Planning for Wet Road Avoidance with an Ackermann Vehicle

Bryson Jones, Evan Schindewolf, Braden Eichmeier, Shaun Ryer
School of Computer Science
Carnegie Mellon University
Pittsburgh, PA 15213
Email: {bkjones, eschinde, beichmei, sryer}@andrew.cmu.edu

***Abstract*—Weather related adverse road conditions can be fatal to passenger traffic and costs logistics companies billions of dollars in delays annually. Modern vehicle safety systems are almost purely reactive, in that they only mitigate the damage after losing control, and thus are not able safely traverse or avoid hazardous patches of road by planning a traversal method ahead of time. Here, we propose a proactive planning-based solution which uses forward-looking information to mitigate the risk of crashing due to wet road conditions. In order to demonstrate this capability, we present results from a variety of wet road conditions in simulation as well as physical demonstration on an autonomous 1/5th scale vehicle.**

I. Introduction

Reactive safety measures such as traction control and air bags have reduced road accident fatalities since their introduction. However, every year, 5,000 people die and more than 400,000 are injured in weather-related accidents on US roads [1]. Similarly, the trucking industry loses $3.5 billion due to weather-induced delays [2]. Reactive safety measures cannot eliminate these losses where proactive systems would be able to, given sufficient warning. In this paper we describe the planning algorithm for the ADAPT system, an end to end software pipeline to perceive and navigate through adverse road conditions for a 1/5-scale ackermann vehicle. A computer vision system scans the road conditions in front of the vehicle for water. These regions of water, or puddles, are segmented and converted into a top-down grid representation of the road. This paper focuses on developing a planner to effectively navigate through this 2D grid environment. Maximum velocity for the vehicle is 4 m/s. Analysis of the planning complexity shows the planner must run within 0.5 seconds. To this end, we propose implementing a motion primitive-based A* search, loosely based on recent work in lattice planners [3], to navigate through the road environment.

II. Approach

A. Map Generation

The map of the environment is generated by taking in a 1D row-major map vector, which is an occupancy grid where the values represent confidence of a wet-area or puddle to be in that cell. This occupancy grid data is assumed to be provided information for the implementation purposes of this method. The cell size is specified to be 0.1 meters, with the map being of variable height and width bounds depending on the vector size.

Along with this map vector, a set of waypoints is provided, which act as the features to define the road that the robot traverses. The total road width is set to be 1.5m, and every cell outside of that width from the segments that connect the provided waypoints is marked as inaccessible, to prevent the robot from planing off-road trajectories.

B. Heuristic and Cost Computation

The heuristic computation is based on the time to move to each cell in the map from the goal. To get the minimum time to arrive at each cell, a 2D A* method was used. The A* was composed of a weighted 8-connected wavefront method that started from the target and expanded in a breadth-first manner starting with the four adjacent cells and then the four diagonal cells until the entire map was filled. Dry ground traversal uses the maximum vehicle velocity of 4 m/s to compute the time cost. Puddle traversal assumes a velocity of 2 m/s.

C. Motion Primitive Generation

Whenever the planner expands a node in the search process, the set of next possible states is drawn from a set of pre-generated motion primitives. Primitives are pre-generated using a set of possible steering angles and a set of velocities. To generate a primitive, a fixed steering angle drives from the current velocity to each of the possible velocities for a fixed time interval. The entire motion primitive set consists of all pairs of starting velocity, ending velocity, and steering angle. The set of available primitives is selected from a dictionary of primitive sets with the vehicle's current velocity as the key. Figure 1 shows an example motion primitive set with 11 steering angles and 4 velocities.

After selecting the available set of primitives, the primitives are transformed into the vehicle frame and checked for safety. A primitive is deemed unsafe if any part of the primitive path navigates outside of the map boundaries or crosses adverse terrain at an excessive speed. A new search node is produced for each remaining primitive and the search iteratively continues until the goal is reached.

D. Real-time Modifications

Running a planning algorithm on hardware requires several modifications due to noisy sensor data and imperfect path

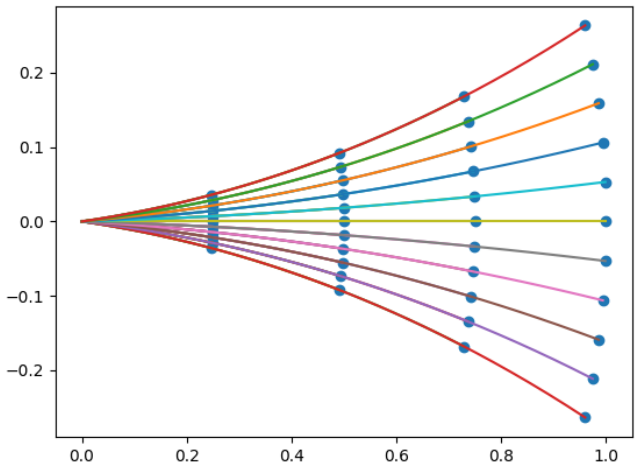

Fig. 1. Example motion primitive visualization.

execution. To adapt to these situations, we make two adjustments to the A* algorithm. The first adjustment is a maximum planning time allotment. By setting a maximum planning time, the algorithm regularly receives updated plans even if the planning does not reach the goal. A second adjustment is implementing a maximum distance horizon. The algorithm ends by selecting the path with the lowest heuristic value once either horizon condition is met.

III. Experiment Analysis

A. Simulation

The first set of testing used a python environment to visualize the vehicle's motion through the environment. Figures 2-3 show the final path in two different environments. The planning time for both maps was about 0.3s. In Fig. 2, the environment was designed to force the car through two large puddles, and drive around several others. In this setting, the vehicle experienced heavy oscillations when avoiding the puddles towards the end of the road. This led to producing a new motion primitive set with more possible steering angles. Figure 3 shows the results on a new map with a single puddle and more complex road layout. The plan shows effective behavior in minimizing the path through the puddle, and cutting the distance in the sharp turn. The oscillatory behavior in the previous map was also alleviated. However, there is still a slight oscillation that occurs due to the restricted set of primitives.

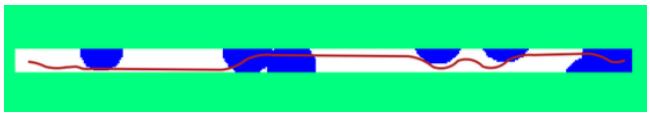

Fig. 2. Straight line road with cluttered puddle set shows.

B. Hardware

The system hardware consisted of a Nvidia Jetson AGX Xavier inside an enclosure on a 1/5 scale remote control

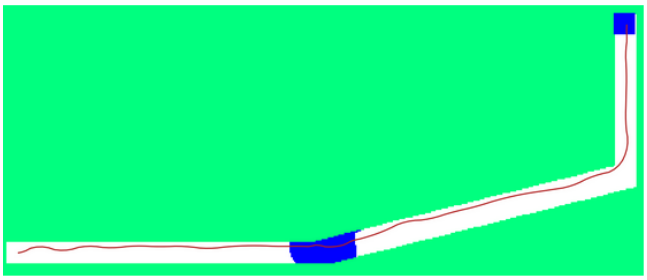

Fig. 3. Large road environment with a single puddle.

camera car, Figure 4. Software infrastructure included a ROS architecture of nodes which provided planning with a live or pre-generated puddle map and the current state of the vehicle, and then passed the resulting trajectory on to the controls system. See Final Presentation in Supplemental Materials for video demonstration.

Fig. 4. Vehicle platform used to test the ADAPT planner.

IV. Conclusion

In this paper, we presented a planning algorithm for autonomously navigating wet road conditions. In order to demonstrate this capability we validated the ADAPT planner in both simulation and on a physical autonomous vehicle over a range of different scenarios. These exhibited that the ADAPT planner can be used to plan safe trajectories that avoid or navigate wet road conditions all but eliminating the risk of accidents due to hazardous road scenarios. [1]

[1]Supplemental Materials:
Final Presentation:
https://drive.google.com/file/d/181r5lTKvuIPyGW7NBhPP_pe9FSVAhDax/view?usp=sharing
Live Demo Recording:
https://drive.google.com/file/d/1w2VxUKgAcJdA1kxtFAgp0NfdIfVupmP3/view?usp=sharing
Initial Project Implementation Outline:
https://docs.google.com/document/d/1gNLvt6GAUcdBbpjJZ5dIw6nGurDKnUftPWfW6ecjPE
edit?usp=sharing

REFERENCES

[1] How do weather events impact roads? URL https://ops.fhwa.dot.gov/weather/q1_roadimpact.htm.

[2] Sean Kilcarr. Mitigating the weather's impact on trucking, November 2016. URL www.fleetowner.com/blog/mitigating-weather-s-impact-trucking.

[3] Mikhail Pivtoraiko. *Differentially Constrained Motion Planning with State Lattice Motion Primitives*. PhD thesis, Carnegie Mellon University, Pittsburgh, PA, February 2012.