Integrated Task and Motion Planning for Instances of Autonomous Driving

Jorge A. Delgado*, Edgar Granados*, Eduardo D. Martínez*, Karen L. Poblete*, Marco Morales
Department of Digital Systems, ITAM
Río Hondo 1, Ciudad de México, 01080, México
{jdelgad7, edgar.granados, eduardo.martinez, kpobleter, marco.morales }@itam.mx

I. Introduction

Autonomous transportation, particularly autonomous driving, is receiving a lot of interest in the research community since the DARPA challenge showed it to be viable with current technologies (e.g. Levinson et al. [5]). It involves a highly dynamic environment where cars need to plan and actuate fast and be responsive to unexpected situations. In such scenarios, cars frequently switch between tasks such as parking, passing a car, switching lanes. Also, they need to reason about the long term effects of their actions.

Here we present ongoing work motivated by an autonomous driving competition where a model car drives on a two-lane road with intersections, everything at a scale 10:1. We address this challenge with a modular system built on ROS [7], that allows us to focus on smaller problems. The car is the *AutoNOMOS mini* robot (Rojas and Boroujeni [6]), a model at scale 10:1 equipped with the typical perception, computing, and acting capabilities found in current autonomous cars.

We are interested in using a minimal set of sensors without relying on global information. Instead, we make a probabilistic localization of the road in the vicinity of the robot using the RGB data of its frontal camera (Intel SR300) and the range reported by its laser scanner (RPLiDAR 360°). We aim to complete high-level tasks given uncertain motion and a probabilistic localization on an incomplete map. A global planner, generated from Linear Temporal Logic specifications (Bhatia et al. [2]), feeds goals to a local planner that attempts to reach them through basic skills, encoded as Partially Observable Markov Decision Processes (Bai et al. [1]).

II. Problem Definition

We address a simplified version of the problem of autonomous driving for a car on a two-way road. It may also have crossroads. There is no global map available, thus decisions need to be made locally only with the information captured on-the-fly. Basically, we want the car to traverse the road as it would be expected from a human driver, but we limit its behaviors. First, it should traverse the road on its lane when it is the only car. Second, if it catches up with a car that goes at a very low speed, it should overtake it. Third, if it finds a crossroads, it should stop and wait for its turn to pass it.

Research supported by Asociación Mexicana de Cultura A.C.
*Equal contribution, ordered alphabetically

III. Environment

A. Perception

1) Local probabilistic localization: We localize the robot with respect to the road based on the visual input using a standard probabilistic estimation of states. We feed the camera data to a road line recognition program (Blahut [3]) that performs an inverse perspective mapping to obtain a bird's eye view of the road, applies a Canny edge detector, and applies Random Sample Consensus (RANSAC) for fast line fitting into Newton Polynomials. We assume that the car can see at least one road line and that there are no significant reflections. Obstruction or partial visibility of lines can be problematic.

We estimate the state of the car with respect to the road as a discrete probabilistic distribution (Fig. 1). First, we compute the distance in pixels between the car and the lines. Next, we mark the states consistent with the observation as hit and the others as $miss$. Finally, the Bayes Rule and normalization are applied over the distribution.

2) Obstacle Detection: We detect obstacles, such as cars, through the LiDAR which produces a 2D point cloud in a range of 6m in our scale model. We identify clusters to map to grid cells in our environment model.

B. Environment Model

We model the environment locally around the robot as a grid as shown in Fig. 1. On one dimension we have the same information as gathered from the localization system. On the other dimension we have the equally spaced values for distance from the car. In this way each cell represents a pair of lane localization, distance from the car. The perception system is also used to determine the cells that may be occupied by obstacles (for our current purposes, other cars) within the perception range (shown as a shaded circle in the figure).

The resulting map that models the environment is not complete nor accurate. We construct it while moving on it, and we have uncertainty over the robot pose since we are relying on a probabilistic localization.

IV. Global and Local planning

We propose a system that interleaves global and local planning. At the global level, we describe the tasks that the car should execute using Lineal Temporal Logic (LTL). At the local level, we propose to keep a set of skills encoded as

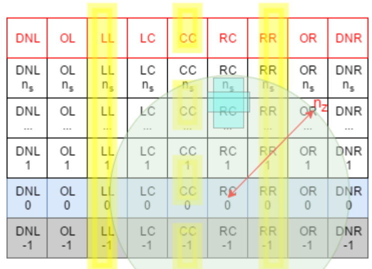

Fig. 1. States defined for the road lanes: Do not know left (DNL), Out left (OL), Left left (LL), Left center (LC), Center center (CC), Right center (RC), Right right (RR), Out right (OR), Do not know right (DNR)

Partially Observable Decision Processes (POMDPs) that may be labelled as *inactive* or *running*, and one of the *running* ones is in control of the robot. In our proposed approach, the local planners also provide feedback to the global planner regarding the potential validity of the future states so that the local planner may disambiguate alternate courses of action.

A. Global planner and decision making

LTL has been successfully used in task representation for motion planning systems (e.g., Bhatia et al. [2], Fainekos et al. [4]). For the global planner, we propose taking decisions based on the probabilistic state of the car and using LTL to describe the tasks for the robot to execute in real time, (e.g. staying in the right lane at constant velocity, or passing a car).

The LTL functions are translated to minimized Deterministic Finite Automatons (DFA) using a simplified method based on (Bhatia et al. [2]) double layered planner. The DFAs are designed so the regions of interest of the LTL propositions are related to the environment model described in Sect. III-B.

The global planner is designed around the main goal of driving forward at the center of the right lane. With this purpose, we can define basic skills such as *speeding up*, *slowing down*, *passing a car*, and *stopping* in relation to the states of the environment model discussed in Sect. III-B and shown in Fig.1. Most of the time, the car needs to be on the right center (RC) state or on the left center state (LC) when overtaking another car. Other states are generally avoided. When we are approaching a car on the right lane, we are getting closer to the region just behind the other car (e.g., RC_2). While in this region, the car either slows down or starts passing the other car. The decision of what action to take may be based on the feedback provided by the local planners regarding the validity of future states in the corresponding skill. If the decision is to slow down, the task would be defined as moving to the adjacent region RC_1 staying behind it always on the right lane. If the decision is to overtake the car ahead, the task would be defined as visiting a sequence of regions starting on the right lane behind the car, then getting in the left lane (LC_n), and merging again in the right lane in front of the other car (RC_5) $\beta = \diamond(RC_1 \wedge \diamond(LC_n \wedge RC_5))$.

The decisions made by the global planner are based on the environment model processed by the perception system and also in simple outputs from the sensors and controller that can be mapped to simple boolean values.

B. Local planning with POMDP

The use of POMDPs in autonomous driving has shown good results but it is costly (Bai et al. [1]). Some hierarchical approaches have had good results. In our case, we propose to define our POMDPs as follows: The states are the ones discussed in Sect. III-B and illustrated in Fig. 1 (with the robot in state *RC0*), the observations come from our sensors, the actions allow the robot to move to the adjacent cells, the transition probabilities come from the joint probability of failure in sensors and the uncertainty of action execution, the estimation of observations given actions and states come from experiments in the robot, and the discount factor for immediate and future rewards is a user-defined constant to be adjusted.

V. Control

We use a Proportional and Integral (PI) controller which receives a coordinate of the target state in pixels and through a vision feedback from the camera provides a degree of direct steering to the vehicle.

VI. Conclusions

We present ongoing work on a system that integrates Task and Motion Planning for Autonomous Driving. Our modular system is implemented in model car to drive on a two-way road with intersections on a scale 10:1. We focus on the completion of high-level tasks based on low-level skills.

References

[1] Haoyu Bai, Shaojun Cai, Nan Ye, David Hsu, and Wee Sun Lee. Intention-aware online pomdp planning for autonomous driving in a crowd. In *Robotics and Automation (ICRA), 2015 IEEE International Conference on*, pages 454–460. IEEE, 2015.

[2] Amit Bhatia, Lydia E Kavraki, and Moshe Y Vardi. Sampling-based motion planning with temporal goals. In *Robotics and Automation (ICRA), 2010 IEEE International Conference on*, pages 2689–2696. IEEE, 2010.

[3] Vaclav' Blahut. FU line detection. https://github.com/AutoModelCar/AutoModelCarWiki/wiki/FU-line-detection, 2017. [Online; accessed 17-April-2017].

[4] Georgios E Fainekos, Antoine Girard, Hadas Kress-Gazit, and George J Pappas. Temporal logic motion planning for dynamic robots. *Automatica*, 45(2):343–352, 2009.

[5] Jesse Levinson, Jake Askeland, Jan Becker, Jennifer Dolson, David Held, Soeren Kammel, J Zico Kolter, Dirk Langer, Oliver Pink, Vaughan Pratt, et al. Towards fully autonomous driving: Systems and algorithms. In *Intelligent Vehicles Symposium (IV), 2011 IEEE*, pages 163–168. IEEE, 2011.

[6] Raul Rojas and Zahra Boroujeni. Automodelcar. https://github.com/AutoModelcar, 2017.

[7] ROS. Ros.org, 2016. URL http://wiki.ros.org/.