CS221 project proposal: Motion Planning for Autonomous Driving

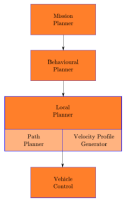
Philippe Weingertner, Minnie Ho

October 21, 2019

1 Problem definition

We study the motion planning problem for Autonomous Driving. The behavioural planner defines a short term objective: where the ego vehicle shall go, with associated constraints. The local planner then computes a set of paths that are kinematically feasible and collision free with respect to static obstacles. For every candidate path, a velocity profile is generated to avoid dynamic obstacles. This two steps decomposition of local planning is typical to make the problem simpler to solve in real time.

Given a path to follow and information about dynamic obstacles, we have to define a velocity profile such that the trajectory is collision free, comfortable (without too strong accelerations or decelerations) and efficient (as close as possible to the target velocity). An autonomous vehicle has to deal with many sources of uncertainties. We consider sensor uncertainty and driving model uncertainty which translate to dynamics uncertainties.



2 Proposed Approach

To define our Oracle we consider there is no uncertainty: sensors are perfect and we know the driving models of every car. The Oracle is a search algorithm without real-time constraints. To define our baseline we set a simple rule for decision making: if the smallest Time To Collision with respect to other cars is below 10 seconds we decelerate by $-2m.s^{-2}$ (as long as we see this collision risk) otherwise we accelerate at $1m.s^{-2}$ (as long as we are below the speed limit) to reach our target as fast as possible.

Ultimately we will handle this problem as a MDP search problem as we are dealing with uncertainty and we want to investigate how we can improve real-time performances. One possible track, as in [2], is to combine planning and learning [1], in a similar way to AlphaZero [4], to have a good heuristic to

efficiently guide the search of a MCTS tree search [3] used for solving online our MDP problem.

2.1 Model

The state space is continuous whereas the action space is discrete. The model used by our Oracle is the following:

- **startState**: random initialization with 5 cars on the left and 5 cars on the right with trajectories crossing the middle path, of the ego vehicle. All the speed are initialized around $20ms^{-1}$. The ego vehicle is $200m$ away from its target point: it shall find a velocity profile that is efficient (fast), comfortable (limiting strong decelerations and accelerations) and collision free for the proposed path.
- **isEnd**: when target is reached by the ego vehicle
- **states**: a vector in $\mathbb{R}^{44}$ with $[x, y, \dot{x}, \dot{y}]$ for 10+1 objects
- **actions**: every 250 ms we choose an acceleration command out of $\{-2, -1, 0, 1, 2\}$ $m.s^{-2}$
- **succAndCost**: the new state is computed based on a Constant Acceleration (over a time step) model for the cars. The cost is 1000 for a collision, 1 for an acceleration command $\in [-1, 0, 1]$ ms^{-2} and 2 for uncomfortable accelerations ± 2 ms^{-2}. The evolution from one state to the next state is deterministic: with probability 1 the surrounding vehicles use an acceleration of 0. We will introduce uncertainty later on.

2.2 Algorithms

We use Dynamic Programming and Uniform Cost Search algorithms to solve the above problem for the Oracle. We use a simple rule, at every time step, for the baseline decision making. We will investigate MDP search in the next steps.

3 Experimental Setup and Status

The source code is available here: CS221 Project

- Baseline: the decision is fast, immediate, but we find a collision free velocity profile only in 35% of the cases.
- Oracle: the search is challenging as the state space is big. With a time step of 250 ms, looking for a collision free velocity profile over the next 100 meters we find in 100% of the cases a collsion free solution; but in 190.7 seconds with UCS running on an iCore9. We explored the graph with a depth of 24, corresponding to a 6 seconds lookahead. Our branching factor is 5.

References

[1] Carl-Johan Hoel, Krister Wolff, and Leo Laine. "Automated Speed and Lane Change Decision Making using Deep Reinforcement Learning". In: *CoRR* abs/1803.10056 (2018). arXiv: 1803.10056. URL: http://arxiv.org/abs/1803.10056.

[2] Carl-Johan Hoel et al. "Combining Planning and Deep Reinforcement Learning in Tactical Decision Making for Autonomous Driving". In: *CoRR* abs/1905.02680 (2019). arXiv: 1905.02680. URL: http://arxiv.org/abs/1905.02680.

[3] Mykel J. Kochenderfer. *Decision Making Under Uncertainty: Theory and Application*. MIT Press, 2015.

[4] David Silver et al. "Mastering Chess and Shogi by Self-Play with a General Reinforcement Learning Algorithm". In: *CoRR* abs/1712.01815 (2017). arXiv: 1712.01815. URL: http://arxiv.org/abs/1712.01815.