Learning MDP-ProbLog Programs for Self-Driving Cars†

Alberto Reyes[1], Héctor Avilés[2], Marco Negrete[3], Rubén Machucho[2], Karelly Rivera[2], Gloria Isabel de-la-Garza-Terán[2]
[1]National Institute of Electricity and Clean Energies, [2]Polytechnic University of Victoria, Mexico, [3]Faculty of Engineering, National Autonomous University of Mexico

†This work is supported by UNAM-DGAPA under grant TA101222 and Mexican Consortium of AI CIMAT-CONAHCYT

Outline

1. Motivation
2. Description of the self-driving system
3. Research proposal
4. Conclusions and future work

Introduction

- Self-driving cars promise some advantages for the future of human mobility (e.g., safety, economy, health, pollution)[1]
- These machines must perceive and take critical driving decisions in complex environments with several objects around (e.g., pedestrians, other vehicles, various obstacles)
- The components of traditional hardware-software arquitectures can roughly be divided into perception, planning and control
- An important component is the behavior selection module that is responsible of the repeated selection of reactive or short-term driving actions (stop, go, pass, brake)

[1] Motivational Tesla video

Introduction

- Our work[2] is focused on the development of different capabilities for a simulated self-driving[3] car that include:
  - Visual perception for lane detection, and obstacle detection and speed estimation using 3D-LiDAR
  - Lane tracking and the execution of driving behaviors via speed and steering control
  - Recurrent selection of one of four driving behaviours accordingly to the current environment: stop, cruise, keep distance, overtake

[2] Avilés, H., Negrete, M., Machucho, R., Rivera, K., Trejo, D., & Vargas, H. Probabilistic Logic Markov Decision Processes for Modeling Driving Behaviors in Self-driving Cars. In Ibero-American Conference on Artificial Intelligence (pp. 366-377). Springer.

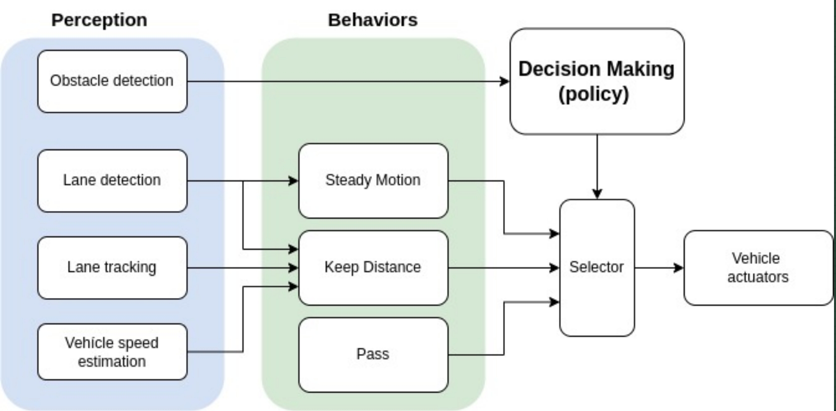
[3] Webots, "http://www.cyberbotics.com." Open-source Mobile Robot Simulation Software.

Introduction

Example(s) of the driving system

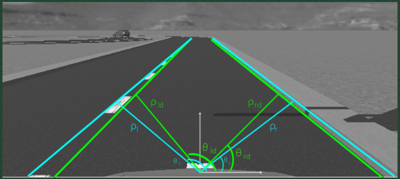
Video(s)

General architecture

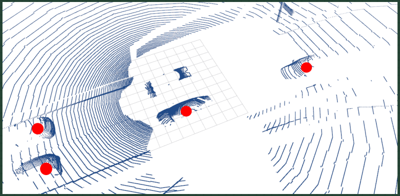


Self-driving system

Lane detection based on edge detection and Hough Transform



Obstacle detection based on Lidar sensor



Lane tracking and keep distance using proportional control

$$v = C_b \quad (1)$$

$$\delta = K_\rho e_\rho + K_\theta e_\theta \quad (2)$$

with

$$e_\rho = ((\rho_{ld} - \rho_l + \rho_{rd} - \rho_r)/2)$$

$$e_\theta = ((\theta_{ld} - \theta_l + \theta_{rd} - \theta_r)/2)$$

Pass behavior using Finite State Machines

Self-driving system

- The behavior selection module is based on an action policy obtained from probabilistic logic description of a factored Markov decision process designed by hand
- Causal discovery
  - Explainability <-> Accident causes identification.
  - RL -> Learned policies for causal discovery.

Factored MDPs

- FMDPs are composed by a 5-tuple $(\mathrm{X}, \mathbf{X}, \mathcal{A}, p, R)$, in which:
  i) $\mathrm{X} = \{X_i\}_{i=1}^{n}$ is a set of n discrete state random variables,
  ii) $\mathbf{X}$ is a set of all possible n-tuples $\mathbf{x} = (x_i)_{i=1}^{n}$, such that x_i is a particular value of the random variable X_i, for all $i = 1, ..., n$ (each tuple $\mathbf{x}$ defines a state of the system)
  iii) $\mathcal{A}$ is a set of possible actions the decision maker can choose,
  iv) $p(\mathbf{x}'|\mathbf{x}, a) \in [0, 1]$ is a discrete, joint probability transition function, where $\mathbf{x}, \mathbf{x}' \in \mathbf{X}$, and,
  v) $R(\mathbf{x}, a)$ is the reward model.

Factored MDPs

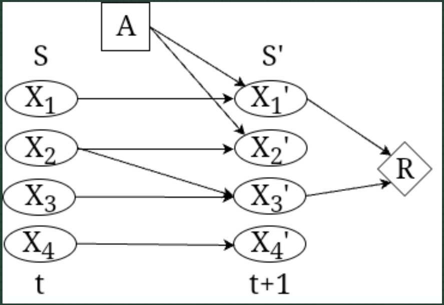


Figure: An example of a influence diagram representing a transition function.

MDP-Problog program

- An MDP-ProbLog program[4] is a tuple $PL = (\mathrm{X_F}, \mathcal{A}, \mathcal{U}, \mathcal{T}_r, \mathcal{R}_r, \mathcal{C})$, such that:
  1) $\mathrm{X_F}$ is a finite set of n *state fluents*
  2) $\mathcal{A}$ is a finite non-empty set of m actions
  3) $\mathcal{U}$ is a finite set of utilities
  4) $\mathcal{T}_r$ is a finite non-empty set of factored transition rules
  5) $\mathcal{R}_r$ is a finite set of reward rules that compose the *reward model*
  6) $\mathcal{C}$ is a finite set of complementary atoms (atoms that are neither state fluents nor actions)

[4] T. P. Bueno, D. D. Mau´a, L. N. De Barros, and F. G. Cozman, "Markov decision processes specified by probabilistic logic programming: representation and solution," in 2016 5th Brazilian Conference on Intelligent Systems (BRACIS), pp. 337–342, IEEE, 2016.

States, reward function and actions

- The state of the system is described using four binary state fluents North, North-West, West, and South-West
- A positive reward is assigned whenever the self-driving car has free space in front of it, and different negative rewards for rear crashes and side swipe crashes are considered
- Actions are: change_lane, overtaking, keep_distance

States of the system

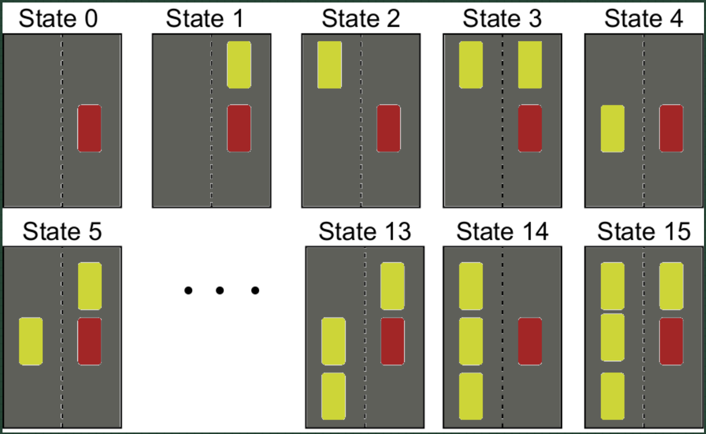


Figure: MDP States of the system. The red rectangle represents the self-driving car and the yellow rectangles are nearby (obstacle) vehicles.

Factored transition rules

- An example of factored transition rules for the future state fluent $free\_N'$ given the current state and action $overtaking$ is:

$$0.9 :: free\_N(1)\ :- free\_NW(0), free\_W(0), overtaking.$$
$$0.05 :: free\_N(1)\ :- (not(free\_NW(0)); not(free\_W(0))), overtaking.$$

(each rule represents probability value $p(x'|\mathbf{x}, a)$, where $x' \in \mathrm{X}_F$, $\mathbf{x} \in \mathbf{X}$, and $a \in \mathcal{A}$)

Self-driving system

- The reward function is based on (independent) additive utilities assigned to actions and state fluents:

$$utility(free\_N(1), 5).$$
$$utility(rear\_crash(1), -30).$$
$$utility(side\_crash(1), -10).$$
$$utility(keep\_distance, -10).$$
$$utility(overtaking, -1).$$

$$0.99 :: rear\_crash(1) \; :- not(free\_N(1)), \\ steady\_motion, not(keep\_distance).$$
$$0.95 :: rear\_crash(1) \; :- not(free\_NW(1)), overtaking.$$
$$0.95 :: side\_crash(1) \; :- not(free\_W(1)), overtaking.$$

Dataset recording

- The following data are logged:
  - The value of the current state fluents
  - The action selected
  - Simulation time
  - Accelerometer data (for crash detection)
  - Success or failure of each maneuver
  - (x, y, θ) of the self-driving car (obtained from the simulator)
  - The relative distance to other vehicles
  - Speed and steering of the self-driving car (relative speed and steering of other cars is forthcoming)

Learning causal relationships

- To learn an MDP-PL, data are collected from random driving actions performed by the self-driving car on the environment.
- This data will be partitioned and sequentially registered in ordered tuples $d_t(\mathbf{x}, a, r)$ indexed in time $t \in \{1, ..., T\}$, such that $T \in \mathbb{N}$, $\mathbf{x} \in \mathbf{X}$ is the current observed state, $a \in \mathcal{A}$ is the current performed action, and r is a numerical reward value assigned to $\mathbf{x}$ and a.

Learning causal relationships (2)

- In the first learning stage, the K2 algorithm will be used to learn the transition function $p(\mathbf{x}'|\mathbf{x}, a)$.
- The reward function will be obtained by using J48 to generate a reward decision tree .
- In the second learning stage, the ID will be converted into an MDP-PL.
- The SPI software [5] allows to approximate different types of MDPs from data

[5] Reyes A., Ibargüengoytia P.H., Santamaría G. (2019) SPI: A Software Tool for Planning Under Uncertainty Based on Learning Factored MDPs. In: Martínez-Villaseñor L., Batyrshin I., Marín-Hernández A. (eds) Advances in Soft Computing. MICAI 2019. Lecture Notes in Computer Science, vol 11835. Springer, Cham. https://github.com/albreyes/factoredMDPs

Conclusions and future work

- We presented a proposal to learn causal relationships between state variables that model a sequential decision process using MDPs for a self-driving car
- Our work aims to develop perceptual and control capabilities of a self-driving car
- A realistic robot simulator is used as a test bed of the self-driving car and to construct a database of driving examples
- The database includes information about time, relative speed of other cars, the current and future state variables, the reward received in each state, the success or failure of the action
- Future work: Extend the simulated car with other sensors such as motor enconders, and other capabilities such as human detection

Thank you!

Questions?

areyes@ineel.mx,
{havilesa,rmachuchoc,1930435,2130071}@upv.edu.mx,
marco.negrete@ingenieria.unam.edu