Intention-Aware Online POMDP Planning for Autonomous Driving in a Crowd

Haoyu Bai Shaojun Cai Nan Ye David Hsu Wee Sun Lee

***Abstract*— This paper presents an intention-aware online planning approach for autonomous driving amid many pedestrians. To drive near pedestrians safely, efficiently, and smoothly, autonomous vehicles must estimate unknown pedestrian intentions and hedge against the uncertainty in intention estimates in order to choose actions that are effective and robust. A key feature of our approach is to use the partially observable Markov decision process (POMDP) for systematic, robust decision making under uncertainty. Although there are concerns about the potentially high computational complexity of POMDP planning, experiments show that our POMDP-based planner runs in near real time, at 3 Hz, on a robot golf cart in a complex, dynamic environment. This indicates that POMDP planning is improving fast in computational efficiency and becoming increasingly practical as a tool for robot planning under uncertainty.**

I. Introduction

Imperfect robot control, sensor noise, and unexpected environment changes pose significant challenges to efficient and reliable robotic systems. The partially observable Markov decision process (POMDP) [16] is a powerful mathematical model that captures such uncertainties for robust decision making and planning. However, the use of POMDPs for robot planning under uncertainty is not widespread. Many are concerned about its reputedly high computational cost. In this work, we apply DESPOT [17], a state-of-the-art approximate online POMDP planning algorithm, to autonomous driving in a complex, dynamic environment (Fig. 1).

There are many successful autonomous vehicles today (see, *e.g.*, [18], [20]), but it is still uncommon to see autonomous vehicles driving among many pedestrians. To drive near pedestrians safely, efficiently, and smoothly, autonomous vehicles must estimate unknown pedestrian *intentions* and hedge against the *uncertainty* in intention estimates in order to choose actions that are effective and robust. They must also reason about the long-term effects of immediate actions. POMDP planning provides a systematic approach to achieve these objectives. Simple reactive control methods are inadequate here. They choose actions based on sensor data on the *current* system state. They do not hedge against sensing uncertainty and do not consider an action's long-term effects. This often results in overly aggressive or conservative actions, or attractive short-term actions with undesirable longer-term consequences. See Section III for an example.

The authors are with the Department of Computer Science, National University of Singapore, Singaore 117417, Singapore.

This work is supported in part by the SMART IRG grant R-252-000-527-592, the MoE grant MOE2010-T2-2-071, and the AOARD grant FA2386-12-1-4031. The views and conclusions contained herein are those of the authors and should not be interpreted as necessarily representing the official policies or endorsements, either expressed or implied, of the Air Force Research Laboratory or the U.S. Government.

Fig. 1. The autonomous golf cart drives in a crowded, dynamic environment.

We build a POMDP for autonomous driving among many pedestrians. We model the intention of a pedestrian as a subgoal location [8] and model pedestrian behavior conditioned on the intention as a hidden variable. The model captures uncertainty in pedestrian goal estimation as well as uncertainties in vehicle control and sensing.

To achieve real-time performance with limited computational resources, we adopt a two-level hierarchical planning approach and exploit POMDP planning only in the critical part of the system to hedge against the uncertainty in predicting pedestrian behaviors. At the high level, we apply the hybrid A* algorithm [18] to search for a path through less crowded regions, based on a simplified predictive model of pedestrian motions. At the low level, we perform online POMDP planning in near real time to control the speed of the vehicle along the planned path. Online POMDP planning interleaves planning and plan execution. It maintains a *belief*, *i.e.*, a probability distribution over system states, as a representation of uncertainty. At each time step, the algorithm performs a lookahead search for the best action at the current belief, and the vehicle executes the chosen action immediately. The algorithm then updates the belief by incorporating new sensor data received. We replan at both levels in each time step in order to handle dynamic changes in the environment.

We implemented our approach on an autonomous golf cart and tested it extensively in a plaza on our university campus. Experiments show that the vehicle is capable of driving safely and smoothly in a crowded unstructured environment. To our knowledge, POMDP planning has not been applied to complex environments of this scale before. One main contribution of this work is to demonstrate the latest progress in online POMDP planning, in particular, the DESPOT algorithm [17], as a tool for robot planing under uncertainty.

POMDPs have been used before for intention-aware motion planning [1]. The earlier work builds a discrete-state POMDP model and solves the model *offline* for a policy. By exploiting online POMDP planning, specifically, the DESPOT algorithm, our work makes several advances. First,

online planning allows us to handle dynamic environment changes, *e.g.*, sudden appearance of new obstacles, while the earlier work assumes a fixed environment. Second, the DESPOT algorithm allows continuous state variables in POMDP models, which enable us to model vehicle and pedestrian dynamics more accurately and conveniently. Third, due to the limitation on computational efficiency, the earlier algorithm computes the action for each pedestrian in isolation and then chooses the most conservative one. The new DESPOT algorithm scales much better and allows for a more realistic model that treats the pedestrians jointly and captures their interactions.

II. Related Work

There are two main approaches to POMDP planning: offline policy computation and online search. In offline planning, we compute beforehand a policy contingent upon all possible future scenarios, and the robot executes the computed policy based on the sensor data received (see, *e.g.*, [14]). It is difficult to scale up offline planning to very large POMDPs due to the exponential number of scenarios. It is also difficult to handle environment changes, as all such changes must be anticipated and modeled in advance. In contrast, online planning interleaves planning and plan execution (see, *e.g.*, [6], [12], [15], [17]). The robot searches for the best action for the current belief only, executes the action, and updates the belief. The process then repeats at the new belief. These online algorithms apply several algorithmic techniques for computational efficiency, including heuristic search, branch-and-bound pruning, and Monte Carlo sampling [12]. POMCP [15] and DESPOT [17] are among the fastest online POMDP algorithms available today. Both can handle POMDPs with very large number of states, but DESPOT has a much stronger worst-case performance bound. In this work, we build a two-level hierarchical online planning approach based on DESPOT for real-time control of a robot vehicle in a crowded, dynamic environment.

Today there are many successful systems for autonomous driving (see, *e.g.*, [11], [20]). It is beyond the scope of this paper to survey this new and exciting field of robotics. So far, these systems have paid relatively little attention to close interaction with pedestrians, especially, in unstructured environments. Such technology is useful for providing on-demand, personalized transportation in densely populated urban localities, such as university campuses, business parks, downtown shopping districts, ..., and for handling the first and the last mile of public transportation systems [24].

One main difficulty of autonomous driving among pedestrians is to incorporate pedestrian intentions and behaviors into decision making. There are two related, but orthogonal issues here. One is pedestrian intention and behavior modeling. There are various modeling approaches based on, *e.g.*, linear dynamic systems with Gaussian noise, hidden Markov models (HMMs) [2], [22], Gaussian processes (GPs) [5]. This work focuses on the orthogonal issue, decision making, which determines the vehicle action given a predictive pedestrian behavior model. The simplest approach is reactive control (see, *e.g.*, [4], [13]). It does not require a sophisticated predictive model and basically ignores prediction uncertainty during decision making. It is fast, but often results in sub-optimal decisions over the long term. A more sophisticated approach is to compute an open-loop plan, *i.e.*, a path, using a predictive model and then execute the plan with a feedback controller (see, *e.g.*, [9], [21], [22]). POMDP planning goes one step further: it reasons about uncertainties systematically and computes a *close-loop* plan that handles all future contingencies. In addition to pedestrian behavior uncertainty, the POMDP approach can also incorporate vehicle control and sensing uncertainties into decision making systematically. Conceptually, POMDP planning is analogous to computing an optimal controller for a linear quadratic Gaussian (LQG) system, but POMDP planning is more general and does not assume linear dynamics or Gaussian noise distribution. The advantages of POMDP planning comes at the cost of higher computational complexity. One purpose of this work is to argue that POMDP planning is improving fast in computational efficiency and is becoming more practical as a tool for robot planning under uncertainty. Of course, it does not replace other approaches. The trade-off between the optimality of decision making and computational efficiency remains, but the gap is narrowing.

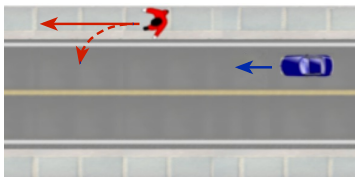


Fig. 2. The robot vehicle approaches a pedestrian on the sidewalk.

III. Overview

To drive near pedestrians effectively, a robot vehicle must infer pedestrian intentions and hedge against the uncertainty in intention estimates for robust decision making. We may capture this uncertainty in a belief, *i.e.*, a probability distribution over possible intentions. This is, however, not sufficient. We must also reason about the long-term effects of vehicle actions and sensor observations. Consider, for example, the vehicle approaches a pedestrian walking on the sidewalk (Fig. 2). If the pedestrian stays on the sidewalk, the vehicle may accelerate and pass him. If the pedestrian crosses the road, the vehicle may slow down to yield. Let $(p, 1-p)$ denote the belief over these two intentions. Assume, for the sake of argument, that the initial belief is $(0.51, 0.49)$. Should the vehicle then accelerate and attempt to pass the pedestrian? Based on the maximum likelihood consideration, it should. Unfortunately, soon after, the vehicle receives sensor data indicating that the pedestrian is turning and may step off the sidewalk. The belief gets updated to $(0.35, 0.65)$, and the vehicle brakes to slow down. However, the earlier acceleration results in high speed, and the vehicle fails to stop in time to prevent an accident. The reason behind this failure is the greedy nature of maximum-likelihood decision making at the very beginning. It does not reason about the long-term effects of robot actions. It does not consider the effects of future sensor observations on the belief, *i.e.*, the *value of information* [7]. This limitation is common to most greedy decision-making approaches, including, in particular, reactive control. In contrast, POMDP planning performs such

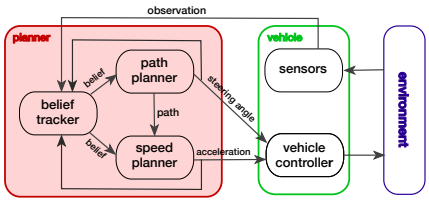


Fig. 3. Two-level POMDP-based planning for autonomous driving.

reasoning in a principled manner.

The benefit comes at a cost. Online POMDP planning searches a *belief tree* rooted at the current belief (Fig. 4). The tree branches on all actions and observations. In the worst case, it has an exponential size $\mathcal{O}(|A|^H|Z|^H)$, where $|A|$ and $|Z|$ denote the sizes of action and observation spaces, respectively, and H is the maximum tree height. For autonomous driving, both the action space and the observation space can be large.

For computational efficiency, we decompose the planning task into two levels. The high level controls vehicle steering. It builds a cost map, which encodes distributions of static obstacles and pedestrians, and plans a path that minimizes the expected collision cost. The low level controls vehicle speed along the planned path. It applies online POMDP planning to handle uncertainties in pedestrian intention estimates, imperfect vehicle control, *etc.*. This decomposition substantially reduces the action space for POMDP planning, while the DESPOT algorithm for POMDP planning is capable of handling large observation spaces. Additionally, the decomposition simplifies system implementation, as it enables independent tests of separate system components.

Our overall planner consists of three modules: a POMDP speed planner (Section IV), a path planner (Section V), and a belief tracker (Section VI). See Fig. 3. The belief tracker maintains a belief over system states. At each time step, it performs Bayesian updates to incorporate new information from vehicle actions and sensor data into the belief. The path planner then plans a minimum-cost path for the current belief and outputs the vehicle steering angle for the current step. Given the path and the current belief, the POMDP speed planner computes a conditional plan over the next H steps. However, it outputs only the first step of the plan and sends the desired acceleration to the vehicle controller for execution. To handle environment changes, we replan at both the high and the low levels at each time step.

IV. Intention-Aware Online POMDP Planning

A. POMDP Preliminaries

A POMDP models a system taking a sequence of actions under uncertainty to maximize its total reward. Formally, a POMDP is a tuple $(S, A, Z, T, O, R, \gamma)$, where S, A, and Z denote the system's state space, action space, and observation space, respectively. At each time step, the system takes an action $a \in A$ to move from a state $s \in S$ to $s' \in S$ and then receives an observation $z \in Z$. A conditional probability function $T(s, a, s') = p(s'|s, a)$ models the state-transition dynamics. It specifies the probability distribution of the new system state when the system takes action a in state s. Similarly, the conditional probability function $O(s', a, z) = p(z|s', a)$ models noisy sensor observations. The system gets an immediate reward $R(s, a)$ for taking action a in state s.

In a partially observable system, the system state is not completely known. We maintain a belief over possible states. Let b_{t-1} be the belief at time $t-1$. If the system takes action a_t and receives observation z_t at time t, we then apply the Bayes' rule to obtain the new belief b_t:

$$b_t(s') = \eta O(s', a_t, z_t) \sum_{s \in S} T(s, a_t, s') b_{t-1}(s), \qquad (1)$$

where η is a normalizing constant. This defines a belief update function τ such that $b_t = \tau(b_0, a_1, z_1, a_2, z_2, \ldots, a_t, z_t)$, where b_0 is an initial belief.

A key concept in POMDP planning is a *policy*, a mapping π that specifies the action $a = \pi(b)$ at belief b. In online POMDP planning, we seek a policy that maximizes the *value*, *i.e.*, the expected total reward at the *current belief* b:

$$V_\pi(b) = \mathrm{E}\Big(\sum_{t=0}^{\infty} \gamma^t R\big(s_t, \pi(b_t)\big) \;\Big|\; b_0 = b\Big), \qquad (2)$$

where $\gamma \in (0, 1)$ is a discount factor, which places preferences for immediate rewards over future ones.

B. A POMDP Model for Pedestrian Avoidance

Our POMDP planner controls the vehicle acceleration along a given path.

1) Vehicle Modeling: The vehicle state contains the position (x, y), orientation θ, and instantaneous speed v. Our POMDP model has three discretized actions that control the acceleration: Accelerate, Maintain, and Decelerate. The separate path planner controls the steering angle. Given the chosen acceleration and steering angle, we integrate the system forward for a fixed time duration Δt and add a small amount of noise to form a predictive model of vehicle dynamics [19]. This simple dynamic model satisfies the nonholonomic constraint of a car.

2) Pedestrian Modeling: The state of pedestrian i contains the position (x_i, y_i), goal g_i, and instantaneous speed v_i towards the goal. The goal captures the pedestrian's intention. Typically, a goal g represents a specific location (x_g, y_g) in the environment. We assume that at each time step, pedestrian i walks a distance $v_i \Delta t$ in the direction of the goal, with a small amount of Gaussian noise in the direction. The speed v_i remains constant within a single planning cycle, but may change from one cycle to the next (see Section VI). There is a special goal, which represents the pedestrian standing still, with only small random movement. In summary, these lead to a probabilistic dynamic process $p(x_i', y_i'|x_i, y_i, g_i, v_i)$ describing pedestrians' movement from the current position (x_i, y_i) to the next position (x_i', y_i') given the goal g_i and speed v_i. Our model is currently very basic, but we plan to build richer and more accurate probabilistic dynamic models through learning [2], [22] in the future.

The pedestrian model does not consider that the intention of a pedestrian may change. However this can be handled by online POMDP planning through belief update and replanning. See Section VII-B for experimental results.

3) Sensor Modeling: An observation is a vector consisting of discretized values of vehicle position, vehicle speed, and positions of all pedestrians. We do not model observation noises for these variables, as they are small and do not

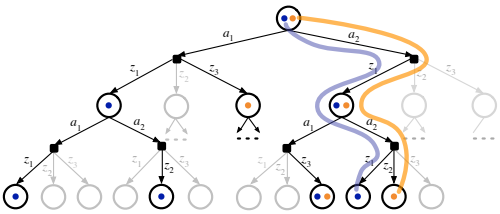


Fig. 4. A belief tree of height $H = 2$ (gray) and a corresponding DESPOT tree (black) obtained with 2 sampled scenarios, shown in blue and orange. The blue and orange curves indicate the execution paths of a same policy under the two scenarios.

affect decision making substantially. However, there is no direct observation on pedestrian intentions, and they are the key partially observable variables in our model. We must infer the pedestrian intentions from the observations received over time, and also hedge against the estimation uncertainty during decision making.

4) Reward Modeling: The reward function prescribes the desirable driving behavior: safe, efficient, and smooth.

- If any pedestrian gets within a small distance D_{col} of the vehicle, there is a very large penalty R_{col}. This is to ensure safety.
- If the vehicle reaches within a small distance D_{goal} of the destination, there is a large reward R_{goal}. This is to encourage the vehicle to reach the destination.
- There is a penalty $R_{\text{speed}} = (v - v_{\max})/v_{\max}$, where v and $v_{\max}$ are the current and the maximum vehicle speed, respectively. This encourages driving at a higher speed when it is safe to do so.
- Finally, there is is a small penalty R_{accel} for the actions ACCELERATE and DECELERATE. This penalizes excessive speed changes and encourages smooth driving.

C. DESPOT

To choose the best vehicle action at the current belief b_0, we perform online POMDP planning. We search a belief tree with root at b_0 and maximum height H (Fig. 4). Each tree node represents a belief, and each edge represents an action-observation pair. If a child node b' is connected to its parent b by an edge (a, z), then $b' = \tau(b, a, z)$ according to (1). To search a belief tree, we perform a post-order traversal. At each leaf node, we simulate a *default policy* to obtain a lower bound on its value. At each internal node, we apply Bellman's principle of optimality to choose a best action:

$$V(b) = \max_{a \in A} \Big\{ \sum_{s \in S} b(s) R(s, a) + \gamma \sum_{z \in Z} p(z|b, a) V\big(\tau(b, a, z)\big) \Big\}, \tag{3}$$

which recursively computes the maximum value of action branches and the average value of observation branches. The result is an approximately optimal policy π for the current belief b_0. The vehicle then executes the first action of the policy, $\pi(b_0)$. The idea is simple. However, the tree grows on the order of $\mathcal{O}(|A|^H |Z|^H)$. It is thus impractical to construct or search the full tree when the action space or the observation space is large.

To overcome this challenge, we use DESPOT, whose key strengths include handling large observation spaces. For completeness, we provide a brief summary of DESPOT's main idea here. See [17] for details. Intuitively, we do not need to examine all observation branches to identify an approximately optimal action, because the second sum in (3) computes an *average* value over observation branches. A sampled subset of observations branches may be sufficient to *estimate* this average. The key idea of DESPOT is to summarize the execution of all policies under K sampled *scenarios*. Under each scenario, a policy traces out a path in the belief tree (Fig. 4). It corresponds to a particular sequence of action chosen by the policy and observation received. We now define a subtree, called *DEterminized Sparse Partially Observable Tree* (DESPOT), which contains only the belief-tree nodes and edges traversed by *all* policy under the sampled scenarios (Fig. 4). A DESPOT tree is a sparsely sampled belief tree. While the original belief tree contains $\mathcal{O}(|A|^H |Z|^H)$ nodes, the DESPOT tree contains only $\mathcal{O}(|A|^H K)$ nodes, leading to dramatic improvement in computational efficiency for moderate K values. Equally importantly, we can prove that a small DESPOT tree is sufficient to produce an approximately optimal policy with bounded regret, provided that an optimal policy admits a compact representation [17].

More precisely, we define a DESPOT tree with root b_0 using a *deterministic simulative model*. Here, a scenario is defined as a random sequence $\boldsymbol{\phi} = (s_0, \phi_1, \phi_2, \ldots)$, in which the start state s_0 is sampled according to the belief b_0 and each ϕ_i is a real number sampled independently and uniformly from $[0, 1)$. The deterministic simulative model is a function $g\colon S \times A \times \mathbb{R} \mapsto S \times Z$ such that if a random number ϕ is distributed uniformly over $[0, 1)$, then $(s', z') = g(s, a, \phi)$ is distributed according to $p(s', z'|s, a) = T(s, a, s')O(s', a, z')$. If we simulate a policy π using this model under the scenario $\boldsymbol{\phi}$, the simulation generates a trajectory $(s_0, a_1, s_1, z_1, a_2, s_2, z_2, \ldots)$ such that $(s_t, z_t) = g(s_{t-1}, a_t, \phi_t)$, $a_t = \pi(b_t)$, and $b_t = \tau(b_0, a_1, z_1, \ldots, a_t, z_t)$. The value of π under the scenario $\boldsymbol{\phi}$ is $V_\pi(seq) = \sum_{t=0}^{\infty} \gamma^t R(s_t, a_t)$. The empirical value $\hat{V}_\pi(b_0)$ of π under a set of K sampled scenarios is the average value over the individual scenarios. This estimate then replaces the exact value $V(b_0)$ when we evaluate (3).

DESPOT is an *anytime* algorithm, which builds its tree incrementally by performing heuristic search guided by a lower bound and an upper bound on the value at each tree node. For the lower bound at a leaf node b_l, we use the empirical value of a default policy under the sampled scenarios. Any valid policy may serve as a default policy and provides a lower bound on the value of an optimal policy. Our current implementation uses a simple reactive controller. It chooses ACCELERATE, MAINTAIN, or DECELERATE based on two distance thresholds D_{far} and D_{near} of the nearest pedestrian to the car. The value of this policy under a set of sampled scenario can be easily calculated by simulation. For the upper bound at b_l, it is computed as the average of the upper bounds for the scenarios. There are two cases for computing the upper for a scenario. If the vehicle and a pedestrian collides under the scenario, the bound is R_{col}. Otherwise, it is $\gamma^t R_{\text{goal}}$, where t is the minimum number of steps to reach the goal, assuming that the vehicle drives at

maximum speed with no pedestrians around. We then take the average over the sampled scenarios to obtain the upper bound at b_1. For the internal nodes of the DESPOT tree, we evaluate their lower and upper bounds recursively using (3).

The speed planner computes a policy over the entire DESPOT tree to hedge against uncertainty, but sends only the first step of the policy, the action at the root of tree to the vehicle for execution.

V. Path Planning

The path planner searches for a minimum-cost path that is potentially safe, fast, and smooth to drive. Compared with standard path planning, our path planner incorporates information on pedestrian movements into the cost.

We represent a path ρ as a sequence of point $(x_0, y_0), (x_1, y_1), \ldots, (x_n, y_n)$ equally spaced along ρ with distance ℓ between successive points. We define the path cost

$$C(\rho) = \sum_{i=0}^{n} \lambda^i C_{\text{st}}(x_i, y_i) + \sum_{i=0}^{n} \lambda^i C_{\text{ped}}(x_i, y_i) + \sum_{i=1}^{n-1} \lambda^i C_{\text{sm}}(\rho, i) \tag{4}$$

for a fixed constant $\lambda \in (0, 1]$. The path cost is the sum of three components. The first one C_{st} penalizes collision with static obstacles, the second one C_{ped} penalizes collision with pedestrians, and the third component C_{sm} penalizes non-smooth paths. Since we replan at each time step, the beginning of the path is closer to execution and more important. So we discount the cost exponentially by λ.

- $C_{\text{st}}(x, y)$ is a cost map of the environment discretized with a fixed-resolution grid. We place a potential field [10] around each static obstacle in the cost map.
- $C_{\text{ped}}(x, y)$ accounts for predicted future pedestrian movements. If the uncertainty on the pedestrian's intention is high, we put a large potential field around the pedestrian's current location to reflect the randomness in the pedestrian's future positions. Otherwise, we construct a segment of the pedestrian's most likely path and put a potential field over this path segment. Essentially, $C_{\text{ped}}(x, y)$ approximates the probability that a pedestrian and the vehicle collides at (x, y), marginalized over time and vehicle trajectories.
- The last component $C_{\text{sm}}(\rho, i)$ is the discrete path curvature of ρ at (x_i, y_i), suitably scaled to match the range of values with other components.

To search for a minimum-cost path, we apply the hybrid A* algorithm [18]. The search state is (x, y, θ), which encodes the vehicle position and orientation. The search action is $(\ell, \Delta\theta)$, which takes the vehicle from (x, y, θ) to a new state $(x + \ell\cos\theta, y + \ell\sin\theta, \theta + \Delta\theta)$. To guide the search, the heuristic function calculates the distance to drive from the current state to the goal, assuming no obstacles in between. One unique feature of the hybrid A* algorithm is that it maintains both the exact continuous state (x, y, θ) and a discretized version of it. This representation allows us to generate "drivable" paths that satisfy the nonholonomic constraint of a car while ensuring computational efficiency.

After finding a minimum-cost path ρ^*, the planner uses the first segment from (x_0, y_0) to (x_1, y_1) of ρ^* to calculate the desired vehicle orientation and the steering angle required to achieve it. It then sends the steering angle to the vehicle for execution and send ρ^* to the speed planner.

VI. Belief Tracking

From the observed pedestrian movements, the belief tracker infers the pedestrians' intentions, which are the key partially observable variables in our system. Since the number of intentions are finite, the belief over all the intentions form a discrete probability distribution. The belief tracker updates the belief with new observations using the Bayes' rule (1). For each pedestrian, we observe their movement from (x, y) to (x', y'). The belief $b(g)$ is then updated using the pedestrian model: $b'(g) = \eta p(x', y'|x, y, v, g) b(g)$, where η is a normalizing factor, and $p(x', y'|x, y, v, g)$ is the pedestrian model specified in Section IV-B.

The change of pedestrian's intention can be captured by the belief tracker. When a pedestrian switches from goal g_1 to g_2, the belief $b(g_1)$ will decrease while $b(g_2)$ will increase. To improve the robustness on handling intention changes, we add a small smoothing factor so that we always have $b(g) > 0$ for every g. This capability of handling intention changes is demonstrated in our experimental results in Section VII-B.

Currently we do not have a probabilistic dynamic model of pedestrian speed change and cannot perform Bayesian tracking. Instead, we use the more primitive method of exponentially weighted average speed to track the pedestrian speed, based on the history of observed pedestrian positions.

VII. Implementation and Experiments

A. The Experimental Vehicle

Our test vehicle is a YAMAHA G22E golf cart retrofitted for autonomous driving. The low-level vehicle controller controls the throttle, brake, and steering. The sensors include a SICK LMS 291 LIDAR, a camera, wheel encoders, and an inertia measurement unit (IMU). The LIDAR has a range of 30 meters and a field-of-view of 180°. The camera is an inexpensive off-the-shelf web camera.

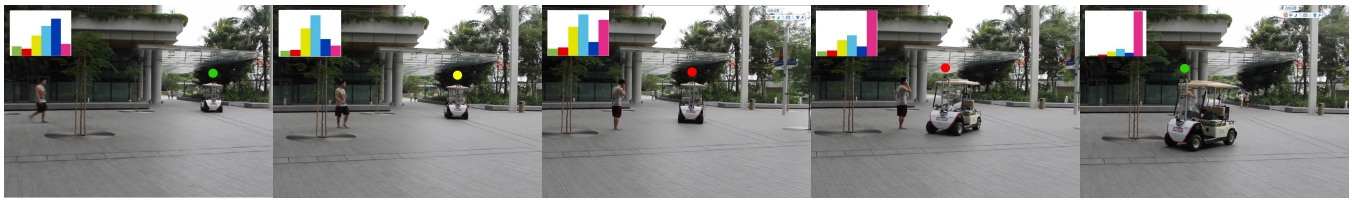
Our autonomous driving system is based on the Robot Operating System (ROS). The vehicle localizes itself in the given map, using the adaptive Monte-Carlo localization algorithm [19], which integrates data from the LIDAR, wheel encoders, and IMU. The vehicle controller regulates vehicle steering by tracking a given path with pure pursuit control [9] and regulates vehicle speed with PID control.

The system detects and tracks pedestrians using a combination of LIDAR and camera data. It clusters LIDAR point clouds to identify candidates and then uses HOG features from camera images to verify the candidates. More details on the system implementation are available in [3].

Our system, including path planning, online POMDP planning, belief tracking, and people tracking, all run in parallel on a single computer with a 4-core Intel processor. It performs path planning at about 2 Hz and performs online POMDP planning at strictly 3 Hz. For safety, the vehicle drives at the maximum speed of 1.5 m/s and triggers emergency brake when a pedestrian gets within 0.5 m.

B. Results on the Autonomous Golf Cart

We carried out experiments on our university campus in a plaza area about 70 m × 60 m in size (Fig. 6). It is a popular spot with large crowds of people passing by. We built a map of the environment from LIDAR data.



(*a*) (*b*) (*c*) (*d*) (*e*)

Fig. 5. The vehicle encounters a pedestrian who stops to make a phone call. Histograms indicate beliefs over pedestrian intentions. See Fig. 6 for color codes. The colored dots indicate vehicle actions: green for ACCELERATE, yellow for MAINTAIN, and red for DECELERATE.

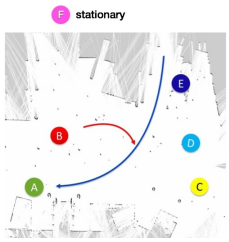


Fig. 6. A top-down view of the plaza area with a map built from LIDAR data. "A"–"F" indicate pedestrian intentions. The blue and the orange lines roughly correspond to the vehicle and the pedestrian paths, respectively, for the test run in Fig. 5.

Pedestrian intentions are modeled as goal locations ("A"–"E" in Fig. 6), which correspond to entrances to office buildings, shops, restaurants, *etc.*, as well as a bus stop.

The path planner uses a discretized cost map with resolution 0.1 m. There are 37 search actions with $\ell = 0.6$ m and $\Delta\theta$ ranging from $-18°$ to $18°$ at $1°$ intervals.

The POMDP speed planner tracks a maximum of six nearest pedestrians for planning, in order to balance between decision quality and computational efficiency. It takes in discretized observations with resolution of 0.5 m for position and 0.03 m/s for speed. Even this moderate setup results in a huge number of observations, about 10^{31}, but our DESPOT algorithm handles it well. The actions ACCELERATE, MAINTAIN, and DECELERATE are $0.5\,\text{m/s}^2$, $0\,\text{m/s}^2$, and $-0.5\,\text{m/s}^2$, respectively. The planning horizon is 90 steps. The maximum planning time per step is $1/3$ s.

We conducted extensive tests in this environment. With limited space, we report some instances here for illustration. For more details, see the video in the supplementary materials or online at http://youtu.be/jdkzV0lu2Bc.

In the first instance, the vehicle interacts with a single pedestrian (Fig. 5). Initially the vehicle starts up, and the pedestrian walks towards most likely D or E (Fig. 5*a*). The pedestrian then changes his intention. He slows down and eventually stops to make a phone call (Fig. 5*b*–*d*). Observe the change in the belief in the embedded histograms: the probabilities of D and E decrease, and the probability of F (pedestrian standing still) increases. In Fig. 5*d*, the probability of F has already risen to be the highest. The vehicle, however, chooses to decelerate, instead of accelerating to pass the pedestrian. Why? The residue uncertainty of intention estimation is still substantial, and the consequence of choosing a wrong action is severe. The vehicle must hedge against this uncertainty. This is exactly the reason given at the beginning of Section III. Finally, the vehicle is confident of the pedestrian intention to stand still and thus accelerates to pass him (Fig. 5*e*). This test also clearly demonstrates our system's ability to handle changing pedestrian intentions.

In the second instance, the vehicle drives amongst a crowd of almost 30 pedestrians (Fig. 7). Here understanding pedestrian intentions is important for driving safety and efficiency.

TABLE I

COMPARISON OF POMDP PLANNING AND REACTIVE CONTROL.

	Risk	Time (s)	Total Acceleration (m/s^2)
POMDP	0.0043 ± 0.0013	38.57 ± 0.16	6.31 ± 0.03
Reactive	0.0192 ± 0.0021	48.43 ± 0.27	7.85 ± 0.03

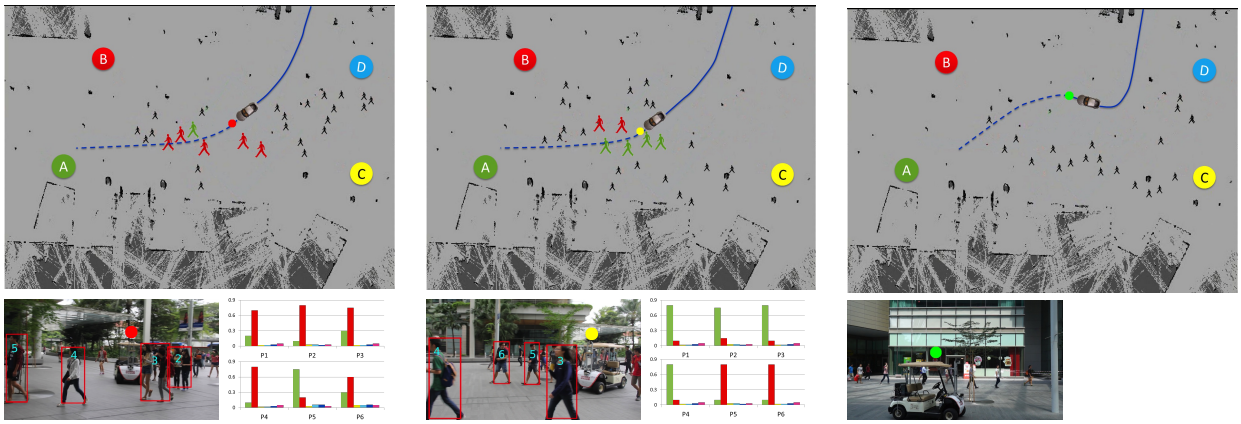
In Fig. 7*a*, the vehicle decelerates as several pedestrians intend to cross its path. In contrast, there are just as many pedestrians nearby in Fig. 7*b*, but they intend to walk away from the vehicle path. The vehicle thus maintains its speed. At another time (Fig. 7*c*), the pedestrians are mostly distributed in the lower-right side of the environment. The path planner replans a path going through a less congested region in order to reach the destination faster. See the video for more on path replanning.

C. Results in Simulation

The golf cart experiments indicate that the planner performs well in a crowded, dynamic environment. However, it remains difficult to quantify the performance benefits of our POMDP-based planner over simpler alternatives, because it is not possible to repeat the same dynamic environment exactly and evaluate multiple approaches. We thus performed a preliminary simulation study to compare our planner and a simple reactive controller.

We implemented our simulator to match up the setting in Section VII-B as closely as possible. The simulator is also based on ROS. It uses Stage [23] to simulate the vehicle dynamics and uses the same environment map. It simulates pedestrian motions by playing back data recorded in the plaza. This improves the accuracy of pedestrian simulation in some aspects, *e.g.*, by capturing the interactions among multiple pedestrians in a crowd. However, the potential downside is that these simulated pedestrians do not react to vehicle movements. This is not a major issue, as we do not model pedestrian reaction in the POMDP or take advantage of it in planning.

We compare our POMDP speed planner with a simple reactive controller, which is the default policy in Section IV-C, in more than $10,000$ trials. In this comparison, all the other system components, including, in particular, the path planner and the belief tracker are identical. This directly quantifies the benefits of the more powerful, but more expensive POMDP planner. The results are reported in Table I. The second column of the table shows the risk, which is the fraction of *near misses*. A near miss occurs when the vehicle gets within 0.5 m of a pedestrian, with speed greater than 1.0 m/s. The third column is the average time for the vehicle to reach the destination. The last column is the average total acceleration, which measures driving smoothness. Excessive



(*a*) (*b*) (*c*)

Fig. 7. The vehicle drives amongst a dense pedestrian crowd. The dashed blue line indicates the planned path. Each figurine indicates a pedestrian detected. The six pedestrians closest to the vehicle are tracked for the planning purpose, and the colors of their corresponding figurines indicate their most likely intentions. Each histogram on the lower right of a subfigure indicates the belief over the intentions of a tracked pedestrian. The last subfigure contains no histograms, as the pedestrians are all far away and not tracked.

acceleration or deceleration causes passenger discomfort. The total acceleration for each trial is the sum of the absolute values of all speed changes over time.

The result in Table I clearly shows that compared with the reactive controller, our POMDP-based planner enables driving that is safer, faster, and more smooth. Of course, this simple reactive controller uses a very weak model for decision making. See [1] for comparison of POMDP planning with other stronger, but still greedy approaches.

VIII. Conclusion

This paper presents an intention-aware online POMDP planner for autonomous driving among many pedestrians. Experiments show that our planner runs in almost real time on a golf cart and enables effective autonomous driving in a complex, dynamic environment. These experiments provide two important lessons. First, a principled decision framework such as the POMDP is essential for safe, efficient, and smooth autonomous driving. Second, despite the concern about the complexity of POMDP planning, it is improving fast in computational efficiency and becoming increasingly practical as a tool for robot planning under uncertainty. However, we need to understand better the gap in decision quality between POMDP planning and its greedier and faster alternatives, in the context of autonomous vehicles interacting with pedestrians or other vehicles.

References

[1] T. Bandyopadhyay, K. Won, E. Frazzoli, D. Hsu, W. Lee, and D. Rus, "Intention-aware motion planning," in *Algorithmic Foundations of Robotics X—Proc. Int. Workshop on the Algorithmic Foundations of Robotics (WAFR)*, 2012.

[2] M. Bennewitz, W. Burgard, G. Cielniak, and S. Thrun, "Learning motion patterns of people for compliant robot motion," *Int. J. Robotics Research*, vol. 24, no. 1, pp. 31–48, 2005.

[3] Z. Chong, B. Qin, T. Bandyopadhyay, T. Wongpiromsarn, E. Rankin, M. Ang, E. Frazzoli, D. Rus, D. Hsu, and K. Low, "Autonomous personal vehicle for the first-and last-mile transportation services," in *Proc. IEEE Int. Conf. on Cybernetics & Intelligent Systems*, 2011.

[4] D. Fox, W. Burgard, and S. Thrun, "The dynamic window approach to collision avoidance," *IEEE Robotics & Automation Magazine*, vol. 4, no. 1, pp. 23–33, 1997.

[5] C. Fulgenzi, C. Tay, A. Spalanzani, and C. Laugier, "Probabilistic navigation in dynamic environment using rapidly-exploring random trees and gaussian processes," in *Proc. IEEE/RSJ Int. Conf. on Intelligent Robots & Systems*, 2008.

[6] R. He, E. Brunskill, and N. Roy, "Efficient planning under uncertainty with macro-actions," *J. Artificial Intelligence Research*, vol. 40, no. 1, pp. 523–570, 2011.

[7] R. Howard, "Information value theory," *IEEE Trans. on Systems Science & Cybernetics*, vol. 2, no. 1, pp. 22–26, 1966.

[8] T. Ikeda, Y. Chigodo, D. Rea, F. Zanlungo, M. Shiomi, and T. Kanda, "Modeling and prediction of pedestrian behavior based on the sub-goal concept," in *Proc. Robotics: Science & Systems*, 2012.

[9] Y. Kuwata, J. Teo, G. Fiore, S. Karaman, E. Frazzoli, and J. How, "Real-time motion planning with applications to autonomous urban driving," *IEEE Trans. on Control Systems Technology*, vol. 17, no. 5, pp. 1105–1118, 2009.

[10] J. Latombe, *Robot Motion Planning*. Kluwer Academic Publishers, 1991.

[11] M. Montemerlo *et al.*, "Junior: The Stanford entry in the urban challenge," *J. Field Robotics*, vol. 25, no. 9, pp. 569–597, 2008.

[12] S. Ross, J. Pineau, S. Paquet, and B. Chaib-Draa, "Online planning algorithms for POMDPs," *J. Artificial Intelligence Research*, vol. 32, no. 1, pp. 663–704, 2008.

[13] B. Schiller, V. Morellas, and M. Donath, "Collision avoidance for highway vehicles using the virtual bumper controller," in *Proc. IEEE Int. Symp. on Intelligent Vehicles*, 1998.

[14] G. Shani, J. Pineau, and R. Kaplow, "A survey of point-based POMDP solvers," *Autonomous Agents and Multi-Agent Systems*, vol. 27, no. 1, pp. 1–51, 2013.

[15] D. Silver and J. Veness, "Monte-Carlo planning in large POMDPs," in *Advances in Neural Information Processing Systems (NIPS)*, 2010.

[16] R. Smallwood and E. Sondik, "The optimal control of partially observable Markov processes over a finite horizon," *Operations Research*, vol. 21, pp. 1071–1088, 1973.

[17] A. Somani, N. Ye, D. Hsu, and W. Lee, "DESPOT: Online POMDP planning with regularization," in *Advances in Neural Information Processing Systems (NIPS)*, 2013.

[18] S. Thrun *et al.*, "Stanley: The robot that won the DARPA grand challenge," *J. Field Robotics*, vol. 23, no. 9, pp. 661–692, 2006.

[19] S. Thrun, W. Burgard, and D. Fox, *Probabilistic robotics*. MIT press, 2005.

[20] C. Urmson *et al.*, "Autonomous driving in urban environments: Boss and the urban challenge," *J. Field Robotics*, vol. 25, no. 8, pp. 425–466, 2008.

[21] J. van den Berg, P. Abbeel, and K. Goldberg, "LQG-MP: Optimized path planning for robots with motion uncertainty and imperfect state information," *Int. J. Robotics Research*, vol. 30, no. 7, pp. 895–913, 2011.

[22] D. Vasquez, T. Fraichard, and C. Laugier, "Growing hidden Markov models: An incremental tool for learning and predicting human and vehicle motion," *Int. J. Robotics Research*, vol. 28, no. 11–12, pp. 1486–1506, 2009.

[23] R. Vaughan, "Massively multi-robot simulation in stage," *Swarm Intelligence*, vol. 2, no. 2-4, pp. 189–208, 2008.

[24] D. Zax, "Can driverless pod cars solve the "last-mile problem"?" *MIT Technology Review*, 17 August 2011.