Please do not disturb!
Minimum Interference Coverage for Social Robots

Gian Diego Tipaldi Kai O. Arras

***Abstract*— In this paper we address the problem of human-aware coverage planning. We first present an approach to learn and model human activity events in a probabilistic spatio-temporal map using spatial Poisson processes. We then propose a coverage planner for paths that minimize the interference probability with people. To this end, we pose the coverage problem as an asymmetric traveling salesman problem with time-dependent costs (ATDTSP) derived from the information in the map. The approach enables a noisy robotic vacuum in a home scenario, for instance, to learn to avoid busy places at certain times of the day such as the kitchen at lunch time. We evaluate the planner using a simulator of people in a home environment to generate typical weekday activity patterns. In the experiments with a regular TSP planner and two modified TSP heuristics, the proposed coverage planner significantly reduces interference with people in terms of number of disturbed persons and overall disturbance time.**

I. Introduction

Robots that operate in human environments require the ability to sense people and recognize their activities. But beyond that, they also need the ability to reason about the places and times when and where people are engaged into which activity. This knowledge will enable a robot to coordinate its actions, plans and schedules with the patterns of human activities, giving it the ability to smoothly blend into the workflows and daily routines of people. We believe that this ability is key in the attempt to build socially acceptable robots.

In this paper, we approach this problem with a planner of coverage paths that minimize the probability of encountering people. To this end, we propose the *spatial affordance map*, a model that represents human activity events as a rate function of a non-homogenous spatial Poisson process. We present how the model is learned and used for the task of human-aware coverage. This is an under-explored planning problem to our knowledge, relevant in all coverage applications in populated environments in which the operation of the robot causes a disturbance to people, or vice-versa, where the presence of people compromises the robot's task. In fact, many coverage tasks occur in inhabited environments. Examples include vacuum cleaning in domestic scenarios or floor care, transportation, mail delivery or inspection in populated office scenarios.

The paper relates to the problem of coverage path planning in that it takes an exact cellular decomposition of the environment with cells corresponding to 'rooms' connected in a graph topology. We assume that the robot is equipped with a coverage algorithm for each cell/room, either randomized or systematic. We then extend the state of the art by addressing the novel question how robots can optimally cover an inhabited environment. Unlike previous works on coverage in robotics where authors optimized time to completion, path length or the number of turns (see [1] for a survey), we seek the most socially acceptable way to fulfill this task based on a previously learned probabilistic model of people's space usage.

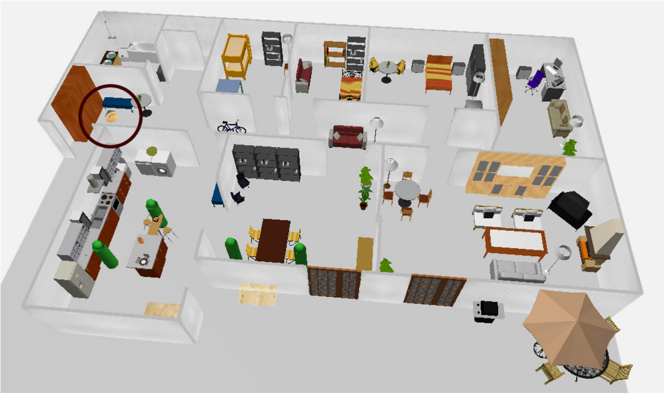

Fig. 1. Four agents (shown as green-colored cylinders) engaged in a dining activity in our simulator. Two agents are in the dining room, two agents are in the kitchen. The vacuum robot (circle) is in the hallway since it has learned that at this particular place and time, the probability to encounter someone is minimal.

The paper is structured as follows: section III gives the theory of the spatial affordance map and describes learning and inference in the map. Section IV develops the planning algorithm followed by Section V that describes the people simulator we used in the experiments. Section VI contains the experimental results and section VII concludes the paper.

II. Related Work

With growing presence of robots in inhabited environments, human-aware planning is a field of increasing activity. The problem has a motion planning and a task planning aspect. Most authors address the problem as a local motion planning problem over short time and space scales. Their purpose is typically to produce acceptable or safe robot motion for interaction, navigation, or manipulation among people [2, 3, 4]. Human-aware task planning, where this work is to be placed, is a relatively new field. Closest are the works presented in [5, 6, 7].

All authors are with the Social Robotics Lab, Department of Computer Science, University of Freiburg, Germany. Email: {tipaldi,arras}@informatik.uni-freiburg.de.

Alami *et al.* [5] propose a decisional framework for human-robot interactive task achievement. The framework relies on known sets of actions that humans and robots can carry out and allows for the specification of social constraints that can encode human preferences or needs. Using a hierarchical task network planner the framework generates plans towards joint robot/human goals under such constraints. Broz *et al.* [6] recognize that the explicit representation of time in models for human-robot interaction is important for achieving good performance. They model interactions as a POMDP with a time index as part of the state and propose time-state aggregation to reduce the resulting very large state spaces. The approach is able to efficiently choose optimal robot policies for simple time-dependent interaction scenarios. The planner of Cirillo *et al.* [7] generates policies conditioned on on-line recognition of human plans. Enumerating all possible human actions in a discrete state space, the planner incorporates domain knowledge as temporal logical expressions that help to prune the (potentially very large) search tree. In a simple vacuum cleaner scenario they also aim at a robot that is able to avoid disturbance of a human. The robot's capacity to anticipate forthcoming actions of humans is delegated to a human plan recognition system assumed to be available.

The major difference to these works is that here we use learning to acquire a model of human activities prior to planning. This enables the robot to reason about and plan with expected forthcoming human actions. The resulting plans are optimal with respect to minimal or maximal occurrence probabilities of the learned human activities and can optionally be combined with sensor-based replanning during plan execution.

Learning human behavior models in robotics has been addressed by several researchers. Mostly, the focus lies on the description of human motion.

Kruse and Wahl [8] propose statistical grids whose cells hold temporal occupancy probabilities of people and stochastic trajectories which are paths of dynamic objects along which their appearance probability is modeled by a Poisson process. The goal is to assess and plan minimal collision probability paths. The grid and the trajectories are learned from ceiling-mounted cameras. Bruce and Gordon [9] learn goal locations in an environment from trajectories obtained by a laser-based people tracker. Based on the assumption that people move in a goal-oriented fashion, paths are planned from the location of a person being tracked to the goal locations. Bennewitz *et al.* [10] learn typical motion patterns that people follow in an environment. The approach collects trajectories of people with multiple statically mounted laser scanners and combine similar trajectories to motion patterns using EM clustering. From each pattern a Hidden Markov Models is derived which enables a mobile robot to predict the motion of people and to adapt its navigation behavior accordingly.

Not only focused on the aspect of human motion is the work by Ihler and Smyth [11]. The authors presents a non-parametric approach to learn time profiles of human activities. The rate function of a Poisson process is learned using non parametric Bayesian models: the infinite mixture model with a Dirichlet process prior. Although interesting and related to our work, their approach does not consider the *spatial* variation of activities.

These works consider either the special case of human motion or lack the ability to make inference in both time and space. In contrast, the spatial affordance map is a single representation for inference about spatio-temporal behavior of people that coherently relates time, space and occurrence probability of activity events. It is able to process queries that ask for occurrence probability, time and space, respectively. The map has been first introduced to learn spatial priors of human motion that lead to considerably more accurate tracking of people [12, 13] and then extended to address more complex inference problems for planning. In an accompanying paper [14] the map has been used for the dual problem of learning to find paths that maximize the probability to encounter humans, solved with a variant of an MDP planner.

III. Spatial Affordance Map

In this section we briefly summarize the spatial affordance map to introduce the background and notation needed to fomalize the minimum interference coverage problem. More details on the map can be found in [13, 14].

The spatial affordance map is a non-homogeneous spatial Poisson process. Under the assumption that events in time occur independently of one another, a *Poisson process* can deal with distributions of time intervals between events. Concretely, let $N(t)$ be a discrete random variable to represent the number of events occurring up to time t with rate λ. Then we have that $N(t)$ follows a Poisson distribution with parameter λt

$$P(N(t)=k) = \frac{e^{-\lambda t}(\lambda t)^k}{k!} \qquad k = 0, 1, \ldots \tag{1}$$

In general, the rate parameter may change over time. In this case, the generalized rate function is given as $\lambda(t)$ and the expected number of events between time a and b is

$$\lambda_{a,b} = \int_a^b \lambda(u)\, du. \tag{2}$$

A homogeneous Poisson process is a special case of a non-homogeneous process with constant rate $\lambda(t) = \lambda$.

The *spatial* Poisson process introduces a spatial dependency on the rate function given as $\lambda(\vec{x}, t)$ with $\vec{x} \in X$ where X is a vector space such as $\mathbb{R}^2$ or $\mathbb{R}^3$. For any subset $S \subset X$ of finite extent (e.g. an area in space), the number of events occurring inside this area can be modeled as a Poisson process with associated rate function $\lambda_S(t)$ such that

$$\lambda_S(t) = \int_S \lambda(\vec{x}, t)\, d\vec{x}. \tag{3}$$

Learning the spatio-temporal distribution of events in an environment is equivalent to learn the generalized rate function $\lambda(\vec{x},t)$. However, learning the full continuous function is a highly expensive process. For this reason, we approximate the non-homogeneous spatial Poisson process with a piecewise homogeneous one. The approximation is performed by discretizing the environment into a tridimensional grid, where each cell represents a local – in terms of space and time – homogeneous Poisson process with a constant rate,

$$P_{ij\tau}(N(t)=k) \quad = \quad \frac{e^{-\lambda_{ij\tau}(t-t_\tau)}(\lambda_{ij\tau}(t-t_\tau))^k}{k!} \qquad (4)$$

with $k = 0, 1, \dots$ and $t_\tau \leq t < t_{\tau+1}$ and where $\lambda_{ij\tau}$ is assumed to be constant and the indices refer to a particular cell in the discretization. Finally, the spatial affordance map represents the generalized rate function $\lambda(\vec{x},t)$ using a grid approximation,

$$\lambda(\vec{x},t) \simeq \sum_{ij\tau} \lambda_{ij\tau} \mathbf{1}_{ij\tau}(\vec{x},t) \qquad (5)$$

with $\mathbf{1}_{ij\tau}(\vec{x},t)$ being the indicator function. Other tessellation schemes in space and time such as octrees, regions of homogeneous Poisson rates or function approximators [11] can equally be used.

We take a Bayesian learning approach using Gamma priors to estimate the Poisson rate parameter of each cell. We discard a maximum likelihood approach since, without priors, it cannot properly initialize never observed cells. The Bayesian approach leads to easy-to-implement counting expressions in a grid. This makes life-long learning particularly simple as new information can be added at any time by one or multiple robots. The map is then learned from human activity observations $k_{1,\dots,n}$ that can be obtained either from ceiling-mounted cameras [8], wearable devices or the exteroceptive sensors of the mobile robot [13]. Here they will come from a people simulator described in Section V.

The spatial affordance map (whose name lends itself from the view of human activities as affordances of an environment) can model all perceived types of human activities, each type in a separate layer of cells. Here, we only consider the activity of *being* at a given place and time. This information allows to reason about robot actions that minimize (or maximize) the time a robot interferes with a person.

IV. Minimum Interference Coverage

As stated in Section I, we decompose the environment into cells that, without loss of generality, correspond to *rooms* in the environment. The rooms are nodes in a graph whose optimal coverage in terms of execution time is obtained by solving a Traveling Salesman Problem (TSP). Here, we intend to minimize the encounter probability of people based on the learned map. This problem can be modeled as an asymmetric traveling salesman problem with time-dependent costs (ATDTSP). The asymmetry here arise from the time-dependent structure of the costs: going from node a to node b is different than going from b to a since the nodes are visited at different times and thus have different costs.

Formally, let $\mathcal{G} = \{\mathcal{V}, \mathcal{E}\}$ be a weighted graph with weights on the nodes and the edges. We define the set of vertices, $\mathcal{V}$, to be the set of cells in the environment and $\mathcal{E}$ the set of edges to encode the connectivity among them. The weight of a vertex v_i is computed considering the time when the vertex is visited, t, and the duration to perform the coverage task, t_V. The resulting cost is then

$$W_i(t) = \int_t^{t+t_V} \int_{A(v_i)} \lambda(\vec{\chi},t)\, d\vec{\chi}\, dt, \qquad (6)$$

where $A(v_i)$ represents the area occupied by the vertex v_i. Similarly, the weight of an edge $e_{ij} = (v_i, v_j)$ is computed as follows

$$\begin{aligned} W_{ij}(t) \quad = \quad & \int_t^{t+t_E/2} \int_{A(v_i)} \lambda(\vec{\chi},t)\, d\vec{\chi}\, dt \,+ \\ & \int_{t+t_E/2}^{t+t_E} \int_{A(v_j)} \lambda(\vec{\chi},t)\, d\vec{\chi}\, dt, \end{aligned} \qquad (7)$$

where the two integrals represent the fact that the robot moves from the center of one cell to the center of the other cell. Note that the size of a room is not a sensible parameter, since the cost of a node already accounts for the room size.

The obtained graph can not be directly used in the ATDTSP setting due to its sparse structure. The graph is transformed into a fully connected one using a modified version of the Floyd-Warshall algorithm that takes into account time variable costs, see Algorithm 1. In order to reduce the computational complexity and to avoid the search in the full continuous range of the time variable, the time axis is discretized into regular intervals. This is equivalent to assume that the time spent to travel between two nodes is negligible with respect to the time spent in a particular node. Note that this assumption do not relax the asymmetry of the problem, since time is spent in visiting a node.

The final ATDTSP problem can then be solved by modifying the dynamic program approach for the standard TSP. Let v_0 be the initial vertex and $\mathcal{S} \subseteq \mathcal{V}$ such that $v_0 \in \mathcal{S}$. Given $v_i \neq v_0$, $v_i \notin \mathcal{S}$, we define $C(\mathcal{S}, i)$ be the cost of the optimal path that starts at v_0, visits all nodes in $\mathcal{S}$ and ends at v_i. With $t_i = t_0 + i(t_R + t_l)$, the optimal solution can then be defined in the following recursive way

$$C(\mathcal{S},k) = \min_{m \in \mathcal{S}} \{C(\mathcal{S},m) + W_{mk}(t_{|\mathcal{S}|}) + W_k(t_{|\mathcal{S}|+1})\} \qquad (8)$$

with the initial condition

$$C(\{v_0\},k) = W_0(t_0) + W_{0k}(t_0) + W_k(t_1). \qquad (9)$$

The resulting program is shown in Algorithm 2.

The complexity of the optimal algorithm is high both from the computational, $O(n^2 2^n)$, and the memory point of view, $O(n2^n)$. This reduces the applicability of the algorithm to small domains with $\simeq 20$ rooms for standard PCs. For this reason, we also consider two well-known

Algorithm 1: Time dependent Floyd-Warshall

```
In: A partial connected graph $\mathcal{G} = \{\mathcal{V}, \mathcal{E}\}$
Out: The fully connected graph $\mathcal{G} = \{\mathcal{V}, \mathcal{E}\}$
// The edge costs for each timestep
float C[ ][ ][ ]
// The predecessor node for path recovering
int P[ ][ ][ ]
Initialize C from (7)
for $t \leftarrow 0$ to $|V|$ do
    for $k \leftarrow 0$ to $|V|$ do
        for $v \leftarrow 0$ to $|V|$ do
            for $u \leftarrow v + 1$ to $|V|$ do
                if $C[v][k][t] + C[k][u][t] < C[v][u][t]$ then
                    C[v][u][t] = C[v][k][t] + C[k][u][t]
                    P[v][u][t] = k
                end
            end
        end
    end
end
Update the new edge costs from C
return $\mathcal{G}$
```

heuristics for the TSP. We modified the nearest neighbor and the greedy heuristic to work with time dependent costs.

The nearest neighbor heuristic builds a path incrementally, always visiting the unvisited node with the minimum cost at each step. To make it work with time dependent costs, we simply use the costs computed from the actual time step to choose the next room.

The greedy heuristic builds a path by choosing the edge with the minimum cost that leads to a valid path. The chosen edge does not need to connect the last visited node, as in the nearest neighbor case, but needs to satisfy a set of conditions:

- It connects two unconnected nodes,
- It does not create a loop,
- It is the only edge selected for that time step,
- It follows the *direction* of the previously selected edges (each node in the path must have only one incoming and one outcoming edge),

The last two properties are specific for the ATDTSP. The computational complexity of the two heuristics is $O(n^2)$ for the nearest neighbor and $O(n^3)$ for the greedy method. Both have linear memory requirements.

V. Simulator

For evaluating the model and the planner, we developed a people simulator that models human activities of agents in a home environment (Fig. 3). Simulation is required in our case since with real humans, experiments cannot be reproduced and simulated agents do not change their behavior in the presence of robots.

The simulator models typical home activities of multiple agents during a work day from 8 am to 11 pm. The patterns of each agent on each day are generated by sampling from a discrete distribution that describes the occurrence probabilities of a number of predefined activities. These activities are scripted by hand and reflect typical work day activities of a family with two kids according to the best of our knowledge. Fig. 4 depicts example activity profile of the four agents. The day starts with breakfast and bathroom activities before three agents leave to house for work and school. At lunch time, the family gets together. Two agents leave the house again during the afternoon. The evening includes cooking, a common dinner, and several distraction-related activities before everyone goes to bed. While the exact courses of activities may strongly vary in practice, we found no indication in our experiments that different or more elaborated activity patterns would influence the results. The subdivision of the environment into rooms and the graph topology is shown in Fig. 2.

Algorithm 2: Dynamic programming for ATDTSP

```
In: A fully connected graph $\mathcal{G} = \{\mathcal{V}, \mathcal{E}\}$, an initial node $v_0$
    and time $t_0$
Out: The minimum cost path $\mathcal{P}$
// The cost of optimal paths (see (8))
float C[ ][ ]
// The predecessor node of the optimal paths
int P[ ][ ]
Initialize C from (9)
for $i \leftarrow 2$ to $|V|$ do
    foreach $\mathcal{S} \subseteq V, |S| = i$ do
        foreach $u \in \mathcal{S}, u \neq v_0$ do
            Find $v$ that minimizes (8)
            Update $C[\mathcal{S} \setminus \{u\}][u]$ according to (8)
            $P[\mathcal{S} \setminus \{u\}][u] \leftarrow v$
        end
    end
end
$v^* \leftarrow argmin_v \, C[\mathcal{V} \setminus \{v\}][v]$
$\mathcal{S} \leftarrow \mathcal{V} \setminus \{v^*\}$
while $|\mathcal{P}| \neq |\mathcal{V}| - 1$ do
    $v^* \leftarrow P[\mathcal{S}][v^*]$
    $\mathcal{P} \leftarrow v^* \cup \mathcal{P}$
    $\mathcal{S} \leftarrow \mathcal{S} \setminus \{v^*\}$
end
$\mathcal{P} \leftarrow v_0 \cup \mathcal{P}$
return $\mathcal{P}$
```

The simulator engine follows the three-layered agent architecture from [15] that in our case consists in the layers

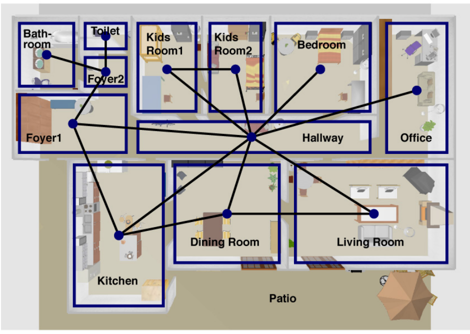

Fig. 2. The graph and room annotations of the home environment.

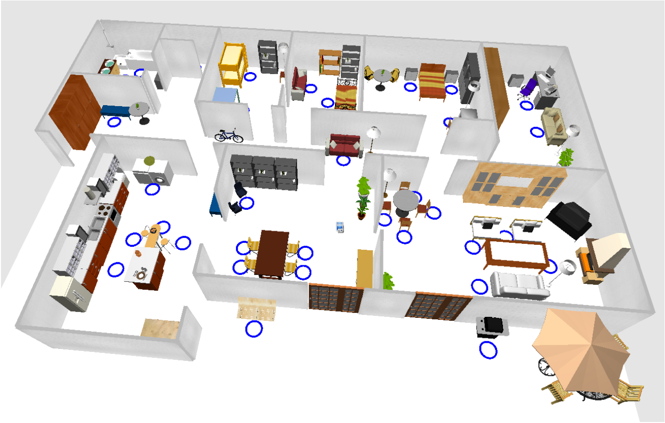

Fig. 3. The simulated home environment inhabited by four agents that engage in activities typical for a family with two kids.

activity scheduler, *activity executor* and *action executor*. At the beginning of the day, the activity scheduler generates a fixed schedule of the day for each agent. By randomly sampling from the discrete distributions, each day has a different schedule leading to variability in the patterns. Both the distribution and the sampled schedules are not known to the planning algorithm.

Every activity is composed of a set of actions such as enter, move, stay or leave, which in turn are activated and deactivated by the activity executor. Once an action is activated, the action executor takes care of its progress and signals back when it reached its final state. The actual plans are generated using A* with action costs that are randomly perturbed to simulate some motion variability. The simulator is also given the designated places at which activities are carries out (the patio for smoking, the kitchen for cooking, etc.). Finally, each time an agent is engaged into an activity, its type and place form an activity observation k_i to learn the map.

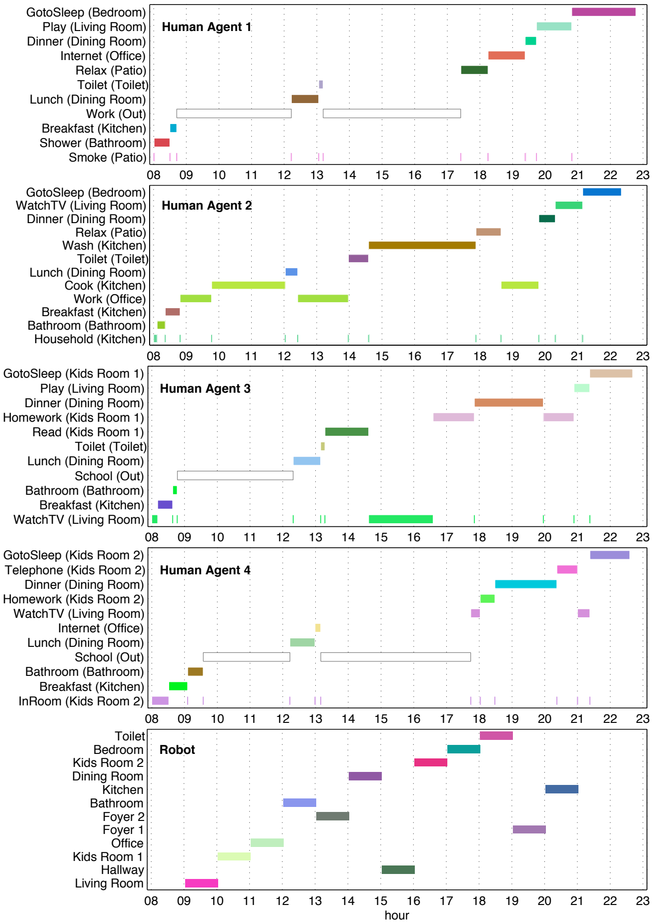

Fig. 4. Example activity profiles of the agents in the home environment and the robot. For the activities *Work* and *School* the human agents leave the house. All others are carried out at designated places in the environment shown as blue circles in Fig. 3. The bottom diagram shows an example coverage path of the robot which chooses the rooms in a way to avoid people. The kitchen, for instance, is cleaned last when the agents mainly stay in their bedrooms or in the living room.

VI. Experiments

For the experiment, the map is learned from activities of four agents over ten days, followed by a series of ten testing days. We use a grid resolution of $0.25m$ in x and y and 1 hour in time resulting in 15 time slots for the household scenario. For our simulated vacuum robot, we generate 500 randomized start locations and times such that the coverage task can be finished before the end of the day at 11 pm. For testing, we vary the cleaning time per room from 15 minutes to 1 hour and measure the amount of time the robot is in the same room with a person and how many people have been disturbed along its path. Disturbing the same person in two different rooms is counted twice.

The experiment demonstrates the ability of the AT-DTSP planner to generate optimal paths in terms of minimal interference while being able to cover the entire environment. We compare the optimal paths computed with the dynamic programming approach (DP) with the two heuristic strategies, nearest neighbor (NN) and greedy (GR), and a regular TSP strategy that minimizes the time to completion ignoring the map (CV). Comparison with the latter reveal the impact of the spatial affordance map.

Figure 5 shows the results. The DP approach outperforms the two heuristic planners and the optimal Euclidean path both in terms of the average number of disturbed people and the average overall disturbance time. For the maximum considered cleaning time per room, our solution yields a three-fold improvement of the number of disturbed people and an almost five-fold improvement of the disturbance time. Further, the DP planner is very weakly correlated with the length of the coverage task, showing that it can effectively account for the time variance of the problem. An interesting aspect is that the two modified heuristics (NN and GR) are relatively sensitive to changes in the coverage time and perform poorly with respect to the optimal solution. This demonstrates that classic TSP heuristics do not work well with time variability of costs and asymmetry in the graph and that new heuristics should be developed for this class of problems.

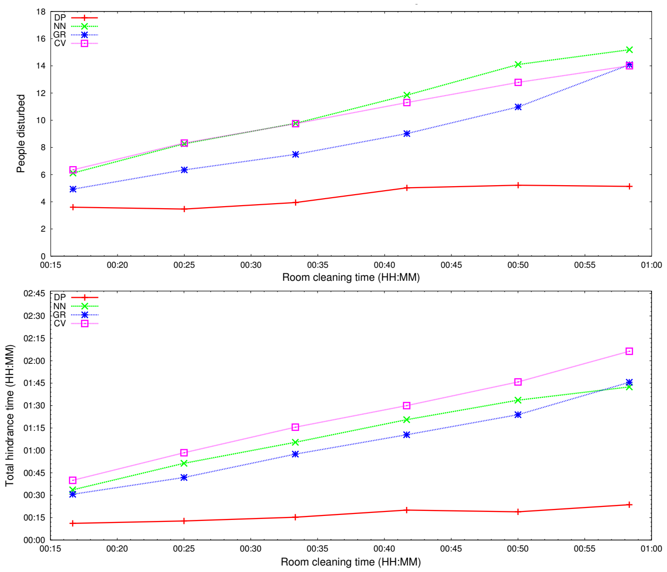

Fig. 5. Minimum interference coverage experiment. The picture shows the average number of people disturbed (top) and the average total interference time (bottom) versus the time to clean an individual room. The optimal solution to the ATDTSP problem outperforms the other alternatives in all aspects.

Lastly, the poor performance of the regular TSP strategy confirms clearly that the spatial affordance map provides valuable and effective information on human space usage that should not be ignored for coverage tasks in populated environments.

VII. Conclusions

In this paper, we developed a human-aware coverage planner to generate paths that minimize the interference probability with people. This problem is relevant for both, robots that seek to increase their social acceptability and robots whose efficiency is compromised by the presence of humans.

To this end, we proposed the *spatial affordance map*, a model that represents human activity events as a learned rate function of a spatial Poisson process and showed how the map can serve as a cost model for planning. Using this model, we formulated the minimal interference coverage problem as an asymmetric traveling salesman problem with time varying costs (ATDTSP) and developed a dynamic programming approach to solve the problem optimally. In the experiments, we compared the optimal solution to two standard heuristics in the TSP literature and a regular uninformed TSP planner.

The results showed very clearly that the proposed approach outperforms both the heuristics and the regular TSP planner in terms of the number of disturbed persons and the total disturbance time. The outcome confirms the appropriateness of the spatial affordance map as a model to forecast human actions in time and space. It also shows that the classical TSP heuristics perform poorly on the ATDTSP since the latter is much harder in terms of combinatorial structure than the regular symmetric TSP.

Currently, the optimal DP planner takes several seconds to complete a coverage plan in the considered home environment. Since this can be a limitation for low-cost robots with constraint embedded CPU power, future work should focus on novel heuristics for the ATDTSP for more efficient yet near-optimal solutions in large domains.

Acknowledgment

The authors would like to thank Malte Helmert and Vincenzo Bonifaci for valuable input. This work has been supported by the German Research Foundation (DFG) under contract number SFB/TR-8.

References

[1] H. Choset, “Coverage for robotics – A survey of recent results,” *Annals of Mathematics and Artificial Intelligence*, vol. 31, pp. 113–126, 2001.

[2] E. Sisbot, L. Marin-Urias, R. Alami, and T. Simeon, “A human aware mobile robot motion planner,” *IEEE Transactions on Robotics*, vol. 23, no. 5, pp. 874 – 883, October 2007.

[3] J. Müller, C. Stachniss, K. O. Arras, and W. Burgard, “Socially inspired motion planning for mobile robots in populated environments,” in *International Conference on Cognitive Systems (CogSys’08)*, Karlsruhe, Germany, 2008.

[4] T. Kruse, A. Kirsch, E. Sisbot, and R. Alami, “Exploiting human cooperation in human-centered robot navigation,” in *IEEE International Symposium in Robot and Human Interactive Communication (ROMAN)*, Viareggio, Italy, 2010.

[5] R. Alami, A. Clodic, V. Montreuil, E. Sisbot, and R. Chatila, “Toward human-aware robot task planning,” in *AAAI Spring Symposium “To Boldly Go Where No Human-Robot Team Has Gone Before”*, Stanford, USA, 2006.

[6] F. Broz, I. Nourbakhsh, and R. Simmons, “Planning for human-robot interaction using time-state aggregated pomdps,” in *Proc. of the National Conference on Artificial Intelligence (AAAI)*, Chicago, USA, 2008.

[7] M. Cirillo, L. Karlsson, and A. Saffiotti, “A human-aware robot task planner,” in *Proc of the 11th Int Conf on Automated Planning and Scheduling (ICAPS’09)*, Thessaloniki, Greece, 2009.

[8] E. Kruse and F. M. Wahl, “Camera-based observation of obstacle motions to derive statistical data for mobile robot motion planning,” in *Proc. of the IEEE Int. Conf. on Robotics & Automation (ICRA)*, Leuven, Belgium, 1998.

[9] A. Bruce and G. Gordon, “Better motion prediction for people-tracking,” in *Proc. of the IEEE Int. Conf. on Robotics & Automation (ICRA)*, Barcelona, Spain, 2004.

[10] M. Bennewitz, W. Burgard, G. Cielniak, and S. Thrun, “Learning motion patterns of people for compliant robot motion,” *Int. Journal of Robotics Research*, vol. 24, no. 1, 2005.

[11] A. T. Ihler and P. J. Smyth, “Learning time-intensity profiles of human activity using non-parametric Bayesian models,” in *Neural Information Processing Systems 19*. Cambridge, MA: MIT Press, 2007, pp. 625–632.

[12] M. Luber, G. D. Tipaldi, and K. O. Arras, “Spatially grounded multi-hypothesis tracking of people,” in *Proc. of the ICRA’09 Workshop on People Detection and Tracking*, Kobe, Japan, 2009.

[13] ——, “Place-dependent people tracking,” *International Journal of Robotics Research*, vol. 30, no. 3, March 2011.

[14] G. D. Tipaldi and K. O. Arras, “I want my coffee hot! Learning to find people under spatio-temporal constraints,” in *Proc. IEEE International Conference on Robotics and Automation (ICRA’11)*, Shanghai, China, 2011.

[15] R. P. Bonasso, D. Kortenkamp, D. P. Miller, and M. Slack, “Experiences with an architecture for intelligent, reactive agents,” *Journal of Experimental and Theoretical Artificial Intelligence*, vol. 9, pp. 237–256, 1995.