Date: 4/25/23

To: The Ohio State University & Jessica Thomas

From: Will Rueter

Subject: Autonomous Steering Simulation

Introduction

The purpose of this project was to create a simulation of a car driving down a road given steering inputs using MATLAB. The problem was to first simulate a car driving given a constant steering input, which would either lead to circular motion with a constant steering input above zero, or a straight line path, given a steering input of zero. Then, using the provided autonomous steering differential equations, the group had to derive a method of simulating the autonomous steering of the car and plot that over top of the actual road, as represented by a polynomial. This was then to be implemented into a graphical user interface. This report will go over the results of this project, discuss successes and failures, and outline what might be considered moving forward for improvement of the simulation.

Results & Discussion

The group first created the simulation with the ability to take a constant steering input by solving the provided differential equations using numerical methods, specifically Euler's method, and implementing this into the syntax of MATLAB. Once this had been tested thoroughly and successfully simulated the car as the provided document stated, the group added in the option for autonomous steering using the provided equations. Additionally, the group added the method of polynomial creation for the car to follow, using MATLAB's symbolic variable system. After finalizing this and reducing bugs, the created simulation function had the following inputs:, Open loop Steering constant (u) Constant Speed (v), Steering Constants (A, B, C)Polynomial Constants (third, second, first, zeroth), and Final Time (tfinal). These led to the following outputs: X position (x), Y position (y), and Error (E) (Figure 6). No supporting functions beyond MATLAB built-in functions were used.

This simulation function was used in two ways. First, it was implemented into a test script, which tested the function against known cases and ensured it produced values within expectation. Second, it was implanted into a MATLAB app with a GUI that displayed the plotted road vs the simulated driving route, and allowed users to change the steering constants, polynomial constants, speed, and time, as well as switch between manual and autonomous modes. It also displays the error of the simulated route from the polynomial if the autonomous mode is selected.

Using the test script and the app, key results were found. First, the simulation is well within acceptable limits for all manual test cases, and follows the behavior outlined in the document. A steering constant of zero leads to a straight line, and increasing values of u lead to decreasing radius circular paths (Figures 1-2). It also withstands autonomous routes, such as linear lines and simple polynomials (Figures 3-4). However, more complex polynomials can lead to higher error or crash the program (Figure 5). Additionally, the best steering parameters for the simulation change with differing velocities and polynomials.

Figure 1: Straight Path with Zero Steering Value

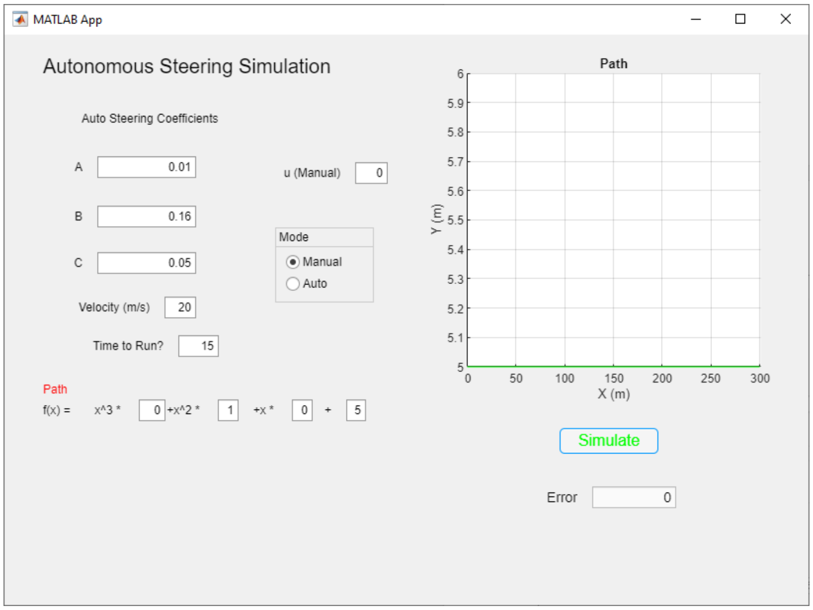


Figure 2: Circular Path with Constant Steering Value

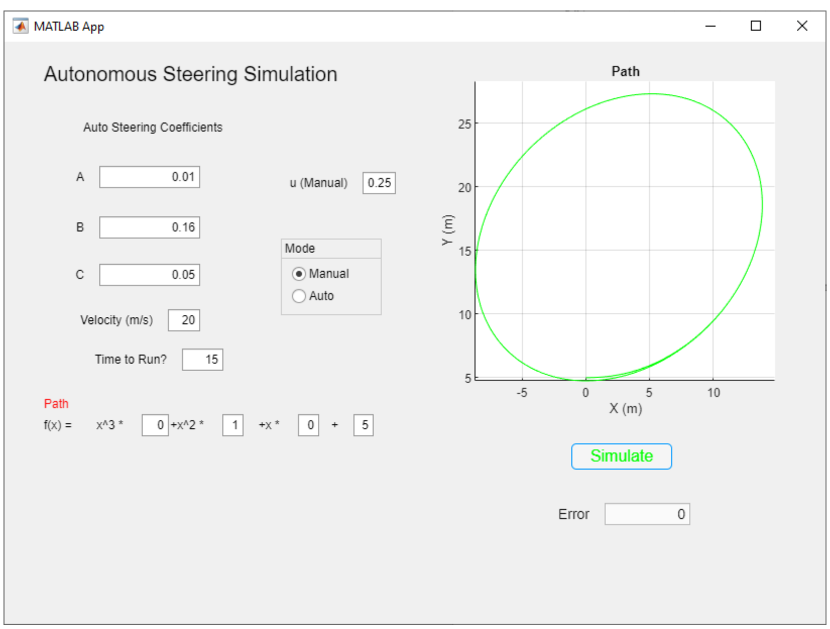


Figure 3: Acceptable Error with a 2nd Degree Polynomial Path Autonomously Followed

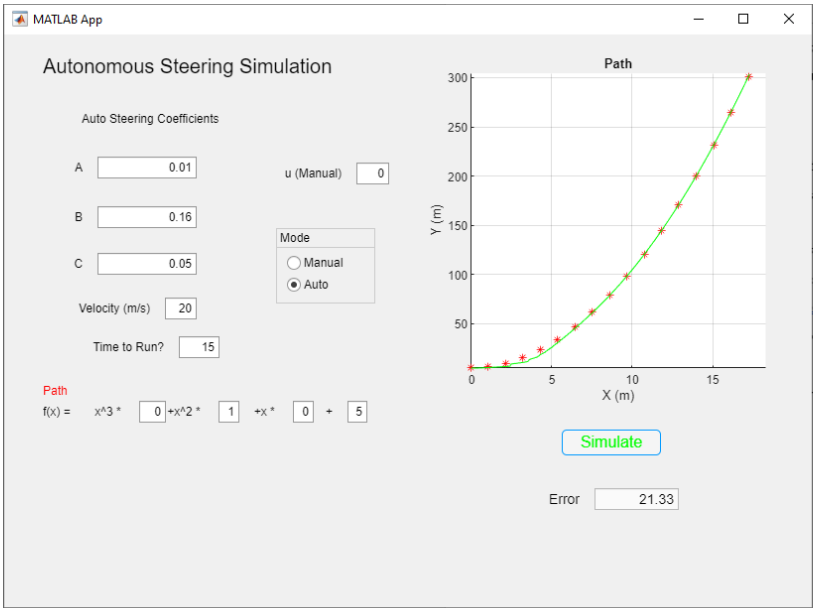


Figure 4: Acceptable Error with a Third Degree Polynomial Autonomously Followed

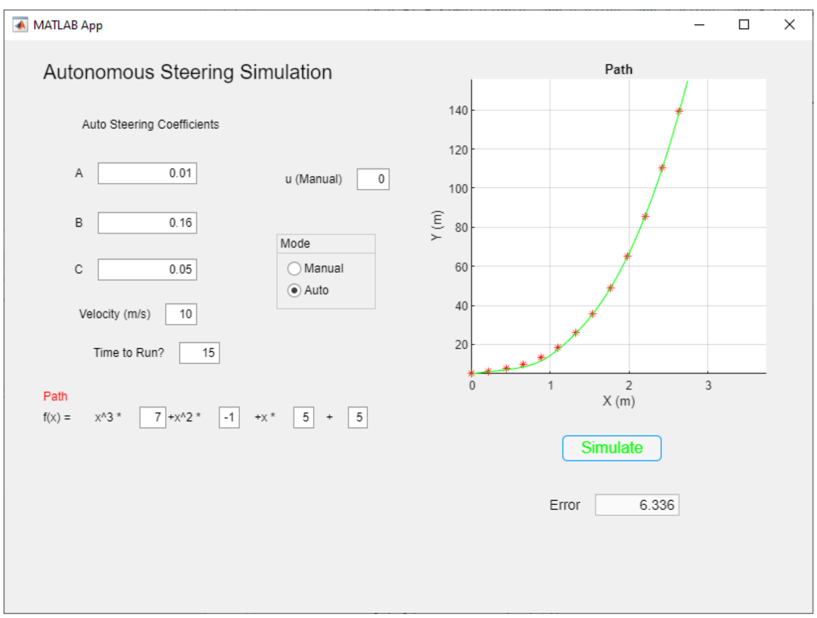
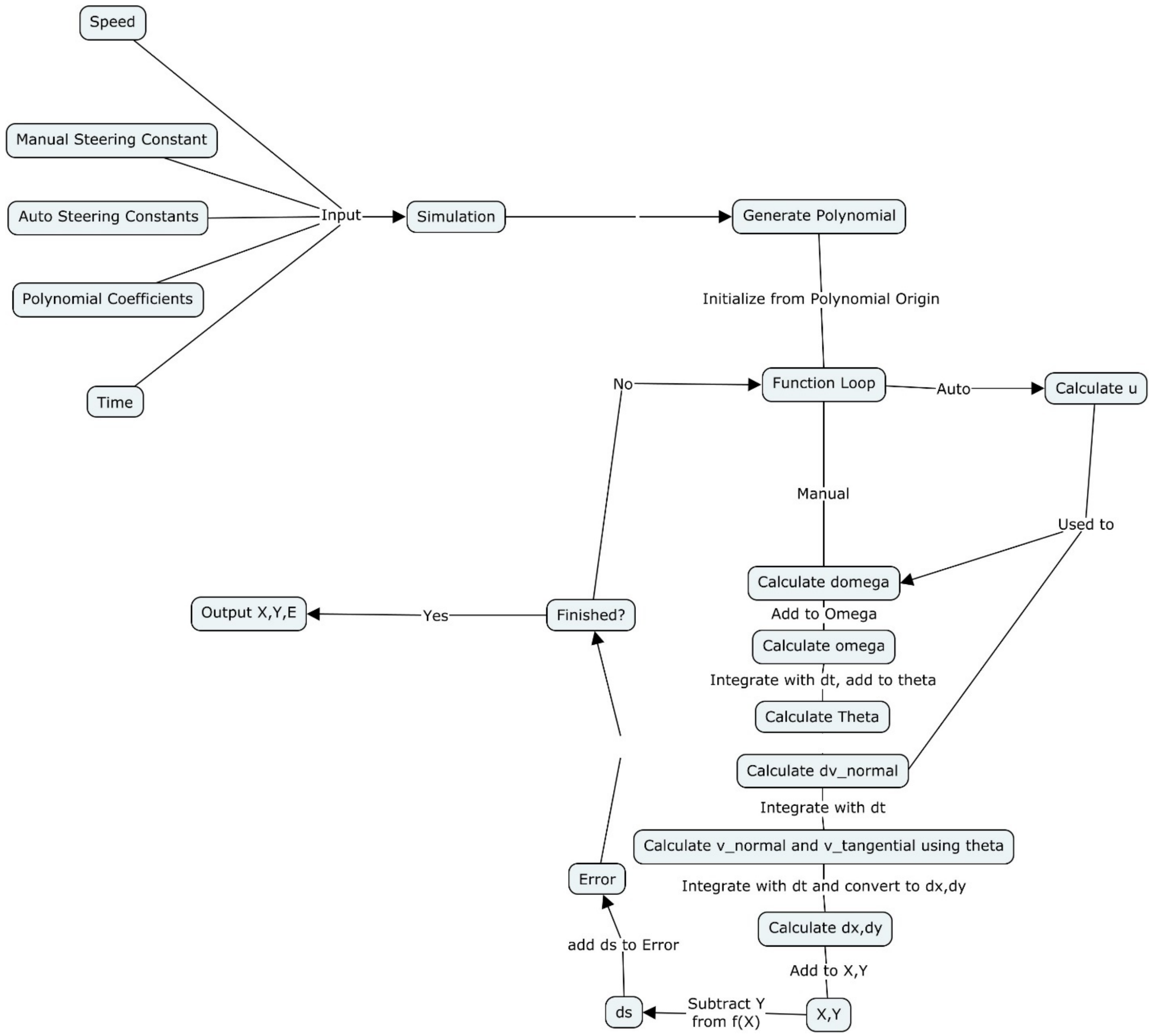


Figure 5: Program Crashed from Complex Polynomial

The qualitative results outlined lead the group to believe that the simulation as it exists now is robust enough to withstand simple polynomial routes of two degrees or less, with manual optimization. Additionally, the manual mode is complete to the group's satisfaction. The changing of optimal steering parameters with differing velocities and polynomials means that the autonomous routine is not finished to the group's satisfaction. However, optimizing the steering parameters was too time-intensive for the group to accomplish, as doing so would have required extreme amounts of computational time, given that the simulation takes at minimum five seconds to run and thousands of different combinations would be required. This could be solved in a few ways. One would be finding more information about the provided steering algorithm, and understanding reasonable limits for the steering constant, as no limits were given. The second is to reduce computation time by improving the efficiency of the simulation. The third is to consider different steering algorithms altogether, as a different algorithm may be better documented, with established literature to base testing on. The group's poor understanding of the steering constants is likely the cause of the current high error of the autonomous simulation, so solving that would significantly improve the overall application and user experience.

Figure 6: Flowchart



Conclusions & Recommendations

In conclusion, the project successfully created a simulation of a car driving down a road, with both manual and autonomous steering. Both are acceptable, but the autonomous steering needs more research and editing before it is fully accurate. Additionally, performance needs to be improved, as the simulation takes an unacceptable amount of time.

The group recommends exploring the given steering algorithm more to better understand the steering constants. Alternatively, other steering algorithms should be explored, as they may be more efficient or accurate, and better documented. It is also recommended to research methods of improving performance, perhaps by increasing the timestep. Additional recommendations include adding an option to create a polynomial on the graph from multiple points by using the polyfit command, adding preset options for different cars to compare steering performance, and introducing different road conditions to increase the complexity of the simulation.

All work was done by Will Rueter.