ENGR 16100
Design Challenge: Autonomous Collision Avoidance

Objective:
This design challenge requires identifying issues and opportunities for controlling the dynamic behavior of a system.

Total Time-on-Task: 75 minutes

Background:

Autonomous control of cars has long been a vision of engineers interested in advancing transportation opportunities in cities and highways. At the basic level we have cruise control allowing us to relax our feet while the car maintains control of our desired speed. Still the car needs the driver to monitor the road conditions and take responsibility for maintaining a safe distance from other vehicles and perform emergency braking or lane changes when necessary. New cars are being equipped with proximity sensors that provide drivers with information about objects in their blind spots. These systems can give warnings to drivers or actually take control of the brakes and gas to safely decelerate the car and keep the driver and passengers from mayhem. The future of transportation could involve completely autonomous vehicles that will improve and transform mass transit in urban areas, which is one of the Engineering Grand Challenges. For example, Google developed a self-driving car which lead to a prototype that allowed a legally blind person to have a personal driving experience [3]. Auto manufacturers are developing technologies to make these possibilities available, leveraging our current transportation infrastructure (e.g. [1][2][4]).

Specification:
Complex technologies are required to produce a safe and reliable system. In this challenge teams will explore the design of a cruise control collision avoidance system. The objective is to create a car that can maintain a minimum cruise distance from any other cars or obstacles in front of it. Additionally, the car should have a crash avoidance buffer distance, where the car takes action to avoid an imminent collision if crossed. The crash avoidance distance should be lower than the minimum cruise distance. Both the cruise distance and crash avoidance distance are user defined (chosen by the designing team); however, the cruise distance cannot exceed 35 cm, and the collision avoidance distance should not exceed 20 cm.

To start, the car will be placed on a highway and should accelerate to a predetermined cruising speed of 5 cm/s. While on the highway, the car should be capable of detecting

any obstacles in front of it. If an obstacle is detected within the cruise distance, the car should slow down to a minimum of 1 cm/s until the obstacle distance increases above the minimum cruise distance again. If the obstacle distance increases above the minimum cruise distance, the car should accelerate and resume traveling at the cruising speed. If the obstacle distance decreases below the crash avoidance distance, the car should turn right, drive off the highway, and stop. Under no circumstances should the car come to a complete stop on the highway.

Teams should take into account factors that influence passenger comfort and system reliability. Teams are responsible for generating their own specifications for cruising distance, collision avoidance distance, and minimum speed; however, they should be prepared to explain their choices and provide supporting evidence. Success of this design challenge requires identifying the relevant variables to control, deciding on which sensors to use and devising a control system capable of adapting to the changes on the track.

Optional - Proportional Control:
The above method for maintaining the cruise distance (lowering the speed) is effective, but inefficient and impractical for an actual highway. For a more efficient system, finer control of the car's velocity, based on the distance to the obstacle in front, is necessary.

An alternative to simply reducing the speed to a fixed lower speed when needed would be to develop a control algorithm that continually adjusts the speed in order to maintain a desired distance. This control method is called proportional control. A simple control system that tries to keep the car at a desired distance (variable `distDesired)` will measure the actual distance (variable `distMeasured)` and will calculate a motor power change (variable `powerReduction)` according to:

$$powerReduction = K\ (distDesired - distMeasured)\ ,$$

where `K` is a constant called the control gain. The calculated `powerReduction` value will then be subtracted from the initial motor power. Changing the value (and the sign) of `K` will impact the performance of your control system. The higher the gain, the more sensitive your system is to the error between desired distance and measured distance. However, too high of a gain might cause the vehicle to be unstable.

Equipment:
1. Project Kits
2. Python Code Template
    - Available in Brightspace module *Design Challenges > Design Challenge 2*
3. Whiteboards (simulating an obstacle)
4. Classroom tools (whiteboards, markers, post-its, rulers, scissors, tape)
    - Note: Rulers and tape cannot be used as physical parts of the car.

Procedures:

Task 1 – Development and Testing (70 minutes)

1. Develop a rolling vehicle and control system to meet the specified performance. Validate the performance and reliability of your system through quantitative analysis and report the data in your design notebook.

2. To document the conceptual design process and the construction of your prototype, each team will generate an electronic design notebook (`DC2_DesignNotebook_TeamXX.pdf`). All aspects of the prototype design process should be documented in this design notebook, including, but not limited to: mechanisms used to understand the problem, listings of technical requirements, assumptions made, evidence of meeting specifications, project management decisions, idea generation, decision making strategies, comparing alternatives, control algorithms, etc.

3. As a part of your design process, you should create a flow chart of the algorithm you will use to perform the control and document it in your design notebook. Once you have created your algorithm, you should create a Python script (`DC2_Code_Team`*`XX`*`.py`) that will implement the control algorithm, using the supplied template. Once completed, the code for the control algorithm should be appended to the end of the design notebook. **Note: you should retain this code for future use.**

4. Conduct application testing prior to presenting. For simulating a highway lane, the design team may use any stretch of flat ground around 1 m in length to demonstrate the car capabilities. The car should run independently of human control (either physically or through laptop). The team should collect evidence that the vehicle meets the specified design constraints and criteria. Testing data should be reported truthfully – falsifying results is considered academic dishonesty.

5. If time permits, take your car to the highway track for demonstration.

Task 2 – Cleanup (5 minutes)

1. Power off the Pi and disassemble your robot.
2. Return any material that your team borrowed to a TA.

Deliverables: Submit the following to Gradescope (see site for deadline information):

1. Notebook (with code appended): `DC2_DesignNotebook_TeamXX.pdf`. This is a team submission.

References:

[1] Mercedes Autonomous car - https://www.youtube.com/watch?v=4jW0fJ80VG8

[2] Volvo Autonomous Emergency Braking (accessed September, 2016) https://www.youtube.com/watch?v=WH4F7J7AAFE

[3] Google Self-driving car test: Steve Mahan. [video] accessed April 5, 2014 - https://www.youtube.com/watch?v=cdgQpa1pUUE

[4] National Academy of Engineering. Engineering Grand Challenges - http://www.engineeringchallenges.org/cms/8996/9136.aspx (accessed April, 2014)