**ME227: Vehicle Dynamics and Control** **Spring 2021**

# Assignment 6: Linearization and Project Reflection

Due Thursday, May 27 at 5:00pm PST 

## Purpose

For this homework, we will study linearization, reflect on the project, and start thinking about racing.

## Instructions

This homework assignment will be submitted using Gradescope.

All written portions must be turned in through Gradescope. See the Piazza post on homework guidelines for more instructions on the different homework resources available to you. Whatever format you decide to use, please **BOX** all of your final answers.

# Problem 1 – Linearization

In the last homework, you used your intuition about vehicle dynamics to determine signs of feedback gains to stabilize around a drift equilibrium. This week, we are going to learn how we can compute a linearization and use linear control techniques to determine these gains, rather than by inspection.

For all the following problems, consider a drift equilibrium consisting of $r_{eq} = 0.7486$ rad/s, $U_{y,eq} = -8.661$ m/s, $U_{x,eq} = 10.3224$ m/s, $\delta_{eq} = -0.5132$ rad, and $F_{xr,eq} = 7317$ N.

(This is derived from drifting an 18 meter radius path ($\kappa = 1/18$) with constant sideslip of $-40°$)

## Question 1.A – Linearization of our Velocity Dynamics (MATLAB Grader)

Up until now, we have linearized our dynamics via linear *assumptions*, such as small angle approximations and linear tires. We will now directly linearize our dynamics (which allows analysis and control of cool vehicle behavior such as drifting).

Consider just our velocity states, so $x = [r, U_y]^T$. We can write our dynamics as being a nonlinear function of our state and steering input (forget $F_{xr}$ for now - assume we are using the same longitudinal controller from HW5):

$$\dot{x} = f(x, \delta)$$

We can linearize the dynamics and write our system as:

$$\dot{x} = f(x_{eq}) + A(x - x_{eq}) + B(\delta - \delta_{eq}) = A(x - x_{eq}) + B(\delta - \delta_{eq})$$

$$\dot{\bar{x}} = A\bar{x} + B\bar{\delta}$$

$$A = \begin{bmatrix} \frac{\partial \dot{r}}{\partial r} & \frac{\partial \dot{r}}{\partial U_y} \\ \frac{\partial \dot{U}_y}{\partial r} & \frac{\partial \dot{U}_y}{\partial U_y} \end{bmatrix}_{x=x_{eq}} \tag{6.1}$$

$$B = \begin{bmatrix} \frac{\partial \dot{r}}{\partial \delta} \\ \frac{\partial \dot{U}_y}{\partial \delta} \end{bmatrix}_{x=x_{eq}} \tag{6.2}$$

To calculate the partial derivatives, we need to carefully consider what each term in our dynamics is a function of. Based on the Fiala lateral tire model:

$$F_{yf} = f(\alpha_f) = f(U_y, r, \delta)$$

Generally while drifting, $\alpha_r > \alpha_{sl}$ so:

$$F_{yr} = f(F_{xr})$$

So to calculate the first entry in the A matrix:

$$\left.\frac{\partial \dot{r}}{\partial r}\right|_{x=x_{eq}} = \left.\frac{1}{I_z}(a\frac{\partial F_{yf}}{\partial r}cos(\delta) - b\frac{\partial F_{yr}}{\partial r})\right|_{x=x_{eq}} = \left.\frac{acos(\delta)}{I_z}\frac{\partial F_{yf}}{\partial r}\right|_{x=x_{eq}}$$

To calculate $\frac{\partial F_{yf}}{\partial r}$, we then need to take the derivative of each term in the Fiala model with respect to $r$. The rest of the derivatives can be computed similarly.

For this two state model we have computed the A & B matrices for you (so you won't need to take the derivative of the Fiala model). However, make sure you understand how one could compute each term in A & B, as you will be finding some of the derivatives in the next problem.

$$A = \begin{bmatrix} -1.1963 & -0.744 \\ -11.5025 & -0.7339 \end{bmatrix} \tag{6.3}$$

$$B = \begin{bmatrix} 13.1451 \\ 12.967 \end{bmatrix} \tag{6.4}$$

Finally, lets use the control law $\bar{\delta} = -K\bar{x}$ (full state feedback).

**Find the gain vector K that puts the poles of the closed loop system at -4 and -8. Compute this by hand and show your work. Submit your gain vector to MATLAB Grader.**

**Solution:**

The CL system is given by

$$\dot{x} = (A - BK)x$$

The poles are the eigenvalues of $(A - BK)$

$$|A - BK - sI| = 0$$

$$\left| \begin{bmatrix} -1.1963 & -0.744 \\ -11.5025 & -0.7339 \end{bmatrix} - \begin{bmatrix} 13.1451 \\ 12.967 \end{bmatrix} \begin{bmatrix} K_1 & K_2 \end{bmatrix} - \begin{bmatrix} s & 0 \\ 0 & s \end{bmatrix} \right| = 0$$

$$\left| \begin{bmatrix} -1.1963 & -0.744 \\ -11.5025 & -0.7339 \end{bmatrix} - \begin{bmatrix} 13.1451K_1 & 13.1451K_2 \\ 12.967K_1 & 12.967K_2 \end{bmatrix} - \begin{bmatrix} s & 0 \\ 0 & s \end{bmatrix} \right| = 0$$

$$\left| \begin{bmatrix} -1.1963 - 13.1451K_1 - s & -0.744 - 13.1451K_2 \\ -11.5025 - 12.967K_1 & -0.7339 - 12.967K_2 - s \end{bmatrix} \right| = 0$$

$$0.878 + 15.5124K_2 + 1.1963s + 9.6475K_1 + 170.455K_1K_2 + 13.1451K_1s + 0.7339s + 12.967K_2s + s^2 - \left[ 8.5579 + 9.6475K_1 + 151.202K_2 + 170.455K_1K_2 \right] = 0$$

$$-7.68 - 135.69K_2 + (1.9302 + 13.1451K_1 + 12.967K_2)s + s^2 = 0$$

For poles at $s = \{-4, -8\}$, we have

$$(s+4)(s+8) = 0 \implies s^2 + 12s + 32$$

Therefore,

$$-7.68 - 135.69K_2 = 32 \implies K_2 = -0.292$$

$$1.9302 + 13.1451K_1 + 12.967K_2 = 12 \implies K_1 = 1.0545$$

$$\boxed{K = \begin{bmatrix} 1.0545 & -0.292 \end{bmatrix}}$$

## Question 1.B – Incorporating Path Tracking (MATLAB Grader)

Drifting by itself is pretty cool, but what if we wanted to incorporate a reference path? Let's define a new state $x = [r, U_y, e, \Delta\psi]^T$.

**Find a value for $\Delta\psi_{eq}$ given the drift equilibrium we are using in this problem. Do not make any small angle assumptions.**

Let's linearize these dynamics as before and then try to find gains that stabilize the system. This time, our linearized matrices are:

$$A = \begin{bmatrix} \frac{\partial \dot{r}}{\partial r} & \frac{\partial \dot{r}}{\partial U_y} & \frac{\partial \dot{r}}{\partial e} & \frac{\partial \dot{r}}{\partial \Delta\psi} \\ \frac{\partial \dot{U}_y}{\partial r} & \frac{\partial \dot{U}_y}{\partial U_y} & \frac{\partial \dot{U}_y}{\partial e} & \frac{\partial \dot{U}_y}{\partial \Delta\psi} \\ \frac{\partial \dot{e}}{\partial r} & \frac{\partial \dot{e}}{\partial U_y} & \frac{\partial \dot{e}}{\partial e} & \frac{\partial \dot{e}}{\partial \Delta\psi} \\ \frac{\partial \dot{\Delta\psi}}{\partial r} & \frac{\partial \dot{\Delta\psi}}{\partial U_y} & \frac{\partial \dot{\Delta\psi}}{\partial e} & \frac{\partial \dot{\Delta\psi}}{\partial \Delta\psi} \end{bmatrix}_{x=x_{eq}} \tag{6.5}$$

$$B = \begin{bmatrix} \frac{\partial \dot{r}}{\partial \delta} \\ \frac{\partial \dot{U}_y}{\partial \delta} \\ \frac{\partial \dot{e}}{\partial \delta} \\ \frac{\partial \dot{\Delta\psi}}{\partial \delta} \end{bmatrix}_{x=x_{eq}} \tag{6.6}$$

**Find $\Delta\psi_{ss}$ and the new linearized dynamics matrices. Using full state feedback as before, place the poles at [-4 +/- 4.5j, -0.15 +/- 0.75j]. You are encouraged to use the 'place' function in MATLAB. Report your gain vector and $\Delta\psi_{ss}$ on MATLAB Grader**

**Solution:**

Our dynamics equations are:

$$\dot{r} = \frac{1}{I_z}\left(aF_{yf}\cos\delta - bF_{yr}\right)$$

$$\dot{U}_y = \frac{1}{m}\left(F_{yr} + F_{yf}\cos\delta\right) - rU_x$$

$$\dot{e} = U_y\cos\Delta\Psi + U_x\sin\Delta\Psi$$

$$\Delta\dot{\Psi} = r - K\dot{s} = r - K\left(\frac{1}{1-eK}\right)\left(U_x\cos\Delta\Psi - U_y\sin\Delta\Psi\right)$$

Solving for $\Delta\Psi_{eq}$

$$\dot{e}_{eq} = 0 = U_{y_{eq}}\cos\Delta\Psi_{eq} + U_{x_{eq}}\sin\Delta\Psi_{eq}$$

$$\tan\Delta\Psi_{eq} = -\frac{U_{y_{eq}}}{U_{x_{eq}}}$$

$$\Delta\Psi_{eq} = \operatorname{atan}\left(-\frac{U_{y_{eq}}}{U_{x_{eq}}}\right)$$

$$\boxed{\Delta\Psi_{eq} = 39.9983^\circ}$$ (or $219.9983^\circ$, but we take the first solution)

Our Jacobians are:

$$A = \begin{bmatrix} \frac{a\cos\delta}{I_z}\cdot\frac{\partial f}{\partial r} & \frac{a\cos\delta}{I_z}\cdot\frac{\partial f}{\partial U_y} & 0 & 0 \\ \frac{\cos\delta}{m}\cdot\frac{\partial f}{\partial r} - U_x & \frac{\cos\delta}{m}\cdot\frac{\partial f}{\partial U_y} & 0 & 0 \\ 0 & \cos\Delta\Psi & 0 & -U_y\sin\Delta\Psi + U_x\cos\Delta\Psi \\ 1 & \frac{K}{1-eK}\sin\Delta\Psi & \frac{-K^2}{(1-eK)^2}\left(U_x\cos\Delta\Psi - U_y\sin\Delta\Psi\right) & \frac{K}{1+eK}\left(U_x\sin\Delta\Psi + U_y\cos\Delta\Psi\right) \end{bmatrix}$$ same as prev. A

$$B = \begin{bmatrix} \frac{a}{I_z}\left(F_{yf}(-\sin\delta) + \frac{\partial f}{\partial\delta}\cos\delta\right) \\ \frac{1}{m}\left(F_{yf}(-\sin\delta) + \frac{\partial f}{\partial\delta}\cos\delta\right) \\ 0 \\ 0 \end{bmatrix}$$ same as prev. B

Plugging in the equilibrium values, we get:

$$A = \begin{bmatrix} -1.1963 & -0.744 & 0 & 0 \\ -11.5025 & -0.7339 & 0 & 0 \\ 0 & 0.76606 & 0 & 13.4746 \\ 1 & 0.035709 & -0.041588 & 0 \end{bmatrix}$$

$$B = \begin{bmatrix} 13.1451 \\ 12.967 \\ 0 \\ 0 \end{bmatrix}$$

Using `place`, we get

$$\boxed{K = \begin{bmatrix} 0.9318 & -0.4534 & 0.0606 & -1.1586 \end{bmatrix}}$$

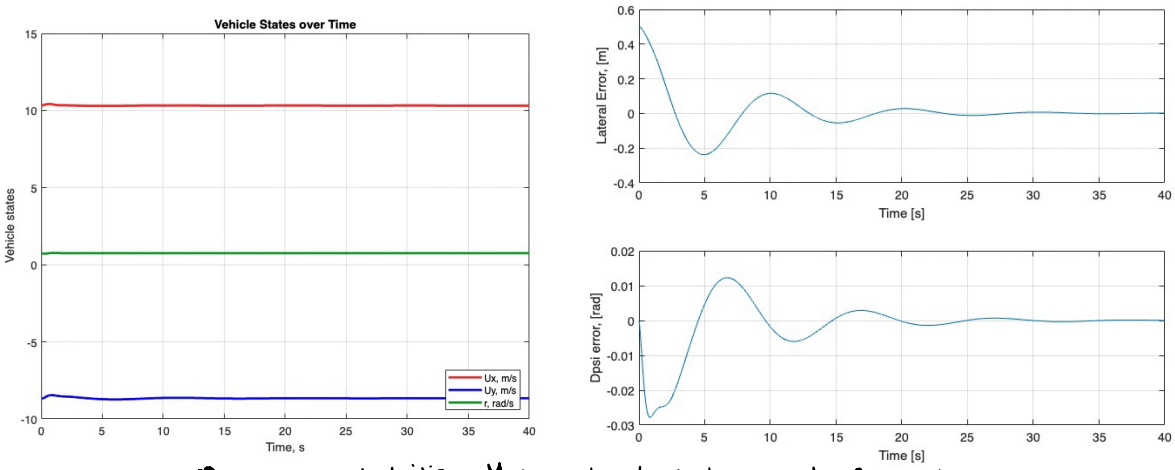
## Question 1.C – Simulation (MATLAB Online / Gradescope)

Let's see if the gains for combined drifting and path tracking work. Implement our new controller using the provided simulator framework.

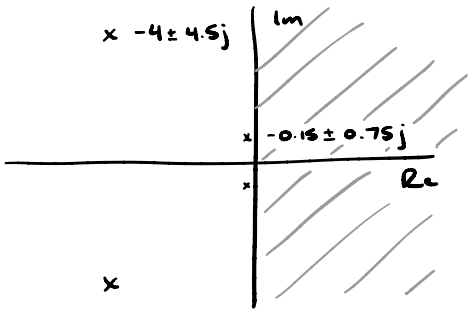
We have provided the 18m radius reference path, as well as the lower level functions that you have implemented several times by now. You only need to edit the file called ***mainproblem_1c***.

***Provide two plots, one of the velocity states ($U_x$, $U_y$, $r$) and one of our path tracking errors (Fig's 2 & 4 in provided plotting script). Do the results match what you expected, based on our closed loop pole locations? Why or why not.***

**Solution:**




We very effectively stabilize the vehicle into a drift state.
Yes, the error dynamics follow the pole locations: stable but oscillatory.



## Problem 2 – Reflection

You've completed your project. You've watched your lookahead and group controller executed on a real vehicle. You're officially a vehicle dynamicist!

In the project, you developed feedforward and feedback controllers for steering Niki along a defined path. Having had a chance to try those controllers on the actual car, you now have additional information about the appropriateness of your design choices beyond what you could gain from simulation. We want you to reflect on your project experience in light of this new information. This should be an individual response, not a team response. If you do find something wrong in your code when answering the first question, however, you can (and should) point this out to your teammates.

With the data and video from your vehicle testing, let's take a closer look at how your simulated performance compares to the actual performance on the vehicle.

## Question 2.A – Reflection of Lookahead Controller (Gradescope)

We'll start with a reflection on your lookahead controller. Let's take a look at your code and make sure that everything was actually implemented correctly. A couple of teams had some implementation issues so, if your results seemed strange, check again that your code was written correctly.

Next, consider whether you think the amount of feedback and feedforward was appropriate. Feedforward enables you to lower feedback gains by adjusting the steering to the value you expect you need for a turn. It can, however, lead to errors if the feedforward steering is not exact (which is very possible if there is even a slight misalignment of the steering). Feedback – particularly proportional feedback – reduces the steady-state error but also makes the system response faster, potentially amplifying noise or the effects of delay.

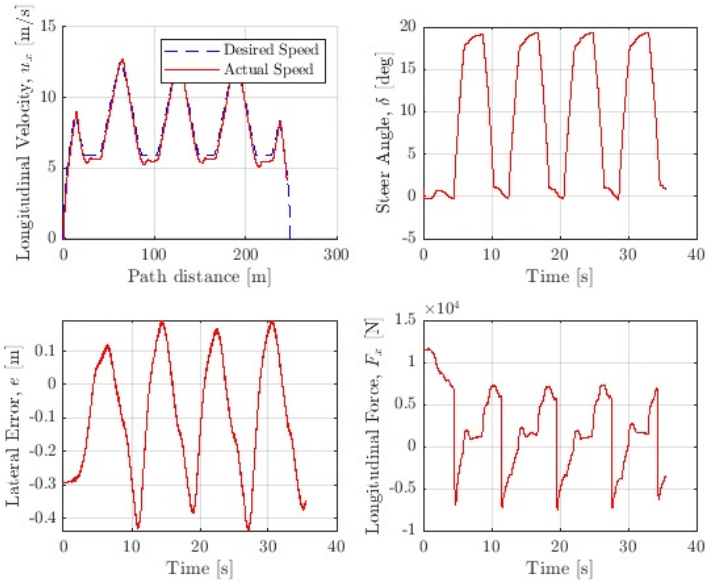
**(1) Was everything implemented correctly? If not, what implementation challenges did your team run into?**

**(2) Did each of your controllers effectively use feedback and feedforward? Support your conclusion with plots of your performance.**

**Solution:**

Everything was implemented correctly for our lookahead controller.

Yes, our controllers effectively used feedback and feedforward. While there was lateral error of 40 cm on one side, our controller managed to stabilize the car and track the path.



## Question 2.B – Reflection of Group Controller (Gradescope)

Now let's move on to your group's controller. Let's take a look at your code once more to make sure that everything was actually implemented correctly. Let's again consider whether you think the amount of feedback and feedforward was appropriate in this case.

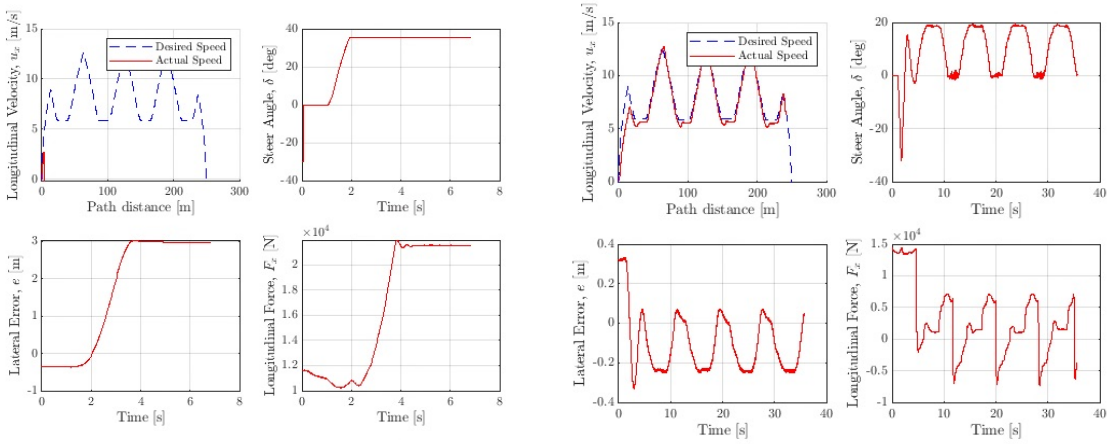
**(1) Was everything implemented correctly? If not, what implementation challenges did your team run into?**

**(2) Did each of your controllers effectively use feedback and feedforward? Support your conclusion with plots of your performance.**

**Solution:**

Our PID controller did not work as expected the first time. There was a bug in our error modeling that led to one-sided integrator windup. While our simulations looked fine for the initial conditions, the different initial conditions in reality exposed the bug, which led to the poor performance in the left set of plots. Once this bug was fixed, we were able to successfully run our controller on the car.

Our updated/fixed PID controller did use feedback and feedforward well. There were some significant oscillations in the steering angle on the straightaways due to the increasing speed and sensor noise. If we were to redo this, we would lower these gains to decrease the sensitivity to noise and better track the path. Overall though, we track well, getting close to 10 cm lateral error on the left and close to 20 cm lateral error on the right side.



## Question 2.C – Reflection of Project (Gradescope)

Now that we've reflected on both the lookahead controller and your group's controller, let's reflect on what we've learned from the design, analysis, and testing of these controllers.

**(1) What did you take away from the project?**

**(2) What, if anything, is still unclear to you about the controllers you used or linear vehicle dynamics after this project?**

**Solution:**

It was really nice to have an opportunity to test our controller in a high-fidelity simulator and in real-life. I learned a lot about the real-world challenges in control systems (noise, error, saturation, etc.) and how to better consider these effects during the design process. This might have been the first time I went all the way from basic physical modeling to dynamics to control design to real-world testing - thanks for such a great experience!

I have a great understanding of vehicle dynamics after the project!