Seeds for Principles of Safe Autonomy (ECE498SM) Projects

Sayan Mitra (mitras@illinois.edu) and Katie Driggs-Campbell (krdc@illinois.edu)

These project ideas are provided to get you started. More pointers, software, etc., may be provided once you home-in on a project. Your project can build-up on previous work, but you have to clearly state what is new, nontrivial, interesting, and useful.

You will work on the project with one other person.

1. Important Dates

Jan 23 Team formation; initial project discussion
Mar 13 Intermediate progress report
Apr 29 Poster and demos to course staff
May 11 Final report

2. Project Ideas

2.1 Code an autonomous campus shuttle

Develop the complete software pipeline for an autonomous shuttle that travels on a fixed route in a campus-like environment. You will implement lane detection, pedestrian detection, decision making, path planning, and control in several MPs for this class and test them using the RightHook simulator. In this project, you will improve some of these existing pieces and build new ones to make the shuttle work in *more interesting scenarios*. You can make the scenarios interesting along several dimensions:

- Road geometry with multiple lanes, curved roads, road signs
- Static and mobile obstacles parked cars, traffic cones, and pedestrians
- Driving under visibility, night, and weather conditions

In your project demonstration and report, you should at least cover the following aspects: (i) the technical advancements you have accomplished in the pipeline, (ii) the range of scenarios for which your shuttle software can be assured to work autonomously and safely, and (iii) the argument (design, testing, verification, etc.) supporting the claim in (ii).

2.2 Path planning with dynamic agents

One of the key challenges in autonomous driving pertains to the fact that the intention of other agents on the road have to be inferred by the software in the ego vehicle. The goal of this project is to develop path planners that take into account the possible intentions of all the other agents in the environment of the ego vehicle.

Let us model the intention of each agent by a finite set of paths. Suppose at time t_0 the state of the system is $x(t_0)$---this includes the positions of the agents as well as their intentions. The dynamic path planning algorithm takes $x(t_0)$ as input and computes a path $p(t_0)$ for the ego vehicle. Then, it follows the first

segment of $p(t_0)$ over time δ and repeats the operation with the new state $x(t_0 + \delta)$. Note that the intentions of the agents in $x(t_0 + \delta)$ will have to be updated according to the newly observed states.

You may take some inspiration from this paper: *Intention-Aware Online POMDP Planning for Autonomous Driving in a Crowd* by H. Bai et al. You may also consider taking this one step further and you develop a methodology for better predicting an agent's intent? How does this effect your planning algorithm?

For this project you can use Righthook, other simpler traffic simulators (e.g. Sumo), or build your own simulator. In your project demonstration and report, you should at least cover the following aspects: (i) pseudocode for the algorithm you have implemented, (ii) experimental results exploring how common knowledge about intentions can make driving more efficient, and (iii) correctness argument or proof.

2.3 Controller verification

Verifying that the low-level controller software works correctly in a variety of scenarios is a challenging problem. Tools like C2E2 and DryVR can be used to address this problem and you will learn about them in the second half of the course. This project will be about exploring new applications of these tools in testing autonomous vehicles (AV).
http://dryvr.readthedocs.io/en/latest/publications.html
http://publish.illinois.edu/c2e2-tool/
Possible directions: (a) improve usability by building AV-specific interfaces for C2E2 or DryVR, (b) building a DryVR-RightHook bridge, (c) explore parallelization of verification algorithms.

2.4 Lane Detection in Real World

In an MP, you will implement simple lane detection in a simulation environment. However, the real world is more complicated. We will offer you several video clips filmed from an actual car and the goal of this project will be for you to develop a lane detection module that works well on these and other real-world videos.

Aspects you need to consider: (i) under different weather and light conditions the color of the lane markers will be different, (ii) lines could be erodedd, faded, or even non-existent sometimes, (iii) the camera might vibrate. Your code should give stable output disregarding those difficulties.

Here are some tips to improve robustness: (i) Try to remember previous results. When current frame is not recognizable, just use the most recent results for a period of time. (ii) Try deep learning techniques. Strictly speaking, lane detection is a segmentation problem. You can find many interesting papers in this area such as https://arxiv.org/pdf/1802.05591.pdf. You can implement one of them, make sure it works on our dataset.

2.5 Reinforcement Learning in Righthook

Reinforcement Learning (RL) is a popular method for learning decision making policies in robotics. In one MP, you'll learn about RL and apply it to decision making tasks in a simple simulator. The goal of this project will be to take what you've learned in the RL MP and implement this in Righthook. Once you've defined the scenario, you'll have can explore: (i) different reward functions to achieve good performance, (ii) policy representations and architectures; (iii) algorithmic approaches to guarantee safety, and/or (iv) limitations of the RL methodology and how to interface with other planning and control algorithms.

You'll need to carefully consider the level of abstraction are you considering (e.g., will the input features be image data or processed semantic information? Will you be giving continuous control inputs or high-level commands?), which will guide many of the design decisions you make (e.g., can the problem be solved with Q-Learning? Will you use a deep learning policy representation?). How general of an approach can you design using RL? How does it compare to traditional planning and control approaches? In what sorts of scenarios does this method perform best and where does it suffer?

2.6 Platooning and Multi-Agent Systems

One of the earliest motivations for autonomous vehicles is in the trucking industry. To optimize efficiency for long distance travel, trucks often platoon, an extension of car following where high coordination is required as the following distance is minimized to improve energy efficiency.

This project would require you to design a multi-vehicle platoon, simulate the system in Righthook or another simulator, and then examine the safety aspects of such a cooperative system. What are the safe ranges of operation that you can achieve? What happens when the sensors / communication fails? What are the trade-offs between safety and efficiency?

You further extend the methods for single vehicle planning and control to multi-agent settings. A fun example would be to configure different multi-agent formations that are executed while achieving another goal (e.g., reaching the end of a track by time T, while executing a weaving pattern). Again, rigorous analysis must be completed conveying the real-world justification and impact of this work.

2.7 Mobility as a Service

Once fully autonomous vehicles are ready for the public domain, mobility as a service will be possible. When adding a layer of complexity, is it possible to increase risk? Here are some applications to consider:

- The taxi driver problem: if any autonomous system is trying to get a passenger from point A to point B the most efficient way possible, how should the autonomous system balance efficiency (trying to get to the destination quickly) and safety (driving conservatively)?
    - Check out the Taxi Driver games for motivation.
- Most of the projects we've considered have only considered an ego-centric view of the problem and will not closely look at route planning or fleet planning (like Uber/Lyft). What the assumptions that need to be made about low-level behaviors to make such a system work? Are there security risks in having a centralized controller?
- Are there any unique problems for autonomy that may be used by underserved populations?

For projects following these ideas, a fleshed out analysis must be performed about how the system is to be developed, a working prototype, examples of failure, and measures we can take to improve the system.

2.8 Emergency Scenarios and Switched Systems

No single policy or approach can solve every problem an autonomous vehicle may encounter, especially when considering potentially dangerous situations: The same approach for regular driving may not be

effective in a near collision scenario. Often very complex systems have different modes that operate under certain conditions, and can be thought of as a finite state machine or a hybrid system.

Can you define different modes of operation (e.g., lane keeping, lane changing, car following, near collision, approaching crosswalk, etc.) and combine multiple approaches into a complete, meaningful system? How do you handle the extreme cases and scenarios where there many feasible actions? Is there an optimal switching strategy? Can you guarantee expected, safe operation of this hybrid automata?

3. Expectations and Deliverables

In your project demonstration and report, you should at least cover the following aspects: (i) the technical advancements and algorithms, (ii) the range of scenarios for which your detection can robustly work and where it fails, and (iii) arguments (design, testing, etc.) supporting the claim in (ii).

We will provide Latex templates for the project reports and the posters.