An Application of POMDPs for Autonomous Driving

Jorge A. Delgado*, Edgar Granados*, Marco Morales
Department of Digital Systems, ITAM
Río Hondo 1, Ciudad de México, 01080, México
{jdelgad7,edgar.granados,marco.morales}@itam.mx

***Abstract*—We present ongoing work on a system for autonomous driving with a global planner that arbitrates the activation, execution, and control of a set of local skills. The local skills, to be implemented as POMDPs, set the control inputs for speed and steering. Our system relies on a probabilistic localization of the robot on the road. We are at an early stage of our research, but we already have a working probabilistic perception system and a working controller. We focus on the definition of the skill POMDPs to be implemented.**

I. Introduction

In this paper we present an application of POMDPs to the problem of skill representation for autonomous driving. Autonomous driving has captured a lot of interest since the DARPA challenge (e.g., Levinson et al. [8]) showed it to be viable with recent technologies. Moreover, fully autonomous vehicles may significantly reduce accidents. Also, networks of autonomous vehicles could alleviate traffic congestion.

Autonomous driving involves a highly dynamic environment where cars need to plan and actuate fast while being responsive to unexpected situations. In such scenarios, cars frequently switch between tasks such as parking, passing a car and switching lanes. Also, they need to reason about the long term effects of their actions.

Here we present ongoing work on autonomous driving where we propose a global planner for tasks that selects active and controlling skills that set the control inputs for speed and steering. We do not rely on a global map nor on global sensors. Instead, we make a probabilistic localization of the road in the vicinity of the robot using on-board sensors.

The global planning part of this approach is discussed in work submitted to a separate workshop. This paper focuses on the definition of skills that we propose to implement through Partially Observable Markov Decision Processes Bai et al. [1].

Although this is a very early version of our work, we already have working components for perception, localization, and low-level control. We are currently working on the motion planning component with a scale model of the car and the road. The *AutoNOMOS mini* robot version 1.0 Rojas and Boroujeni [9] equipped with a laser scanner (RPLiDAR 360°), Intel 3D Camera SR300, an IMU and an on-board Linux system running ROS [10]. We currently only use its proprioceptive motor sensors, the RGB data of its frontal camera and the LiDAR (in the future we plan to only use the camera).

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

II. Related Work

Different levels of driving autonomy were defined by SAE [11]. Each of these levels require increasing capacities on perception and planning in order to detect objects and respond to events according to the situation giving priority to safety. Most current approaches for this depend on a combination of vision, laser scanner and GPS.

When driving a vehicle there are several tasks that have to be accomplished simultaneously such as crashing and keeping the vehicle on its lane. This is a suitable application for integrated task and motion planning Fainekos et al. [5], Bhatia et al. [2] that exploits the ability of symbolic task planners to express tasks in terms of conditions and sequences of actions with the effectiveness of motion planners to define motions required to achieve a specific configuration.

Sampling based motion planners have been applied to vehicle driving. One case is Kuwata et. al. Kuwata et al. [7] who apply nonlinear control and RRTs.

POMDPs have been applied to autonomous driving. POMDPs are costly but they are better than greedy alternatives for long term planning. Foka and Trahanias Foka and Trahanias [6] propose a hierarchical POMDP for localization, planning and local obstacle avoidance with good real time results in both simulated and real environments. Vitus and Tomlin Vitus and Tomlin [14] model the behaviour of human drivers to determine the best action for the robot. Ulbrich and Maurer Ulbrich and Maurer [13], implement lane changes using signal processing networks which define eight states, keeping complexity low. Bai et. al Bai et al. [1] address the problem of driving in a crowd through an online POMDP as a speed planner that runs in almost real time.

III. Problem Definition

Here we address a simplified version of the problem of autonomous driving for a car on a two-way road using a scale model. The road has one lane for each direction. It may also have crossroads. There is no global map available of the road, 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. It should traverse the road on its lane when it is the only car. If it catches up with a car that goes at a very low speed, it should overtake it. At crossroads it should stop and wait for its turn to cross.

IV. The Robot and its Environment

Here we describe the perceptual components of our approach and how we model the environment.

A. Local probabilistic localization

We first localize the robot with respect to the road based on the visual input and a standard probabilistic estimation of states. We feed the camera data to a road line recognition program [3] that performs an inverse perspective mapping to obtain a bird's eye view of the road. Next, it applies a Canny edge detector. Finally, it 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. Insufficient data points due to obstruction or partial visibility of lines can be problematic.

Once we have a set of lines, 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 observed lines. Next, based on these distances we determine the states consistent with the observation marking them as hit and the others as $miss$. Finally, the Bayes Rule and normalization are applied over the distribution.

B. Obstacle Detection

In order to detect obstacles, such as cars, we use the LiDAR which produces a 2D point cloud data around the car in a range up to 6m in our scale model. So far we only identify clusters that we use to map to grid cells in our environment model described in Sect. IV-C.

C. 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. Thus, each cell represents a pair of lane localization, distance from the car. The perception system also marks the cells that may be occupied by obstacles (e.g., other cars) within the perception range (a shaded circle in the figure). The resulting map is not complete nor accurate since the robot is moving and the probabilistic localization provides a robot pose with high uncertainty.

V. Global and Local planning

We propose a system that interleaves global and local planning based on Rodney Brooks' subsumption architecture [4] which keeps a set of behaviors as a referee picks one to execute while subsuming all the other behaviors. At the global level, we describe the tasks that the car should execute. At the local level, we propose to keep a set of skills encoded as Partially Observable Decision Processes (POMDPs) that may be labelled as *inactive* or *running*. Out of the *running* skills, one and only one is in *control* of the robot. The local planners also provide feedback to the global planner regarding the potential validity of the future states.

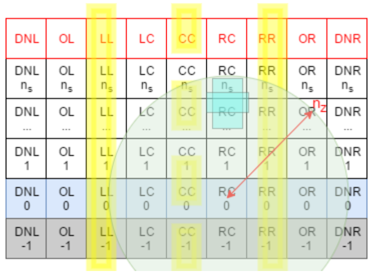
DNL OL LL LC CC RC RR OR DNR
DNL n_s OL n_s LL n_s LC n_s CC n_s RC n_s RR n_s OR n_s DNR n_s
DNL ... OL ... LL ... LC ... CC ... RC ... RR ... OR ... DNR ...
DNL 1 OL 1 LL 1 LC 1 CC 1 RC 1 RR 1 OR 1 DNR 1
DNL 0 OL 0 LL 0 LC 0 CC 0 RC 0 RR 0 OR 0 DNR 0
DNL -1 OL -1 LL -1 LC -1 CC -1 RC -1 RR -1 OR -1 DNR -1
n_z

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)

The available skills model clearly different behaviors for the car including *driving the road*, *negotiation of an intersection*, *parking*, *changing lanes*, *emergency stop*.

A. Global planner

The global planner is in charge of coordinating the skills. We currently propose a Task Planner based on LTLs, which we are describing in a poster submitted to another workshop in RSS. Here we focus on the interaction between the global and local planner.

The global planner is designed around the main goal of identifying the most relevant skill to use according to the current situation. It determines the next goal based on the states of the environment model processed by the perception system that, in addition to the states of the environment, also provides information about the presence of other objects in the surrounding area. Then, it keeps *running* the skills that are consistent with that goal and perceptual input and it makes a plan of the next cells to visit. Then, it gives *control* to the skill that is the most suitable to achieve the next goal.

B. Local planning with POMDP

We propose to define a set of POMDPs as skills so that the global planner *decides* which ones are in control, running or inactive depending on sensors information. Some of these POMDPs will share the following elements of the tuple $\langle S, Z, A, T, O, R, \gamma \rangle$.

- S: The set of discrete states (squares in our grid world) to consider as function of the sensors observations, such that $|S| = N_S$.
- Z: The set of observations made by our sensors, used to partially build our grid world, such that $|Z| = N_Z$.
- A: The set of actions the car can execute. For simplification we consider high-level definitions, with seven possible move actions: $A = \{\nwarrow, \uparrow, \nearrow, \searrow, \downarrow, \swarrow, \oslash\}$, where $\oslash$ means not to move. These actions are defined by the non-holonomic constraints of the car.
- $\gamma \in [0, 1]$. We use a discount factor of $\gamma = 0.5$ in $E[\sum_{t=0}^{\infty} \gamma^t r_t]$ to give equal weight to immediate and future rewards.

- $T : S \times A \times S \rightarrow [0,1]$ such that $T(s,a,s') = P(s'|a,s) = p_{sas'}$, with $p_{sas'}$ assigned by a joint probability of failure in the car sensors and mechanical imprecision when executing an action.
- $O : Z \times A \times S \rightarrow [0,1]$ such that $O(z,a,s) = P(z|a,s) = p_{zas}$, where p_{zas} is an estimation of the current state of the car.

The main difference between these two POMDPs are the rules to assign *rewards* to the states as follows:

- $R_i : S \times A \rightarrow \mathbb{R}$, $i = 1, 2$. We assign high *costs* c_h to states with obstacles, *medium* costs c_m to states outside the lanes and low costs c_p to states matching obstacle-free lanes. Thus, we define the *reward* function for each for each POMDP in order to achieve the following behaviors:
    1) R_1 *(Drive and overtake):* Generally drive in the right lane and overtake obstacles when possible.
    2) R_1 *(Deal with a crossroad):* Stop ($\oslash$) at crossroads and move on when its turn comes.

Figure 1 illustrates this tuple with the car at state *RC0*.

Remark 1. $0 < N_Z \leq N_S$.

Remark 2. *In order to work on different types of roads we will need to estimate $N_S - N_Z$ states.*

Remark 3. $r_h < r_m < r_p$.

For any additional *skill* a new POMDP can be defined (e.g., *Decisions for parking* and *Passing roundabouts*). Important aspects when added POMDPs are their state definitions and the global planner mechanisms to choose among them.

This approach seeks to reduce computing costs by reducing the size of one POMDP. An important algorithmic challenge is to determine a solution. At [12] an efficient way to exploit the sparsity of the matrices and avoidance of solving linear programs for value iteration reduces computing time. Parallel solution computing addresses the issue, but may result on resources not used efficiently

.

VI. Control

We control the driving speed and steering angle. The driving speed is constant. The steering angle is determined through a Proportional and Integral (PI) controller which receives the target state in pixels and vision feedback from the camera.

VII. Conclusions

Here, we present ongoing work on a system for autonomous driving with a global planner that arbitrates local skills to be represented as POMDPs. The local skills set the control inputs for speed and steering. Our system relies on a probabilistic localization of the robot on the road. We are at an early stage of our research, but we already have a working probabilistic perception system and a working controller. Here we focus on the way we describe the POMDPs which we are currently implementing.

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] Rodney Brooks. A robust layered control system for a mobile robot. *IEEE journal on robotics and automation*, 2(1):14–23, 1986.

[5] 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.

[6] Amalia Foka and Panos Trahanias. Real-time hierarchical pomdps for autonomous robot navigation. *Robotics and Autonomous Systems*, 55(7):561–571, 2007.

[7] Yoshiaki Kuwata, Justin Teo, Gaston Fiore, Sertac Karaman, Emilio Frazzoli, and Jonathan P How. Real-time motion planning with applications to autonomous urban driving. *IEEE Transactions on Control Systems Technology*, 17(5):1105–1118, 2009.

[8] 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.

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

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

[11] Taxonomy SAE. Definitions for terms related to on-road motor vehicle automated driving systems-j3016. *Society of Automotive Engineers: On-Road Automated Vehicle Standards Committee*, 2013.

[12] Trey Smith and Reid Simmons. Point-based pomdp algorithms: Improved analysis and implementation. *arXiv preprint arXiv:1207.1412*, 2012.

[13] Simon Ulbrich and Markus Maurer. Probabilistic online pomdp decision making for lane changes in fully automated driving. In *Intelligent Transportation Systems-(ITSC), 2013 16th International IEEE Conference on*, pages 2063–2067. IEEE, 2013.

[14] Michael P Vitus and Claire J Tomlin. A probabilistic approach to planning and control in autonomous urban driving. In *Decision and Control (CDC), 2013 IEEE 52nd Annual Conference on*, pages 2459–2464. IEEE, 2013.