Mini-Project 1 - Part A

CSCI 513: Autonomous Cyber-Physical Systems

Instructor: Jyotirmoy V. Deshmukh
TA: Aniruddh Puranic

Due: 11/09/2020, Time: 23:59.59 AoE (UTC -12)

1 Problem Statement

In this part of the mini-project, we will design an adaptive cruise control system. A car (ego car or host car) equipped with ACC typically has a sensor, such as radar, that estimates the distance to the vehicle in front in the same lane (lead car).

The basic operation of ACC is as follows: the driver sets a cruising speed reference *vref_host* that is assumed to be constant across this experiment. If the distance between the host car and the lead car is greater than *dSafe*, then the host car attempts to travel at the driver-set cruising speed. If the distance between the host car and the lead car is less than *dSafe*, then the host car tries to maintain at least the safe distance *dSafe* from the lead car.

The inputs to the ACC are the driver-set reference for the host car, the estimated distance between the lead car and the host car, and the host car's velocity. The output of the ACC-based controller the acceleration value for the host car.

2 Provided Files

The zip file contains the following source files:

1. Incomplete Simulink Model: `ACC_MP1_a.slx`
2. Breach driver script: `ACCBreach_MP1_a.m`

`ACC_MP1_a.slx` To assist your control design, I have provided you a skeleton of the closed-loop control system in `ACC_MP1_a.slx`. This Simulink model file has the dynamical models for the lead car and the host car. These models are identical, and essentially capture the relation between the acceleration, velocity and position for a car. Typically, there is some delay between applying acceleration and increase in the car's velocity. This is modeled through a 'first order delay'

or a 'transfer function block'. Furthermore, we do not want to model a car traveling backward in an ACC setting: thus, when the velocity is close to zero, we prevent the velocity from further decreasing by introducing a saturation block.

The block named 'Adaptive Cruise Control System' has no Simulink blocks inside, and produces a constant acceleration of 1 m/s^2. Clearly, this is incorrect. You will have to design the internals of this block using any appropriate control scheme.

The choice of control algorithm is entirely yours! Do not hesitate to experiment with different controllers.

`ACCBreach_MP1_a.m` In this part, you will be expected to use the tool Breach for your experiments. You can download Breach from this URL:

`https://github.com/decyphir/breach`.

Once you download and install Breach, you can run the command `GenDoc()` to generate the documentation of Breach. Please check the Breach github page for further instructions. The provided file `ACCBreach_MP1_a.m` is a driver script for Breach. Please read the comments in the file for information on each command in the file. For now, you can use the file to just simulate your models and check results.

Later, once we finish discussion on STL, you will use the following requirements to test the quality of your design: `ACC_Requirements.stl`

We will discuss in class how to integrate the requirements with your design. Note that your grade will depend on your design doing a good job satisfying the requerments!