Task-Aware Waypoint Sampling for Planning Robots

Sarah Keren[1,2*], Gerard Canal[3] and Michael Cashmore[4]
[1]Harvard University, School of Engineering and Applied Sciences
[2]Hebrew University, School of Computer Science and Engineering
[3]Department of Informatics, King's College London
[4]University of Strathclyde
skeren@seas.harvard.edu, gerard.canal@kcl.ac.uk, michael.cashmore@strath.ac.uk

Abstract

To achieve a complex task, a robot often needs to navigate in a physical space to complete activities in different locations. For example, it may need to inspect several structures, making multiple observations of each structure from different perspectives. Typically, the positions from which these activities can be performed are represented as *waypoints* – discrete positions that are sampled from the continuous physical space. Existing approaches to waypoint selection either iteratively consider the entire space or each activity separately, which can lead to task planning problems that are more complex than is necessary or to plans of compromised quality. We offer an approach that produces more efficient plans by performing a one-time computation of the connectivity graph and by prioritizing waypoints from which multiple activities can be performed. In addition, we support user specified *performance preferences* that represent preferences a system operator may have about the generated task plan but that cannot be directly represented in the map used for navigation, such as areas near doorways where it is preferable that the robot does not stop to perform activities. We demonstrate the performance benefits of our approach on simulated manufacturing tasks in an automated factory.

Introduction

Robots are typically assigned complex missions that require performing various activities in different locations. To complete the overall mission, a mobile robotic agent must reason over a physical space and decide which activities must be performed as well as how to navigate between the positions from which it can perform the activities. Since the physical space is continuous, task planning is typically performed using an abstraction of the space. A common approach is to use a finite set of discrete *waypoints* that represent specific configurations (positions) in the space. The waypoints represent nodes in a *probabilistic road map* (PRM) (Kavraki et al. 1996), in which the edges represent feasible paths between waypoints and their estimated navigation costs.

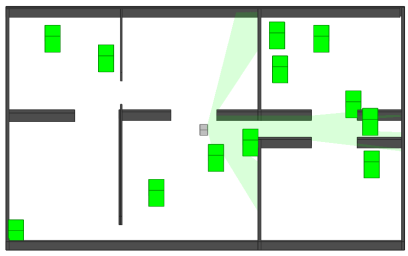
In generating waypoints there exists a trade-off between the complexity and completeness of the resulting representation. Intuitively, a small set of waypoints is a coarse abstraction of the physical space that limits the positions that can be used to perform the task, potentially leading to lower quality plans or unsolvable problems. On the other hand, a larger set of waypoints will lead to a higher probability of finding a plan, but may exceed the capacity of the task planner.

Generally, there are two common approaches to waypoint generation. With *Fixed Waypoint Generation* (FWPG), a single waypoint is generated for each possible activity (Edelkamp et al. 2018). This approach provides a good coverage of the space, but may yield problems that are too big for the planner to handle. On the other end, with a *Pure PRM* (PPRM) approach (Kavraki et al. 1996), a PRM is created by randomly sampling waypoints. The size of the graph can be set to comply with the planner's capacity, but since the placement of waypoints is random, the coverage of the space may be insufficient, which requires iteratively generating a new PRM, until a solution is found.

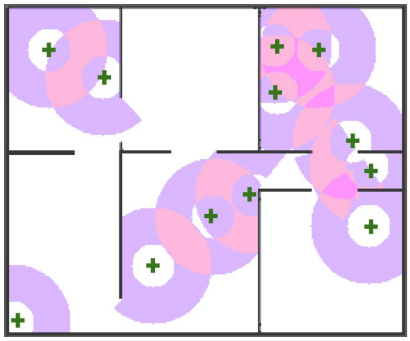
In this work we suggest a novel approach to waypoint generation which bridges the gap between the two common approaches to sampling and provides good coverage of the space, while accounting for the planner's capacity. Our approach, which we call Task-Aware Waypoint Sampling (TAWS), first generates a very *Dense PRM* (DPRM) that captures a fine representation of the reachability information in the space, and includes with very high probability a representation of a solution to the task. To find a plan, waypoints are sampled from the DPRM according to probabilities that are induced by the task description. If a plan cannot be found with the sampled set of waypoints, a new bigger set is re-sampled from the DPRM. Similarly, if the planner's capacity is exceeded, a smaller set is re-sampled.

As with PPRM, TAWS relies on sampled waypoints for task planning. However, it avoids the need to reconstruct a PRM at every planning iteration. As with FWPG, TAWS incorporates domain knowledge into the waypoint selection process, but instead of using it to fix a set of waypoints, it uses it to set the probabilities according to which waypoints are sampled from the DPRM. This not only can be used to increase efficiency by prioritizing waypoints from which more than one activity can be performed, but it also makes it possible to account for *performance preferences*, arbitrary user-defined preferences over positions from which the robot can perform its activities but that cannot be directly represented in the map used by the robot for navigation. Such preferences can reflect, for example, social norms (e.g.,

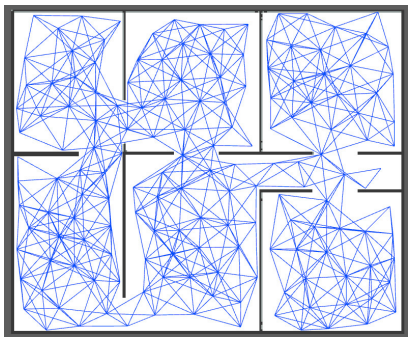
*Contact Author



(a) 3D representation of an example factory environment, with the robot (gray square in the center), and the machines in green.

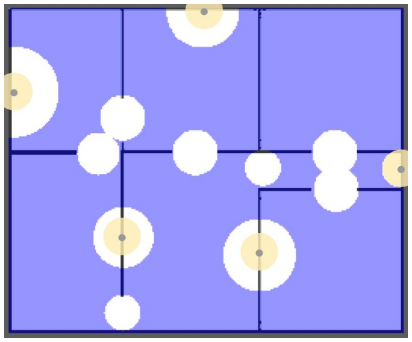


(b) A map of the physical space highlighted by the areas from which the robot can interact with each of 12 different machines (green crosses).

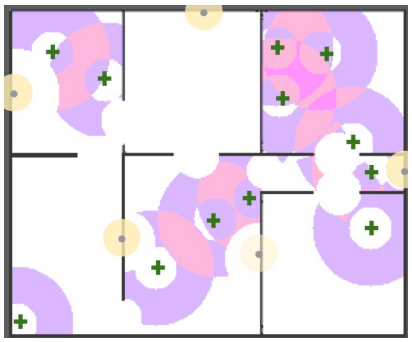


(c) The dense PRM imposed over the map. Each collision-free edge (blue lines) connects two waypoints (configurations).

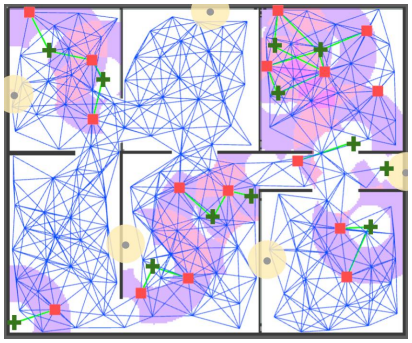
Figure 1: An example setting from the RCLL domain



(a) Map of the physical space specifying first aid stations (yellow) and doorways (white), which represent performance preferences of areas where robots should not stop.



(b) Map representing the combination of the performance preferences and the machine interaction areas.



(c) Waypoints (red squares) are sampled from the DPRM for task planning. Machines that can be accessed from each waypoints are highlighted by green edges.

Figure 2: Integrating Performance preferences into the sampling process

areas where some social event is taking place and should be avoided by noisy robots), safety and efficiency constraints (e.g., a carpeted area that is hard for robots to traverse, or an area near a first aid kit robots shouldn't block), and areas where performance is enhanced (e.g., it is preferable for a robot to operate near a charging station since it will be able to recharge and recover if it's battery is unexpectedly depleted). TAWS can account for these arbitrarily defined preferences by changing the probability of sampling certain positions according to the specified preferences.

Example 1 *Consider the scenario used for the Robocup Logistics League (RCLL) (Niemueller, Lakemeyer, and Ferrein 2015) and depicted in Figure 1a in which robots must navigate in a factory in order to collect items from a set of machines and deliver them to their destinations. In such scenarios, if the environment is fixed and known, FWPG can be used to prescribe a finite set of waypoints, including a waypoint for each activity robots may need to perform, such as picking up an item from a machine. The result may include many redundant waypoints or waypoints that cannot be connected, or lead to inefficient plans since each activity is considered separately. Also, FWPG does not allow for iterations if the planner's capacity is exceeded. On the other hand, the PPRM approach might require many iterations to solve problems of realistic size or produce inefficient plans since the size of the PRM and the accuracy of its cost estimations is limited by the planner's capacity.*

TAWS takes a hybrid approach by first producing a single dense PRM (DPRM) that is used throughout the search for a plan. This is likely to lead to plans that are more efficient at execution since the DPRM provides more accurate navigation cost estimates. Moreover, by prioritizing positions from which more than one machine can be accessed or waypoints at machines from which more than one item, it minimises extraneous navigation costs. It can also be used to make sure robots do not block doors or increase the probability that robots avoid slippery areas of the factory floor.

Our contributions are threefold. First, we suggest to perform a one-time computation of a connectivity graph in a given environment, thus decoupling between the connec-

tivity analysis and the task planning process. This new approach to modeling planning problems for robots navigating in physical space, can find the minimal number of waypoints that are needed to solve a given problem and produce more reliable plans. Secondly, we suggest to increase efficiency by prioritizing waypoints from which multiple activities can be performed. More generally, we support any performance preferences that induce the waypoint sampling probabilities. Finally, we use a set of simulated manufacturing tasks in an automated factory to show that our approach is able to scale to larger tasks, and produce more efficient plans when compared to current approaches.

Related Work

Typical robotic control systems must determine which activities must be performed, and how to navigate between those activities. Common approaches to planning for robots combine *motion planning* and *task planning* (Gravot, Cambon, and Alami 2005; Cambon, Alami, and Gravot 2009; Kaelbling and Lozano-Pérez 2011; Dornhege, Hertle, and Nebel 2013; McMahon and Plaku 2014; Srivastava et al. 2014; Toussaint 2015; Fernández-González, Karpas, and Williams 2017; Canal et al. 2018). Motion planning is the process of finding a plan to perform a basic activity, such as picking up an item or moving between two adjacent locations. Task planning is the search for a sequence of activities that is predicted to achieve the goal, while minimising duration and other costs such as energy use.

When planning in complex scenarios, task planning typically uses an abstraction of the space. One way to abstract the space is by using geometric computations that help the high-level planner make appropriate choices. For example, (Kaelbling and Lozano-Pérez 2011) handle the integration of continuous geometric planning with task planning by using geometric "suggesters", which construct configurations dynamically during an "aggressively" hierarchical planning process. Another approach integrates the motion planner's geometric search for positions into the symbolic forward-search of a task planner. For example, (Cambon, Alami, and Gravot 2009) devise an integrated task and motion planner that reasons about geometric constraints that describe the positions from which it is possible to accomplish some action as sub-manifolds of the configuration space of the robot. These sub-manifolds are mapped within the solver to high level symbols. McMahon and Plaku (McMahon and Plaku 2014) combine task planning with sampling-based motion planning to plan trajectories that satisfy constraints in LTL.

Another approach to abstraction uses *waypoints* that represent discrete positions (Cashmore et al. 2014; McMahon and Plaku 2014; Edelkamp et al. 2018). This reduces the complexity of the problem, making it possible to focus on the task-planning aspect of the problem, i.e., selecting and scheduling activities, while using heuristic approximations to estimate navigation and motion costs. Once a high-level task plan is produced, motion planning is delegated to a dedicated low-level motion planner.

In this paper we focus on waypoint-based approaches and and on the selection of waypoints for task planning. Waypoints can be selected randomly, for example using a PRM (Kavraki et al. 1996), or can be generated using knowledge of the space and task (Plaku and Hager 2010; Edelkamp et al. 2018). The disadvantage of the random approach is that in order to ensure coverage of all interesting areas, a large number of waypoints might be required. For simple problems, such as inspection missions (Cashmore et al. 2014), this can be feasible. However, in a more complex task this will result in problems that are too hard to solve within a reasonable time.

On the other hand, generating fixed waypoints means that for each affordance in the physical space (corresponding to a non-navigation action that the robot might make) a set of waypoints of fixed size is generated. To demonstrate, in the RCLL setting in Example 1, the approach by (Edelkamp et al. 2018) generates a separate waypoint for each item pickup activity by randomly sampling a position around the machine the item is positioned at, even if the machine has more than one item. These waypoints are connected together using a PRM, adding additional waypoints to cover the space, if needed. The resulting representation is guaranteed to include a solution. However, as the number of activities increases, the corresponding task planning problem may unjustifiably exceed the planner's capacity, even though it may contain many redundant waypoints.

Moreover, the fixed approach relies on domain knowledge that may not be always available. In domains with complex configuration spaces, it may not be possible to explicitly prescribe in advance the region from which an activity can be performed, making it necessary to sample waypoints and determine whether an activity is achievable from them. For example, consider a mobile base carrying an arm with 5-degrees of freedom, performing a picking task in a cluttered scene. Due to the clutter, it is not possible describe in advance a region for the base from which it is guaranteed that the arm can reach the target. However it is possible instead to sample a position and orientation for the base and use a motion planner to determine if there is a collision-free path for the arm to the target.

We suggest a new approach to waypoint sampling that combines the benefits of random sampling with the use of domain knowledge. TAWS is in an anytime approach that iteratively improves solution quality and that is agnostic to the specific task and motion planners used. Most notably, all approaches mentioned above only consider geometric constraints that can be imposed on the waypoint selection process. TAWS is the first approach that also accounts for arbitrary performance preferences, thus making it possible to prioritize or discourage specific behaviors.

Task Aware Waypoint Sampling (TAWS)

The input to the waypoint sampling problem is a tuple $p = \langle M; A; F \rangle$, where

M is the set of configurations $m \in \mathbb{R}^n$, where n represents the dimensions of the space,

A is a set of non-navigation activities that can be performed, and

F is a set of performance preferences.

Each sampled waypoint corresponds to a configuration $m \in \mathbb{R}^n$. Each activity $a \in A$ is associated with a function $!_a : M \to [0,1]$ specifying the probability of successfully executing a from configuration m. Typically, these probability functions are generated using prescribed templates for each activity type the robot can perform. Each preference f is a score function $f : M \to \mathbb{R}$ that is used to describe areas from which it is (un)desirable that the robot operates.

In Example 1, a robot navigates the factory floor and can interact with a number of stationary machines. For simplicity, we ignore the orientation of the robot. The configuration space is therefore described by a 2-dimensional map (the floorplan) ,i.e., $m \in \mathbb{R}^2$. The activity set represents the possible interactions of the robot with each machine (e.g., picking up an item from a machine). The function $!_a : \mathbb{R}^2 \to \{0,1\}$ of each machine is defined by a prescribed template that defines the probability of successfully completing the activity in a given configuration, taking into account adjacent obstacles (e.g., walls) and the extent of the robot's arms. Figure 1a shows an example setting with 12 stationary machines, with a single activity per machine. In this setting, each the activity is deterministically mapped to configurations from which it can be achieved, which forms a ring around the machine (Figure 1b). The areas in pink are those from which more than one activity can be achieved. The performance preferences can prioritize sampling from these areas to produce more efficient plans.

Sampling Procedure

Our TAWS approach decouples the connectivity analysis of a domain and the task planning process. First, it generates a Dense PRM (DPRM) over the configuration space. The process starts from the robot's initial position. The PRM is constructed by iteratively selecting a waypoint from the existing PRM for expansion. A set of new waypoints is cast from the chosen waypoint. Waypoints that are not in collision, are added to the graph. The coordinates of each node and the length of each edge are stored so that they can be used to estimate the cost of traveling between the waypoints during the task planning process. A DPRM for the factory domain is shown in Figure 1c.

After completing the generation of the DPRM, the iterative task planning stage begins. At each iteration, a number x of waypoints is selected from the DPRM and sent to the task planner. If the planner is unable to solve the problem within a time bound, the number of waypoints x is decremented. If the planner claims that the problem is unsolvable, the number is incremented. If a plan is found, it is recorded, and the number of waypoints is incremented in order to find a more efficient solution. This process is repeated iteratively until timeout is reached. TAWS automatically finds the number of waypoints that can be handled by the task planner.

Note that the waypoints are iteratively selected from the DPRM which contains all the connectivity information, and not from the underlying map. The selection of waypoints at each iteration is done according to the following procedure:

1. A sampling probability is assigned to each waypoint in the DPRM, using a task specific score which is based onperformance preferences and discussed in detail in the next section.
2. A waypoint is sampled from the DPRM and added to the task plan's model. The distance between the new waypoint and all existing waypoints is calculated by finding the shortest path through the edges of the DPRM. This value is added to the planning model as an estimate of the path costs. In addition, the planing model is updated with information about all activities that can be performed from the new waypoint.
3. The score function is updated to reduce the probability of sampling more waypoints near the one sampled.

If a task plan is found it is passed to the executor, which uses a motion planner to compute plans between the waypoints of the plan.

Combined Score

In order to account for both the probability of successfully accomplishing an activity from a given configuration as well as the performance preferences, we suggest using a single *combined score* (CS) that associates a score to each waypoint (corresponding to a sampled configuration) in the DPRM. This score, that can be defined arbitrarily to account for different settings, is normalized over the waypoints in the DPRM, and is used to specify the probability of sampling each waypoint for inclusion in the task planning problem.

Specifically, the CS we suggest in Equation 1 is designed for the type of settings we consider here and uses a weighted sum over the different activities, while considering preferences multiplicatively.

$$CS(m) = \prod_{f \in F} f(m) \sum_{a \in A} !_a(m) \qquad (1)$$

Our scoring approach increases the score (and corresponding probability) of waypoint m from which multiple actions can be achieved by summing the probabilities $!_a(m)$ of successfully completing each activity from m.

The application specific preferences can be used to account for anything from breaking ties between otherwise equally probable waypoints, to imposing hard constraints that prevent sampling in certain regions. The former case can be achieved by setting $f(m)$ to vary between 1 and 1 for some small value , so that the score of a waypoint is scaled down by up to 1 in areas where it is preferred not to sample a waypoint. This may be relevant, for example, in settings where a noisy robot should avoid getting close to a working station of a human worker. In the latter case, hard constraints, such as for ensuring that a robot never blocks access to a first-aid station, can be enforced by setting $f(m)$ to 0 in the critical area. This ensures that no waypoint can be sampled in that area, as its sampling probability will be 0.

In Figure 2a, critical areas represent doorways and first aid stations, where a robot shouldn't stop. The combined score is assigned according to CS, the cost function in Equation 1, that considers both the hidden preferences and the activity information (Figure 2b). The score is normalized and used to sample a set of waypoints, which is used to find a plan for the task (Figure 2c).

Evaluation

Our empirical evaluation was designed to answer two main questions: (1) what is the benefit of using a Dense PRM (DPRM) over existing approaches to waypoint selection, and (2) what is the best way to account for performance preferences in the waypoint selection process.

To address these questions we used a dataset that consisted of automated factory scenarios similar to those used for the Robocup Logistics League (RCLL) (Niemueller, Lakemeyer, and Ferrein 2015) and described in Example 1. In these scenarios, a robot must move between machines and benches to pick up and place work-pieces. The work-pieces are combined at the machines to produce a complete order that can be delivered at a delivery window. The problem description is temporal, such that each action is described by it's estimated duration. In contrast with the RCLL scenario, orders here do not change during execution nor are constrained by deadlines. We varied the number of machines from 1 to 40 with 1 10 work-pieces per order. Each machine could have several work-pieces, and the same work-piece could be found in different machines. For each machine count, we generated 10 different problems, varying the types and positions of objects, for a total of 400 problems.

The Benefits of Using a DPRM

To assess the benefits of using a DPRM, we compared TAWS that produces a single DPRM as a preprocessing step, against FWPG and PPRM. We used the FWPG implementation from (Edelkamp et al. 2018), in which each activity is associated with a template which prescribes the area from which it can be performed. A waypoint is randomly sampled from this area for each activity. For PPRM we used the implementation in (Kavraki et al. 1996), which reconstructs a (sparse) PRM at every iteration. In this part of our evaluation, no performance preferences were considered beyond the task description, so we are only evaluating the benefit of decoupling the connectivity analysis from the task planning process with a one time generation and reuse of the DPRM.

We embedded all three approaches in ROS using the ROSPlan framework (Cashmore et al. 2015) with the POPF temporal planner for task planning (Coles et al. 2010). All approaches solved the same problems with a total time bound of 10 minutes to compute a solution. For FWPG, this meant 10 minutes of planner time. For PPRM and TAWS, each call to the planner was limited to 10 seconds[1]. The initial sample set size for TAWS and PPRM was 1 and the sampling step size was 4 (for FWPG it is fixed by definition).

To compare the performance of the approaches we measured (1) the number of instances that were solved by each approach within different time bounds, (2) the quality of the first and best solutions found within seven time bounds (5 and 10 seconds and 1, 2, 4, 8 and 10 minutes), and (3) the amount of time it took to compute the first and best solutions. A quality of a plan is measured according to the total duration of the plan (shorter is better). For each approach, Table 1 shows the number of instances solved within each time bound. For problems solved by all approaches, the table shows the mean time to solution and mean plan duration in seconds. The standard deviation is indicated in brackets.

The results show that while FWPG solves the largest number of problems within the 10 minute bound, and that the most successful approach changes within the different time intervals. In terms of computation time, FWPG outperforms the other two approaches on the instances solved by all approaches. The notable achievement of TAWS is in terms of plan quality. For all time intervals, TAWS finds shorter solutions, with up to 40% reduction.

Since PPRM only solved a limited number of instances, the results in Table 1 only reflect the performance of the approaches on the smallest instances. We therefore compared the performance of TAWS against FWPG on instances solved by both of these approaches. In Figure 3 we compare plan duration, and in Figure 4 we compare the time to solution in seconds for an increasing problem size for FWPG, and for the first and best solutions of TAWS. In Figure 3, where the instances below the line are those for which TAWS achieved a better result, we can see that TAWS achieved better plan quality over most instances, with an average 88% plan duration compared to FWPG across all problems solved by both approaches.

We have seen in Table 1 that for the small instances solved by all three approaches, FWPG outperforms the other two approaches in terms of the mean computation time. By investigating problems according to their size, Figure 4 reveals FWPG's computation time tends to be fixed around either 600 or 5 seconds for any problem size. In contrast, TAWS's time to best solution increases with problem size, and is much lower than FWPG for the majority of instances. TAWS reaches the first solution in 17% and the best solution in 62% of the time taken by FWPG solution.

These results can be explained by recalling that FWPG generates a single and potentially unnecessarily large problem. In contrast, TAWS is an anytime approach, that iteratively generates and sends to the planner representations of the problem of increasing size. Our results show that despite the fact that TAWS is only allowed 10 seconds of planner time per iteration, the anytime approach is still able to find solutions more quickly than the fixed approach and those solutions are of higher quality.

Accounting for Performance Preferences

To examine the best way to account for performance preferences we introduced two kinds of preferences to the automated factory domain: *hard constraints*, and *soft preferences*. *Hard constraints* represent restrictions that cannot be violated, such as disallowing dwelling at locations (waypoints) in doorways and near first aid kits, as exemplified in Figure 2a. These constraints are implemented by assigning a zero probability for sampling in specific areas of the model. Soft preferences represent areas of the model in which it is preferable that the robots avoid dwelling at, such as areas

[1]Due to space considerations we only report here some of our results. In practice, we experimented with different time allocations, and different settings. The code and benchmark set can be found in (*URL omitted for blind review*). The additional results can be found in the appendix.

	Solved Instances			Mean Time to Solution (s)			Mean Plan Duration (s)		
Time elapsed	PPRM	FWPG	TAWS	PPRM	FWPG	TAWS	PPRM	FWPG	TAWS
5 secs	**6**	0	0	-	-	-	-	-	-
10 secs	16	**20**	14	**3.88 (0.00)**	9.47 (0.36)	9.07 (0.30)	5.80 (0.80)	**5.00 (0.00)**	**5.00 (0.00)**
1 min	27	**103**	65	17.41 (14.66)	**11.14 (1.11)**	18.95 (7.48)	7.12 (1.27)	12.99 (4.12)	**5.00 (0.00)**
2 min	40	**113**	103	36.40 (31.64)	**11.44 (1.24)**	31.22 (20.60)	8.48 (2.31)	14.19 (3.62)	**5.33 (0.64)**
4 min	47	113	**159**	58.76 (45.79)	**11.72 (1.50)**	48.58 (44.78)	10.01 (3.54)	14.03 (3.79)	**6.21 (2.10)**
8 min	56	113	**261**	110.68 (99.93)	**11.60 (1.43)**	54.09 (51.10)	10.38 (3.42)	13.69 (3.88)	**6.17 (2.05)**
10 min	63	**338**	300	174.31 (174.12)	**11.39 (1.42)**	53.55 (52.43)	10.63 (3.37)	13.35 (4.55)	**6.25 (2.15)**

Table 1: Performance per approach; standard deviation in brackets.

	PPRM	FWPG	TAWS
Number of solved instances	63	**337**	304
Mean time to first solution	276.38 (284.23)	57.13 (52.31)	**15.96 (4.80)**
Mean first solution quality - duration	10.23 (2.76)	**8.94 (4.51)**	10.53 (4.46)
Mean time to best solution	276.38 (284.23)	**57.13 (52.31)**	61.00 (61.68)
Mean best solution quality - duration	10.23 (2.76)	8.94 (4.51)	**6.05 (1.86)**

Table 2: Accounting for hard constraints.

	PPRM	FWPG	TAWS
Number of solved instances	61	**335**	311
Mean Time to First solution	215.84 (209.21)	59.66 (56.71)	**16.52** (5.00)
Mean first solution quality - duration	**10.21** (2.70)	10.46 (4.96)	10.76 (4.21)
Mean First solution quality - preferences	9.80 (3.09)	16.60 (9.56)	**0.63** (1.08)
Mean Time to best solution	215.84 (209.21)	59.66 (56.71)	**42.74** (33.65)
Mean Best solution quality - duration	10.21 (2.70)	10.46 (4.96)	**5.61** (1.11)
Mean Best solution quality - preferences	9.80 (3.09)	16.60 (9.56)	**0.08** (0.16)

Table 3: Accounting for constraints and preferences

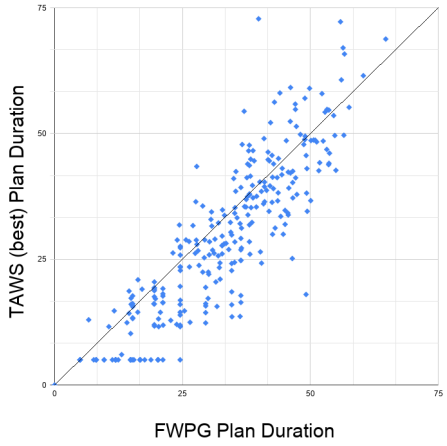


Figure 3: Comparison of best plan quality (plan duration in seconds) for problems solved by both TAWS(vertical axis) and FWPG (horizontal axis). Points below the line indicate higher quality for TAWS.

near workbenches. This is implemented by allowing the user to associate a score to each area of the model which reduces the probability of sampling a waypoint from that area. As depicted in our score function in Equation 1, we also account for *overlap information* by increasing the probability of sampling a waypoint according to the number of activities that can be performed from it. In the factory domain, this prioritizes waypoints from which more than one machine can be accessed and waypoints at machines from which more than one work-piece can be collected.

Since neither FWPG nor PPRM account for performance preferences, we extended both approaches to do so. For FWPG, in which waypoints are chosen randomly from within the area from which each activity can be performed, we used performance preferences to set the probability of sampling a specific position within each area. Performance preferences were used in two different stages of the PPRM generation. The first variant uses the performance preferences to influence which existing node in the PRM is chosen for expansion, while the alternative uses them during expansion to set the probability of sampling a new waypoint around the selected waypoint. As the results were similar for both variants we only report the results of the latter.

In addition to assessing plan quality according to its duration and keeping track of the times to solution, we used the following measure to evaluate a plan .

$$P(\ ) = \sum_{m 2 M_{\pi}} \left(dur(m) \sum_{f 2 F} (1 \quad f(m)) \right) \qquad (2)$$

where M are the set of waypoints visited in the plan, $dur(m)$ is the total time spent at waypoint m, and $f(m)$ 2 $f0; 1g$ is the normalized performance preferences score of that waypoint's position. This measure penalizes plans according to the time spent in undesirable locations.

Note that the performance preferences score is accounted for by the waypoint selection process and not directly modelled in the resulting task planning domain. This allows our approach to be used with any task planner, such as temporal, probabilistic, or contingent planners. Specifically, POPF optimises plan duration.

Table 2 shows the results achieved for instances for which only hard constraints were specified. For each approach, the table shows the number of instances solved within the time bound. For problems solved by all three approaches, we show the mean time to the first and best solution and the quality of the solutions in terms of duration. Table 3 shows the results for instances for which both hard constraints and soft preferences were specified. In addition to measuring plan duration, we include the mean performance score according to Equation 2 (the breakdown according to timestamps is shown in the appendix).

The results show that adding performance preferences does not have a substantial effect on the number of problems solved by each approach. For instances with only hard constraints, the mean quality of the first solution is best

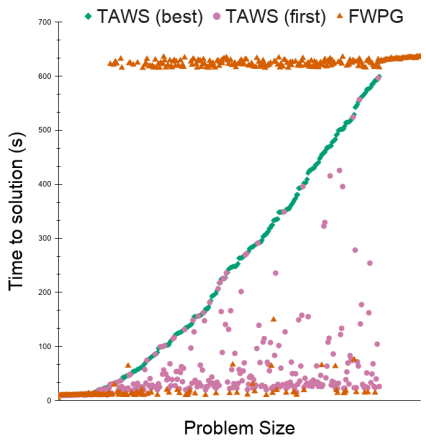


Figure 4: Comparison of time to solution for FWPG and time to first and best solutions for TAWS, on problems solved by both approaches, for increasing problem sizes.

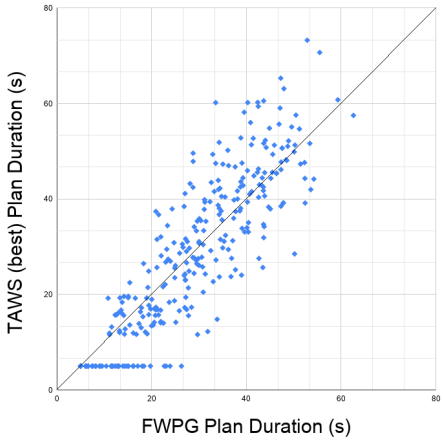


Figure 6: Comparison of plan duration for FWPG and TAWS in problems with soft preferences. Plan quality in terms of plan duration is comparable.

for FWPG, but TAWS reaches the first solution much faster. Moreover, TAWS achieves the best plan quality within the time bound. As shown in Table 3, when soft constraints are added, TAWS achieves both the lowest plan duration and the best preference score according to Equation 2.

Again, since PPRM solves only a limited number of instances, our analysis in tables 2 and 3 only accounts for smaller instances. In figures 5 and 6 we therefore exclude PPRM, and compare solution quality for the instances solved by FWPG and TAWS.

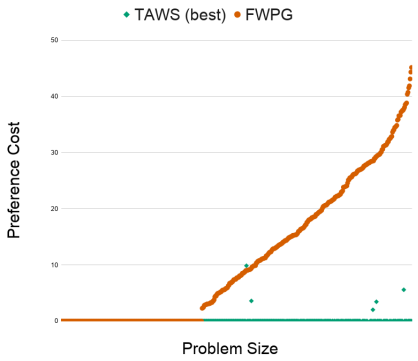


Figure 5: Comparison of preference cost for FWPG and TAWS, for increasing problem sizes.

The results in Figure 5 show that FWPG is limited in its ability to account for the performance preferences when compared to TAWS, which manages to achieve a penalty of 0 (according to Equation 2) for most instances. In Figure 6 we see that the better score in terms of performance preferences achieved by TAWS doesn't compromise the plans' quality in terms of plan duration.

The superior performance of TAWS compared to FWPG is due to fact that TAWS samples a smaller set of waypoints that respect the performance preferences. In contrast, FWPG may result in many redundant waypoints that do not always respect the performance preferences, not only making for a more complex problem, but also increasing the probability that the resulting plan will make use of those undesirable areas.

Conclusion

We presented Task-Aware Waypoint Sampling (TAWS) as a new approach to selecting waypoints for task planning. TAWS's novelty is in the way it decouples the connectivity analysis of a domain from the task planning process, and its ability to account for user defined performance preferences.

The connectivity information is captured through a dense PRM, which is generated once and reused through the anytime iterative planning process to estimate navigation costs. The task planning model at each iteration is constructed by sampling waypoints from the dense PRM according to probabilities that are defined by both the activities and performance preferences.

Our empirical evaluation on a set of automated factory problems shows that TAWS finds solutions that maximize compliance with the specified preferences, without compromising computation time and the robot's ability to achieve the goal efficiently.

In the future, we intend to evaluate TAWS on other settings beyond the factory use case. Specifically, we intend to investigate exploration scenarios, in which the sampling probability of TAWS can be used to specify areas in which a more meticulous search is desired. Also, in this work, desirable behaviors were induced by changing the robot's task planning procedure. As a next step, we intend to account for settings in which robots are treated as 'black boxes' and their inner implementation cannot be modified. In such settings, their behavior can instead be influenced by changing the information that is provided to them, such as the map that is used for navigation.

References

Cambon, S.; Alami, R.; and Gravot, F. 2009. A Hybrid Approach to Intricate Motion, Manipulation and Task Planning. *The International Journal of Robotics Research* 28 (1): 104–126.

Canal, G.; Pignat, E.; Alenyà, G.; Calinon, S.; and Torras, C. 2018. Joining high-level symbolic planning with low-level motion primitives in adaptive HRI: application to dressing assistance. In *IEEE International Conference on Robotics and Automation (ICRA)*, 3273–3278. ISSN 2577-087X. doi: 10.1109/ICRA.2018.8460606.

Cashmore, M.; Fox, M.; Larkworthy, T.; Long, D.; and Magazzeni, D. 2014. AUV mission control via temporal planning. In *Proceedings - IEEE International Conference on Robotics and Automation*. ISSN 10504729. doi: 10.1109/ICRA.2014.6907823.

Cashmore, M.; Fox, M.; Long, D.; Magazzeni, D.; Ridder, B.; Carrera, A.; Palomeras, N.; Hurtós, N.; and Carreras, M. 2015. Rosplan: Planning in the robot operating system. In *Proceedings International Conference on Automated Planning and Scheduling, ICAPS*, volume 2015-Janua. ISBN 9781577357315. ISSN 23340843.

Coles, A.; Coles, A.; Fox, M.; and Long, D. 2010. Forward-Chaining Partial-Order Planning. In *ICAPS*, 42–49.

Dornhege, C.; Hertle, A.; and Nebel, B. 2013. Lazy evaluation and subsumption caching for search-based integrated task and motion planning. In *IROS workshop on AI-based robotics*.

Edelkamp, S.; Lahijanian, M.; Magazzeni, D.; and Plaku, E. 2018. Integrating Temporal Reasoning and Sampling-Based Motion Planning for Multigoal Problems With Dynamics and Time Windows. *IEEE Robotics and Automation Letters* 3(4): 3473–3480.

Fernández-González, E.; Karpas, E.; and Williams, B. C. 2017. Mixed Discrete-Continuous Planning with Convex Optimization. In *AAAI*.

Gravot, F.; Cambon, S.; and Alami, R. 2005. aSyMov: A Planner That Deals with Intricate Symbolic and Geometric Problems. *Robotics Research. The Eleventh International Symposium. Springer Tracts in Advanced Robotics* 15: 100–110.

Kaelbling, L. P.; and Lozano-Pérez, T. 2011. Hierarchical task and motion planning in the now. In *2011 IEEE International Conference on Robotics and Automation*, 1470–1477.

Kavraki, L. E.; Svestka, P.; Latombe, J. .; and Overmars, M. H. 1996. Probabilistic roadmaps for path planning in high-dimensional configuration spaces. *IEEE Transactions on Robotics and Automation* 12(4): 566–580.

McMahon, J.; and Plaku, E. 2014. Sampling-based tree search with discrete abstractions for motion planning with dynamics and temporal logic. In *2014 IEEE/RSJ International Conference on Intelligent Robots and Systems, Chicago, IL, USA, September 14-18, 2014*, 3726–3733.

Niemueller, T.; Lakemeyer, G.; and Ferrein, A. 2015. The RoboCup logistics league as a benchmark for planning in robotics. *Planning and Robotics (PlanRob-15)* 63.

Plaku, E.; and Hager, G. D. 2010. Sampling-Based Motion and Symbolic Action Planning with geometric and differential constraints. In *IEEE International Conference on Robotics and Automation*, 5002–5008.

Srivastava, S.; Fang, E.; Riano, L.; Chitnis, R.; Russell, S.; and Abbeel, P. 2014. Combined task and motion planning through an extensible planner-independent interface layer. In *2014 IEEE International Conference on Robotics and Automation (ICRA)*, 639–646.

Toussaint, M. 2015. Logic-Geometric Programming: An Optimization-Based Approach to Combined Task and Motion Planning. In *IJCAI*, 1930–1936.