Safe Predictive Mobile Robot Navigation in Aware Environments

Michael Arndt[1] and Karsten Berns[1]
[1]*Robotics Research Lab, Dept. of Computer Sciences, University of Kaiserslautern, Kaiserslautern, Germany*
{*arndt, berns*}*@cs.uni-kl.de*

Keywords: Mobile Robot Path Planning, Safety, Human Motion Prediction, Human Aware Planning, Ambient Intelligence, Smart Environment

Abstract: It is a common goal to improve safety and performance of mobile indoor robots by predicting the movements of people in the surroundings. In contrast to many related works which exclusively employ sensors mounted on mobile robots, this work shows a method to achieve this goal in a smart environment where external sensors are used to sense people's positions. By using probabilistic models and filters, the evolution of the environment's state is predicted and optimal paths with respect to safety and performance are planned. Experiments in reality and in a simulation environment show the applicability in real-world scenarios and the advantages over classical path planning approaches.

1 INTRODUCTION

It has been acknowledged by many researchers that safety and human awareness are important topics for mobile robots employed in human environments. As (Alami et al., 2006, sec. I) state, safety is an important key to successfully introduce robots into such domains. The concept of human aware motion planning has been researched a lot in the past (Sisbot et al., 2005; Sisbot et al., 2007; Bennewitz et al., 2005; Guzzi et al., 2013) to achieve robot motion that does interfere with human motion as little as possible.

At the same time, so called *smart*, *aware*, *ambient* or *ubiquitous* technologies are becoming more and more important and present in everyday life (Augusto et al., 2010; Lukowicz et al., 2012) and also the field of robotics tries to make use of that technology (Saffiotti and Broxvall, 2005; Coradeschi and Saffiotti, 2006; Sanfeliu et al., 2008; Amato et al., 2012).

Previous work by the authors has already shown that the risk, a mobile robot imposes on human inhabitants of *smart* environments, can be decreased by making use of environmental sensor information. By assessing the current situation, as perceived by wireless sensor nodes, distributed in the environment, the path planner of a mobile robot is able to find paths that are safe according so a safety metric, but also efficient subject to the length of the path (Arndt and Berns, 2014). However, the previous method yields suboptimal results, as it only evaluates possible paths according to the *current* situation at the time of planning. While it provides good results for setups that remain static after the path has been planned, this assumption is of course not true in general, as environments inhabited by people are typically highly dynamic.

In reality, people frequently change their locations, thus the environments evolve during the traversal of a path of the mobile robot (i.e. while it is driving), thus the environment is *time variant*. This fact will now be taken into account within this work by making use of *prediction*.

When planning with respect to time, the dimensionality of the planning problem increases. This leads to an increase in complexity, which makes the search for exact solutions often unfeasible (Lindemann and LaValle, 2005). However, the method presented later in this work is able to find exact solutions in typical application scenarios with acceptable computation times.

The rest of the paper is structured in the following way: First, work related to this one is introduced in the next section. Afterwards, a brief overview about the probabilistic state estimation methods that are employed, is given, followed by a description of the essence of this work, the prediction of future states and the planning using the outcome. Section 5 describes the experiments that have been conducted in reality as well as in a simulation environment to validate the proposed algorithms in detail. The experimental results are followed by a conclusion and an outlook over possible future works.

2 RELATED WORK

Path planning in time variant, or dynamic environments has been known for quite a while and has already been researched in different domains. This section aims to give an overview about some techniques and algorithms that can be found in the literature.

The work by (Alvarez et al., 2004) has presented a solution for a predictive path planner for autonomous underwater vehicles in ocean environments. Due to the fact that littoral waters are highly variable and routes for the underwater vehicles should be energy efficient, forecasts are used to predict the conditions of the ocean in advance (Alvarez et al., 2004, sec. I) to achieve better plans. In their work, the authors acknowledge the high dimensionality of the temporal planning problem and thus the difficulty of finding computationally feasible exact solutions. Instead, in their work, they make use of genetic algorithms and accept possibly non-optimal solutions.

Regarding mobile robots in indoor environments, the tracking of people (by the robot itself) is an important foundation of planning in general (Bruce and Gordon, 2004) and human aware planning in particular. The authors acknowledge that people usually follow trajectories between points of interest and do not just walk randomly (Brownian motion) or with a constant velocity through the environment. In that work, they have used training data (trajectories of people in the environment) to learn goal locations of people in the environment. Later, in their motion model, they calculate paths from people's current locations to the possible goal locations to be able to predict their motion and improve their tracking, compared to a simple Brownian motion model.

The work by (Bennewitz et al., 2005) has similar goals as the one by Bruce et al. and is also highly related to this work. It describes a way to learn motion patterns of people from laser scanner data and to derive hidden Markov models (HMMs) from the patterns. They later use the HMMs to estimate current as well as also *future* positions of people. The authors evaluated their approach in different settings. Experimental results show that the HMMs can be used to estimate the position of a single person and also multiple persons in the environment, even during times of occlusions (Bennewitz et al., 2005, 5.2.1 - 5.2.2).

Especially related to this work is their experiment to improve the navigation behavior of the robot using the motion patterns (Bennewitz et al., 2005, 5.3). An A^* search is performed for the least-cost path in the time-space configurations of the robot with predicted people trajectories, leading to a significant reduction in the time needed for the robot to perform a navigation task with prediction enabled.

An approach with a two-step prediction method was presented by (Foka and Trahanias, 2010). They describe an approach to enhance the path planning capabilities of a mobile indoor robot by trying to predict future motion of humans in the environment. Their solution makes predictions on two levels. The short-term prediction only tries to predict the position of a sensed human in the next time step. More interesting in the context of this work is the long-term prediction, which works with the help of having defined so-called *hot points* in the environment. Those are points where people will most likely end their paths through the environment, e.g. stairs, elevators or seats. These points can be either manually defined or automatically learned. The connections between tracked people and hot points in their field of view will be interpreted as possible paths they take and those position in the map will be penalized for the robot, so it is less likely that robot and person will meet.

Another very interesting work by (Luber et al., 2010) employs the concept of *social forces* to predict the future actions of people, especially in highly populated environments. The social forces model the behavior of people in crowds and are used to improve predictions during multi-target tracking in their work. The authors have proven the concept by evaluating the number of data association errors in their multi-target tracker during occlusion of targets. Using the social force model instead of just a constant velocity model enabled them to reduce the number of data association errors up to 50%, depending on the scenario. However, in that work, their predictions are only used for improving the tracking (by improving the data association step), and not for the long-term path planning, in contrast to this work.

3 STATE ESTIMATION

The state of the smart environment is estimated using Bayesian filters. The whole process is described in detail in the previous work (Arndt and Berns, 2012b, Section 2), so this section will just give a brief overview about the process and it explains modifications and improvements relevant to this work. For the full details, the reader is referred to the previous work.

People in the environment are detected using AmICA wireless sensor nodes (Wille et al., 2010) equipped with passive infrared (PIR) sensors. These sensor-events are fed to a probabilistic filter which tracks the positions of people in the environment.

The internal state space of the probabilistic filter is an undirected graph $G(V,E)$ similar to the one used

by (Liao et al., 2003). Let the vertices of this graph be $v_i \in V$ and the edges $e_{ij} \in E$. The people that are tracked are assumed to be moving on these edges (edge e_{ij} is the one connecting vertices v_i and v_j).

While the possibility of having complex models to describe people's movements in the environment has been mentioned previously (Arndt and Berns, 2012b), the authors have only used simple ones until recently. For this work, the probabilities for moving from one edge to another edge have been specified for interesting vertices of the motion graph. Formally, the probabilities $p(e_{jl} \,|\, e_{ij})$ for transferring from edge e_{ij} to edge e_{jl} are now initialized non-uniformly for some vertices. This allows to model paths that are much less likely to be traversed by humans as others.

For the experiments in this work, the graphs as well as the transition probabilities have been manually defined. However, it should be noted, that there are various methods available in the literature to infer the layout of such a graph and the respective transition probabilities from observations of the environment. For example, (Bennewitz et al., 2005) define *resting places* between which trajectories are learned and (Ikeda et al., 2012) use a very similar *sub-goal* concept to learn positions at which people take decisions on their further paths.

4 PREDICTIVE PATH PLANNING

4.1 State Prediction

To understand how the evolution of the environment is predicted, it should first be recapped how probabilistic filters are used to track objects. They generally work in two steps: first, the current state is *predicted* using models for motion and afterwards the belief is *updated* with new incoming sensor information.

This separation into two steps is the reason why these filters are very well suited for the long term prediction of the evolution of the environment. If the *update* step is simply skipped, the filter will continuously *predict* the state of the environment. For this case, the well-known recursive Bayesian update rule can then be re-written as in (1). In the formula, η is the normalizing constant while x_{t-1} and x_t represent the previous and the current state of the environment, respectively.

$$\text{belief}(x_t) = \eta \int p(x_t \,|\, x_{t-1}) \, \text{belief}(x_{t-1}) \, \mathrm{d}x_{t-1} \quad (1)$$

In the implementation of this work, this process is decoupled from the on-line tracking of the people. When a new path has to be planned, a copy of the current belief of the on-line tracker is created and only this copy is then iteratively updated to "see" into the future, leaving the rest of the tracking framework unaffected.

4.2 Evaluation of Risk

To find safe paths in the environment, the notion of safety is important. This work will define the risk imposed on a human being by the robot exactly as in (Arndt and Berns, 2014, sec. 3.1), as it has shown to be a useful metric. To recap, the function that gives the cost in terms of risk is a parameterized piecewise polynomial according to (2).

$$f_{a,b} : [0;\infty] \to [0;1]$$
$$f_{a,b}(d) = \begin{cases} 1 - (a^{-1}d)^b & d \leq a \\ 0 & d > a. \end{cases} \quad (2)$$

The function assigns a risk-cost for each distance d between the human and the robot. It will be maximal at $d = 0$, i.e. when robot and person coincide and monotonically decrease till zero. The parameters a and b are used to control the exact shape of the curve. Parameter a reflects the distance from which on the cost will be zero, i.e. it defines which distance is assumed to be perfectly safe. The second parameter $b > 0$ defines the steepness of the cost curve. For the experiments later in this work, the cost function $f_{8,2}(d)$ has been used. An illustration of that instance of the function is given in Figure 1.

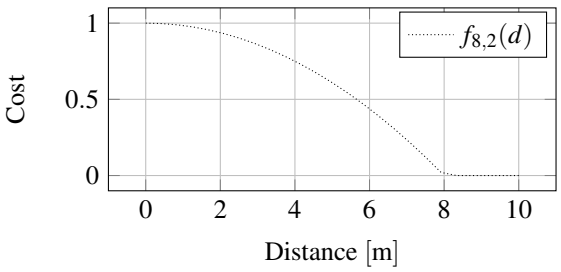


Figure 1: Plot of the cost function for $a = 8$ and $b = 2$.

As outlined in section 3, the state estimation of people's positions is done on a graph structure. The risk function is thus applied to the belief (e.g. represented by particles) and the outcome is a separate graph structure, the cost graph, with edge weights representing the risks associated with driving along the corresponding edge (Arndt and Berns, 2014, sec. 3.3.2).

In a last step, the edge weights of this risk cost graph must be fused with the ones of the classic dis-

tance cost graph, to obtain a resulting graph that allows to find paths that have minimal costs in terms of both safety and distance. This fusion takes two weights α and β as arguments which can be used to adapt the preference for safe or short paths, see (Arndt and Berns, 2014, sec. 3.4) for details.

4.3 Planning

Path planning for the mobile robot takes place on the same graph which is also used to constrain the human motion and where the tracking of people takes place. This has shown to be very convenient, as only a single representation of the state space has to be maintained.

It was noted before, that the complexity of temporal path planning is high and thus not computationally feasible for all applications. However, for typical graphs, our research has shown that it is perfectly feasible to do the prediction for *all* simple (without having cycles) paths from start to goal and to finally select the one with the least predicted costs. This subsection goes more into detail of how the candidate paths are evaluated for the resulting cost.

For each candidate path[1] $(v_1, \ldots, v_n)$ between the start vertex v_1 and the goal vertex v_n, the algorithm **AnalyzePath** (see Figure 2) will be called with the current belief at the time of planning. This algorithm further needs the two parameters l_{step} and v_{robot} which define the step length for discretization and the assumed robot velocity, respectively. The step length is important, because it defines the granularity of the planning process in the time-dimension.

Roughly speaking, the algorithm will simulate the robot driving a distance of l_{step} and then predict the future at the next point of time. Setting l_{step} to a smaller value increases the time resolution of the prediction, but will of course also slow the algorithm down. Reasonable values for the two parameters are e.g. $l_{\text{step}} = 1\text{m}$ and $v_{\text{robot}} = 0.5\frac{\text{m}}{\text{s}}$, leading to a time of two seconds between predicted steps.

As long as the end of the path has not yet been reached, the filter is advanced and the next segment (the distance l_{step}) will be analyzed with the algorithm **AnalyzeSegment** (see Figure 3). This algorithm takes the current edge e_{ij}, the current linear position t on the vertex, the distance to travel d and the current accumulated cost c as arguments. It traverses the current edge for the specified distance d. If it can stay on the edge, the fraction of the traversed distance of the edge's weight is summed up to the cost c and the control is returned to the higher level algorithm. If, however, the end of the current edge is reached it will first check if the end of the path has been reached in which case it returns and also the **AnalyzePath** algorithm will terminate. If the path is not ending, the algorithms calls itself recursively for the next edge till the requested distance has been traversed or the end of the path has been reached. At the end, when the single costs of all candidate paths are known, the one with the lowest overall costs (as returned in Figure 2 line 13) is selected as the resulting path.

To make these steps more intuitively accessible, the rest of this section is dedicated to an example of the algorithm applied to a real-world path-planning task. Figure 4 shows the output of the "classic" path planner that uses probabilistic information (as described in (Arndt and Berns, 2014)), but does *not* do any prediction about the evolution of the state.

From first glimpse, this path does indeed look quite reasonable, as it aims to avoid the region of high probability for a person being present (red patches on the heatmap). If, however, the prediction is enabled, the path looks different and at first "worse" (i. e. more expensive), see Figure 5 which depicts the situation at the time of planning. In this and the following figures, the blue circle will indicate the predicted robot position.

In the subsequent figures 6 and 7, the filter is advanced as the robot (hypothetically) moves along the planned path. Based on the motion model of the probabilistic filter, the belief takes certain "paths" of the

Require: $l_{\text{step}} > 0$ {*the step length for discretization*}
Require: $v_{\text{robot}} > 0$ {*the assumed robot velocity*}
Require: $(v_0, \ldots, v_n),\ n > 1$ {*the path to be analyzed*}
Require: belief(x) {*the initial belief*}
Returns: Total cost of the path
1: **AnalyzePath**$((v_0, \ldots, v_n), \text{belief}(x))$:
2: **InitializeFilter**(belief(x)) {*initialize the probabilistic filter with the initial belief*}
3: time_delta $= l_{\text{step}} / v_{\text{robot}}$
4: {*start on the beginning of first edge of the path*}
5: $t \leftarrow 0$
6: $e \leftarrow e_{v_0 v_1}$
7: $c \leftarrow 0$ {*start with zero costs*}
8: end_reached $\leftarrow$ false
9: **while** $\neg$end_reached **do**
10: belief(x) $\leftarrow$ **AdvanceFilter**(time_delta) {*advance probabilistic filter for the simulated amount of time*}
11: (end_reached, e, t, c) $\leftarrow$ **AnalyzeSegment**$(e, t, l_{\text{step}}, c)$
12: **end while**
13: **return** c

Figure 2: Function to analyze a whole path

[1] Every *simple* path from start to goal qualifies as a candidate path.

Require: $(v_0, \ldots, v_n)$ {*the whole path*}
Require: $d > 0$ {*remaining distance to travel*}
Require: $\forall e_{ij} \in E : w_{e_{ij}} \geq 0$ {*all edge weights according to the current belief*}
Require: **Successor**(e) {*function that finds the successor of edge e on the current path*}

1: **AnalyzeSegment**(e_{ij}, t, d, c) :
2: $l =$ **EdgeLength**(e_{ij})
3: $t' = t + (d/l)$ {*calculate the new position on edge*}
4: **if** $t' > 1$ **then**
5: {*new position is beyond current edge*}
6: $c \leftarrow c + (1-t) \cdot w_{e_{ij}}$ {*sum up the fraction traversed on current edge to the costs*}
7: **if** $\neg\exists$**Successor**(e_{ij}) **then**
8: **return** $(\text{true}, e_{ij}, t, c)$ {*there is no successor of edge e_{ij}, the path is ending*}
9: **end if**
10: $e' =$ **Successor**(e_{ij}) {*find the next edge*}
11: $t' = 0$
12: $d' = d - (1-t) \cdot l$ {*calculate remaining length*}
13: **if** $d' < 0$ **then**
14: **return** $(\text{false}, e', t', c)$ {*no more distance to travel, end recursion*}
15: **end if**
16: **return** **AnalyzeSegment**(e', t', d', c)
17: **else**
18: $c \leftarrow c + (d/l) \cdot w_{e_{ij}}$ {*sum up the fraction traversed on current edge to the costs*}
19: **return** $(\text{false}, e_{ij}, t', c)$ {*as we are still on the same edge e_{ij}, just use new linear parameter t'*}
20: **end if**

Figure 3: Function to analyze a segment of a path

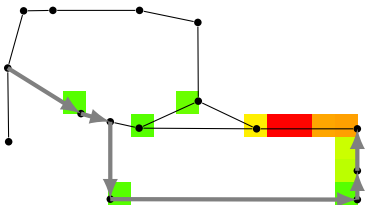


Figure 4: Shortest path found by Dijkstra's algorithm without predicting the future states of the environment.

graph. It can be easily seen that the chosen path mainly follows edges of the graph that have very little probability of a person being present at the predicted time in the future. If the original shortest path found without prediction (Figure 4) is re-evaluated using the predicted situation depicted in Figure 7, it can be realized that the classic "shortest path" does not look that good after all.

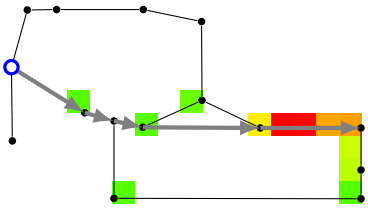


Figure 5: Evaluation of the path found with predictive path planning at $t = 0$s.

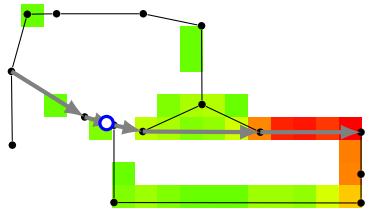


Figure 6: Evaluation of the path found with predictive path planning at $t = 10$s.

5 EXPERIMENTS

5.1 Experimental Setup

All of the conducted experiments are set in the facilities of the authors' research group, a common University building. It was equipped with a total of six AmICA wireless sensor network nodes to provide ambient sensor data. A schematic view of the environment can be found in Figure 8. For each of the experiment runs, the mobile indoor robot ARTOS was placed at a fixed start position and then given the instruction to plan a path to a fixed goal position with and without enabled state prediction. The path was then traversed. For the cost function in Equation 2, the parameters have been set to $a = 8$ and $b = 2$.

Experiments were conducted in two sets. The first one was done with the real robot in the real environment to prove the applicability of the proposed approach. While these experiments are of great importance, as no simulation environment can replace the reality, they are unfortunately unable to provide statistically relevant data, as the count of runs is limited.

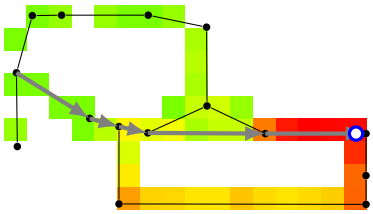


Figure 7: Evaluation of the path found with predictive path planning at $t = 36$s (goal reached).

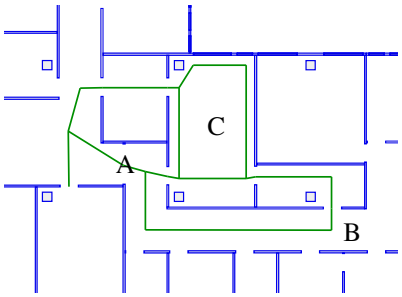


Figure 8: Layout of the environment within which the experiments have been conducted. The graph for tracking, prediction and path planning has been overlaid. Important locations that are referenced in the text: A: start position of the robot; B: goal position; C: meeting room.

For this reason, a second set of experiments was conducted in a simulated version of the same environment. This simulation was based on the SimVis3D framework (Wettach et al., 2010) and handled everything from simulation of vehicle physics to the behavior of the passive infrared motion sensors as well as basic radio communication of the sensor nodes (Arndt and Berns, 2012a).

5.2 Experimental Results

5.2.1 Reality

In this section, two experiment runs with the real robot in the real smart environment will be presented. During the first run, a person (the author) was being present in the meeting room all the time, "blocking" the path through that room. As the models indicate that, if someone is present in that room, the person is unlikely to leave it soon, the robot predicts the situation and thus takes the path through the hallway. Figure 9 shows the initial situation at the time of planning. The current belief as estimated by the particle filter shows a high probability in the area of the meeting room, which reflects the situation well. The position of the robot itself, as predicted or localized by its own localization system, respectively, is marked by a dot and the resulting best path is visualized by arrows.

Figure 10 shows the robot while traversing the hallway. It is important to note that the first part of the illustration (Subfigure 10(a)) represents the *prediction* at the time of planning while the second part (Subfigure 10(b)) is a representation of the *actual* situation at the time the real robot passes that point. It can be seen that the belief in reality differs from the predicted belief, but this is due to the fact that the *real* belief is updated using new sensor values while the prediction does not have these information available.

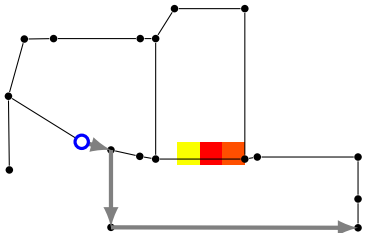
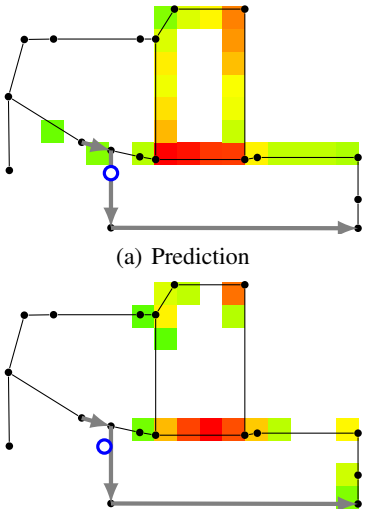


Figure 9: Experiment run one, initial situation at $t = 0$ with resulting best path (according to the prediction).

Nevertheless, the similarities are clearly visible. Figure 11 shows an actual photo of the robot at approximately the marked position.



(a) Prediction

(b) Reality

Figure 10: Experiment run one, robot traversing the hallway.

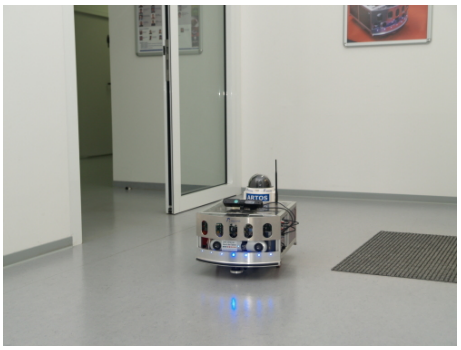
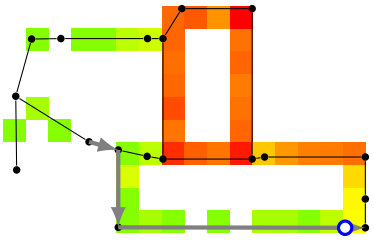
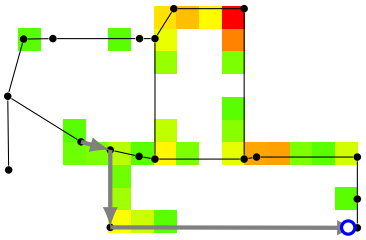


Figure 11: ARTOS following the path through the hallway, situation roughly corresponding to Figure 10.

Fast forwarding further, the robot can be seen reaching its goal position in Figure 12. Again, this



(a) Prediction



(b) Reality

Figure 12: Experiment run one, robot reaching the goal.

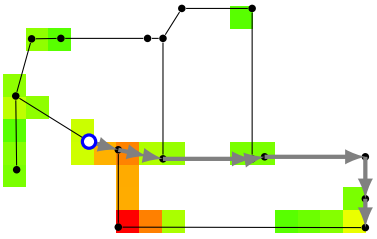
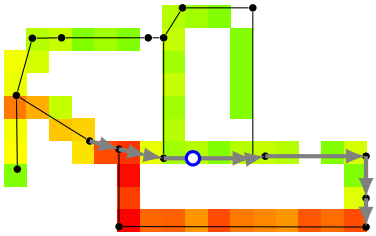


Figure 13: Experiment run two, initial situation at $t = 0$ with resulting best path (according to the prediction).

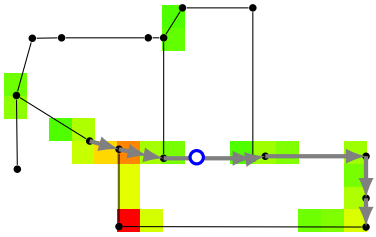
figure can be consulted to evaluate the accuracy of the prediction. By comparing it to reality, it can be seen that both beliefs about the situation differ more strongly than in in the previous illustration, but they are still close to each other. In both of them, the probability for the person still being present in the meeting room is rather large.

For the second experiment run in reality, a person was walking through the southern part of the hallway at the time of planning (the exact ground truth is not known and also not indicated in the figures). For this situation, the models indicate that it is very likely that the person will continue its path through the hallway and consequently, the predicted evolution of the state leads the robot's path planner to choose a path through the meeting room to minimize the chances of collision and maximize the safety of the path. The initial situation shows the belief focused in the hallway, although not as highly concentrated as for the previous run, see Figure 13.

The situation after some time has passed is illustrated in Figure 14. The belief in reality has changed compared to the initial situation, but not as drastically as predicted by the planner. Still, the prediction is not wrong, as it states that the person is most likely located in the hallway.



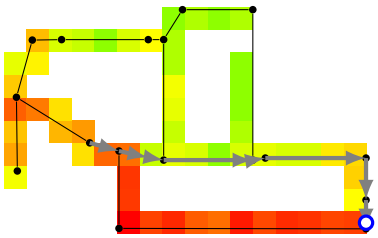
(a) Prediction



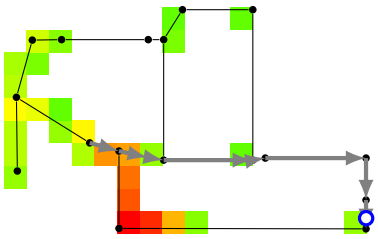
(b) Reality

Figure 14: Experiment run two, robot traversing the meeting room.

Finally, in Figure 15 the situation when the robot reaches the goal is depicted. Just as in the previous run, the prediction further deviates from the reality, slowly evolving into a uniform distribution over the whole state space.



(a) Prediction



(b) Reality

Figure 15: Experiment run two, robot reaching the goal.

Regarding the computational feasibility it should be noted that during the experiments, everything[2] was calculated on-line and on-board of the robot. A WiFi connection to the robot was only used for visualization and initialization purposes. The calculation of the best path took about six seconds for the shown graph on an (at the time of writing) eight-year old ultra-low voltage processor[3].

5.2.2 Simulation

As already outlined at the beginning of this section, the simulation experiments were used to conduct a large number (compared to the manually conducted experiments in reality) of runs with prediction enabled and also disabled. These runs were then statistically analyzed to compare the traditional path planning with the proposed temporal path planning in terms of safety.

Due to the use of the simulation environment, the ground truth of the person's position was available and thus the evaluation of the safety, computed just as in (Arndt and Berns, 2014, sec. 4.2.2), was much easier. Four different trajectories for people have been simulated, two of them represent the same situations as with the real robot: walking through the southern hallway (situation 1) and staying in the meeting room (2). The other two are different as they do *not* behave like modeled in the filter, but take paths that are less likely: starting in the meeting room and leaving it through the door at the west (3) or the door at the east (4), respectively.

For all the classes, the mean duration of the traversal $\overline{d}$, the mean distance between robot and person $\overline{x}$ as well as the mean risk-cost $\overline{c}$ are given in Table 1. The standard deviations of the corresponding values are suffixed with a subscript σ. A total sum of 113 experiment runs have been used to calculate these values.

In all regarded situations, the active prediction *increases* the mean distance x between robot and person and thus *decreases* the risk-costs c associated with that path, although the effect is not that distinct in situation three. However, it must also be noted that the reduced risk often comes at the cost of an increased duration of traversal, although not always as prominent as in situation one.

[2]Including the reception of frames from the wireless sensor network, the processing of the probabilistic filters as well as the planning with prediction

[3]Intel Core Duo U2500 at 1.20GHz

Table 1: Comparison of the effects of prediction using simulated experiment runs

Sit.	Pred.	$\overline{d}$ [s]	d_σ	$\overline{x}$ [m]	x_σ	$\overline{c}$	c_σ
1	Off	74	27.8	10.2	1.8	0.20	0.10
	On	132	23.7	12.3	1.5	0.09	0.07
2	Off	64	3.4	7.8	0.8	0.27	0.06
	On	63	4.6	9.9	0.6	0.16	0.04
3	Off	62	3.5	9.5	0.3	0.17	0.01
	On	66	3.5	9.8	0.1	0.17	0.01
4	Off	63	5.0	8.5	1.2	0.23	0.12
	On	69	4.0	10.1	0.1	0.09	0.01

6 CONCLUSIONS AND FUTURE WORKS

Building on previous work that defines metrics for safety and how to find a trade-off between short and safe paths, a method to *predict* the evolution of the environment of a mobile robot, using external sensor information from a smart environment, has been introduced. The theoretical concept has been validated with a real mobile robot in a smart environment. Results of the experiments show that (given good models) the evolution of the environment can indeed be predicted to a degree that allows to optimize the path planning in mobile robots. The simulation experiments aim to analyze the concept in a more statistical fashion, because one may of course argue that the experiments in reality lack diversity. Nevertheless, the results of these experiments also validate the principle.

The accuracy of the prediction could in the future be increased using more sophisticated models that also take into account the *intentions* people have while moving through their environment, similar to the goal-orientation described by (Bruce and Gordon, 2004). More complex models could then also be trained using learning algorithms (as outlined by e.g. (Foka and Trahanias, 2010)) instead of being handcrafted.

Additionally, it would be very interesting to decrease the computational complexity of the proposed approach. While being perfectly feasible for the examined examples, there might be situations with much larger graphs or search-spaces in which it might be needed to switch to heuristics or to find more efficient algorithms to be still able to plan paths in the spatio-temporal domain.

REFERENCES

Alami, R., Albu-Schaeffer, A., Bicchi, A., Bischoff, R., Chatila, R., Luca, A. D., Santis, A. D., Giralt, G., Guiochet, J., Hirzinger, G., et al. (2006). Safe and dependable physical human-robot interaction in anthropic domains: State of the art and challenges. In *Proc. IROS*, volume 6.

Alvarez, A., Caiti, A., and Onken, R. (2004). Evolutionary path planning for autonomous underwater vehicles in a variable ocean. *Oceanic Engineering, IEEE Journal of*, 29(2):418–429.

Amato, G., Broxvall, M., Chessa, S., Dragone, M., Gennaro, C., López, R., Maguire, L., Mcginnity, T., Micheli, A., Renteria, A., O'Hare, G. M., and Pecora, F. (2012). Robotic ubiquitous cognitive network. In Novais, P., Hallenborg, K., Tapia, D. I., and Rodríguez, J. M. C., editors, *Ambient Intelligence - Software and Applications*, volume 153 of *Advances in Intelligent and Soft Computing*, pages 191–195. Springer Berlin Heidelberg.

Arndt, M. and Berns, K. (2012a). Mobile robots in smart environments: The current situation. In Levi, P., Zweigle, O., Häußermann, K., and Eckstein, B., editors, *Autonomous Mobile Systems 2012*, Informatik aktuell, pages 39–47. Springer Berlin Heidelberg. 10.1007/978-3-642-32217-4_5.

Arndt, M. and Berns, K. (2012b). Optimized mobile indoor robot navigation through probabilistic tracking of people in a wireless sensor network. In *Proceedings of the 7th German Conference on Robotics (Robotik 2012)*, pages 355–360, Munich, Germany. VDI Verlag, Berlin.

Arndt, M. and Berns, K. (2014). Risk-reduced mobile robot path planning in smart environments. In *Proceedings for the joint conference of ISR 2014 and ROBOTIK 2014*, pages 582–589. VDE VERLAG GMBH.

Augusto, J. C., Nakashima, H., and Aghajan, H. (2010). Ambient intelligence and smart environments: A state of the art. In *Handbook of Ambient Intelligence and Smart Environments*, pages 3–31. Springer.

Bennewitz, M., Burgard, W., Cielniak, G., and Thrun, S. (2005). Learning motion patterns of people for compliant robot motion. *The International Journal of Robotics Research*, 24(1):31–48.

Bruce, A. and Gordon, G. (2004). Better motion prediction for people-tracking. In *Proceedings of the International Conference on Robotics and Automation (ICRA)*, New Orleans, USA.

Coradeschi, S. and Saffiotti, A. (2006). Symbiotic robotic systems: Humans, robots, and smart environments. *Intelligent Systems, IEEE*, 21(3):82–84.

Foka, A. F. and Trahanias, P. E. (2010). Probabilistic autonomous robot navigation in dynamic environments with human motion prediction. *International Journal of Social Robotics*, 2(1):79–94.

Guzzi, J., Giusti, A., Gambardella, L., Theraulaz, G., and Caro, G. D. (2013). Human-friendly robot navigation in dynamic environments. In *Robotics and Automation (ICRA), 2013 IEEE International Conference on*, pages 423–430.

Ikeda, T., Chigodo, Y., Rea, D., Zanlungo, F., Shiomi, M., and Kanda, T. (2012). Modeling and prediction of pedestrian behavior based on the sub-goal concept. In *Proceedings of Robotics: Science and Systems*, Sydney, Australia.

Liao, L., Fox, D., Hightower, J., Kautz, H., and Schulz, D. (2003). Voronoi tracking: Location estimation using sparse and noisy sensor data. In *Intelligent Robots and Systems, 2003. (IROS 2003). Proceedings. 2003 IEEE/RSJ International Conference on*, volume 1, pages 723 – 728.

Lindemann, S. R. and LaValle, S. M. (2005). Current issues in sampling-based motion planning. In Dario, P. and Chatila, R., editors, *Robotics Research. The Eleventh International Symposium*, volume 15 of *Springer Tracts in Advanced Robotics*, pages 36–54. Springer Berlin Heidelberg.

Luber, M., Stork, J., Tipaldi, G., and Arras, K. (2010). People tracking with human motion predictions from social forces. In *Robotics and Automation (ICRA), 2010 IEEE International Conference on*, pages 464–469.

Lukowicz, P., Pentland, S., and Ferscha, A. (2012). From context awareness to socially aware computing. *Pervasive Computing, IEEE*, 11(1):32–41.

Saffiotti, A. and Broxvall, M. (2005). PEIS ecologies: Ambient intelligence meets autonomous robotics. In *Proc of the Int Conf on Smart Objects and Ambient Intelligence (sOc-EUSAI)*, pages 275–280, Grenoble, France.

Sanfeliu, A., Hagita, N., and Saffiotti, A. (2008). Network robot systems. *Robotics and Autonomous Systems*, 56(10):793 – 797. Network Robot Systems.

Sisbot, E., Alami, R., Simeon, T., Dautenhahn, K., Walters, M., and Woods, S. (2005). Navigation in the presence of humans. In *Humanoid Robots, 2005 5th IEEE-RAS International Conference on*, pages 181–188.

Sisbot, E., Marin-Urias, L., Alami, R., and Simeon, T. (2007). A human aware mobile robot motion planner. *Robotics, IEEE Transactions on*, 23(5):874–883.

Wettach, J., Schmidt, D., and Berns, K. (2010). Simulating vehicle kinematics with simvis3d and newton. In *2nd International Conference on Simulation, Modeling and Programming for Autonomous Robots*, Darmstadt, Germany.

Wille, S., Wehn, N., Martinovic, I., Kunz, S., and Göhner, P. (2010). Amica - design and implementation of a flexible, compact and low-power node platform. Technical report, University of Kaiserslautern. `http://ems.eit.uni-kl.de/uploads/tx_uniklwehn/AmICATechReport2010.pdf`.