. Verification will be done through the following methods

- 100 random roads will be generated and the robot does not fail to complete any of them within 50 seconds
- A plot of 100 runs will be generated and at least 95 of the runs must be completed with the average time being under 50 seconds.

2 Model

The motion of the robot is governed by ordinary differential equations with the form

$$\begin{bmatrix} \ddot{\phi} \\ \dot{v} \\ \dot{w} \end{bmatrix} = f(\phi, \dot{\phi}, v, w, \tau_R, \tau_L) \tag{1}$$

where ϕ is the pitch angle of the chassis, $\dot{\phi}$ is the pitch angular velocity, v is the forward speed, w is the turning rate, and τ_R and τ_L are the torques applied by the chassis to the right and left wheel, respectively .

In the process of solving this problem, a state, input and output are defined. Shown in Equation (2), the system has six states, two inputs and four outputs.

$$\text{state} = \begin{bmatrix} \dot{\phi} \\ v \\ w \\ \phi \\ e_{\text{lateral}} \\ e_{\text{heading}} \end{bmatrix} \quad \text{input} = \begin{bmatrix} \tau_R \\ \tau_L \end{bmatrix} \quad \text{output} = \begin{bmatrix} v \\ w \\ e_{\text{lateral}} \\ e_{\text{heading}} \end{bmatrix} \tag{2}$$

To begin the linearization of the system, an equilibrium for the input and state must be determined. The function DP4fsolve in controller.m finds the equilibrium values for θ, ϕ, $\dot{x}$ and $\dot{z}$. The equilibrium state and equilibrium input for the system found are as follows

$$eq_{\text{state}} = \begin{bmatrix} \dot{\phi} \\ v \\ w \\ \phi \\ e_{\text{lateral}} \\ e_{\text{heading}} \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ 0.5 \\ 0 \\ 0 \\ 0 \end{bmatrix} \qquad eq_{\text{input}} = \begin{bmatrix} \tau_R \\ \tau_L \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \end{bmatrix} \tag{3}$$

Using matlabs symbolic tools and after defining a state and finding equilibrium values A,B,C and D can now be found. The following code was used to do so.

```
1
2 %Linearization
3 A = double(subs(jacobian(fsym,state),[state; input; vroad; wroad],[eqstate;eqinput;
      vroade;wroade]))
4 B = double(subs(jacobian(fsym,input),[state; input; vroad; wroad],[eqstate;eqinput;
      vroade;wroade]))
5 C = [0 0 (1/parameters.r) ((parameters.b)/(2*parameters.r)) 0 0;
6     0 0 (1/parameters.r) (-(parameters.b)/(2*parameters.r))  0 0;
7     0 0 0 0 1 0;
8     0 0 0 0 0 1;]
```

which contain the matrices A, B and C for the state-space model found are as follows. C is non-trivial in this case and was calculated as shown above.

$$A = \begin{bmatrix} 0 & 1 & 0 & 0 & 0 & 0 \\ 27.4427 & 0 & 0 & 0 & 0 & 0 \\ -3.4540 & 0 & 0 & 0 & 0 & \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & -0.5 \\ 0 & 0 & 0 & 1 & 0 & 0 \end{bmatrix}$$

$$B = \begin{bmatrix} & \\ 0 & 0 \\ -2.2255 & -2.2255 \\ 0.5874 & 0.5874 \\ -2.4814 & 2.4814 \\ 0 & 0 \\ 0 & 0 \end{bmatrix} \tag{4}$$

$$C = \begin{bmatrix} 0 & 0 & 5 & 1 & 0 & 0 \\ 0 & 0 & 5 & -1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix}$$

The D matrix is simply 0 since the output does not rely on the input of the system. The resulting state-space model is

$$\dot{x} = Ax + Bu$$
$$y = Cx$$

The D matrix is simply 0 since the output does not rely on the input of the system.

3 Controller and Observer

3.1 Controller

Verification for controllability and stability of the model is performed through the following code. Since all real eigen values are negative, the model is stable and controllable. A linear feedback control design Implementing LQR introduces a Q and R matrix. The selected Q and R matrices are represented as follows

$$Q_c = 1.5 * \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 30000 & 0 \\ 0 & 0 & 0 & 0 & 0 & 2750 \end{bmatrix} \qquad R_c = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \tag{5}$$

$$K = \begin{bmatrix} -13.7553 & -2.8711 & -0.8660 & -4.6842 & 150.0000 & -52.5845 \\ -13.7553 & -2.8711 & -0.8660 & 4.6842 & -150.0000 & 52.5845 \end{bmatrix} \tag{6}$$

```
%Verifies System is Controllable and observable
controllability = rank(ctrb(A,B))-length(A)
observability = rank(obsv(A,C)) -length(A)

%Define Gains Controller
Qc = 500*[1 0 0 0 0 0;0 15 0 0 0 0;0 0 1 0 0 0;0 0 0 1 0 0;0 0 0 0 10 0;0 0 0 0 0
    10];
Rc = [1 0;0 1];
K = lqr(A,B,Qc,Rc);

%Verifies if Controller is Asymptoticly Stable
eig(A-B*K)
```

3.2 Observer

As proved in the code in the previous section, the system is observable. Implementing the LQR introduces a Q and R matrix. As shown in the code in the previous subsection the L

matrix can be found and thereby allowing us to check the stability by making sure A-LC has all negative real parts. The Matrices are as follows

$$R_o = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix} \qquad Q_o = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{7}$$

$$L = \begin{bmatrix} -3.8360 & -2.8360 & 0 & 0 \\ -19.7637 & -19.7637 & 0 & 0 \\ 1.7748 & 1.7748 & 0 & 0 \\ 0.6809 & -0.6809 & -0.0260 & 0.2683 \\ -0.0260 & 0.0260 & 1.0918 & -0.2619 \\ 0.2683 & -0.2683 & -0.2619 & 1.1507 \end{bmatrix} \tag{8}$$

Since all eigen values have negative real parts the system is asymptotically stable.

4 Implementation

For the robot I tried implementing 3 controllers on the system and the following is a comparison on how they performed. Controller 2 and 3 can be found in Controller2.m

4.1 Controller 1

Controller 1 is the best implementation and is mentioned in section 3 of this report. The system performed well on 100 different roads with no failures and completed all runs in within 50 seconds. The reason behind this would be carefully tuned weights in the R and Q matrix. Results in Figure 1

4.2 Controller 2

Controller 2 has the R and Q matrix to be identity matrices. The controller failed to complete any road and usually crashed within the first 10 seconds since it could not perform hard turns and always went of the road at the first bend it came up on. This is because the gains on the controller are not tuned at all. The Gains section matrices remained the same i.e. the identity matrices mentioned in section 3.2. Results in Figure 2 and 3

4.3 Controller 3

For the third controller the gains are improved as compared to the identity matrix but are not as large as those in Controller 1. The Gains section matrices remained the same i.e. the identity matrices mentioned in section 3.2. Results in Figure 4 and 5. The matrices are in equation (9)

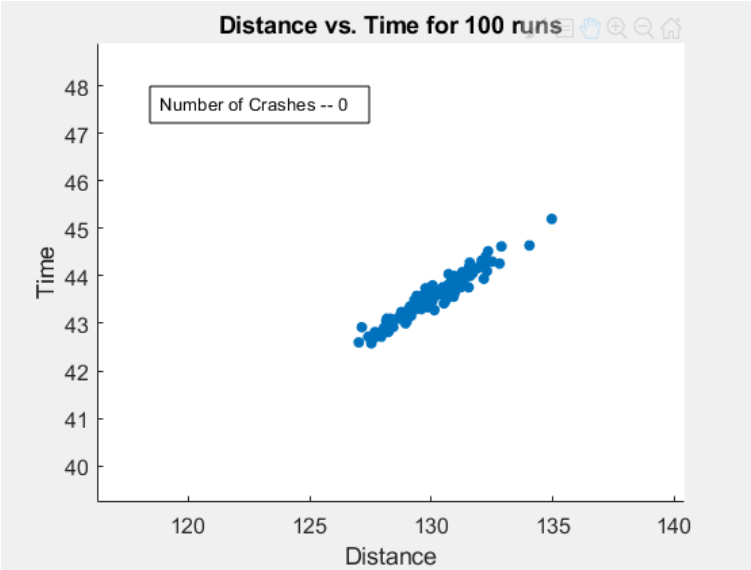

Figure 1: Results of 100 Roads with Controller 1

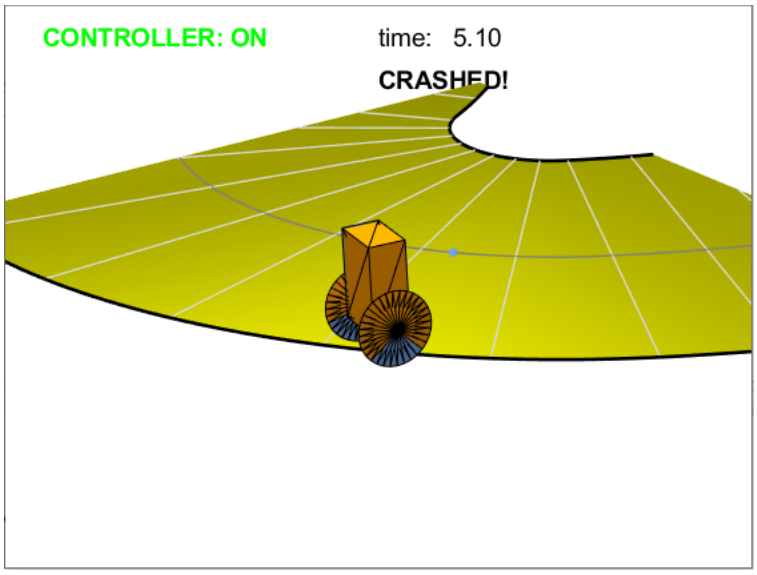

Figure 2: Controller 2 causing a crash within 10 seconds

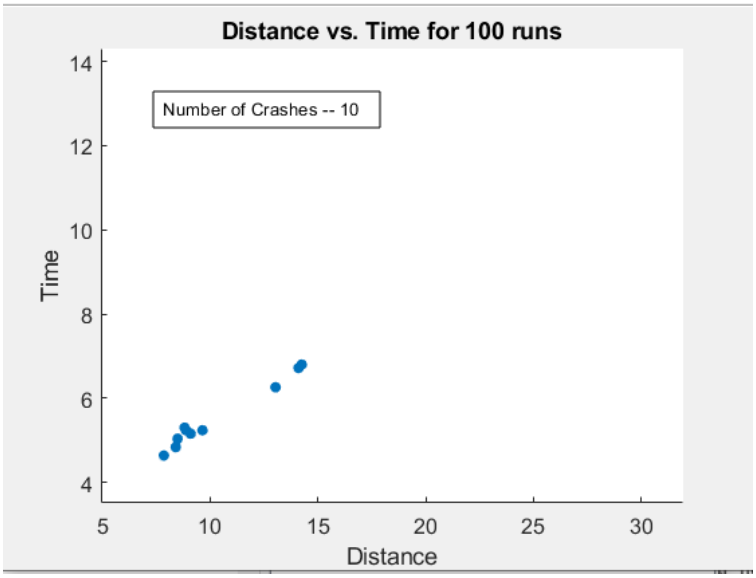

Figure 3: Results of 10 Roads with Controller 2

$$Q_c = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 300 & 0 \\ 0 & 0 & 0 & 0 & 0 & 100 \end{bmatrix} \qquad R_c = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \tag{9}$$

The results for controller 3 are as follows

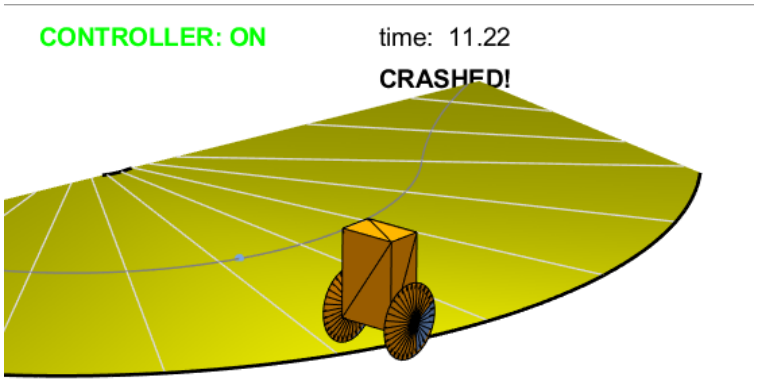

Figure 4: Controller 3 causing a crash within 15 seconds

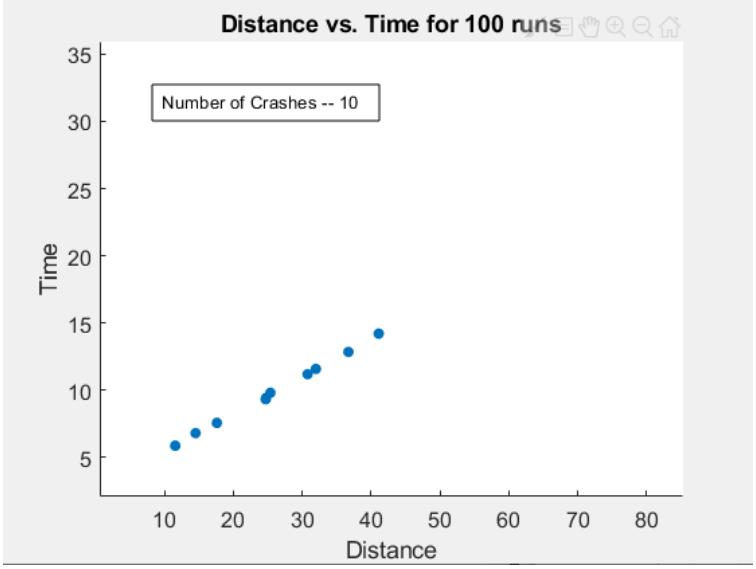

Figure 5: Results of 10 Roads with Controller 3

5 Analysis

The results of the Controllers as seen above can be explained by the tuning of the gain matrices. Controller 2 performed the worst and usually did not make it past the first bend. It failed all 10 times when run on 10 different roads. Controller 3 was next, even though it was slight improvement due to some tuning in the matrices and thus took turns yet the turns were too wide and the robot was slightly slower. Even though it travelled a greater distance than Controller 2 it still failed all 10 roads that it was tested on. The best controller was Controller 1 which is the one followed in the rest of the report. Even though it faced the same issues faced by Controller 3 The Gain matrices were tuned and increased to the point where it penalized variables a lot stronger than that done by Controller 3 and thus performed the way it did. Of the 100 trials it completed the road all 100 times and all tries were below 50 seconds.

6 Conclusion

As we saw the Controller with the properly tuned gains performed the best and thus was used to complete the requirements. As per the Verification the simulation ran on 100 different roads and completed all 100 roads. As seen in the graph the average time was far below 50 seconds since none of the roads took more than 46 seconds to complete thus the requirement is verified.

References

[1] K. J. Aström and R. M. Murray. *Feedback Systems: An Introduction for Scientists and Engineers* Princeton University Press, 2010

http://www.cds.caltech.edu/ murray/amwiki/index.php/First$_E$*dition*

[2] T. Bretl. *Design Problem 4 Description* UIUC, 2017
https://piazza-resources.s3.amazonaws.com/iy352lpu44t1zm/j1pjoc8j474q3/DesignProblem0

[3] Y. X. Mak *Realization of mini segway robot using NI MyRIO* University of Twente, 2015
http://purl.utwente.nl/essays/67004

7 Acknowledgements

The following of my colleagues helped me in the process of finding a good gain matrix to meet my requirements as well as in helping me plot the 100 different trials of the controllers.

- Ashvat Bansal
- Pratik Nistala