# Optimization of PID controller parameters for simulation of autonomous vehicle path-tracking

## Problem Description

As part of the ME 227 Vehicle Dynamics and Control class offered by Professor Gerdes in Spring 2021, we were tasked with developing a **speed controller** and a **steering controller** and evaluating the performance on simulators of varying fidelity.

We were provided a path that defines 2 oval-shaped laps in the Searsville parking lot. The information consisted of a vector of distances along the path, s, the path curvature κ at each distance, the location of each of these points on the path in terms of their East and North positions, and the local heading.

The requirements for the controller design were:

1. Produce no more than 20 cm of lateral error at the center of gravity

2. Track the speed profile within 0.75 m/s

Chart, line chart

Description automatically generated

Figure 1 Desired speed profile matching two oval-shaped laps

Simulator code was provided by the teaching staff to allow us to test our controller in 3 varying levels of robustness, with the most difficult simulator including a simulated signal noise. The controller would have to perform accurately and reliably on each of these simulators.

Our group designed a PID controller that would use feedback control to reduce the lateral error and the speed error using the state space equations (Figure 2). The problem, however, was that when it came to tuning the three PID parameters (Kp, Ki, and Kd) we couldn’t find a way to optimize the parameters in such a way that our steady state lateral and velocity error would match the requirement on each of the 3 simulators. Our approach was to apply trial and error and evaluate the results visually in MATLAB and store them in an Excel sheet. This took multiple days of fine tuning to reach a point where our controller would meet the requirements in the simpler simulators but not in the noisy one.

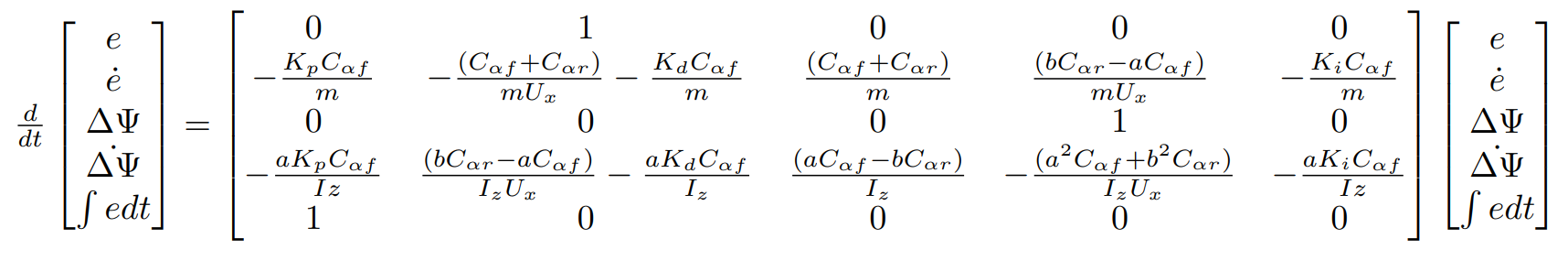


Figure 2 State space representation with PID control

## Problem Approach

## Measuring Performance