2009

Planning-based Prediction for Pedestrians

Brian Ziebart
Carnegie Mellon University

Nathan Ratliff
Carnegie Mellon University

Garratt Gallagher
Carnegie Mellon University

Kevin Peterson
Carnegie Mellon University

J. Andrew Bagnell
Carnegie Mellon University

See next page for additional authors

Authors

Brian Ziebart, Nathan Ratliff, Garratt Gallagher, Kevin Peterson, J. Andrew Bagnell, Martial Hebert, Anind K. Dey, and Siddhartha Srinivasa

Planning-based Prediction for Pedestrians

Brian Ziebart[1]   Nathan Ratliff[1]   Garratt Gallagher[1]   Kevin Peterson[1]
J. Andrew Bagnell[1]   Martial Hebert[1]   Anind K. Dey[1]   Siddhartha Srinivasa[2]

[1]*School of Computer Science*
Carnegie Mellon University
{*bziebart, ndr, ggallagh, kp, dbagnell, hebert, anind*}*@cs.cmu.edu*

[2]*Intel Research*
Pittsburgh, PA
siddhartha.srinivasa@intel.com

Abstract— In this paper, we describe a novel uncertainty-based technique for predicting the future motions of a moving person. Our model assumes that people behave purposefully – efficiently acting to reach intended destinations. We employ the Markov decision process framework and the principle of maximum entropy to obtain a probabilistic, approximately optimal model of human behavior that admits efficient inference and learning algorithms. The method learns a cost function of features of the environment that explains previously observed behavior. This enables generalization to physical changes in the environment, and entirely different environments. Our approach enables robots to plan paths that balance time-to-goal and pedestrian disruption. We quantitatively show the improvement provided by our approach.

I. INTRODUCTION

Determining appropriate robotic actions in environments with moving people is a well-studied [13], [1], [4], but often difficult task due to the uncertainty of each person's future behavior. Certainly a robot should never collide with a person [10], but preventing collisions alone is often unsatisfactory because the disruption of almost colliding can be burdensome to the person and sub-optimal for the robot. Instead, the robot should predict the future locations of people and plan routes that will avoid such hindrances (*i.e.*, situations where the person's natural behavior is disrupted due to the robot's proximity) while still efficiently achieving the robot's objectives. For example, given the origins and target destinations of the robot and person in Figure 1, the robot's hindrance-minimizing trajectory would take the longer way around the center obstacle (a table) and leave the pedestrian a clear path.

One common previous approach for predicting people's trajectories is to project the prediction step of a tracking filter [8], [11], [9] forward over time. For example, using a Kalman filter [6], future positions are predicted according to a Gaussian distribution with growing uncertainty, and, unfortunately, often high probability for physically impossible locations (*e.g.*, behind walls, within obstacles). Particle filter approaches [14] can incorporate more sophisticated constraints and non-Gaussian distributions, but degrade into random walks of feasible motion over large predictive time horizons rather than purposeful, goal-based motion. Closer to our research are approaches that directly model the policy [5]. These approaches assume that previously observed trajectories capture all purposeful behavior, and the only uncertainty involves determining to which previously observed class of trajectories the current behavior belongs. Models based on mixtures of trajectories and conditioned action distribution modeling (using hidden Markov models) have been employed [15]. This approach often suffers from over-fitting to the particular training trajectories and context of those trajectories. When changes to the environment
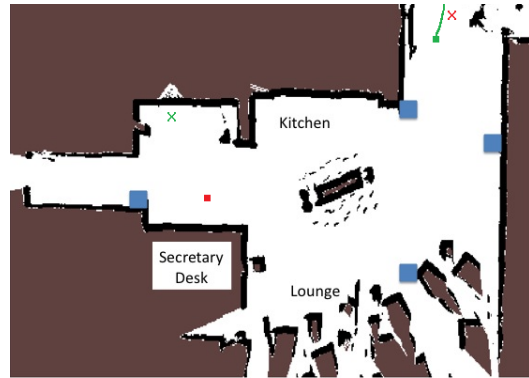

Fig. 1.   A hindrance-sensitive robot path planning problem in our experimental environment containing a person (green square) in the upper right with a previous trajectory (green line) and intended destination (green X) near a doorway, and a robot (red square) near the secretary desk with its intended destination (red X) near the person's starting location. Robot-person hindrances are likely if the person and robot both take the distance-minimizing path to their intended destinations. Laser scanners are denoted with blue boxes.
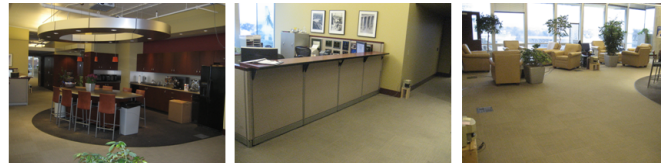

Fig. 2.   Images of the kitchen area (left), secretary desk area (center), and lounge area (right) of our experimental environment.

occur (*e.g.*, rearrangement of the furniture), the model will confidently predict incorrect trajectories *through* obstacles.

We present a novel uncertainty-based planning approach to this problem. In traditional planning, given a *cost function* mapping features of each state and action to a cost, the optimal trajectory is easily obtained for any endpoints in any environment described using the same features. Our approach learns the cost function that best explains previously observed trajectories. Unfortunately, traditional planning is *prescriptive* rather than *predictive* – the sub-optimality typically present in observed data is inexplicable to a planner. We employ the principle of maximum entropy to address the lack of decision uncertainty using a technique we previously developed called maximum entropy inverse reinforcement learning[16]. Employing this approach yields a *soft-maximum* version of Markov decision processes (MDP) capable of accounting for decision uncertainty. As we shall show, this soft-max MDP generalizes MDPs, and it supports

efficient algorithms for learning the cost function that best explains previous behavior, and for predicting a person's future positions and trajectory.

Importantly, the cost function employed by our modeling approach is feature-based, which enables generalization. Specifically, the cost function employed is a linear combination of a given set of features computed from the environment (*e.g.*, obstacles and filters applied to obstacles). Once trained, the cost function applies to any configuration of these features. Therefore if obstacles in the environment move, or the environment otherwise changes, or we consider an entirely different environment, our model generalizes to this new setting. We consider this improved generalization to a major benefit of our approach over previous techniques.

It is important in robotics applications to reason about the behavior and navigational decisions of humans when acting within their environment. One natural approach to leverage our predictive model is to employ a planner that uses time-dependant costs. Unfortunately, the increased dimensionality of the planning problem can be prohibitive. Instead, we present a simple, incremental "constraint generation" approach to planning that enables real-time performance. This approach initially employs a cost map that ignores the predictions of people's future locations. Each time the algorithm receives new pedestrian predictions, it starts with a prior cost map and iterates the following procedure: 1. plan under the current cost map, 2. simulate the plan to find locations where the plan will likely interfere with the persons predicted path (both spacially and temporally), and 3. update the cost map to penalize those points of interference. The time-independent cost function that this procedure produces accounts for the time-varying predictions, and ultimately yields a high quality, hindrance-free robot trajectory, while requiring much less computation than a time-based planner.

We evaluate the quality of our combined prediction and planning system on the trajectories of people in a lab environment using the opposing objectives of maximizing the robot's efficiency in reaching its intended destination and minimizing robot-person hindrances. An inherent trade-off between these two criteria exists in planning appropriate behavior. We show that for any chosen trade-off, our prediction model is better for making decisions than an alternative approach.

II. MODELING PURPOSEFUL MOTION

Accurate predictions of the future positions of people enable a robot to plan appropriate actions that avoid hindering those people. We represent the sequence of actions that lead to a person's future position using a deterministic Markov decision process (MDP) over a grid representing the environment. Unfortuntely, people do not move in a perfectly predictable manner, and instead our robot must reason probabilistically about their future locations. By maximizing the entropy of the distribution of trajectories, $H(P_\zeta) = -\sum_\zeta P(\zeta) \log P(\zeta)$ subject to the constraint of matching the reward of the person's behavior in expectation, we obtain a distribution over trajectories [16].

In this section, we present a new interpretation of the maximum entropy distribution over trajectories, and algorithms for obtaining it. This is framed as a softened version of the value iteration algorithm, which is commonly employed to find optimal policies for MDPs. We first review the Bellman equations and value iteration for solving MDPs. We next relax these equations, obtaining a distribution over actions and trajectories. We then employ Bayes' Rule using this distribution to reason about unknown intended destinations. We then compute expected visitation counts, $D_{x,y}$, across the environment, and finally obtain time-based visitation counts, $D_{x,y,t}$, that can be used for hindrance-sensitive planning purposes.

A. Relaxing Maximum Value MDPs for Prediction

Consider as a model of planning, a Markov decision process with deterministic action outcomes. This class of MDPs consist of a state set, S, an action set, A, an action transition function, $T : S \times A \to S$, and a reward function, $R : S \times A \to \mathbb{R}$. A trajectory, ζ, is a sequence of states and actions, $\{s_0, a_0\}$ that satisfies the transition function (*i.e.*, $\forall_{s_{t+1}, s_t, a_t \in \zeta} \ T(s_t, a_t) = s_{t+1}$). Goal-based planning is modeled in the MDP by assigning costs (*i.e.*, negative rewards) for every action except for a self-transitioning action in the *absorbing* goal state that has a cost of 0. MDPs are *solved* by finding the state and action values (*i.e.*, future reward of that action or state), $V^*(s)$ and $Q^*(s, a)$, for the maximum future reward policy, $\pi^* : S \to A$. The Bellman equations define the optimal quantities:

$$Q^*(s, a) = R(s, a) + V^*(T(s, a)) \qquad (1)$$
$$V^*(s) = \max_a Q^*(s, a). \qquad (2)$$

The value iteration algorithm produces these optimal values by alternately applying Equations 1 and 2 as update rules until the values converge. The optimal policy is then: $\pi^*(s) = \text{argmax}_a Q^*(s, a)$. While useful for prescribing a set of actions to take, this policy is not usually predictive because observed trajectories are often not consistently optimal.

We employ a "softened" version of MDPs that incorporates trajectory uncertainty into our model. In this setting, trajectories are probabilistically distributed according to their values rather than having a single optimal trajectory for the solution. We accomplish this by replacing the *maximum* of the Bellman equations with a *soft-maximum* function, $\text{softmax}_x f(x) = \log \sum_x e^{f(x)}$.

$$Q^{\approx}(s, a) = R(s, a) + V^{\approx}(T(s, a)) \qquad (3)$$
$$V^{\approx}(s) = \text{softmax}_a Q^{\approx}(s, a) \qquad (4)$$

In this case, the solution policy is probabilistic with distribution: $\pi(a|s) = e^{Q^{\approx}(s,a) - V^{\approx}(s)}$. The probability of a trajectory, ζ, can be shown [16] to be distributed according to $P(\zeta) \propto e^{\sum_{(s,a) \in \zeta} R(s,a)}$. Trajectories with a very high reward (low cost) are exponentially more preferable to low reward (high cost) trajectories, and trajectories with equal reward are equally probable. The magnitude of the rewards, $|R(s, a)|$, is meaningful in this softmax setting and corresponds to the certainty about trajectories. As $|R(s, a)| \to \infty$, $\text{softmax}_a Q^{\approx}(s, a) = \max_a Q^*(s, a)$, and the distribution converges to only optimal trajectories. An analogous $O(T|S||A|)$ time value-iteration procedure is employed (with appropriately chosen T) to solve this softmax policy distribution – terminating when the value function has reached an acceptable level of convergence. Note that this softmax value distribution over trajectories is very different than the softmax action selection distribution that has been employed for reinforcement learning: $P_\tau(a|s) \propto e^{Q^*(s,a)/\tau}$ [12].

B. Learning Unknown Cost Functions

For prescriptive MDP applications, the reward values for actions in the MDP are often engineered to produce

appropriate behavior, however for our prediction purposes, we would like to find the reward values that best predict a set of observed trajectories, $\{\tilde{\zeta}_i\}$. We assume the availability of a vector of *feature values*, $\mathbf{f}_{s,a}$, characterizing each possible action. For our application, these features are obstacle locations and functions of obstacle locations (*e.g.*, blurring and filtering of obstacles). We assume that the reward is linear in these features, $R(s,a) = \theta^\top \mathbf{f}_{s,a}$, with unknown weight parameters, θ. We denote the first state, s_0, of trajectory $\tilde{\zeta}_i$ as $s_0 \in \tilde{\zeta}_i$. The learning problem is then[1]:

$$\theta^* = \operatorname*{argmax}_{\theta} P(\tilde{\zeta}_i|\theta) \tag{5}$$

$$= \operatorname*{argmax}_{\theta} \sum_i \left(\left(\sum_{(s,a)\in\tilde{\zeta}_i} \theta^\top \mathbf{f}_{s,a} \right) - V^\approx(s_0 \in \tilde{\zeta}_i) \right).$$

The gradient of the function in Equation 5 has an intuitive interpretation as the difference between the feature counts of observed trajectories and expected feature counts according to the model: $\sum_{i,(s,a)\in\tilde{\zeta}_i} \mathbf{f}_{s,a} - E_{P_\theta(\zeta)}[\mathbf{f}_{s,a}]$. We employ gradient-based optimization on this convex function to obtain θ^* [16]. We refer the reader to that work for a more detailed explanation of the optimization procedure.

C. Destination Prior Distribution

Up until this point, our model has been conditioned on a known destination location. However, this destination location is not known at prediction time. Our predictive model must reason about all possible destinations to predict the future trajectory of a person. We address this problem in a Bayesian way by first obtaining a prior distribution over destinations using previously observed trajectories and the features of the environment.

In this work, we base our prior distribution on the goals of previously observed trajectories. We *smooth* this probability to nearby cells and also add probability for previously unvisited locations to avoid overfitting. When little or no previous data is available for a particular environment, a feature-based model of destinations with features expressing door, chair, and appliance locations could be employed.

D. Efficient Future Trajectory Prediction

In the prediction setting, the robot knows the person's partial trajectory from state A to current state B, $\zeta_{A\to B}$. and must infer the future trajectory of the person, $P(\zeta_{B\to C})$, to an unknown destination state, C, given all available information. First we infer the posterior distribution of destinations given the partial trajectory, $P(\text{dest } C|\zeta_{A\to B})$, using Bayes' Rule. For notational simplicity, we denote the softmax value function of state X to destination state Y as $V^\approx(X \to Y)$ and the reward of a policy as $R(\zeta) = \sum_{(s,a)\in\zeta} R(s,a)$. The posterior distribution is then:

$$P(\text{dest C}|\zeta_{A\to B}) = \frac{P(\zeta_{A\to B}|\text{dest C})P(\text{dest C})}{P(\zeta_{A\to B})}$$

$$= \frac{\frac{e^{R(\zeta_{A\to B})+V^\approx(B\to C)}}{e^{V^\approx(A\to C)}}P(\text{dest } C)}{\sum_D \frac{e^{R(\zeta_{A\to B})+V^\approx(B\to D)}}{e^{V^\approx(A\to D)}}P(\text{dest } D)}. \tag{6}$$

The value functions, $V^\approx(A \to D)$ and $V^\approx(B \to D)$, for each state D are required to compute this posterior

[1]We assume the final state of the trajectory is the goal destination and our probability distribution is conditioned on this goal destination.

(Equation 6). The naïve approach is to execute $O(|D|)$ runs of softmax value iteration – one for each possible goal, D. Fortunately there is a much more efficient algorithm. In the hard maximum case, this problem is solved efficiently by modifying the Bellman equations to operate backwards, so that instead of $V(S)$ representing the future value of state S, it is the maximum value obtained by a trajectory reaching state S. Initializing $\forall_{s\neq A}V(s) = -\infty$ and $V(A) = 0$, the following equations define $V(A \to D)$ for all D.

$$Q(s,a) = R(s,a) + V(s) \tag{7}$$

$$V(s) = \max_{(s',a):T(s',a)=s} Q(s',a) \tag{8}$$

For the soft-max reward case, the $\max$ is replaced with softmax (Equation 8) and the value functions, $V^\approx(A \to D)$, are obtained with a value-iteration algorithm. Thus, with two applications of value-iteration to produce $V^\approx(A \to D)$ and $V^\approx(B \to D)$, and $O(|D|)$ time to process the results, the posterior distribution over destinations is obtained.

We now use the destination posterior to compute the conditional probability of any continuation path, $\zeta_{B\to C}$:

$$P(\zeta_{B\to C}|\zeta_{A\to B}) \tag{9}$$

$$= \sum_D P(\zeta_{B\to C}|\zeta_{A\to B}, \text{dest } D)P(\text{dest } D|\zeta_{A\to B})$$

$$= P(\zeta_{B\to C}|\text{dest } C)P(\text{dest } C|\zeta_{A\to B})$$

$$= e^{R(\zeta_{B\to C})-V^\approx(B\to C)}P(\text{dest } C|\zeta_{A\to B}).$$

This can be readily computed using the previously computed posterior destination distribution and $V^\approx(B \to C)$ computed as part of generating that posterior distribution.

For planning appropriate robotic movement, the expected occupancy of different states, D_x, is of greater interest than the probability of a particular trajectory. This is obtained by marginalizing over all paths that contain D_x. For this class of paths, which we denote $\Xi_{B\to x\to C}$, the path can divided[2] into a path from B to x and a path from x to C.

$$D_s = \sum_C \sum_{\zeta\in\Xi_{B\to x\to C}} P(\zeta_{B\to C}|\text{dest } C)P(\text{dest } C|\zeta_{A\to B})$$

$$= \sum_C P(\text{dest } C|\zeta_{A\to B})$$

$$\sum_{\zeta_1\in\Xi_{B\to x}} \sum_{\zeta_2\in\Xi_{x\to C}} e^{R(\zeta_1)+R(\zeta_2)-V^\approx(B\to C)}$$

$$= \left(\sum_{\zeta_1\in\Xi_{A\to x}} e^{R(\zeta_1)} \right) \tag{10}$$

$$\left(\sum_C \sum_{\zeta_2\in\Xi_{x\to C}} e^{R(\zeta_2)+\log P(\text{dest } C|\xi_{A\to B}-V^\approx(B\to C))} \right)$$

The first summation of Equation 10 equates to $e^{V^\approx(A\to x)}$, which we can easily obtain from our previously computed value functions. We compute the second double summation by adding a final state reward of $(\log P(\text{dest } C|\xi_{A\to B}) - V^\approx(B \to C))$ and performing soft value iteration with those modified rewards. Thus with one

[2]This derivation assumes paths containing state s at most once. The resulting solution is also correct for paths with more than one occurance of state s, but in the interest of space we do not present the less compact derivation for this general case.

Algorithm 1 Incorporating predictive pedestrian models via predictive planning

1: **procedure** PREDICTIVEPLANNING($\sigma > 0$, $\alpha > 0$, $\{D_{s,t}\}$, D_{thresh})
2:     Initialize cost map to prior navigational costs $c_0(s)$.
3:     **for** $t = 0, \ldots, T$ **do**
4:         Plan under the current cost map.
5:         Simulate the plan forward to find points of probable interference with the pedestrian $\{(s_i)\}_{i=1}^{K_t}$ where $D_{s,t} > D_{\text{thresh}}$.
6:         If $K = 0$ then break.
7:         Add cost to those points
8:         $c_{t+1}(s) = c_t(s) + \alpha \sum_{i=1}^{K_t} e^{-\frac{1}{2\sigma^2}\|s-s_i\|^2}$.
9:     **end for**
10:    **return** The plan through the final cost map.
11: **end procedure**

---

additional application of the soft value iteration algorithm and combining the results (which is constant time with respect to the number of goals), we obtain state expected visitation counts.

E. Temporal Predictions

To plan appropriately, our robot must not only predict where people will be, but when they will be there as well. More formally, we need predictions of expected future occupancy of each location during the time windows surrounding fixed intervals: $\tau, 2\tau, \ldots, T\tau$. We denote these quantities as $D_{s,t}$.

In theory, time can be added to the state space of a Markov decision process and explicitly modeled. In practice, however, this expansion of the state space significantly increases the time complexity of inference, making real-time applications based on the time-based model impractical. We instead consider an alternative approach that is much more tractable.

We assume that a person's movement will "consume" some cost over a time window t according to the normal distribution $N(tC_0, \sigma_0^2 + t\sigma_1^2)$, where C_0, σ_0^2, and σ_1^2 are learned parameters. Certainly $\sum_t D_{s,t} = D_s$, so we simply divide the expected visitation counts among the time intervals according to this probability distribution. We use the cost of the optimal path to each state, $Q^*(s)$, to estimate the cost incurred in reaching it. The resulting time-dependent occupancy counts are then:

$$D_{s,it} \propto D_s e^{\frac{-(C_0 t - Q^*(s))^2}{2(\sigma_0^2 + t\sigma_1^2)}}. \quad (11)$$

These values are computed using a single execution of Dijkstra's algorithm [2] in $O(|S|\log|S|)$ time to compute $Q^*(.)$ and then $O(|S|T)$ time for additional calculation.

III. PLANNING WITH PEDESTRIAN PREDICTIONS

Ideally, to account for predictive models of pedestrian behavior, we should increase the dimensionality of the planning problem by augmenting the state of the planner to account for time-varying costs. Unfortunately, the computational complexity of combinatorial planning is exponential in the dimension of the planning space, and the added computational burden of this solution will be prohibitive for many real-time applications.

We therefore propose a novel technique for integrating our time-varying predictions into the robot's planner. Algorithm 1 details this procedure; it essentially iteratively shapes a time-independent navigational cost function to remove known points of hindrance. At each iteration, we run the time-independent planner under the current cost map and simulate forward the resulting plan in order to predict points at which the robot will likely interfere with the pedestrian. By then adding cost to those regions of the map we can ensure that subsequent plans will not interfere at those particular location. We can further improve the computational gain of this technique by using efficient replanners such as D* and its variants [3] in the inner loop. While this technique, as it reasons only about stationary costs, cannot guarantee the optimal plan given the time-varying costs, we demonstrate that it produces good robot behavior in practice that efficiently accounts for the predicted motion of the pedestrian.

By re-running this iterative replanner every 0.25 seconds using updated predictions of pedestrian motion, we can achieve intelligent adaptive robot behavior that anticipates where a pedestrian is heading and manuvers well in advance to implement efficient avoidance. The accompanying movie demonstrates the behavior that emerges from our predictive planner in select situations. In practice, we use the final cost-to-go values of the iteratively constructed cost map to implement a policy that chooses a good action from a predefined collection of actions. Since our algorithm can detect when a good plan with sufficiently low probability of pedestrian hindrance cannot be found, we can varying the robot's speed according to the uncertainty in pedestrian behavior. Additionally, when the robot is too close to a pedestrian, all actions that take the robot within a particular radius of the human are removed to retain robustness to unpredictable behavior. Section IV-F presents quantitative experiments demonstrating the properties of this policy.

IV. EXPERIMENTAL EVALUATION

We now present experiments demonstrating the capabilities of our prediction model and its usefulness for planning hindrance-sensitive robot trajectories[3].

A. Data Collection

We collected over one month's worth of data in a lab environment. The environment has three major areas (Figure 1): a kitchen area with a sink, refrigerator, microwave, and coffee maker; a secretary desk; and a lounge area. We installed four laser range finders in fixed locations around the lab, as shown in Figure 1, and ran a pedestrian tracking algorithm [7]. Trajectories were segmented based on significant stopping time in any location.

From the collected data, we use a subset of 166 trajectories through our experimental environment to evaluate our approach. This dataset is shown in Figure 3 after post-processing and being fit to a 490 by 321 cell grid (each cell represented as a single pixel). We employ 50% of this data as a training set for estimating the parameters of our model and use the remainder for evaluative purposes.

B. Learning Feature-Based Cost Functions

We learn a function over simple features of the environment, which we argue are easily transferable to other environments. The first feature is a constant feature for every

---

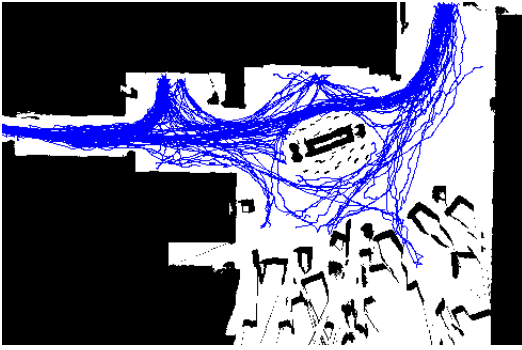[3]Additional videos and images are available at: http://www.cs.cmu.edu/~lairlab/planpredict/

Fig. 3. Collected trajectory dataset.

grid cell in the environment. The remaining functions are an indicator function for whether an obstacle exists in a particular grid cell, and four "blurs" of obstacle occupancies, which are shown in Figure 4.
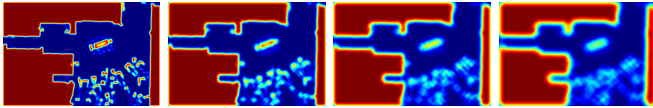

Fig. 4. Four obstacle-blur features for our cost function. Feature values range from low weight (dark blue) to high weight (dark red).

We then learn the weights for these features that best explain the demonstrated data. The resulting cost function for the environment is shown in Figure 5. Obstacles in the cost function have very high cost, and free space has a low cost that increases near obstacles.
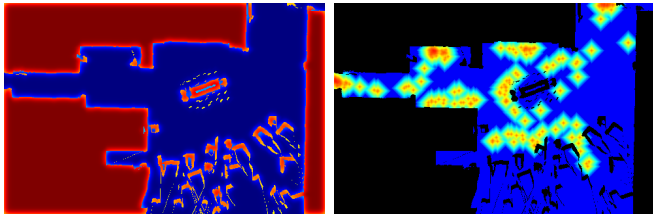

Fig. 5. Left: The learned cost function in the environment. Right: The prior distribution over destinations learned from the training set.

The prior distribution over destinations is obtained from the set of endpoints in the training set, and the temporal Gaussian parameters are also learned using the training set.

C. Stochastic Modeling Experiment

We first consider two examples from our dataset (Figure 6) that demonstrate the need for uncertainty-based modeling.
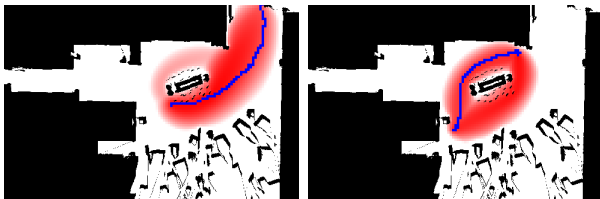

Fig. 6. Two trajectory examples (blue) and log occupancy predictions (red).

Both trajectories travel around the table in the center of the environment. However, in the first example (left), the

person takes the lower pathway around the table, and in the second example (right), the person takes the upper pathway despite that the lower pathway around the table has a lower cost in the learned cost function. In both cases, the path taken is not the shortest path through the open space that one would obtain using an optimal planner. Our uncertainty-based planning model handles these two examples appropriately, while a planner would choose one pathway or the other around the table and, even after smoothing the resulting path into a probability distribution, tend to get a large fraction of its predictions wrong when the person takes the "other" approximately equally desirable pathway.

D. Dynamic Feature Adaptation Experiment

In many environments, the relevant features that influence movement change frequently – furniture is moved in indoor environments, the locations of parked vehicles are dynamic in urban environments, and weather conditions influence natural environments with muddy, icy, or dry conditions. We demonstrate qualitatively that our model of motion is robust to these feature changes.

The top frames of Figure 7 show the environment and the path prediction of a person moving around the table at two different points in time. At the second point of time (top right), the probability of the trajectory leading to the kitchen area or the left hallway is extremely small. In the bottom frames of Figure 7, an obstacle has been introduced that blocks the direct pathway through the kitchen area. In this case, the trajectory around the table (bottom right) still has a very high probability of leading to either the kitchen area or the left hallway. As this example shows, our approach is robust to changes in the environment such as this one.
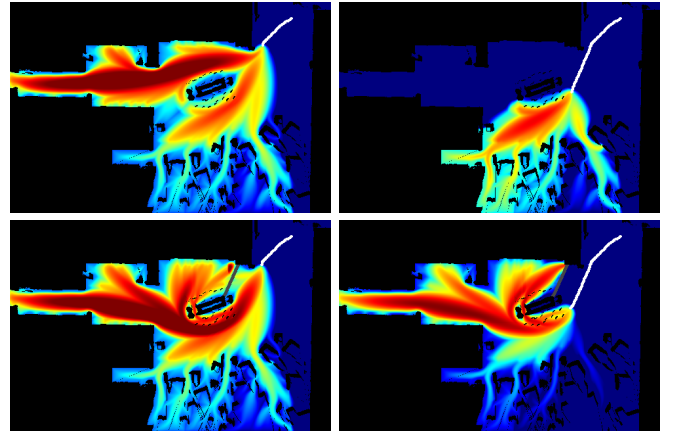

Fig. 7. Our experimental environment with (right column) and without (left column) an added obstacle (gray) between the kitchen and center table. Predictions of future visitation expectations given a person's trajectory (white line) in both settings for two different trajectories. Frequencies range from red (high log expectation) to dark blue (lower log expectation)

E. Comparative Evaluation

We now compare our model's ability to predict the future path of a person with a previous approach for modeling goal-directed trajectories – the variable-length Markov model (VLMM) [5]. The VLMM estimates the probability of a person's next cell transition conditioned on the person's history of cells visited in the past. It is variable length because it employs a long history when relevant training data is abundant, and a short history otherwise.
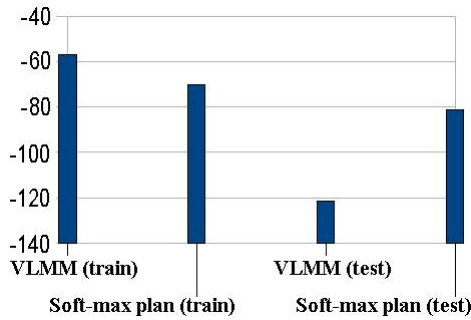
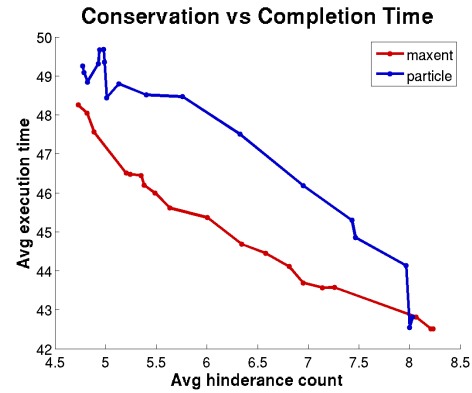Fig. 8. Log probability of datasets under the VLMM and our approach.

Fig. 9. This plot demonstrates the tradeoff in efficiency vs pedestrian hindrance as we vary the degree to which we utilize the predictions during planning under both planning-based predictions and particle-based predictions.

The results of our experimental evaluation are shown in Figure 8. We first note that for the training set (denoted *train*), that the trajectory log probability of the VLMM is significantly better than the plan-based model. However, for the test set, which is the metric we actually care about, the performance of the VLMM degrades significantly, while the degradation in the plan-based model is much less extreme. We conclude from this experiment that the VLMM (and similar directed graphical model approaches) are generally much more difficult to train to generalize well because their number of parameters is significantly larger than the number of parameters of the cost function employed in our approach.

F. Integrated Planning Evaluation

We now simulate robot-human hindrance problems to demonstrate the benefit of our trajectory forecasting approach. We generate 200 hindrance-sensitive planning problems (corresponding to 22 different person trajectories in Figure 3) by selecting problems where naïve planning (disregarding the pedestrian) causes hindrances. We ignore the causal influence of the robot's action on the person's trajectory, and measure the trade-off between robot efficiency and interference with the person's trajectory. Specifically, the *average hindrance count* measures the average number of times the policy removed actions due to proximity to a human, and the *average execution time* measures the number of time steps needed to reach the goal. The trade-off is controlled by varying the degree of the visitation frequency threshold used in Algorithm 1. The robot trajectory planner is provided with person location forecasts at 4Hz[4].

The trade-off curves of planning using our plan-based forecasts and using a straight-forward particle-based forecasting model on this set of problems are shown in Figure 9. For both predictive models, while the execution time varies over a small range, the average hindrance count decreases by a factor of two. Additionally, as the figure shows, the plan-based forecasting is almost uniformly superior to the particle-based approach for any level of the trade-off.

V. CONCLUSIONS AND FUTURE WORK

We have presented a novel approach for predicting the future trajectory of a person moving through an environment using a soft-max version of goal-based planning. As we have shown, the feature-based cost function learned using this approach allows accurate generalization to changes in the environment. We additionally showed the usefulness of this approach for planning hindrance-sensitive routes using a novel incremental path planner.

In future work, we plan to explicitly model interactions between agents – specifically for multiple people and their trajectories within the environment – so that we can better predict the behavior of people in crowded environments. Additionally, we have assumed a fully observed world that many robotics applications lack. Effeciently extending our approach to the setting where human behavior is only partially observable remains as important future work.

REFERENCES

[1] M. Bennewitz, W. Burgard, and S. Thrun. Learning motion patterns of persons for mobile service robots. In *Proc. ICRA*, pages 3601–3606, 2002.
[2] E. W. Dijkstra. A note on two problems in connexion with graphs. *Numerische Mathematik*, 1:269–271, 1959.
[3] D. Ferguson and A. T. Stentz. Field D*: An interpolation-based path planner and replanner. In *Proceedings of the International Symposium on Robotics Research (ISRR)*, October 2005.
[4] A. F. Foka and P. E. Trahanias. Predictive autonomous robot navigation. In *In Proc. of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS*, pages 490–495, 2002.
[5] A. Galata, N. Johnson, and D. Hogg. Learning variable-length markov models of behavior. *Computer Vision and Image Understanding*, 81(3):398–413, March 2001.
[6] R. E. Kalman and R. S. Bucy. New results in linear filtering and prediction theory. *AMSE Journ. Basic Eng.*, pages 95–108, 1962.
[7] R. MacLachlan. Tracking moving objects from a moving vehicle using a laser scanner. Technical Report CMU-RI-TR-05-07, Robotics Institute, Carnegie Mellon University, Pittsburgh, PA, June 2005.
[8] R. Madhavan and C. Schlenoff. Moving object prediction for off-road autonomous navigation. In *SPIE Aerosense Conference*, 2003.
[9] C. Mertz. A 2d collision warning framework based on a monte carlo approach. In *Proceedings of ITS America's 14th Annual Meeting and Exposition*, April 2004.
[10] S. Petti and T. Fraichard. Safe motion planning in dynamic environments. In *Proc. IROS*, pages 3726–3731, 2005.
[11] C. Schlenoff, R. Madhavan, and T. Barbera. A hierarchical, multi-resolutional moving object prediction approach for autonomous on-road driving. *Robotics and Automation, 2004. Proceedings. ICRA '04. 2004 IEEE International Conference on*, 2:1956–1961 Vol.2, 26-May 1, 2004.
[12] R. S. Sutton and A. G. Barto. *Reinforcement Learning: An Introduction*. The MIT Press, 1998.
[13] S. Tadokoro, M. Hayashi, Y. Manabe, Y. Nakami, and T. Takamori. On motion planning of mobile robots which coexist and cooperate with human. *Intelligent Robots and Systems, IEEE/RSJ International Conference on*, 2:2518, 1995.
[14] S. Thrun, W. Burgard, and D. Fox. *Probabilistic Robotics (Intelligent Robotics and Autonomous Agents)*. The MIT Press, September 2005.
[15] D. A. Vasquez Govea, T. Fraichard, O. Aycard, and C. Laugier. Intentional motion on-line learning and prediction. In *Proc. of the Int. Conf. on Field and Service Robotics*, Port Douglas (AU), July 2005.
[16] B. D. Ziebart, A. Maas, J. A. Bagnell, and A. K. Dey. Maximum entropy inverse reinforcement learning. In *Proceeding of AAAI 2008*, July 2008.

---

[4]This rate is within our implementation's real-time capabilities.