A search-based multi-agent Markov Decision Model in Lane Change

Author： Mingmin Song

I. Introduction

The research on automated vehicles has progressed significantly recent years after the DARPA Grand Challenge [1] and Urban Challenge [2] in both academia and industry. Google has been well known for testing its autonomous vehicles through road test and Tesla has equipped the highly automated driving system in his models. These developments have been accelerated by advancement in sensing and computing technologies to improve sensing and perceptions so as to achieve accurate control on vehicles. Among all the system architectures, the planning system is one of the most import component in terms of its function in connection with perception and control. The planning subsystem typically include common path planners, behavior planners, and control [2].

One key challenge in the planning system is the ability to handle uncertainties either due to inaccurate instruments or unknown intentions of all agents participated in the system. The partially observable Markov decision process (POMDP) is a powerful mathematical model that captures uncertainties in robust decision making and planning [3]. To be more precisely, the uncertainties consists of not only the continuous state such as actual position, velocity, accelerate of vehicles, but also discrete uncertainty in intention such as to yield or not yield when observing the signals of lane change of subject vehicles.

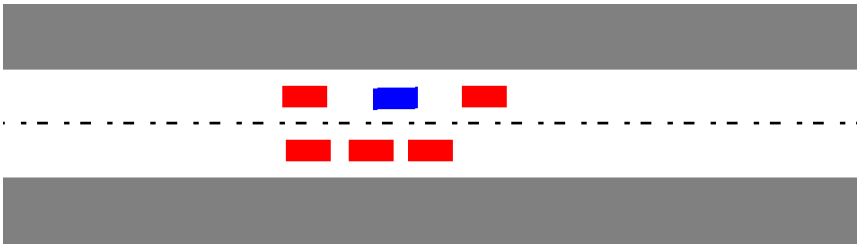

Figure 1 Lane change setting

Direct managing the discrete uncertainly is important to drive safely on the road and make vehicles have human judgement on other vehicle's driving intentions. In the current research, the intention estimate schema is proposed to estimate the likelihood of yield of surrounding vehicles on observing the lane change signal giving by the subject vehicle. Also, the dependence of agent intention on each other is also taken into account, for instance, vehicle tends to follow closely on others which gives rise to the dependency of vehicles such Lane change in the above model while multiple vehicles tends to follow closely with each other as shown in Figure 1.

In the route planning system, the hybrid A* [4] algorithm is applied to search for path through possible lane change routes and generate sets of actions, which is provided to POMDP system model. Unfortunately, solving POMDP models directly is intractable and impractical for online planning. Therefore, approximation and assumptions are made to in the current research work to approximate the solutions. More specifically, the intention probability evolution of surrounding

obstacle vehicles is tracked and further evaluated together with the action sets at any instantons decision point and the best action set with no collision and high utility will be chosen as current decision outcome.

II. Related Work

The decision making in contemporary autonomous driving system is typically hierarchically structured into route planning, behavioral decision making, local motion planning and feedback control [5]. In the highest level of decision making, a vehicle's decision making has to select a plausible collision free path through the current network. The current practical path planning algorithm deployed in field robotic cars differs significantly. In the DARPA Urban Challenge, the CMU boss's [2] has employed a variational techniques for local trajectory generation, while Stanford's team [1] has used hybrid A* search strategy by recursively building search trees, and MIT's team [6] used a sampling-based variant of RRT algorithm. Among all these different path planning techniques, a hybrid A* algorithm proposed by Dmitri Dolgov [4] is simple and able to produce smooth paths for autonomous vehicle and proven to operate well.

In the behavior decision making stage, attentions have been directed on how to find an on-line approximation algorithm for autonomous vehicles. Online search algorithms for POMDP models has been widely studied to deal with large number of states for decision making in games and handle the uncertainties of environment changes [7,8,9,10] in the game strategy search. However, these models are not applied extensively in the automated driving field due to the expensive computation cost and therefore simplified POMDP models with reasonable assumptions are studied based on the specific scenarios. Wei [11] has proposed a point-based MDP model for single-lane autonomous driving behavior control under uncertainties in sensor noise, perception and surrounding vehicles behavior. Simon [12] has used a AND-OR tree search method for the online POMDP model to accommodate sensor noise in traffic scenarios. Enrich and Alexander [13] has showed a multi-policy decision making schema for merging simulation to handle uncertain dynamic environments. Haoyu [14] has presented an intention-aware online POMDP planning approach for autonomous driving amid many pedestrians. W Liu [15] has presented a situation-aware decision making POMDP model and solved it in an online manner.

One key difficulty is to incorporate driver intention and behaviors into the current models. A number of modelling approaches has been studied in literatures for prediction of driver intentions, for instance, M Bennewitz [16] and Vasquez [17] has proposed motion learning approach based on hidden Markov models. Fulgenzi [18] and Joshua [19] has used a Gauss-process pre-learned typical patterns for estimation of moving obstacles, however these process requires extensive training from large number of data. Our current method to predict the driver intention is through Bayesian Inference on the observed vehicles states.

III. Overview

A. Preliminary POMDP model

The general decision process is formulated as a POMDP model and a POMDP is formulized as a tuple $\{S, A, Z, T, O, R\}$, where S is the state space, A is a set of actions and Z denotes the observation space. The transition function $T(s', s, a) = Pr(s|s', a)$: S×A×S models the probability of transiting to state $s \in S$ when the agent takes an action $a \in A$ at state $s' \in S$. The

observation function $O(z,s,a) = Pr\ (z|s,a)$: $S\times A\times Z$, similarly, gives the probability of observing z ∈ Z when action a ∈ A is applied and the resulting state is $s \in S$. The reward function $R\ (s,a)$: $S\times A$ is the reward obtained by performing action $a \in A$ in state $s \in S$. The solution to the POMDP thereby is an optimal policy π^* that maximize the expected accumulated reward $\sum_{t=0}^{H} \gamma^t R(s_t,a)$, where γ ∈ [0, 1) is a discount factor, s_t denotes the agent's state and action at time t. The true state, however, is not fully observable to the agent, thus the agent maintains a belief state b ∈ B, i.e. a probability distribution over S, instead.

B. A POMDP model for lane change decision making

The solution a POMDP problem is to choose an optimal policy $\pi^*$for our vehicle and the policy is a deterministic mapping π^*: $s_t\times z_t \rightarrow a_t$, to generate an action from current state and observation, and the decision process is to choose such policy to maximize the reward. Be more specifically, the nomenclature associated with the lane change model is defined as:

1) State space S: the state for the subject vehicle i can be described as $s_i = (x,y,\theta,v)$, the state for obstacle vehicles can be described as $s_k = (x,y,\theta,v,I)$, and I represents the intent of obstacle vehicles in the system which will be inferred through observation. So the joint state in the current system can be formulated as $s = (s_0, s_1, s_2, s_3 \dots s_k)$
2) Action Space A: the action space for each vehicle can be defined as $A = (acc, dec, normal)$, and each action will be applied with turning angles.
3) Observation Space Z: Similar to the joint state $s \in S$, the joint observation $z \in Z$ consists of the following elements $Z = (z_1, z_2, z_3 \dots z_k)$. To properly update the obstacle vehicle's motion intention belief, each vehicle's position and velocity are observed in the observation function.
4) Transition Model $T\ (s,a,s')$: The transition function describes the stochastic system dynamics driven by both the action applied to subject vehicle and obstacle vehicles. Specifically,

$$\Pr(s'|s,a) = \Pr(s_0'|s_0,a)\prod_{i=1}^{K}\Pr\ (s_i'|s_i,a_i)\Pr\ (a_i|I_i,z_i)$$

where is the action a ∈ A applied to the all vehicles in the system and s, s' is are the old and new joint state respectively, s_0 represents the subject vehicle, thus the transition function following simple vehicle dynamic can be described as.

$$\begin{bmatrix} x' \\ y' \\ v' \\ \theta' \end{bmatrix} = \begin{bmatrix} x \\ y \\ v \\ \theta \end{bmatrix} + \begin{bmatrix} (v+a\Delta t)\cos\ (\theta+\Delta\theta) \\ (v+a\Delta t)\sin\ (\theta+\Delta\theta) \\ a\Delta t \\ \Delta\theta \end{bmatrix}$$

As for the subject vehicle, the action for subject vehicle is available and no inference on action is required. However, for the obstacle vehicles, since the action a is not directly observable, thus the action a has to be inferred through observation of vehicle states.

5) Observation Model $O\ (z,s,a)$

The observation function used here is through direct measurement of the velocity of obstacle vehicles and the observed the velocity information can be used to perform

inference on the intention of other vehicles. To be more specifically, the inference expression can be described ass

$$\Pr(a|I_i, z_i) = \Pr(a_i|I_i)\Pr(I_i|z_i)\Pr(z_i)$$

and $\Pr(z_i)$ can be described as the Gauss Distribution to account for the possible noise $\Pr(z_i) = N(v_i|v_{ref}, v_{std})$, and the mathematic expression to obtain the intention from single vehicle is through

$$\Pr(I_l|z_l) = \iint_{i=0, i\neq l}^{N} \Pr(I_1, I_2, I_3, \dots, I_k|z_1, z_2, z_3, \dots, z_k)\, dz_i dI_i$$

The probability of intentions is obtained from the integration of the joint probability distribution from all vehicles involved.

6) Rewards

The policy selected in the POMDP model is to maximize the total rewards over a given horizon:

$$\pi^* = \underset{\pi}{\operatorname{argmax}} \sum_{t=0}^{H} \lambda^t \int R(s_t)p(s_t)dt$$

IV. Multi-agent policy selection algorithm

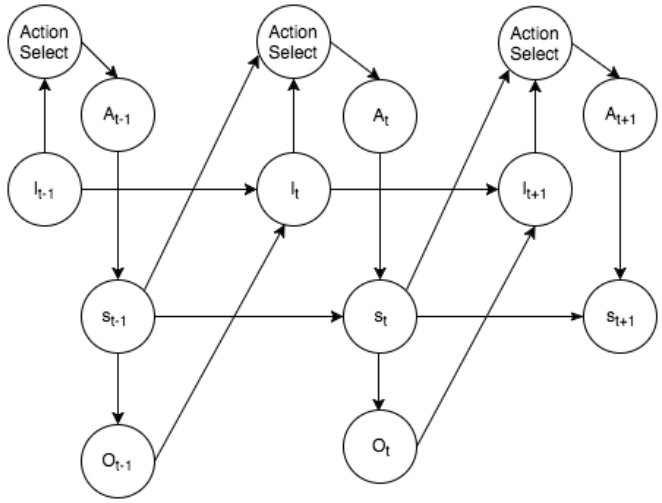

Figure 2 Policy Selection

The proposed multi-agent policy selection algorithm is illustrated in the above Figure 2 Policy Selection, in which the symbols of I, A, S stands for the current intentions, actions, vehicles states at particular time t. And each action is selected through the action select algorithm which incorporate the route planning algorithm and route simulation process with current vehicle intention to evaluate the different routes, then an observation node is also considered to make observation at the current state which will be used to update the vehicle intentions at the next time stamp. A more detailed description of this algorithm is described below:

Algorithm 1: policy selection procedure

Input:

- previous intention I_{t-1}
- current state s_t
- if lane change required

if (isChangeRequired) **then**
    I_t = updatebelief(I_{t-1}) // particle filter update
    scoreset = **Ø** // to save each path, score pair
    paths = generatepaths(s_t) // path generation module to generate multiple path
    foreach path **in** paths **do** // simulate all possible path with estimated belief
        score = simulate (s_t, I_t path) // calculate score for each path
        scoreset = scoreset U {path, score}
    selectbest {path, score}
    makeobservation() // used for later belief update module
    return pathtoactions()
else
    Defaultdriveforward()

Table 1 Multi-agent algorithm

V. Path planning

The path planning searches for a minimum-cost drivable continuous path which is potentially safe, fast and smooth to drive. The cost function for the path planning

$$C(\rho) = min \int_0^H (C^2(s, u(t)) + \lambda \Delta\theta(t)^2) dt$$

Subject to s(0) = s_{start}, and $S_H = S_{goal}$

Here, $\Delta\theta(t)$ is the turning angle for the vehicle executed in each action taken and the desired path should prefer small turns when making turns and λ is the coefficients associated with the turning angles. The objective is to find the series of plausible control input u_t to reach the goal while avoid any static obstacles and dynamic obstacles. To solve the minimal path problem, the graph-based hybrid A* algorithm [4] is employed and proven to be useful in finding the practical paths at any instantaneous time t, a detailed description for this algorithm is in Table 3, and a potential search outcome is listed in Figure 3 for illustration with red color car for obstacle car and blue car for subject car.

Algorithm 2: generatepaths()

Input:

- start state s_t
- current map m_t

legalintentions = generateLegalIntentions
goals = selectgoals(legalintentions)
pathset = ∅
foreach goal **in** goals
    path = findpath(s_t, m_t, goal)
    pathset = pathset U path
return pathset

Table 2 generate paths algorithm

Algorithm 3: findpath

Input:

- start state s_t , goal
- cost for each action c
- current map m_t

```
1   s_t = startstate, c = 0 //initialized cost
2   h = heuristicfunction(s_t)
3   open = ∅ ∪ {s_t, c, c + h} ,  close = ∅
4   while open not empty do
5       s = remove minimal state from open
6       if s is goal then return
7       else
8           if s not in close then
9               insert s into close
10              successors = generateSuccessors(s_t, m)
11              foreach successor in successors do
12                  h = evaluate(successor)
13                  c' = s.c + cost
14                open = open ∪ {sucessor, c', c' + h}
```

Table 3 find path algorithm

In the above algorithm, *geneartepaths* algorithm first check all the legal intentions the subject vehicle could have, for instance, if vehicle is already in the left lane and can't go further left, so left turn intention has to be eliminated from the intention lists, and evaluate function is selected to take into account both the distance to the goal, and also the turning angles for each vehicle might execute, which is used to ensure the continuity.

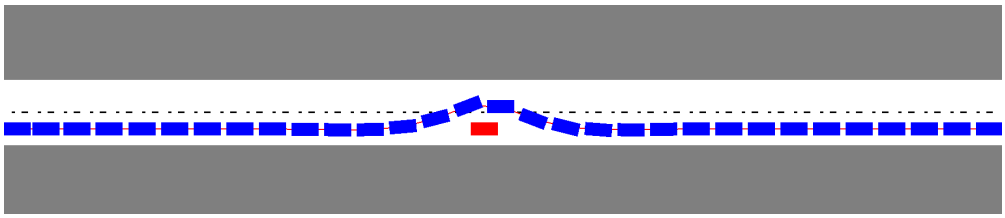

Figure 3 Path Planning Example

VI. Belief Tracking

In the current research works, the focus is concentrated on how to handle the discrete uncertainty of the driver reaction to subject car's lane change intention after giving signal. Thus, the uncertainties from the above-mentioned state can be segregated from the overall state and the uncertainty to be discussed is restricted only to the driver intention of yield or not yield, namely, $I_i = (0, 1)$ with 0 for *yield* and 1 for *not yield*. The other issue should be addressed here is the possible tendency of one vehicle on the other vehicles. Suppose that the vehicles driving on the highway, so the tendency is that one vehicles will be in close distance with others, so the intention for each vehicle is

dependent on the others. Therefore, the general uncertainty for the whole traffic flow is $I = (I_0, I_1, I_2, \dots, I_k)$ and the belief updating function can be described

$$b(I_{t+1}) = b(I_{t+1}|I_t, a)b(a|I_t, z_t)b(z_t|I_t)b(I_t)$$

Here, $b(I_{t+1})$ is the belief probability vector for all obstacle vehicles at time t+1, a is the action vector for each individual vehicle involved and z_t is the observation obtained at time t. It is assumed the action exerted on the vehicle is deterministic once the action is inferred from current belief and observation, which means the term $b(I_{t+1}|I_t, a) = 1$, and the probability model of intention estimation is $b(z_t|I_t) = N(v|v_{vref}, I)$, in which the distribution conforms to the Gauss Noise distribution with given intention and reference velocity. To obtain the current belief, the belief state is updated recursively at each time stamp when decision making is required and particle filter are used to keep tracking the belief distribution update according to the latest observation.

Algorithm 3: beliefupdate

Input:

- start state s_t
- current intention particle set belief of intentions I_t
- current map

```
1  if size of belief is 1 then // for cases with only one particle in belief
2        InitializebeliefUniformly()
3  tempbelief = ∅
4  foreach particle in belief do
5        weight = 1
6        foreach agent in current map do
7              observedv = agent.velocity
8              currentintention = particle[agent]
9              weight = weight*GuassProbl(observ, currentintention) // joint distribution
10       tempbelief = tempbelief U {particle, weight}
11 newbelief = ∅
12 foreach particle in tempbelief do
13       particle = samplingbelief(tempblief) // to sample particles from tempbelief
14       newbelief =  newbelief U particle
15 return newbeliefset
```

Table 4 belief update algorithm

VII. Implementation and Results

The above algorithms are implemented in the simulated environment with car module, model module, path search module, inference module and control module to simulate the vehicle driving in a mixed environment, more specifically, the subject vehicles are surrounded by number of vehicles in congested situations as configured below, vehicle 0 is the vehicle behind the subject vehicle, and vehicle 1 is before, the other vehicle 2,3,4 occupies the right lane and block any right turn maneuvers.

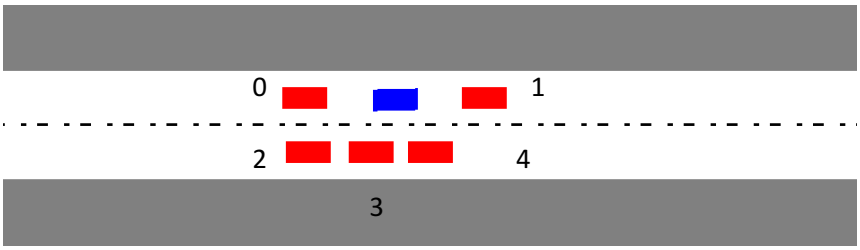

Figure 4 Experiment Setting

In the policy selection algorithms, it is specified that several paths are generated through the path generation module. Here, it is proposed to generate 10 paths when making lane change is required with the goals are distributed along the potential goal areas. For instance, if the current coordinate of the subject vehicle is located at position (x, y), then making turn right means the possible goals will start with (x' = x + grid size, y' = y + grid size), the longitudinal goal x' can also be increased to a series of goals 1, 2, 3, …, 10. The possible lane change trajectories are illustrated in the below figure marked in the triangle area and the candidate lane change trajectory will be evaluated by simulation module to check feasibility.

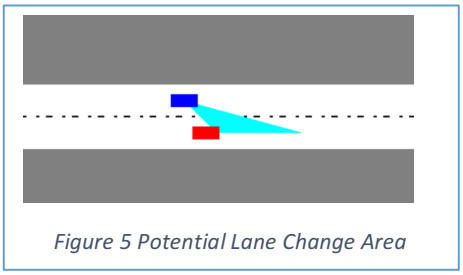

Figure 5 Potential Lane Change Area

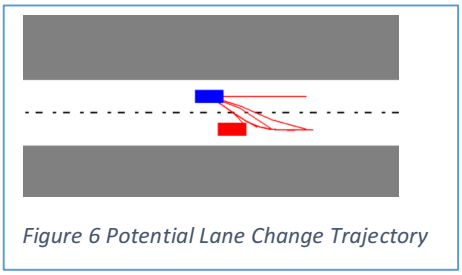

Figure 6 Potential Lane Change Trajectory

In the simulation module, the subject vehicles will simulate the driving forward process through driving simulation in the proposed trajectories while all the other obstacle driving behaviors are also simulated with the estimated driver intention obtained in the belief update module. Thus, if the driver intention is estimated as *yield*, it is assumed that this particular vehicle will yield to the subject vehicle by slowing down it speed.

In the belief update module, the size of particles is set to be 600 and the number of vehicles is set to be 5, and the particles are the combined intention state such as particle p = {0, 0, 1, 1, 0} with 0 for *yield* and 1 for *not yield*, thus the total number of distinct intention combination is 2^5 = 32 and initially, these 32 intention states are uniformly distributed in the 600 particles and particles are updated during each observation with observation function $b(z_t|I_t) = N(v|v_{ref}, I)$ and the v_{ref} is the reference velocity through the observation with added possible noise. Once the observation is made, the weight for each particle is updated and all the particles will be resampled according to the weight of each particle to form a new particle set.

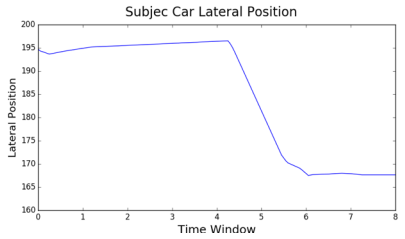

Figure 7 Subject Car Lateral Position

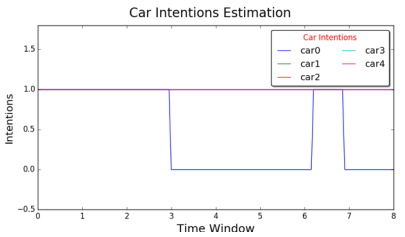

Figure 8 Obstacle Car Intention Estimation

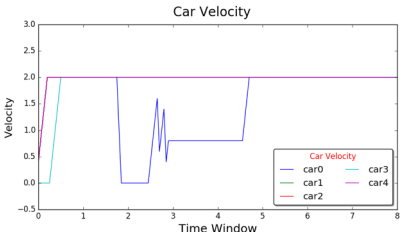

Figure 9 Obstacle Car Velocity Profile

Table 5 Non-Cooperative Driver

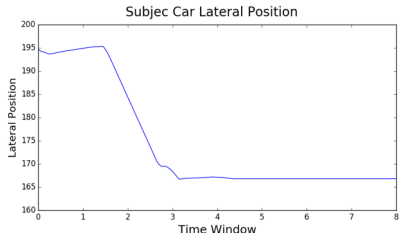

Figure 10 Subject Car Lateral Position

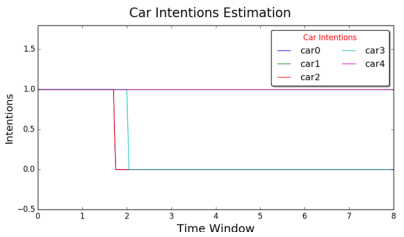

Figure 11 Obstacle Car Intention Estimation

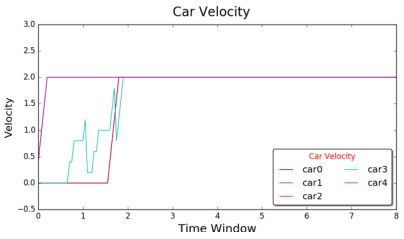

Figure 12 Obstacle Car Velocity Profile

Table 6 Cooperative Driver

In the current experiment settings, two cases are tested for comparison, namely:

- Case I: non-cooperative driver tends to ignore the turning signal as captured in table 5
- Case II: cooperative driver tends to yield when observing signal as captured in table 6

In these two cases, vehicle 0 is the vehicle behind subject vehicle and will is set to be always yield for safety reason and maintain a safety distance between himself and the subject vehicles. Vehicle 1 is the one in front of subject vehicle and subject vehicle will also maintain safety distance with him. The behavior and intentions for remaining vehicles 2,3,4 are unpredictable and thus two test cases are assigned for them with Case I and Case II.

As it can be seen in Case I, the velocity profile in Figure 9 shows only the vehicle 0 slows down and also accelerates due to the slowdown and acceleration of subject vehicle. The lane change occurs between time window 4 and 5 in Figure 7, during which the vehicle intention estimated as *yield* while others estimated as *not yield* in Figure 8. In case 2, the velocity profile in Figure 12 shows vehicle 2 and vehicle 3 slows down when capturing the turning signal, and the corresponding intentions are estimated as *yield* in Figure 11, and the lane change occurs during this period in Figure 10.

VIII. Conclusion

The POMDP is a powerful mathematic tool to solve decision problem with uncertainty involved, however, the exact solution to the POMDP problem is intractable and thus approximation approach is utilized to simply the problem formulation based on the reasonable assumptions. The problem studied in this research is to propose a search based and particle filter based approach to simulate the policy selection process while providing justifiable estimations for the driver intentions. The numerical outcome shows the predication is quite accurate in capturing the obstacle vehicle's driving intentions to make good decision. The future work will further more in how to handle the uncertainty arising in the localization and perception process.

References

[1] Thrun, Sebastian, et al. "Stanley: The robot that won the DARPA Grand Challenge." Journal of field Robotics 23.9 (2006): 661-692.

[2] Urmson, Chris, et al. "Autonomous driving in urban environments: Boss and the urban challenge." Journal of Field Robotics 25.8 (2008): 425-466.

[3] Sondik, Edward J. "The Optimal Control of Partially Observable Markov Decision Processes." PhD the sis, Stanford University (1971).

[4] Dolgov, Dmitri, et al. "Practical search techniques in path planning for autonomous driving." Ann Arbor 1001 (2008): 48105.

[5] Paden, Brian, et al. "A Survey of Motion Planning and Control Techniques for Self-driving Urban Vehicles." arXiv preprint arXiv:1604.07446 (2016).

[6] Leonard, John, et al. "A perception-driven autonomous urban vehicle." Journal of Field Robotics 25.10 (2008): 727-774.

[7] He, Ruijie, Emma Brunskill, and Nicholas Roy. "Efficient planning under uncertainty with macro-actions." Journal of Artificial Intelligence Research 40 (2011): 523-570.

[8] Ross, Stéphane, et al. "Online planning algorithms for POMDPs." Journal of Artificial Intelligence Research 32 (2008): 663-704.

[9] Silver, David, and Joel Veness. "Monte-Carlo planning in large POMDPs." Advances in neural information processing systems. 2010.

[10] Somani, Adhiraj, et al. "DESPOT: Online POMDP planning with regularization." Advances in neural information processing systems. 2013.

[11] Wei, Junqing, John M. Dolan, and Bakhtiar Litkouhi. "A prediction-and cost function-based algorithm for robust autonomous freeway driving." Intelligent Vehicles Symposium (IV), 2010 IEEE. IEEE, 2010.

[12] Ulbrich, Simon, and Markus Maurer. "Probabilistic online POMDP decision making for lane changes in fully automated driving." 16th International IEEE Conference on Intelligent Transportation Systems (ITSC 2013). IEEE, 2013.

[13] Cunningham, Alexander G., et al. "MPDM: Multipolicy decision-making in dynamic, uncertain environments for autonomous driving." 2015 IEEE International Conference on Robotics and Automation (ICRA). IEEE, 2015.

[14] Bai, Haoyu, et al. "Intention-aware online POMDP planning for autonomous driving in a crowd." 2015 IEEE International Conference on Robotics and Automation (ICRA). IEEE, 2015.

[15] Liu, Wei, et al. "Situation-aware decision making for autonomous driving on urban road using online POMDP." 2015 IEEE Intelligent Vehicles Symposium (IV). IEEE, 2015.

[16] Bennewitz, Maren, et al. "Learning motion patterns of people for compliant robot motion." The International Journal of Robotics Research 24.1 (2005): 31-48.

[17] Vasquez, Dizan, Thierry Fraichard, and Christian Laugier. "Growing hidden markov models: An incremental tool for learning and predicting human and vehicle motion." The International Journal of Robotics Research (2009).

[18] Fulgenzi, Chiara, et al. "Probabilistic navigation in dynamic environment using rapidly-exploring random trees and gaussian processes." 2008 IEEE/RSJ International Conference on Intelligent Robots and Systems. IEEE, 2008.
[19] Joseph, Joshua, et al. "A Bayesian nonparametric approach to modeling motion patterns." Autonomous Robots 31.4 (2011): 383-400.