Final Project*

Ding Zhao
24-677 Special Topics: Linear Control Systems

1 Introduction

This project is an individual project in which you are asked to design a controller for your Buggy team(go to https://www.springcarnival.org/buggy.shtml to find out more about Buggy). The goal is to complete a pair of tasks:

1. Linearize the Buggy model
2. Design a controller for a Buggy to finish the track
3. Design a controller(which may or may not be the same controller in the first task) to help your team defeat the baseline controller
4. Race with other Buggy teams in the class

The deadlines of the tasks will be :

- Model linearization is due at 11:00 PM on $Nov.28^{th}$
- Task 2 is due at 11:00 PM on $Dec.7^{rd}$
- The final version of your controller is due at 11:00 PM on $Dec.14^{th}$

2 Model

Here you will use the bicycle model for the vehicle, which is a popular model in the study of vehicle dynamics. Shown in Figure 1, the car is modeled as a two-wheel vehicle in two degree of freedom, described in longitudinal and lateral dynamics separately. The model parameters are defined in Table 1.

*Modified on Nov.22

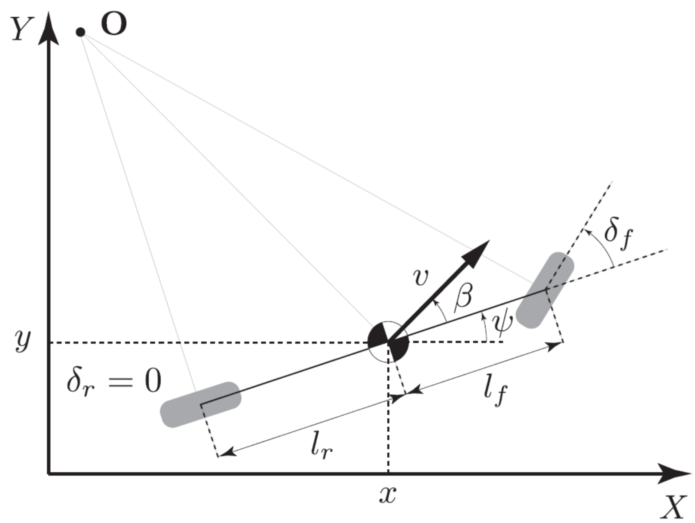

Figure 1: Bicycle model[2]

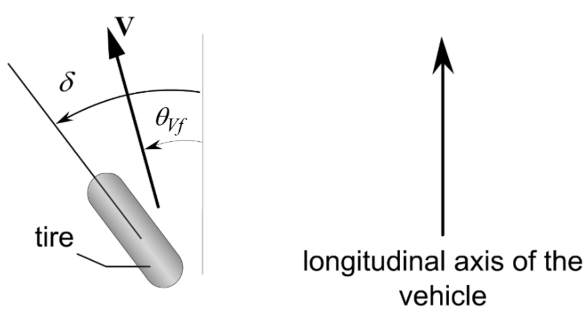

Figure 2: Tire slip-angle[2]

2.1 Lateral dynamics

Ignoring road bank angle and applying Newton's second law for motion along the y axis

$$ma_y = F_{yf} \cos \delta_f + F_{yr}$$

where $a_y = \left(\frac{d^2y}{dt^2}\right)_{inertial}$ is the inertial acceleration of the vehicle at the center of geometry in the direction of the y axis, F_{yf} and F_{yr} are the lateral tire forces of the front and rear wheels respectively and δ_f is the front wheel angle whichi will be denoted as δ later. Two terms contribute to a_y: the acceleration $\ddot{y}$ which is due to motion along the y axis and the centripetal acceleration . Hence

$$a_y = \ddot{y} + \dot{\psi}\dot{x}$$

Combining the two equations, the equation for the lateral transnational motion of the vehicle is obtained as

$$\ddot{y} = -\dot{\psi}\dot{x} + \frac{1}{m}(F_{yf} \cos \delta + F_{yr})$$

Moment balance about the axis yields the equation for the yaw dynamics as

$$\ddot{\psi} I_z = l_f F_{yf} - l_r F_{yr}$$

The next step is to model the lateral tire forces F_{yf} and F_{yr}. Experimental results show that the lateral tire force of a tire is proportional to the "slip-angle" for small slip-angles **when vehicle's speed is large enough**, let's say when $\dot{x} \geq 0.5$ m/s. The slip angle of a tire is defined as the angle between the orientation of the tire and the orientation of the velocity vector of the wheel. the slip angle of the front and rear wheel is

$$\alpha_f = \delta - \theta_{Vf}$$
$$\alpha_r = -\theta_{Vr}$$

where θ_{Vp} is the angle that the velocity vector makes with the longitudinal axis of the vehicle for$p \in \{f, r\}$. A linear approximation of the tire forces are given by

$$F_{yf} = 2C_\alpha \left(\delta - \frac{\dot{y} + l_f \dot{\psi}}{\dot{x}} \right)$$
$$F_{yr} = 2C_\alpha \left(-\frac{\dot{y} - l_r \dot{\psi}}{\dot{x}} \right)$$

where C_α is called the cornering stiffness of tires. If $\dot{x} < 0.5$ m/s, we just set F_{yf} and F_{yr} both to zeros.

2.2 Longitudinal dynamics

Similarly, a force balance along the vehicle longitudinal axis yields

$$\ddot{x} = \dot{\psi}\dot{y} + a_x$$
$$m a_x = F - sign(\dot{x}) F_f$$
$$F_f = f m g$$

where F is the total tire force along x axis, F_f is the force due to rolling resistance at the tires, and f is the friction coefficient. *sign* function returns +1 when $\dot{x} \geq 1$ otherwise -1.

2.3 Global coordinates

In the global frame we have

$$\dot{X} = \dot{x} \cos\psi - \dot{y} \sin\psi$$
$$\dot{Y} = \dot{x} \sin\psi + \dot{y} \cos\psi$$

2.4 System equation

Gathering all the equations, if $\dot{x} \geq 0.5$ m/s we have:

$$\begin{aligned}
\ddot{y} &= -\dot{\psi}\dot{x} + \frac{2C_\alpha}{m}(\cos\delta\left(\delta - \frac{\dot{y}+l_f\dot{\psi}}{\dot{x}}\right) - \frac{\dot{y}-l_r\dot{\psi}}{\dot{x}}) \\
\ddot{x} &= \dot{\psi}\dot{y} + \frac{1}{m}(F - fmg) \\
\ddot{\psi} &= \frac{2l_fC_\alpha}{I_z}\left(\delta - \frac{\dot{y}+l_f\dot{\psi}}{\dot{x}}\right) - \frac{2l_rC_\alpha}{I_z}\left(-\frac{\dot{y}-l_r\dot{\psi}}{\dot{x}}\right) \\
\dot{X} &= \dot{x}\cos\psi - \dot{y}\sin\psi \\
\dot{Y} &= \dot{x}\sin\psi + \dot{y}\cos\psi
\end{aligned}$$

otherwise since the lateral tire forces are zeros we only consider the longitudinal model.

2.5 Observation

The observable states are with some Guassian noise $\epsilon = N(0,\sigma)$, where

$$y = \begin{bmatrix} \dot{x} \\ \dot{y} \\ \dot{\psi} \\ \delta \\ X \\ Y \\ \psi \end{bmatrix} + \epsilon,\ \sigma = \begin{bmatrix} 0.5 & & & \cdots & & & 0 \\ & 0.5 & & & & & \\ & & 0.05 & & & & \\ \vdots & & & 0.05 & & & \\ & & & & 1 & & \\ & & & & & 1 & \\ 0 & & & & & & 0.5 \end{bmatrix}$$

.

2.6 Physical constrain

The system satisfies the constraints that:

$$\begin{aligned}
&|\delta| \leqslant \tfrac{\pi}{6}\ rad/s \\
&|\dot{\delta}| \leqslant \tfrac{\pi}{6}\ rad/s \\
&|F| \leqslant 10000\ N \\
&0 \text{ m/s} \leqslant \dot{x} \leqslant 100 \text{ m/s} \\
&|\dot{y}| \leqslant 10 m/s
\end{aligned}$$

3 Resources

3.1 Buggy Simulator

The Buggy simulator is provided in the file `BuggySimulator.py`. The simulator takes the command, time step, and current buggy state, then output the buggy state after the given time step. Here you are required to use a fixed time step $\Delta t = 0.05s$. Please design your controller in main.py.

Table 1: Model parameters.

Name	Description	Unit	Value
$(\dot{x}, \dot{y})$	Vehicle's velocity along the driection of vehicle frame	m/s	State
(X, Y)	Vehicle's coordinates in the world frame	m	State
ψ, $\dot{\psi}$	Body yaw angle, angular speed	rad	State
δ or δ_f	Front wheel angle	rad	State
$\dot{\delta}$	Front wheel angular speed	rad	Input
F	Total input force	N	Input
m	Vehicle mass	kg	1000
l_r	Length from front tire to the center of mass	m	1.7
l_f	Length from front tire to the center of mass	m	1.1
C_α	Cornering stiffness of each tire	N	15000
I_z	Yaw intertia	kg m^2	3344
F_{pq}	Tire force, $p \in \{x, y\}$,$q \in \{f, r\}$	N	Depend on input force
m	vehicle mass	Kg	2000
f	Friction coefficient	1	0.01

3.2 Trajectory Data

The trajectory is given in `buggyTrace.csv`. It contains the coordinates of the trajectory: (x, y). The way points are given roughly every 0.1 m. The satellite map of the track is shown in Figure 3.

3.3 Supplementary Files

Utility files are written in util.py and the buggy is initiated in initial.py. You should design the controller in controller.py.

4 Tasks and Evaluation

4.1 Model Linearization (20 points)

As we mentioned in the class, model linearization is always the first step for non-linear control. During this assignment, you will figure out how to transfer the given model into a linear model.
Since the longitudinal term $\dot{x}$ is non-linear in the lateral dynamics, we can simplify the controller by controlling the lateral and longitudinal states separately. You are required to write the system dynamics in linear forms as $\dot{s_1} = A_1 s_1 + B_1 u$ and $\dot{s_2} = A_2 s_2 + B_2 u$ in terms of the following given input and states:

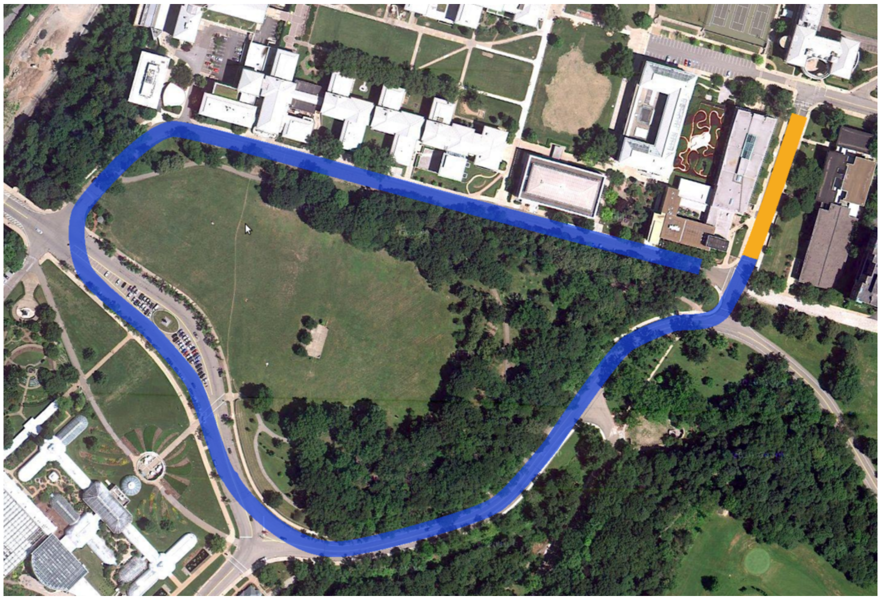

Figure 3: Buggy track[3]

$$u = \begin{bmatrix} \dot{\delta} \\ F \end{bmatrix}, s_1 = \begin{bmatrix} y \\ \dot{y} \\ \psi \\ \dot{\psi} \end{bmatrix}, s_2 = \begin{bmatrix} x \\ \dot{x} \end{bmatrix}$$

This task is due on $Nov.28^{th}$. The answer will be released right after it.

4.2 Finish the Loop(60 points)

The task is to design and implement a controller to manipulate the steering wheel and the traction, such that the Buggy can track a predefined trajectory and finish the loop. Make sure your Buggy's trajectory will pass the evaluation mentioned in 4.5.3. Task 2 will be due on $Dec.7^{rd}$. **You don't need to use the previous result to design your controller.**

4.3 Beat the Baseline(10 points)

This task is to control your Buggy to finish the track faster than the baseline controller. You can use the controller you submitted for previous tasks or develop a new controller. The finishing time of baseline controller will be released right after the deadline of Task 2.

4.4 Competition(10 points)

This task is to compete with other Buggy teams in the class. The score of each team will be decided by its rank among the teams. Task 3 and 4 will be due on $Dec.14^{th}$.

4.5 Evaluation

1. Your project will be graded based on both your code and the result after we run the .py file in your submission. **Changing the main.py, simulator file and any other files other than controller.py will result in unexpected points deduction**, since we will be using the original simulator file during grading. Please read section 6 carefully or your project might be graded as no submission if you fail to follow the submission rules.

2. Your answer to 4.1 should be submitted in the same way as homeworks. You need to submit both a copy to Gradescope and a hard copy in class.

3. The distance between a position p on your trajectory and the track $P_0 = \{p_i = (x_i, y_i) | i = 1, 2, \ldots, n\}$ will be calculated using the following equation:

$$d(p, P_0) = \min_{p_0 \in P_0} |p - p_0|_2$$

the maximum distance between your vehicle's trajectory P and P_0 is defined as

$$D(P, P_0) = \max_{p \in P} d(p, P_0)$$

the average distance between your vehicle's trajectory P and P_0 is defined as

$$d(P, P_0) = \frac{1}{n} \sum_{p \in P} d(p, P_0)$$

To get the full credits for 4.2, your Buggy needs to fulfill the following rules. Also, the metrics of grading are given:

- Your Buggy should complete the whole loop. (20 points)
  The grading metric is:
  $$G_{com} = \frac{20 n_g}{n_{track}}$$
  where $n_g = ||\{p \in P | d(p, P_0) <= d_{threshold}\}||$ is the number of the positions on the provided track for which we can find a position on your trajectory between which the distance is smaller than the maximum distance $C_{max} = 6$; the n_{track} is the number of positions given in the track.

- The average distance C_{avg} between your Buggy's positions and the race track should be less than or equal to 3m. (20 points)
  The grading metric is:
  $$G_{avg} = \begin{cases} 20, & d(P, P_0) \leq 3 \\ -\frac{20}{3} d(P, P_0) + 40 & 3 < d(P, P_0) \leq 6 \\ 0, & 6 < d(P, P_0) \end{cases}$$
  where G_{avg} is the credits you will get if the average distance is larger than C_{avg}. Otherwise, you will get full credits.

- The maximum distance between your Buggy's positions and the track should be less than or equal to 6m. (20 points)
  The grading metric is:
$$G_{max} = \frac{20n_g}{n_{traj}}$$
  where G_{max} is the credits you will get, n_g is the number of the positions on your trajectory that excess the maximum distance constrain $C_{max} = 6$; the n_{traj} is the total number of positions on your trajectory.

To conclude, the credits you will get for task 2 is the sum of your credits separately under the evaluation of each requirement. **The real world test time of your controllers of task 2,3,4 should not be longer than 2 minutes.**

4. In 4.3, at first your controller should complete the track,which means $\frac{n_g}{n_{track}} = 1$, $D(P, P_0) < 6$, otherwise you will not get points for this task.
   Then, your Buggy is to finish the race faster than our baseline Buggy. The controller you submitted for this task will be first graded in the same way as the previous task. Then, the grading metric is:
$$G_{baseline} = \begin{cases} 10 & T_p \le T_{baseline} \\ -\frac{10T_p}{T_{baseline}} + 20 & T_{baseline} < T_p \le 2T_{baseline} \\ 0, & 2T_{baseline} < T_p \end{cases}$$
   where $G_{baseline}$ is the credits you will get if your performance is worse than baseline controller; G_r is the credits you will get under the evaluation of Task 2; $T_{baseline}$ is the baseline time; T_p is your performance time. Otherwise, you will get full credits.

5. The final task is to compete with other teams in the class. The best player will get full credits of this part. Others will be graded proportionally according to their positions on the ranking.

5 Submission

No late submission will be allowed for tasks 2,3,4 of this project. The controller will be written using python in the python file called controller.py. You will submit your controller as a **.zip file** on Gradescope. The name of your .zip file should be:

`24-677_Project_<TaskNumber>_<TeamName>_<Name>.zip`

in which <TaskNumber> should be just one number.
Besides the files originally given in the project foler, the .zip file will contain other two files: a .py file, a .npy file and a .txt file which are named as:

`24-677_Project_< TaskNumber>_BuggyStates_< TeamName>.npy`
`24-677_Project_TeamName.txt`

Where you should replace <TeamName> with your team name. Do not include the brackets in your file name. For example, if you are Team Andrew, your files for task 2 will be named:

`24-677_Project_2_BuggyStates_Andrew.npy`
`24-677_Project_TeamName.txt`

The BuggyController file should be the controller based on the file main.py. The BuggyStates file should be a $n \times 7$ numpy array which contains all the states of your Buggy during the race where $n = T/dt$; T is the time your Buggy takes to finish the track . The TeamName file is a .txt file which only contains your team name, your andrew ID and your name in the following format:

$$< TeamName > \_ < AndrewID > \_ < Name >.$$

For example, if you are Team Andrew, your AndrewID is Tartan, your Name is Carnegie Mellon, the file should only contain this line:

$$Andrew\_Tartan\_Carnegie\ Mellon$$

Please feel free to post questions on Piazza or come to office hours. Good luck!

6 Reference

1. Rajamani Rajesh. Vehicle dynamics and control. Springer Science & Business Media, 2011.

2. Kong Jason, et al. "Kinematic and dynamic vehicle models for autonomous driving control design." Intelligent Vehicles Symposium, 2015.

3. cmubuggy.org, `https://cmubuggy.org/reference/File:Course_hill1.png`