# Predictive Planning for a Mobile Robot in Human Environments

Andrey Rudenko[1], Luigi Palmieri[1,2] and Kai O. Arras[2]

***Abstract*— Generating short and safe paths for mobile robots operating in crowded spaces is a hard task due to the high uncertainty of each person's future behaviour. Classical path planning approaches often result in an over-constrained or overly cautious robot that fails to produce a feasible, safe path in the crowd, or plans a large, sub-optimal detour to avoid people in the scene. This work addresses these issues by exploiting long-term predictions of humans' activities to plan short and safe paths for mobile robots in social spaces. We introduce a Markov Decision Process based motion prediction approach, which extends the baseline works by better adapting online to the observation of the pedestrian speed (*policy cutting technique*) and by introducing a fast random-walk based method (*stochastic policy sampling*) to generate predictions. Moreover, differently from the baselines, we evaluate several ways to incorporate predictions in path planning algorithms, and choose the most promising one. Through an extensive evaluation, using both simulated and real-world datasets, we show that our approach can accurately predict human motion and improve the quality of robot's path planning.**

## I. Introduction

Real-time path and motion planning in crowded and dynamic environments is a well-studied, but still an open problem even in low dimensional configuration spaces. Existing works are unable to cope with dynamic situations in real time and to generate smooth, natural motion without causing inconveniences or hindrances to humans in the scene. The difficulty of the task arises due to high uncertainty of each person's future behaviour. Common problems that motion planning methods face in very complex, dynamic environments include *freezing* and *dancing* of the robot. Being involved in a *replanning* architecture, the path planning method may return a solution that is noticeably distinct from the optimal solution of the previous replanning iteration. In general this prevents smooth transition on-the-fly from the previous plan, sometimes making the robot stop and turn on its place, preparing to execute the new optimal plan. As the complexity of the environment increases, the robot fails to traverse significantly any of the current plans, before it becomes invalid and replaced by the updated replanned path. This problem is referred to as the *"dancing robot"* problem. The limiting case of the *dancing* robot problem is the state of a *"freezing"* robot [1]: the robot is too "polite" to move anywhere and it is therefore blocked into the crowd.

Motion and path planning methods typically rely on a static grid map of obstacles, updated with the current positions of dynamic objects in a periodical replanning fashion. We anticipate that using information about temporal evolution of the map in the future might improve the robot's motion planning and reasoning of its own path, in a way similar to what humans do naturally and intuitively.

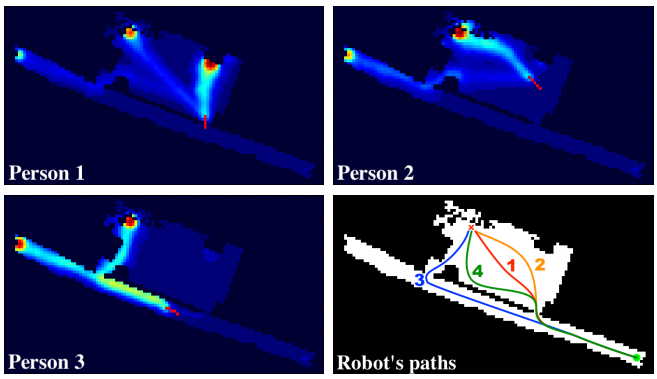


Fig. 1. Predictive planning in the *Social Robotics Laboratory* scenario. There are three people detected in the environment, individual predictions are shown in **top row** and **bottom left**. Observed tracks are depicted in **red**, predicted future trajectory is represented with a **heatmap** (warmer colors correspond to higher probability of visiting a location). **Bottom right:** the robot is located in the top left corner of the room, its goal is in the bottom right corridor. Using the *Inferring Collision Points* predictive planning algorithm, the robot iteratively plans three paths (in **red**, **orange** and **blue**), before it finds the **green** path with no predicted collisions.

In this work we enhance classical path planning techniques with long-term predictions of human trajectories. To this end we present our own method for predicting future locations of humans based on the works of Ziebart *et al.* [2] and Karasev *et al.* [3], review and compare several promising techniques for exploiting the forecast in a path planner [4], [2], [5]. Our prediction method is based on the assumption that people essentially behave like planners by finding an optimal way through the environment. To generate purposeful motion predictions we use a goal-directing motion strategy obtained from solving a Markov Decision Process (MDP) problem. Having estimated the state of the person at the future time moment, we exploit this information in a time-dependent costmap of the environment for predictive planning.

Through an extensive evaluation, using both simulated and real-world recorded human trajectories, we show that our approach can accurately predict human motion and improve the quality of robot's path planning. Following an initial plan of better quality, the *dancing* and *freezing* robot problems are reduced, contributing to more stable and efficient behaviour in social spaces.

We claim the following contributions in this work:

- A simple and efficient method for predicting long-term future trajectories of humans based on [2] and [3].
- Evaluation of several promising methods for incorporating predictions in path planning algorithms.

[1]A. Rudenko and L. Palmieri are with the University of Freiburg, Germany andrey.rudenko@saturn.uni-freiburg.de, palmieri@informatik.uni-freiburg.de

[2]L. Palmieri and K. O. Arras are with Bosch Corporate Research, Stuttgart, Germany, luigi.palmieri, kaioliver.arras@de.bosch.com

- Extensive evaluation with appropriate metrics to show the effectiveness of predictive planning approach, compared to standard non-informed planning methods.

The paper is structured as follows: in Sec. II we review the related works and provide the motivation for our approach, presented in Sec. III. Experiments with the new method and its performance evaluation are given in Sec. IV, followed by the results in Sec. V and the discussion in Sec. VI.

## II. Related Work

Learning typical patterns to model human motions is a widely used method for predicting how people move in known environments. Bennewitz *et al.* [4] propose a technique for learning motion patterns and environment-specific locations that people might be interested in. The method learns the typical patterns from sample trajectories by using Expectation Maximization and it adopts Hidden Markov Models to maintain a belief about the positions of persons. Motion patterns learning methods often suffer from implicit drawbacks: they fail to predict new trajectories unseen in the training set; have poor adaptivity to changes in environment's configuration (predicting trajectories through closed doors) and do not generalize to other environments.

Formalization of social interaction is another approach for predicting future behaviour of humans. Trautman *et al.* [1], making the assumption that people are involved in natural "joint collision avoidance", propose to jointly estimate most likely future trajectories of the humans and the robot with a nonparametric statistical model based on Gaussian processes. A similar approach is used by Kuderer *et al.* [6]: unlike [1], they propose to learn typical human navigation behaviour from a set of sample trajectories. Joint trajectory estimation methods assume cooperation between agents. In reality agents often fail to demonstrate the optimal joint behaviour, predicted by the methods. Additionally, Kuderer's method needs a large learning set to adequately generalize human motion behaviour. It is unclear whether it scales well with the number of humans, given that all topological variants of the scenario are considered.

Rehder *et al.* [7] present a method for probabilistic goal-directed pedestrian prediction, that uses a grid representation of the estimated probability distribution. Dynamics of the pedestrian and environment-based constraints are handled by convolving the probability grid with the dynamics-based transition kernel. Presented results suggest predictive performance on the level of a simple Kalman Filter, only on short-term prediction horizons (up to 2.5 seconds) and with straight ground-truth paths in obstacle-free space. It is not clear whether the method can deal with cluttered environments without degrading into a uniform probability distribution.

Recently, MDP based approaches for predicting human behaviour have received a notable amount of attention. Ziebart *et al.* [2] model the goal-directed trajectories of pedestrians using the maximum entropy inverse optimal control. A soft-maximum version of the MDP, that accounts for alternative ways to reach the goal, is then used to predict future trajectories. The learned feature-based cost function generalizes to changes in the obstacles placement and new environments, described with the same features (obstacles locations). Obtained predictions are used in a novel incremental predictive planner that simulates the predictions forward in time and finds possible points of collision with the planned robot's path.

Kitani *et al.* [8] extend [2] to handle noisy tracker observations and include vision-based physical scene features. Furthermore, Karasev *et al.* [3] provide an interpretation of models from [2], [8] as jump-Markov processes with the goal represented by a hidden variable. Agents' behaviour is interpreted as switching nonlinear dynamical systems, with the latent goal variable governing the switches and the policy describing the nonlinear motion dynamics. Environmental constraints and biases are included as an explicit, hand-crafted semantic map of the environment. Vasquez [9] extends the MDP-based approach by enabling any cost-to-go planning algorithm to represent the uncertainty related to human motion instead of the value function of MDP. The use of a velocity-dependent probabilistic motion model allows estimation of the agent's future position for any given time moment. A gradient-based goal prediction approach, which does not rely on filtering, makes the method capable of quickly recognizing intended destination changes.

Our MDP-based method for predicting human motion is inspired by the works of Karasev *et al.* [3] and Ziebart *et al.* [2]. The approach in [2] assumes a fixed goal and constant-length steps - it does not adapt to the observations of the pedestrian speed. We deal with this limitation using a novel *policy cutting technique* and an efficient *stochastic policy sampling algorithm* based on random walks. For goal inference we adopt an observed track processing strategy similar to [9]. Similarly to [3], we dispense with the IRL method for learning the cost function that humans are assumed to optimize: results from [2] suggest that the learned cost function represents essentially the blurred map of obstacles, i.e. in reality humans are optimizing euclidean length. Moreover, defining a semantic map for a uniform terrain, e.g. indoor environment, is not as straightforward, as it is for an urban scenario [8], [3]. Our prediction method does not need a set of sample trajectories. It provides very general and flexible estimation of the person's future position based on environment-constrained, goal-directed transition modeling, that takes dynamics of the person into account. The resulting probability distribution in the form of predicted occupancy map allows seamless integration with the standard motion planning techniques. Finally, the simple implementation and clear modular structure gives room for the future improvement of method's components.

## III. Our Approach

This section presents our human motion prediction technique and different methods to incorporate the latter into classical path planning algorithms.

### A. MDP-based Predictions

Markov Decision Processes (MDP), that provide a mathematical framework for modeling the decision making problem for a *discrete time stochastic control process*, are well

suited for modeling the path planning task. Formally, the MDP is described with a tuple $\langle \mathcal{S}, \mathcal{A}, \mathcal{T}, \mathcal{R}, \gamma \rangle$ where $\mathcal{S}$ and $\mathcal{A}$ are finite sets of agent's *states* and *actions* respectively. The *transition function* $\mathcal{T}(s, s', a)$ defines the probability of getting to state $s'$ from state $s$ when executing the action $a$. The *reward function* $\mathcal{R}(s, a)$ specifies the immediate reward gained for taking the action $a$ in state $s$. The discount factor $\gamma$ controls the importance of future rewards, relative to immediate rewards. The agent's *policy* $\pi : \mathcal{S} \to \mathcal{A}$, defines the action the agent should take in each state. The *optimal policy* $\pi^*$, which maximizes the cumulative expected future rewards (Eq. 2), is obtained alongside with the state $V^*(s)$ and action $Q^*(s, a)$ *values* by solving the recursive Bellman equations (Eq. 1), e.g. using the *value iteration algorithm*.

$$\begin{cases} Q^*(s,a) = \mathcal{R}(s,a) + \gamma \sum_{s'} \mathcal{T}(s,s',a) V^*(s') \\ V^*(s) = \max_a Q^*(s,a) \end{cases} \tag{1}$$

$$\pi^*(s) = \arg\max_a Q^*(s,a) \tag{2}$$

*1) Our Representation:* The state of the person in our MDP is represented with the 2D $(x, y)$ Cartesian coordinates, which essentially means that the agent's action only depends on the current position. The goal-directed motion of the person is encoded by assigning negative rewards to all states and actions except for the goal state, which only has one self-transitioning action with zero reward (called *the absorbing zero* state). For modeling the action space, we assume that the human motion is unconstrained in terms of orientation and acceleration. Therefore we allow motion in any direction $\theta$ with any speed $v$ below the speed limit $v_{max}$ - a large value that exceeds the expected speed of people in particular environment by a safety margin. Handling the observed person's individual speed is discussed later in Sec. III-A.4. We describe the action space with the $\langle \theta, v \rangle$ orientation-velocity pair, which reads as "making a move in direction $\theta$ with velocity $v$". We assume deterministic action outcome, i.e. $\forall s \in \mathcal{S},\ \forall a \in \mathcal{A}\ \exists! s' : s \xrightarrow{a} s'$. This assumption, common with [2] and [3], implies that humans accurately execute desired actions, i.e. know where they are going. For convenience we denote the transition function with deterministic action outcomes as $\mathcal{T}(s, a) : \mathcal{S} \times \mathcal{A} \to \mathcal{S}$. For the origin state $s = (s_x, s_y)$ and action $a = (\theta, v)$, transition $s \xrightarrow{a} s'$ is calculated as $s'_x = s_x + v\cos(\theta),\ s'_y = s_y + v\sin(\theta)$.

In a fashion similar to probabilistic localization, we frame the task of predicting a person's future location as estimating the probability $p(s|t)$ that the person will be in state $s$ at the time moment $t$, $\forall s_i \in \mathcal{S},\ t_0 < t < T$, where $t_0$ is the current time and $T$ is the prediction horizon. We make the assumption that the person stays in $\mathcal{S}$. As we use a 2D *map* of the environment, the estimated value $p(s|t = t_i)$ is a probability distribution over the map. A *static costmap* $C(s)$ carries the unitary cost of each state, which is set to $\infty$ for occupied cells and to a small value $\epsilon > 0$ for free states.

In the known environment we assume that the possible *goal states* (motion destinations) are known a-priori. Using the set of goals, denoted as $\mathbf{G}$, we prepare and solve $|\mathbf{G}|$ MDP problems with the goal-dependent reward function $\mathcal{R}_g(s, a)$ and acquire the unique optimal policy $\pi^*_g$ for each goal, as well as the goal-dependent $V^*_g(s)$ and $Q^*_g(s, a)$ value functions. $\mathcal{R}_g(s, a)$ is constructed as follows:

$$\mathcal{R}_g(s,a) = \begin{cases} -(w_1 C(s') + w_2 ||s - s'|| + w_3 ||s' - g||), & \text{if } s \neq g \\ 0, & \text{otherwise} \end{cases} \tag{3}$$

where $s' = \mathcal{T}(s, a)$ and $w_1, w_2, w_3 > 0$ control the relative importance of each component: the unitary cost of $s'$, Euclidean distance $||\cdot||$ covered with the action $a$ and estimated distance to the goal to encourage the agent to perform faster actions with larger $v$. Since the reward function is negative everywhere except the goal state, the MDP can be solved with $\gamma = 1$ in Bellman equations [10]. Therefore the $V^*_g(s)$ value of a state is actually the *cost-to-go* from $s$ to $g$.

To predict also alternative paths to the goal and to allow deviations from the optimal policy, we relax the obtained $\pi^*_g$ with the *stochastic Boltzmann policy* (Eq. 4) that assigns to each action $a$ probability to be executed in a particular state $s$ proportional to its value $Q^*_g(s, a)$. Parameter $\alpha$ controls the level of stochasticity, i.e. to what extent sub-optimal actions are considered. We denote the stochastic policy as $\pi_g$.

$$a \sim \pi_g(s) \text{ with prob. } \propto \exp(\alpha(Q^*_g(s,a) - V^*_g(s))) \tag{4}$$

*2) Stochastic Policy Sampling Using Random Walks:* To make predictions using the stochastic policy $\pi_g$, we propose a simple random walk based algorithm (Alg. 1) that samples $K$ paths, each starting in the initial state $s_0$. The position of the particle at each time $t$ is saved in the corresponding time layer $OC_t$ of the occupancy map ($OC$), that is shared among $K$ iterations of the random walk. Each layer is then normalized to properly represent the probability distribution $p(s|t, g)$ of the person's possible location at time $t$. Illustration to the Random Walk Sampling process is given in Fig. 2.

*3) Goal Inference:* Having observed the person moving in the environment, we predict the final destination based on the observed trajectory $\zeta = \langle s(t_0), s(t_1), ..., s(t_i) \rangle$ up to the

**Algorithm 1** Random Walk Stochastic Policy Sampling

**function** SPolicySampling($s_0, \pi_g(s), K, T$)
**for** $i = 1, ..., T$ **do**
  $OC_i \leftarrow$ `zeros`($|\mathcal{S}|$)
**end for**
**for** $k = 1, ..., K$ **do**
  $s_n \leftarrow s_0$
  $t \leftarrow 0$
  **while** $s_n \neq g$ **and** $t < T$ **do**
    $t \leftarrow t + 1$
    **repeat**
      $a \leftarrow$ `sample`($\pi_g(s_n)$)
      $s_{n+1} \leftarrow \mathcal{T}(s_n,\ a)$
    **until** `lineOfSight`($s_n,\ s_{n+1}$)
    $OC_t(s_{n+1}) \leftarrow OC_t(s_{n+1}) + 1$
    $s_n \leftarrow s_{n+1}$
  **end while**
**end for**
**for** $i = 1, ..., T$ **do**
  $OC_i \leftarrow$ `normalize`($OC_i$)
**end for**
**return** $OC$

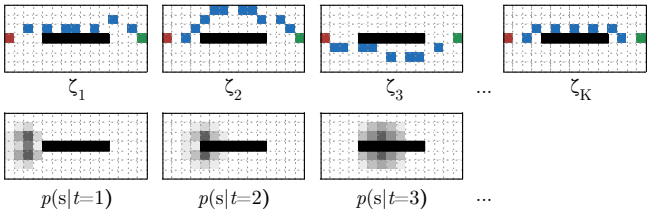


Fig. 2. Example of the random walk stochastic policy sampling. For each of the $K$ sampled paths $\zeta_1 ... \zeta_K$, location at time $t_i$ is added to the corresponding time layer $p(s|t = t_i)$, which is then properly normalized. The path is depicted in **blue**, with the **red** starting point $s_0$ and **green** goal point. Probability distribution is represented with the **shades of gray**, darker areas mean higher probabilities.

current time $t_i$. Similarly to [2] and [9], for each goal $g \in \mathbf{G}$ we estimate the gradient of the value function $V_g^*(s)$ along $\zeta$ as a difference between values at $t_0$ and $t_i$ using a *softmax function*:

$$p(g) \;\propto\; \exp\Big(\tau(V_g^*(s(t_i)) - V_g^*(s(t_1)))\Big) \tag{5}$$

Parameter $\tau$ defines to what extent alternative, less likely goals are considered. Goals' probabilities are combined with the goal-specific prediction, obtained with Alg. 1, to get the final estimation of person's location at time $t$:

$$p(s|t) = \sum_{g \in \mathbf{G}} p(s|t,g)p(g) \tag{6}$$

*4) Policy Cutting For Preferred Speed Selection:* So far we have trained a policy that allows actions up to a very large $v_{max}$. In reality humans are moving with certain *preferred speed*, which is usually less than $v_{max}$. In absence of other cues, it is natural to predict that the human, observed at $v_{obs}$, will continue moving with the same speed. One possible way to handle the observed person's speed correctly is to solve an appropriate individual MDP problem with $v_{max} = v_{obs}$. This approach suffers from two drawbacks: firstly, increased computational burden during online operation and secondly, lack of ability to predict motion faster than $v_{obs}$. Instead, we propose to solve an MDP problem with large $v_{max}$ to obtain $\pi_g$, applicable to all people, and use a simple *policy cutting* technique to incorporate the information about $v_{obs}$ into our prediction algorithm. We redefine the action space for the observed person as $\hat{\mathcal{A}}(v_{obs}) = \langle \theta, v \rangle$ with $\theta \in [0, 2\pi)$ and $v \in (0, 2v_{obs}]$. The policy $\hat{\pi}_g$ is then defined as

$$a = \langle \theta, v \rangle \sim \hat{\pi}_g(s) \text{ with prob. } \propto \begin{cases} \pi_g(\langle \theta, v \rangle), \text{ if } v \le v_{obs}, \\ \pi_g(\langle \theta, 2v_{obs} - v \rangle), \text{ if } v > v_{obs} \end{cases} \tag{7}$$

Intuitively, we assign the probability of faster actions with $v > v_{obs}$ the same as for the symmetrically slower action with $v < v_{obs}$. The original policy $\pi_g$ is "cut" at the point of $v_{obs}$ and "mirrored" backwards, hence the name *policy cutting*. Using the updated $\hat{\pi}_g$ as an input to Alg. 1, we get predictions for each person at the corresponding speed $v_{obs}$.

Alg. 2 summarizes our method for prediction. Its inputs are: the occupancy map $M$ of the environment, set of goals $\mathbf{G}$, observed trajectory $\zeta$. The algorithm has the following parameters: prediction stochasticity $\alpha$, goal uncertainty $\tau$, prediction horizon $T$, and $K$ samples for the stochastic policy sampling. To speed-up the value iteration convergence, we process the states in the order of increasing distance to the goal: convergence is still guarantied in this case [11]. Example of predictions obtained with our method is given in Fig. 1.

**Algorithm 2** Motion Prediction

**function** MotionPrediction($M, \zeta, K, T, \mathbf{G}, \epsilon$)
**for all** $g \in \mathbf{G}$ **do**
  compute $\mathcal{R}_g(s,a)$ as in Eq. 3
  $V_g^*,\ Q_g^*,\ \pi_g^* \leftarrow$ `ValueIteration`$(\mathcal{R}_g(s,a), \epsilon)$
  compute $\pi_g$ as in Eq. 4
**end for**
**for all** $g \in \mathbf{G}$ **do**
  compute $p(g)$ as in Eq. 5
  compute $\hat{\pi}_g$ as in Eq. 7
  $p(s|t,g) \leftarrow$ `SPolicySampling`$(\zeta_{end}, \hat{\pi}_g, K, T)$
**end for**
compute $p(s|t)$ as in Eq. 6
**return** $p(s|t)$

### B. Robot Motion Planning Using Predictions

The general form $p(s|t)$ of the obtained solution to the prediction problem allows simple integration with standard motion planning techniques, such as A* or RRT. The key idea is to generate a *temporal costmap* of the environment that would penalize presence of the robot in the regions probably occupied by humans at that moment. In the following we detail the temporal costmap construction and review several discrete-search based algorithms for predictive planning.

If there are $N$ people present in the environment, we run $N$ times Alg. 2 to obtain prediction $p_i(s|t)$ for each person $i$ individually. These predictions are used to prepare a 3D spatio-temporal costmap ($C_{3D}$) with $T$ future time layers:

$$C_{3D}(t_j) = c_p \sum_{i=1}^{N} p_i(s|t = t_j),\ j = 1...T \tag{8}$$

where $c_p$ is the cost of the person. The $C_{3D}$ is then augmented with appropriate costmap of static obstacles $C_{SO}$.

*1) Spatio-temporal Planning:* The most straightforward way to plan the robot path using $C_{3D}$ is to perform the discrete search (e.g. A*) in $x$,$y$ and $t$ dimensions (e.g. used in [4]). The costmap $C_{DS}$ for spatio-temporal planning is constructed as follows:

$$C_{DS}(t_i) = \begin{cases} C_{SO} + C_{3D}(t_i), \text{ if } i <= T, \\ C_{SO}, \text{ otherwise} \end{cases} \tag{9}$$

While delivering the optimal solution given the available prediction, the 3D A* suffers from increased runtime due to the extra planning dimension. Therefore we consider two alternative approaches: the *Costmap Inflation* and *Inferring Collision Points (ICP)*.

*2) Costmap Inflation:* It is a simple approach to incorporate human motion predictions in the robot motion planning, suggested by Bai *et al.* [5]. Authors propose to run A* planning on the 2D costmap of the environment ($C_I$), augmented with people trajectories predictions, integrated

**Algorithm 3** Inferring Collision Points

1: **function** ICP($s_{start}$, $s_{goal}$, $C_{SO}$, $C_{3D}$, $thrs$, $Z$, $\beta$, $\sigma$)
2: $C \leftarrow C_{SO}$
3: $R_P \leftarrow \emptyset$
4: **for** $i = 1, ..., Z$ **do**
5: $\langle P,\ c\rangle \leftarrow$ DescreteSearch($s_{start}$, $s_{goal}$, $C$)
6: $R_P \leftarrow R_P \cup \langle P,\ c\rangle$
7: $n \leftarrow 0$
8: **for all** $\langle p, t\rangle \in P$ **do**
9: **if** $C_{3D}(p,\ t) > thrs$ **then**
10: $C \leftarrow C + \beta \cdot \mathcal{N}(p, \sigma)$
11: $n \leftarrow n + 1$
12: **end if**
13: **end for**
14: **if** $n = 0$ **then**
15: **break**
16: **end if**
17: **end for**
18: $P_{min} \leftarrow \min_c R_P$
19: **return** $P_{min}$

over a short time period as follows:

$$C_I = C_{SO} + \sum_{i=1}^{T} \lambda^i \cdot C_{3D}(t_i) \tag{10}$$

where $\lambda \in (0, 1]$ is a discounting factor for $C_{3D}$ layers, corresponding to more distant future. Intuitively, the robot avoids regions of the space currently occupied by humans, but is more confident to "cross ways" with people in more distant future.

*3) Inferring Collision Points (ICP):* Another promising approach, presented in [2], provides a balanced trade-off between the solution's quality of the full spatio-temporal planning and the fast operation of the time-independent planning approach. The method, detailed in Alg. 3, essentially iteratively shapes a time-independent navigational cost function to remove known points of hindrance. It is initialized with the static costmap ($C_{SO}$). At each iteration the discrete-search method solution under the current costmap $C$ is obtained (line 5 of Alg. 3) and simulated forward in time to predict the points of possible interference with people, based on the probability of hindrance at each location (lines 8-13). Adding cost (a Gaussian $\mathcal{N}$ with variance $\sigma$ and magnitude $\beta$, centered in the point of collision $p$) to those regions of the map prevents the subsequent plans ($P$ with cost $c$) from visiting those locations. The "planning - map update" cycle is iterated $Z$ times, or until a path with no collision is found. Example of predictive planning with the ICP algorithm is given in Fig. 1.

## IV. Experiments

In this section we present a series of experiments, conducted to evaluate our prediction and predictive planning methods. All algorithms are implemented in Matlab, running on a laptop with 2.3 GHz Intel Core i5 and 4 GB of RAM.

Values of $w_1, w_2, w_3$ in the reward function $\mathcal{R}(s, a)$ are informally estimated to match the expected behaviour of the pedestrian: $w_1 = 1, w_2 = 1, w_3 = 0.01$ and the cost of the free space is $\epsilon = 10^{-10}$. Action space parameters are set as follows: angular discretization of $\theta$ is $\pi/10$ for all simulated scenarios and $\pi/20$ for the *Edinburgh* scenario; translational discretization of $v$ is 0.13 $m/s$, $v \in [0.22, 4.95]$ $m/s$. Physical space per cell discretization is 0.1 $m$. Temporal discretization of predictions is 4.5 $Hz$. Prediction horizon $T$ for each experiment is specified if relevant, otherwise prediction in Alg. 1 is made until the goal state is reached. Number of random walk samples $K$ is typically 50-100 for each goal.

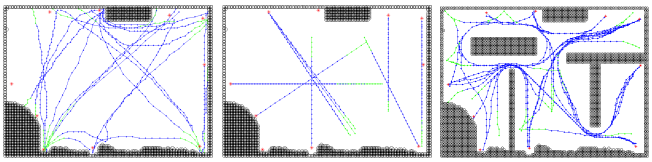

Fig. 3. *Edinburgh* scenario. **Left:** *Testset 1* contains 22 recorded trajectories. **Middle:** 16 simulated trajectories in the *Testset 2*. **Right:** *Edinburgh with obstacles* scenario with 34 simulated trajectories in the *Testset 3*.

### A. Experiment 1: Prediction Evaluation

The first experiment aims to evaluate the predictive ability of our approach and test how well it can predict the future trajectories of humans. For this purpose we have selected 22 diverse trajectories (*Testset 1*) from the Edinburgh[1] dataset, with different starting points, motion speed and destinations, both accounted and unaccounted for. Additionally we have prepared *Testset 2* with 16 hand-crafted trajectories of constant velocity. Finally, to test the ability to predict non-straight paths, we have added several large obstacles to the map of Edinburgh campus and prepared 34 trajectories in this setting (*Testset 3*). Beginning of each trajectory was used as the observed track, and the rest as the ground truth for evaluation. Test sets are presented in Fig. 3.

We evaluate the predictive performance of the algorithm based on the following metrics:

- *Negative Log-Likelihood* (NLL) [9] of the path is a measure of the probability that it is generated by the stochastic policy. Lower value corresponds to higher probability of path generation.
- *Modified Hausdorff Distance* (MHD) [8] is a geometric measure of distance between the ground truth path and the most probable path in the stochastic policy. Lower value corresponds to smaller distance between paths.

Optimal values of $\alpha = 100$ and $\tau = 5$ are determined prior to the main experiment using the paths from *Testset 1*. Next, we evaluate predictions obtained by our method based on the NLL and MHD metrics for all three testsets. We compare the predictive performance of our approach to two baselines: the *Constant linear velocity* and *Random walk* predictions. Constant linear velocity assumes that the person will continue moving in the same direction with average observed speed. Only MHD measure is applied to this non-probabilistic method. Random walk assumes that the person will each step choose the action $a \in \hat{\mathcal{A}}(v_{obs})$ at random. We only calculate the NLL measure for the this method.

In the end we specify the time needed to get predictions in the *Edinburgh* scenario using our algorithm with the following parameters: observation period $OP = 7$, $T = 15$, $K = 50$, $|\mathcal{S}| = 130 \times 94$ states, $|\mathcal{A}| = 40$ orientations $\times$

[1] http://homepages.inf.ed.ac.uk/rbf/FORUMTRACKING/

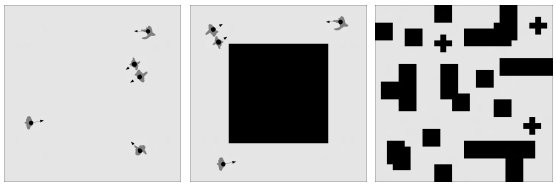

Fig. 4. Scenarios for predictive planning methods comparison. **Left:** *Large empty room*. **Middle:** *Two homotopy classes*. **Right:** *Polygons*.

35 velocities. Additional runtime calculation is made for an action space with coarser discretization $|\mathcal{A}| = 20 \times 18$.

### B. Experiment 2: Planning Methods Comparison

In the second experiment we compare the three predictive path planning methods from Sec. III-B. To this end we use three scenarios of increasing complexity: *Large empty room* with no obstacles, *Two homotopy classes* with a large obstacle in the middle, and *Polygons* cluttered with many polygonal-shaped obstacles. The size of each scenario is $8 \times 8$ $m^2$, discretized into $40 \times 40$ cells gridmap. For each scenario we run 100 randomized trials and plot them on the $X$ axis in 3 separate plots, one for each evaluation criteria. We consider a trial to be "complex" if the ICP method performed at least one iteration of the "planning - map update" cycle. Such trials are marked with a black $*$ sign on the $X$ axis.

We evaluate the paths returned by the three predictive planning methods based on the following metrics: *Euclidean length* of the path, its *social cost*, computed as the sum of $C_{3D}$ values along the path, and the algorithm's *runtime*. Each run of the experiment featured 4-8 people, number chosen at random. People's and robot's start and goal points were also drawn randomly, but spatially well separated, to avoid non-representative experiments where no human-robot interaction occurs. For each person $i$ an A* path $\zeta_i = \langle s(t_0), s(t_1), ..., s(t_{|\zeta_i|}) \rangle$ was computed, position $s(t_k)$ was blurred to imitate uncertain predictions $p_i(s|t = t_k)$.

Parameters used in this experiment: $c_p = 2$ in the $C_{3D}$ (Eq. 8); $\lambda = 0.8$ in the Costmap Inflation method; $thrs = 0.3$, $Z = 10$, $\beta = 0.7$, $\sigma = 0.15$ in ICP.

### C. Experiment 3: Planning Evaluation

In the third experiment we evaluate the solution quality of the ICP predictive planning method, compared to a standard socially-uninformed A* planner. For this purpose we use the *Testset 2* from Experiment 1 and an extended version of *Testset 1* that includes 51 recorded trajectories. Each trial featured 4-5 random people from the testset, beginning of the trajectory was used as the observed track for prediction algorithm and the rest served as the ground truth future trajectory for evaluation. Robot's start and goal points were drawn randomly as in Experiment 2. We counted "complex" trials, until the total number of 200 trials was reached. To obtain the prediction input, we used Alg. 2 with observation period $OP = 7$, $T = 15$, $\alpha = 100$ and $\tau = 5$. Following metrics are used for evaluation: *Euclidean length* and *social cost* of the solution, computed using the ground truth trajectory, *minimum* and *average distance to the closest human* the robot reaches during its planned trajectory execution.

| | | T = 5 | T = 10 | T = 20 |
|---|---|---|---|---|
| *Testset 1* | | | | |
| NLL | Our approach | **10.78** ± 3.52 | **22.27** ± 6.09 | **47.63** ± 11.6 |
| | Random walk | 18.70 ± 2.56 | 37.03 ± 4.84 | 73.63 ± 10.7 |
| MHD | Our approach | **2.69** ± 1.37 | **5.59** ± 3.37 | **10.74** ± 7.99 |
| | Linear | 3.43 ± 1.87 | 6.98 ± 4.33 | 13.62 ± 10.2 |
| *Testset 2* | | | | |
| NLL | Our approach | **8.12** ± 1.44 | **17.34** ± 3.59 | **34.00** ± 7.04 |
| | Random walk | 22.83 ± 5.34 | 44.57 ± 9.80 | 76.70 ± 13.3 |
| MHD | Our approach | **0.15** ± 0.42 | **0.16** ± 0.44 | **0.27** ± 0.55 |
| | Linear | 0.29 ± 0.67 | 0.52 ± 1.16 | 1.36 ± 2.76 |
| *Testset 3* | | | | |
| NLL | Our approach | **17.47** ± 6.10 | **36.18** ± 9.78 | **75.48** ± 18.1 |
| | Random walk | 25.07 ± 1.70 | 49.59 ± 2.15 | 98.30 ± 4.19 |
| MHD | Our approach | 4.43 ± 1.74 | **8.85** ± 4.49 | **25.72** ± 15.2 |
| | Linear | **4.27** ± 2.75 | 13.86 ± 6.24 | 43.99 ± 21.2 |

TABLE I
PREDICTION EVALUATION RESULTS

## V. RESULTS

Table I summarizes results of the first experiment for OP=5. Similar results are present for other values of observation period, which means that we need only a short observation ($\sim 1.1$ sec.) to correctly infer the person's intention. In the *Testset 2* with 16 simulated trajectories our method expectedly delivers much better NLL results, compared to a random walk, while the MHD results are very high for both methods, as the trajectories in the set are straight. Similar results are seen for the real trajectories from *Testset 1*. Results for the scenario with obstacles (*Testset 3*) confirm, that our method performs considerably better than the random walk predictions, and also delivers an improvement over linear constant velocity predictions for longer prediction horizons.

Runtimes of our algorithm using the action space $|\mathcal{A}| =$ 40 orientations $\times$ 35 velocities are as follows: *value iteration* (lines 3 and 4 of Alg. 2) takes $\approx 500$ seconds for each goal; *stochastic policy computation* (line 5 of Alg. 2) takes $\approx 261$ seconds for each goal; *stochastic policy sampling* (lines 7-11 of Alg. 2) takes $\approx 3.9$ seconds for each person, considering 10 goals in the environment. For the coarser discretization of $|\mathcal{A}| = 20 \times 18$, the execution times are $\approx 130$, 67.6 and 3.7 seconds respectively.

Result of the second experiment in the *Two homotopy classes* scenario are presented in Fig. 5, with similar trends present in the other two testing environments. The 3D A*, while delivering the optimal solution, requires a considerably longer runtime. The Costmap Inflation method, on the other hand, quickly delivers solution, but fails to reach high quality in many trials. The ICP method provides a very attractive trade-off between runtime and solution's quality: path length and cost are optimal in most cases and the computational load is greatly decreased as compared to 3D A*, on par with the Costmap Inflation method.

Results of the third experiment are presented in Table II. The predictive planner, which uses the ICP method, finds a much better trajectory of the same length, compared to the uninformed A* planner (social cost of the solution is lower). Additionally the robot, equipped with the predictive planner, on average keeps higher distance from the closest human, which is especially prominent if the motion of humans is more predictable, as seen in the case of simulated trajectories.

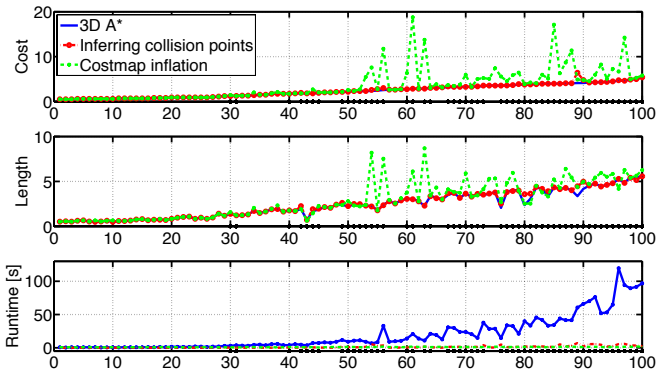


Fig. 5. Predictive planning methods comparison in the *Two homotopy classes* scenario. Same trends are visible in the other scenarios.

| | ICP | Uninformed A* |
|---|---|---|
| *Extended testset 1* | | |
| Social cost | **6.4** ±7.58 | 8.42 ±10.8 |
| Euclidean length | 10.79±2.11 | **10.45** ±2.10 |
| Min. distance to a human | **8.58** ±7.06 | 7.8 ±7.20 |
| Avg. distance to a human | **23.81**±9.38 | 23.33 ±9.31 |
| *Testset 2* | | |
| Social cost | **3.38** ±4.04 | 5.93 ±6.31 |
| Euclidean length | 10.76±2.03 | **10.51** ±2.08 |
| Min. distance to a human | **10.72**±5.63 | 7.58 ±5.82 |
| Avg. distance to a human | **24.62**±9.48 | 23.13 ±9.73 |

TABLE II

PREDICTIVE PLANNING EVALUATION RESULTS

## VI. CONCLUSIONS

In this work we present a method for predicting human trajectories and exploiting predictions to improve the robot path planning in social spaces. For prediction, we model purposeful motion as a goal-directing strategy obtained from solving an MDP problem. Uncertainty in a person's motions is modeled via *stochastic policy*, future locations are estimated with a novel *random walk stochastic policy sampling* algorithm and intention is derived from the observed trajectory. We exploit obtained predictions to build a time-dependent costmap of the free space, that is used for predictive planning. We review three particular discrete-search based solutions for predictive planning and conclude that the ICP algorithm delivers a good trade-off between runtime and solution's quality.

An important benefit of our method is its simplicity: for deployment one needs only a grid-map of obstacles, a value iteration algorithm and a discrete search algorithm (e.g. A*) implementation. Our approach is highly modular: it is possible to update individual components to achieve better performance. The *Policy cutting technique* is another novel aspect of our solution: we solve the MDP problem only once and adapt the obtained policy to the observed speed of each person. Experimental results show that the adapted policy accurately predicts trajectories of people, for a variety of currently observed speeds. Prediction method's runtime scales linearly with the number of observed people and, once the spatio-temporal costmap is built, the ICP runtime is independent of the number of people. Furthermore, our experiments with the replanning framework suggest that the "dancing" problem is reduced. A good, predictive initial plan limits the effect of replanning to small, incremental changes in the path, avoiding large jumps to a plan that belongs e.g. to a different homotopy class.

The workload of the value iteration algorithm and stochastic policy estimation includes processing each action in each state, ($\sim 10^7$ operations in the *Edinburgh* scenario with $94\times 130$ states $\times 40\times 35$ actions). For faster online performance it is possible to pre-compute stochastic policies (lines 2-6 of Alg. 2) and only perform the sampling operations (lines 7-12) online, assuming, of course, that the map of static obstacles does not change. The workload could be considerably reduced when iterating directly over target states $s'$ instead of considering every pair of $\theta$ and $\nu$ from $\mathcal{A}$. In the *Edinburgh* scenario with the given discretizations it would mean a decrease from iterating over $40\times 35 = 1400$ actions to only 253 states, reachable with those actions.

There are several promising points on which the prediction method could be improved. Finding a way to separate angular and translational stochasticity may bring more flexible control of the person's future speed and path uncertainty. Expanding the MDP state-space to include the current orientation $\theta$ may potentially lead to smoother, more precise predictions, obtained with less particles in Alg. 1. The main direction of our future work is a practical C++ implementation of our method, followed by the evaluation in the real-world experiments. We intend to conduct a formal comparison with the state-of-the-art prediction algorithms. An interesting extension of the method would be dealing with joint predictions for all humans in the scene.


## ACKNOWLEDGEMENTS

This work has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 732737 (ILIAD).


## REFERENCES


[1] P. Trautman and A. Krause, "Unfreezing the robot: Navigation in dense, interacting crowds," in *IEEE/RSJ Int. Conf. IROS*, Oct 2010.

[2] B. D. Ziebart, N. Ratliff, G. Gallagher, C. Mertz, K. Peterson, J. A. Bagnell, M. Hebert, A. K. Dey, and S. Srinivasa, "Planning-based prediction for pedestrians," in *IEEE/RSJ International Conference on Intelligent Robots and Systems*, Piscataway, NJ, USA, 2009.

[3] V. Karasev, A. Ayvaci, B. Heisele, and S. Soatto, "Intent-aware long-term prediction of pedestrian motion," in *2016 IEEE International Conference on Robotics and Automation (ICRA)*, May 2016.

[4] M. Bennewitz, W. Burgard, G. Cielniak, and S. Thrun, "Learning motion patterns of people for compliant robot motion," *The International Journal of Robotics Research*, vol. 24, no. 1, 2005.

[5] H. Bai, S. Cai, N. Ye, D. Hsu, and W. S. Lee, "Intention-aware online pomdp planning for autonomous driving in a crowd," in *IEEE Int. Conf. on Robotics and Automation (ICRA)*, May 2015.

[6] M. Kuderer, H. Kretzschmar, C. Sprunk, and W. Burgard, "Feature-based prediction of trajectories for socially compliant navigation," in *Proc. of Robotics: Science and Systems (RSS)*, Sydney, Australia, 2012.

[7] E. Rehder and H. Kloeden, "Goal-directed pedestrian prediction," in *IEEE Int. Conf. on Computer Vision Workshop (ICCVW)*, Dec 2015.

[8] K. M. Kitani, B. D. Ziebart, J. A. Bagnell, and M. Hebert, *Activity Forecasting*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2012.

[9] D. Vasquez, "Novel planning-based algorithms for human motion prediction," in *2016 IEEE Int. Conf. ICRA*, May 2016.

[10] K. Hinderer and K.-H. Waldmann, "Algorithms for countable state markov decision models with an absorbing set," *SIAM journal on control and optimization*, vol. 43, no. 6, 2005.

[11] D. Bertsekas and J. Tsitsiklis, *Parallel and distributed computation: numerical methods*. Prentice hall Englewood Cliffs, NJ, 1989, vol. 23.