Interaction Templates for Multi-Agent Systems

James Motes[1] Read Sandström[1] Will Adams[2] Tobi Ogunyale[3] Shawna Thomas[1] Nancy M. Amato[1]

Technical Report TR18-002

Abstract

This paper describes a framework for multi-robot motion planning problems that require or utilize interactions between robots. Solutions need to consider interactions between agents on a motion planning level to determine the feasibility and cost of each agent's portion of the task. Modeling these problems with current task and motion planning approaches typically requires reasoning about the possible interactions and checking many of the possible robots combinations when searching for a solution.

We present a multi-robot motion planning method called *Interaction Templates* (ITs) which offloads certain types of robot interactions from the task planner to the motion planner. ITs encode iterations between a set of robots and are tiled and connected to robots' individual roadmaps. The resulting combined roadmap allows interactions to be considered by the motion planner. We apply ITs to homogeneous and heterogenous robot teams under both required and optional cooperation scenarios which previously required a task planning method. We show improved performance over the standard TMP approach.

Index Terms

Interaction Template, multi-agent systems, motion planning

[1]James Motes, Read Sandström, Shawna Thomas, and Nancy M. Amato are with the Parasol Lab, Department of Computer Science and Engineering, Texas A&M University, College Station TX 77840, USA. jmotes, readamus, sthomas, amato@tamu.edu

[2]Will Adams is with the University of Texas, Austin TX, USA.

[3]Tobi Ogunyale is with Georgia State University, Atlanta GA, 30302, USA. togunyale1@student.gsu.edu

I. Introduction

Multi-agent systems often provide significant advantages over utilizing a single robot. Heterogeneous systems have the ability to perform numerous tasks requiring different hardware, while homogeneous systems can increase work throughput by executing tasks in parallel. However, detailed planning must be conducted for the system to sufficiently utilize each agent.

One important factor of multi-agent planning is the interaction of robots for task hand-off, or the passing of a task from one agent to another. This can occur because a specific task either requires multiple disjoint agent types or can be executed more efficiently with multiple robots of the same capability. For instance, consider a team of trucks that are transporting boxes from a shipping center to a series of grocery stores. The trucks cannot load or unload the boxes themselves, so some other actor (e.g., a manipulator arm) will be needed for loading and unloading. This shipping scenario defines a set of motion tasks that move each box from a start (the shipping center) to a goal (a grocery store) while requiring interactions for both loading and unloading.

The current state-of-the-art solution for problems with robot interactions is to employ some form of task and motion planning (TMP) to reason about which robots should conduct the interaction [1], [2], [3]. This can potentially involve a large number of motion feasibility tests as various combinations of robots are tried for satisfying a task. Another approach assumes that the interaction(s) can be specified in terms of time-parameterized inter-robot distances [4]. However, this depends on knowing when each interaction should occur in advance.

In this work, we present *Interaction Templates* (ITs) for modeling multi-robot motion planning problems with interactions. ITs are small roadmaps that define the interaction between two or more robots in an isolated setting. They are generated with standard motion planning algorithms as a pre-processing step and are subsequently tiled into the agents' individual roadmaps at appropriate locations to encode potential interactions. The ITs are then connected to form a combined roadmap that encodes possible paths for the robot team which satisfy the motion task.

ITs allow the roadmaps for potentially complex interactions to be computed once and reused for each occurrence. In the grocery store example, single roadmaps for loading and unloading the payload can be used for every terminal at the shipping centers and store terminals, respectively.

This framework moves the responsibility for interaction planning from the 'task' layer to the 'motion' layer. This both allows interaction planning without a task layer and simplifies the task space definition when used in conjunction with a task layer.

Discrete planning work has investigated these types of problems [5]. However, these methods assume a state graph exists to model the transition from one agent type to another. ITs can be viewed as a method for constructing a state graph that describes the entire multi-agent system in a discrete-planning fashion. To our knowledge, this has not been done in a way that incorporates the interactions between agents.

We demonstrate the method on several different simulated problems: one with a homogeneous team of mobile robots and two with heterogeneous teams of mobile robots. The problems span both required and optional cooperation scenarios. We show improved performance over FFRob [1], a current TMP method.

II. Related Work

A. Motion Planning

In this paper, we discuss motion planning in regards to both holonomic and nonholonomic robots. This distinction is in regards to the controllable nature of a robot's *degrees of freedom* (DOFs). Each DOF corresponds to a robot parameter (e.g., position, orientation, joint angles, etc.). The *workspace* is the physical 3-*d* world in which the robots operate. Holonomic robots are robots that have completely controllable DOFs, while nonholonomic robots' controls are dependent on their current state.

A configuration is a single specification of a robot's DOFs. The set of all possible configurations is called the *configuration space* (C_{space}) [6]. The validity of a configuration can be determined relatively efficiently by performing a collision check with the obstacles and other robots in the workspace. If the configuration is valid (i.e., collision-free), then it belongs in the subset of C_{space} called *free space* (C_{free}). Otherwise it is part of the *obstacle space* (C_{obst}).

The problem of motion planning is finding a continuous path through C_{free} from some start configuration to some goal configuration. The start and goal pair define a *query*. In response to the complexity of motion planning [7], sampling based methods have been found to be an efficient way to discover valid paths in C_{free}. Sampling-based planners, such as the various Probabilistic Roadmap Methods [8], attempt to construct a *roadmap*, or a graph that is an approximate model of C_{free}.

These methods work by sampling C_{space} for valid configurations and adding valid configurations as nodes in the roadmap. Paths are found by attempting to connect start and goal configurations by querying the roadmap. This can consist of simply searching through the roadmap for a path, or in the case of Lazy PRM [9], the query re-validates the nodes and edges of the roadmap to check for changes in validity caused by dynamic elements in the environment. This is utilized at various stages in the method presented in this paper to avoid inter-robot collision and account for parts of the roadmap that may have been constructed without knowledge of the problem environment.

B. Task Planning

Task planning requires the computing of subsequent actions that can transition an agent from an initial state to a desired goal state. When considering a multi-agent system, task planning is often comprised of task decomposition and task allocation [10]. In this model, it is the role of the decomposition to create subtasks that can be executed by one agent in order to simplify the task allocation process.

1) Task decomposition: Task decomposition is generally the break down of an abstract task into a set of subtasks based on factors such as environment and agent types [11]. Each of these subtasks are executable by a single agent and define the requirements of individual robots to achieve the complete task. We approach this problem similar to that of [12], [13], [14], where a coordinator-esque agent decomposes a task into smaller subtasks to be allocated to individual agents in a multi-robot team.

2) Task Allocation: Task allocation assigns some set of tasks to some set of agents such that all tasks can be performed. Each task needs to be completed for the problem to be solved. Each task is assigned to only one agent, assuming each agent is capable of completing the assigned task (although it is possible that two or more tasks correspond to a coordinated action). Task assignment additionally attempts to minimize the cost of assigning tasks to agents as well as the utility cost of carrying out the tasks [10].

The most commonly utilized method for task allocation is the auction system [15], [16], [17], which we employ in this work. Our system utilizes a centralized coordinator as auctioneer, though most auction systems are comprised of the same following steps. First, the task is announced to the bidding group. Then, the group members calculate their cost of performing a task and submit their bids. Finally, the task is assigned to the member with the best bid and the auction is closed. This system is extensible to both homogeneous and heterogeneous robot groups, and the centralized coordinator allows task assignment without requiring group members to maintain information about other group members.

C. Integrated Task and Motion Planning

General task planners deal with discrete sets of actions, and cannot reason about geometric motions as motion planners do. Motion planners, on the other hand, have no inherent semantics needed to define a higher-level notion of a task. The integration of task and motion planning provides a complementary solution to complex problems that exist within multi-agent systems. It integrates the high-level decisions of task planning with the low-level geometric constraints of motion planning [18], [19].

Studies on this integration can be split into two groups: integration done at search level and integration done at representation level [2]. There are several methods [20], [21], [22], [23], [24], [1] that utilize a forward-search task planner to build the task plan while checking feasibility with a motion planner at each step. This allows the task planner to focus the motion plan search process. Meanwhile, with integration at the representational level, methods such as [25], [26], [27], [3] use external motion planners to check the feasibility of primitive actions in the motion plan by finding trajectories for the actions. When an action is found to be not feasible, it is common to replan with this knowledge. Not all methods with representational integration have the same extent of integration. Some conduct all feasibility checks in this manner [25], [26] while others only conduct some of their feasibility checks with the integration [27], [3].

D. Group Coordination

There are two commonly used methods for multi-robot group coordination: centralized and decentralized coordination. Centralized coordination requires all team members to communicate with a centralized coordinator, which then assigns tasks to members of the team. Decentralized coordination, on the other hand, requires team members to use distributed algorithms to coordinate amongst themselves. In our method, centralized coordination is only necessary for the task planning portion of the method, and the task allocation can be implemented with either a centralized or decentralized group control method.

1) Centralized Coordination: Centralized coordination provides the advantage of a single agent being able to plan utilizing a global awareness of the system. Because of this, centralized planners have been shown to be very reliable [28]. This is important in the task flow planning stages of the method presented in this paper. The utilization of a centralized group controller is similar to the purpose it has in [12], where a central task planning and allocation system takes advantage of a global view of the problem and environment. The obvious downside of a centralized coordination is the necessity for constant, reliable communication channels between the coordinator and the group members.

2) Decentralized Coordination: Decentralized approaches, such as [29], are more responsive to uncertainty within a system. Agents are able to modify their response as they further explore their environment. With decentralized coordination, failure of a single agent does not necessarily compromise the integrity of the system. However, with improved fault-tolerance and availability comes inherently less reliability, as shown in [28].

III. Problem Definition

Given a homogeneous or heterogeneous team of robots, we wish to solve a problem comprised of motion tasks. Each task contains a start and goal state. A successful team plan is composed of individual robot motions that move the system from

the start to goal state. The grocery store example from the introduction would present tasks such as moving each box from a start location (a shipping center) to a goal (a grocery store). The task plans for these need to consider the cost of agent interactions and utilize the interactions whenever necessary (i.e. loading/unloading the boxes) or beneficial for execution utility (i.e. using multiple trucks to move the boxes faster).

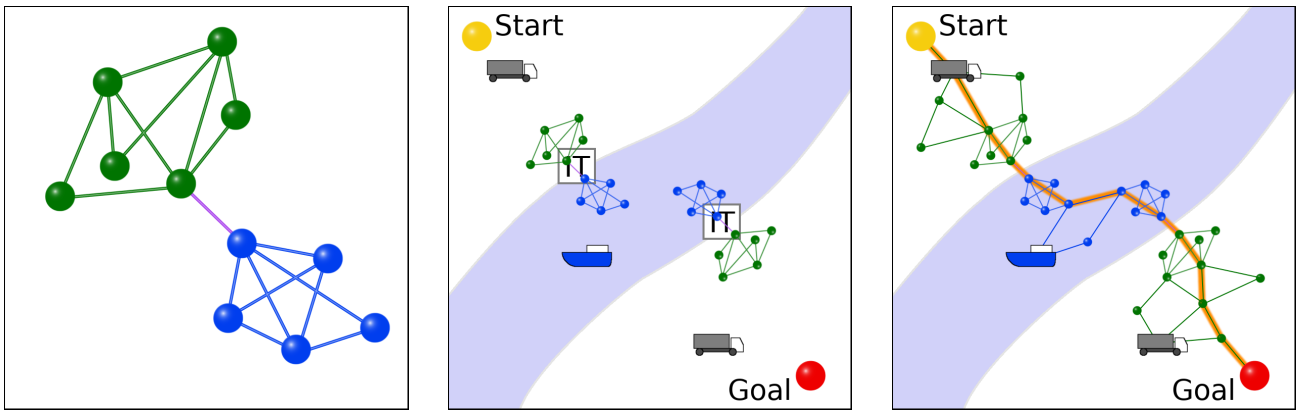
3) Required Cooperation: As described by [5], [30] there exist some problems that require cooperation or coordination amongst the agents in the system. Formally, any problem that is solvable by more than one agent but not solvable by one agent is considered to be a required cooperation problem.

4) Optional Cooperation: In contrast, there exist problems that require only a single robot to execute but can be performed more efficiently by multiple robots. These problems are referred to as optional cooperation problems. In these circumstances, the decision to employ cooperation is determined by a utility function that defines the quality of a task solution (e.g., energy consumption, shortest time, personal reward, etc.). An ideal multi-agent planning system would employ optional cooperation whenever doing so improves the value of the defined utility function.

IV. Method

The Interaction Templates method aims to employ agent interactions in both required and optional cooperation problems through solving a motion planning problem. The driving idea is to generate and search a combined roadmap which represents the task flow through the entire multi-agent system (Fig. 1(c)). The interactions between individual agents are incorporated into this combined roadmap, which enables a standard graph search on this roadmap to naturally use interactions whenever doing so is the best solution (orange path in Fig. 1(c)).

An *Interaction Template* (IT) for a pair of agents is a pair of small roadmaps which are built up around a specific pair of configurations that define the start of an interaction. These configurations are connected with an *interaction edge*, which represents a possible task flow instead of a robot state change. The ITs are first computed in a vacuum (Fig. 1(a)) and then tiled into the environment at appropriate locations (Fig. 1(b)). The appropriate components of the ITs are then connected to the individual agents' roadmaps to encode the interaction motions. The individual roadmaps and interaction edges are then aggregated into a combined roadmap to encode the flow of a task from one robot to another (Fig. 1(c)).



(a) Interaction Template (b) IT Tiling (c) Combined Roadmap and Extracted Task Plan

Fig. 1. Construction and querying of a combined roadmap in an example problem where a payload must be taken across a river by two trucks and a boat. (a) Construction of an interaction template with a purple connecting edge between the maps for each robot. (b) Tiling of the interaction template into the environment at dock locations. The task start and goal are shown in yellow and red, respectively. (c) The combined roadmap including the ITs and task start and goal. The extracted task plan in orange, which spans configurations of three robots.

A. Interaction Template Construction

ITs offer a means of modeling task hand-off interactions for a pair agents in the combined roadmap. These hand-off interactions consist of the passing of a task from one agent to another agent. This can be the physical passing of an object, or the symbolic passing of a role, or any other distinctions that can constitute a task. An IT is defined as a set of small roadmaps, one for each agent involved in an interaction that encodes the relative motions needed to execute the interaction.

The initial step of the method is to build these interaction templates (Alg. 1). For each IT, a set of motion planning problems is created in which each agent must plan from some separated start configuration to its interaction goal while avoiding contact with the other agents at their goals. A roadmap is constructed for each agent in the interaction with a graph-based motion planner in an empty (obstacle-free) environment. The final configurations in the roadmaps are then connected with an edge modeling the cost of the hand-off (Fig. 1(a)).

Algorithm 1 Create an interaction template from set of agent, query pairs Q

Require: Set of agent type, query pairs Q, interaction cost W, graph-based motion planner P
Ensure: Interaction template I

procedure CREATEINTERACTIONTEMPLATES
  create blank interaction template I
  for all agent type, query pair $(a_i, q_i) \in Q$ **do**
    for all agent type, query pair $(a_j, q_j) \in Q$ **do**
      if $i = j$ **then**
        continue
      Configure a_j at q_j.goal
    Solve motion plan for q_i for a_i with P
    Store solution roadmap r_i, final configuration f_i in I
  for all $f_i \in I$ **do**
    for all $f_j \in I$, $j > i$ **do**
      Add edge in I with weight W, between f_i and f_j
  return I

B. Interaction Template Transformation

We assume that the desired location of the interactions is known and tile the ITs into these locations in the environment. This is done by transforming a copy of the IT roadmaps to each desired location (Fig. 1(b)), similar to the method in [31]. The IT roadmaps are thus computed only once per interaction and reused as many times as needed. Lazy validation is performed on each IT roadmap to ensure that the tiled components are valid.

C. Construct Agent Type and Combined Roadmaps

An agent type refers to a subset of homogeneous agents within a multi-agent system. We define an *agent type roadmap* as an initially constructed roadmap connecting all the transformed ITs of the same agent type. Individual agents of a given type can then begin execution using their type roadmap. The agent type roadmaps are constructed by a coordinator agent, which employs a PRM algorithm to connect the IT roadmaps for each agent type. (Alg. 2).

Algorithm 2 Create the agent type roadmaps from set of transformed ITs T

Require: Set of transformed ITs; set of agents types A; lazy graph-based motion planner L
Ensure: Set of agent type Roadmaps R

procedure CREATEAGENTTYPEROADMAPS
  for all agent type a in A **do**
    initialize agent type roadmap r_a
    for all interaction template $it \in T$ **do**
      if $a \in it$ **then**
        r_a.append(r_s)
        plan from existing r_c to r_s with L
    R.append(r_c)
  return R

After the agent type roadmaps are constructed they are distributed to all of the agents of the corresponding agent type. The agents then use these roadmaps as initial solutions to generate their individual paths.

D. Combined Roadmap

We define a *combined roadmap* as an aggregate roadmap representing the entire multi-agent system through agent type roadmaps connected by interaction edges. The combined roadmap is implicitly constructed with the agent type roadmaps because each of the agent type roadmaps are connected to each other through interaction edges (Fig. 1(c)). This enables motion task planning across the combined roadmap to naturally use agent interactions whenever they provide the best solution.

E. Task Planning

A motion satisfying the task presented to the multi-agent team can be extracted by searching the combined roadmap. Start and goal nodes for the task are added to the combined roadmap at the relevant locations. Configurations are then sampled

for each applicable agent type which can occupy these locations and added to their respective agent type roadmaps. These agent start and end configurations are connected to the abstract task start and end nodes by zero weight edges. This enables the team to query the combined roadmap for a single motion plan between the task start and end node, which naturally encodes the needed agent types at each step.

The plan from the task start and end nodes (if successful) models the flow of a task through the environment and across the agent types in the system 1(c). This plan amounts to a set of subtasks for the multi-agent system.

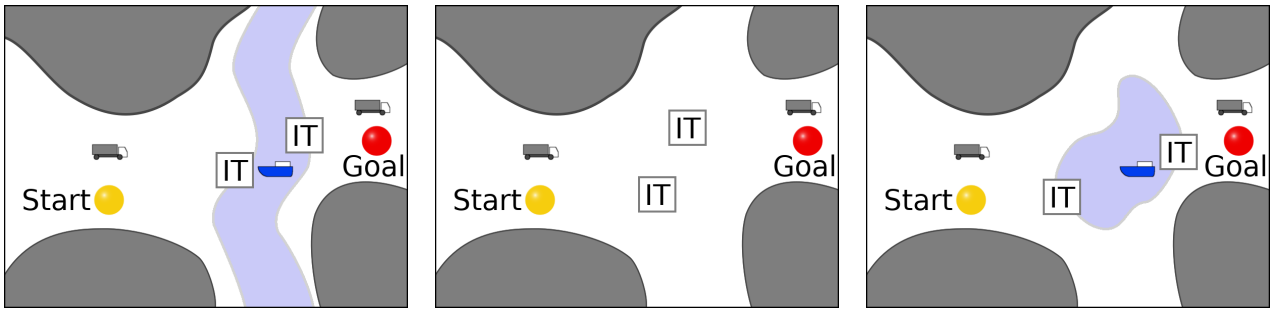
F. Subtask Assignment

The configurations along the combined roadmap path indicate which type of agent is performing the task at that particular stage. Task decomposition occurs at the points along the path where the agent type changes; this indicates that the task has been passed off to another robot via an interaction at that location.

The resulting subtasks are ordered by time priority and assigned using a task assignment algorithm. A simple auction system was used for this in the implementation.

V. Experiments

We examine several different scenarios to exercise ITs for both homogeneous and heterogeneous teams and under both required and optional cooperation schemes. In each scenario, the team must transport a package from a start location to a goal location. The scenarios include: i) a heterogeneous team of land-only and water-only capable robots in an environment where a river separates the land masses containing the start and the goal such that cooperation is required (Fig. 2(a)), ii) a homogeneous team of land-only capable robots in environment without water such that cooperation is optional (Fig. 2(b)), and iii) a heterogeneous team of land-only and water-only capable robots in an environment with a lake that does not separate the start and goal such that cooperation is optional (Fig. 2(c)).



(a) River (Heterogeneous, Req. Coop.) (b) Relay (Homogeneous, Opt. Coop.) (c) Lake (Heterogeneous, Opt. Coop.)

Fig. 2. In each experiment, the task is to move a package from the left side of the environment to the right side.

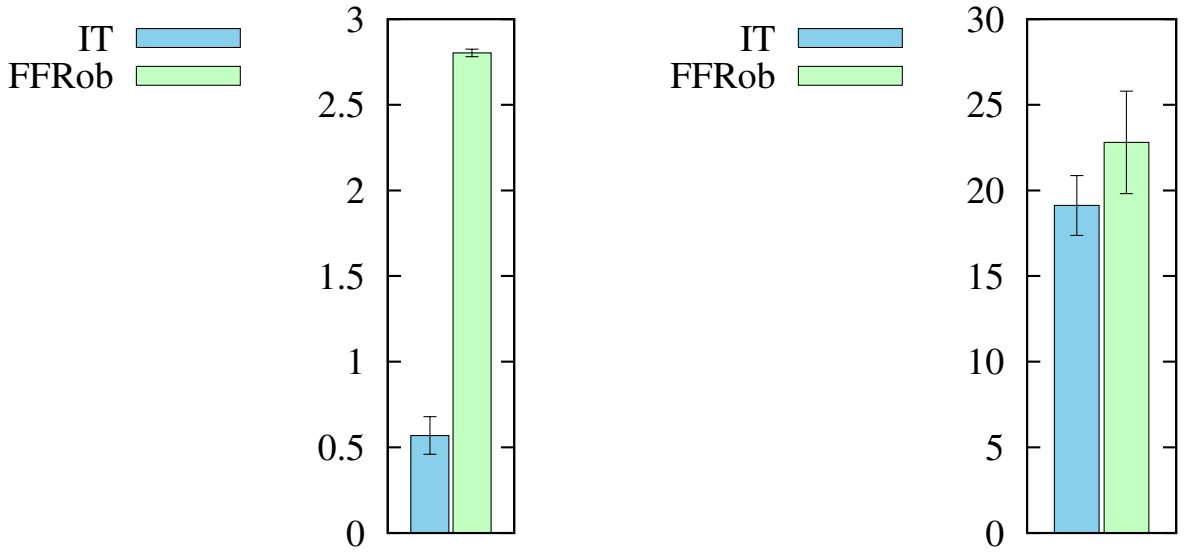
To our knowledge, there are no pure motion planning methods that solve this type of multi agent interaction problem that ITs seek to do. Current solutions require an integrated TMP method to find the desired multi agent interactions. In order to evaluate our method, we implemented the FFRob method [1] to provide a TMP solution for determining multi agent interactions. FFRob is a forward-search planner that uses the Fast Forward task planning method [32] along with motion planning feasibility checks to find a plan. FFRob can handle a broad scope of problems dependent upon the actions the system has to choose from. To solve the same tasks our method seeks to solve, we provided FFRob with the actions 'MoveRobot', 'StartTask', and 'HandoffTask'. These actions allow FFRob to provide solutions similar to the ones produced by the IT method.

A. Required Cooperation

The environment is a flat terrain with a river that divides the land into two components (Fig. 2(a)). The team is comprised of two 'land capability' robots (one positioned in each land component) and one 'water capability' robot. The start and goal reside on separate components, requiring cooperation between different team members.

We place an IT on either edge of the river to describe the handoffs between land and water robots required for a package to cross the river. We ran 19 trials. The average pre-processing time for both ITs and FFRob was 0.331 seconds with a standard deviation of 0.0498 seconds. Fig. 3 provides the planning and execution times for both methods.

The planning time for solving the required cooperation problem was significantly less for the IT method than the TMP method (Fig. 3(a)). This a result of the IT method only needing to make a single motion planning query across the combined roadmap as opposed to the FFRob method's multiple motion planning queries needed to validate potential actions. The difference in the execution time is a result of the short nature of the task (Fig. 3(b)). The variance in task decomposition results in the need for an extra sampling for goal configurations of the subtasks in FFRob. This difference should become negligible with a longer task duration.



(a) River Planning Time

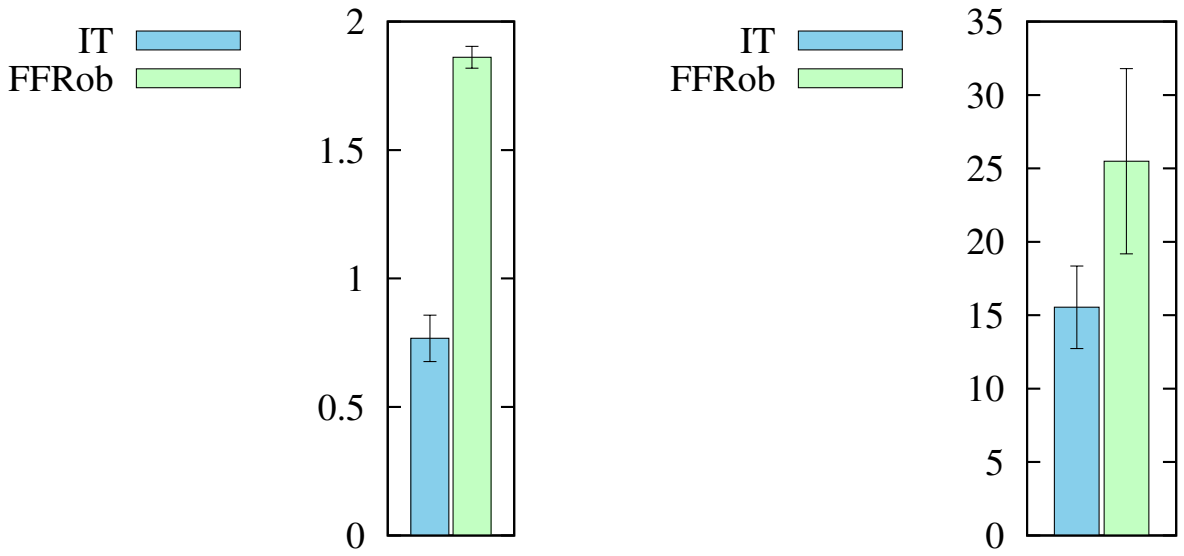
(b) River Execution Time

Fig. 3. These results represent the amount of time that the IT and FFRob methods took to complete the task. The Y-axis represents seconds. The error bars represent standard deviation.

B. Optional Cooperation

1) Homogenous: Here the team is a pair of 'land capability' robots. The environment is a flat terrain with no bodies of water, providing all agents access to all of C_{free} and allowing for optional cooperation (Fig. 2(b)). One robot is placed near the start of the task, and the other near the goal.

We place a single IT in between the start and the goal. We ran 20 trials. The average pre-processing time for both ITs and FFRob was 0.650 seconds with a standard deviation of 0.0742 seconds. Fig. 4 provides the planning and execution times for both methods.



(a) Relay Planning Time

(b) Relay Execution Time

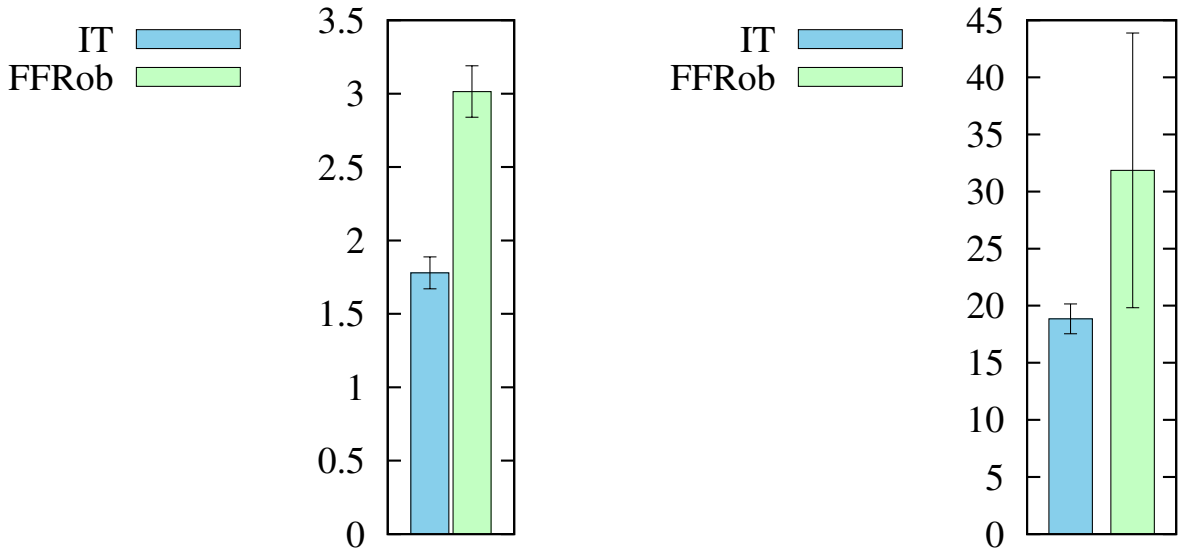
Fig. 4. These results represent the amount of time that the IT and FFRob methods took to complete the task. The Y-axis represents seconds. The error bars represent standard deviation.

The difference in the planning time is again a result of the need for only a single motion planning query in the IT method

as opposed to the multiple queries needed in the FFRob method (Fig. 4(a)). The difference in the execution time is a result of the IT method often finding it advantageous to perform a task handoff (Fig. 4(b)). The cost of moving a single robot from the start to the goal is not in itself is not a significantly longer action, but performing the task handoff resulted in there being no need to avoid inter robot collision. When the task plan developed by either method only utilized a single robot, it had to plan around the second unutilized robot. As the IT method was more likely to produce a plan that involved both robots, the average execution time was significantly less. Note that the resulting difference in execution time is a product of this problem setup and not a general result for all optional cooperation problems.

2) Heterogeneous: This experiment uses the same heterogeneous team as before: two 'land capability' robots and one 'water capability' robot. The environment now has a lake that lies in between the start and goal, but does not entirely impede movement around it creating an optional cooperation scenario (Fig. 2(c)). The land robots begin on either side of the lake near the start and goal.

An IT is placed on the left and right edges of the lake. We ran 15 trials. The average pre-processing time for both ITs and FFRob was 0.404 seconds with a standard deviation of 0.0459 seconds. Fig. 5 provides the planning and execution times for both methods.



(a) Lake Planning Time

(b) Lake Execution Time

Fig. 5. These results represent the amount of time that the IT and FFRob methods took to complete the task. The Y-axis represents seconds. The error bars represent standard deviation.

The difference in planning time in this heterogenous optional cooperation experiment is again a result of the need for a single motion planning query in the IT method (Fig. 5(a)). The difference is less in this experiment than in the previous ones because the task assignment is in a more complicated environment. The ability of the land robots to reach both sides of the body of water provide more possible task assignments that the auction system must consider.

The difference in the execution time is again indicative of the likelihood of the IT method to find a task plan utilizing a handoff (Fig. 5(b)). In this problem, the cost of executing the task with a single robot is much higher than in the prior experiment as the single robot must navigate around the lake rather than simply avoid another robot. The nature of FFRob sometimes results in plans that have a costly first action that produces a better state. This occasionally took the form of finding a single robot plan that utilized the robot on the far side of the lake, forcing it to navigate around the lake twice. These plans were responsible for the outliers in the execution time for the FFRob plans.

VI. Conclusion

Interaction Templates provide a mechanism to handle certain types of robot interactions strictly through the motion planning process, thereby offloading this burden from the 'task' layer to the 'motion' layer. ITs are placed in an environment, with each template containing the motion plan for a potentially complex interaction between agents. The templates are created once and reused for every instance of the interaction, eliminating the cost of computing these interactions at runtime. ITs allow plans to consider the cost of interactions and motion plans while determining task flow.

We demonstrate that ITs can be used for homogeneous and heterogeneous teams and for both required and optional cooperation scenarios. We compared IT planning time and execution time to FFRob, a TMP method and showed that ITs

decrease time by reducing the number of motion planning queries. In optional cooperation scenarios, ITs find opportunities for increased efficiency from cooperation that is often missed as a result of FFRob's greedy nature. ITs may potentially be used as a tool for FFRob and other TMP methods to more effectively determine robot interactions. In the future, we will investigate automatic IT placement to future automate the planning process.

REFERENCES

[1] C. R. Garrett, T. Lozano-Prez, and L. P. Kaelbling, "FFRob: An efficient heuristic for task and motion planning," in *Proc. Int. Workshop on Algorithmic Foundations of Robotics (WAFR)*, pp. 179–195, 2014.

[2] P. Schüller, V. Patoglu, and E. Erdem, "Levels of integration between low-level reasoning and task planning," in *AAAI Workshop on Intelligent Robotic Systems*, pp. 73–78, 2013.

[3] E. Erdem, K. Haspalamutgil, C. Palaz, V. Patoglu, and T. Uras, "Combining high-level causal reasoning with low-level geometric reasoning and motion planning for robotic manipulation," in *Robotics and Automation (ICRA), 2011 IEEE International Conference on*, pp. 4575–4581, IEEE, 2011.

[4] S. Bhattacharya, M. Likhachev, and V. Kumar, "Multi-agent path planning with multiple tasks and distance constraints," in *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)*, IEEE, 2010.

[5] Y. Zhang, S. Sreedharan, and S. Kambhampati, "A formal analysis of required cooperation in multi-agent planning," in *International Conference on Automated Planning and Scheduling (ICAPS 2016)*, June 2016.

[6] T. Lozano-Pérez and M. A. Wesley, "An algorithm for planning collision-free paths among polyhedral obstacles," *Communications of the ACM*, vol. 22, pp. 560–570, October 1979.

[7] J. H. Reif, "Complexity of the mover's problem and generalizations," in *Proc. IEEE Symp. Foundations of Computer Science (FOCS)*, (San Juan, Puerto Rico), pp. 421–427, October 1979.

[8] L. E. Kavraki, P. Švestka, J. C. Latombe, and M. H. Overmars, "Probabilistic roadmaps for path planning in high-dimensional configuration spaces," *IEEE Trans. Robot. Automat.*, vol. 12, pp. 566–580, August 1996.

[9] R. Bohlin and L. E. Kavraki, "Path planning using Lazy PRM," in *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)*, pp. 521–528, 2000.

[10] Z. Yan, N. Jouandeau, and A. A. Cherif, "A survey and analysis of multi-robot coordination," in *International Journal of Advanced Robotic Systems*, November 2013.

[11] J. Chen, Y. Yang, and L. Wei, "Research on the approach of task decomposition in soccer robot system," in *International Conference on Digital Manufacturing & Automation*, December 2010.

[12] P. Caloud, W. Choi, J.-C. Latombe, C. L. Pape, and M. Yim, "Indoor automation with many mobile robots," in *IEEE International Workshop on Intelligent Robots and Systems, Towards a New Frontier of Applications*, July 1990.

[13] R. Simmons, D. Apfelbaum, D. Fox, R. P. Goldman, K. Z. Haigh, D. J. Muslineg, M. Pelican, and S. Thrun, "Coordinated deployment of multiple, heterogeneous robots," in *IEEE/RSJ International Conference on Intelligent Robots and Systems*, October 2000.

[14] R. S. Aylett and D. P. Barnes, "A multi-robot architecture for planetary rovers," in *Proc. ESA Wksp. on Advanced Space Technologies for Robotics and Automation*, 1998.

[15] R. G. Smith, "The contract net protocol: High-level communication and control in a distributed problem solver," in *Transactions on Computers*, December 1980.

[16] M. B. Dias, "Traderbots: A new paradigm for robust and efficient multirobot coordination in dynamic environments," *Robotics Institute*, p. 153, 2004.

[17] B. P. Gerkey and M. J. Mataric, "Sold!: Auction methods for multirobot coordination," in *Transactions on Robotics and Automation*, October 2002.

[18] S. Srivastava, N. Desai, R. Freedman, and S. Zilberstei, "An anytime algorithm for task and motion MDPs," *CoRR*, February 2018.

[19] S. Cambon, R. Alami, and F. Gravot, "A hybrid approach to intricate motion, manipulation and task planning," in *The International Journal of Robotics Research*, January 2009.

[20] F. Gravot, S. Cambon, and R. Alami, "asymov: a planner that deals with intricate symbolic and geometric problems," in *Robotics Research. The Eleventh International Symposium*, pp. 100–110, Springer, 2005.

[21] K. Hauser and J.-C. Latombe, "Integrating task and prm motion planning: Dealing with many infeasible motion planning queries," in *ICAPS09 Workshop on Bridging the Gap between Task and Motion Planning*, Citeseer, 2009.

[22] L. P. Kaelbling and T. Lozano-Pérez, "Hierarchical task and motion planning in the now," in *Robotics and Automation (ICRA), 2011 IEEE International Conference on*, pp. 1470–1477, IEEE, 2011.

[23] E. Plaku and G. D. Hager, "Sampling-based motion and symbolic action planning with geometric and differential constraints," in *Robotics and Automation (ICRA), 2010 IEEE International Conference on*, pp. 5002–5008, IEEE, 2010.

[24] E. Plaku, "Planning in discrete and continuous spaces: From ltl tasks to robot motions," in *Conference Towards Autonomous Robotic Systems*, pp. 331–342, Springer, 2012.

[25] O. Caldiran, K. Haspalamutgil, A. Ok, C. Palaz, E. Erdem, and V. Patoglu, "Bridging the gap between high-level reasoning and low-level control," in *International Conference on Logic Programming and Nonmonotonic Reasoning*, pp. 342–354, Springer, 2009.

[26] A. Hertle, C. Dornhege, T. Keller, and B. Nebel, "Planning with semantic attachments: An object-oriented view.," in *ECAI*, vol. 242, pp. 402–407, 2012.

[27] E. Erdem, E. Aker, and V. Patoglu, "Answer set programming for collaborative housekeeping robotics: representation, reasoning, and execution," *Intelligent Service Robotics*, vol. 5, no. 4, pp. 275–291, 2012.

[28] G. Sanchez and J.-C. Latombe, "Using a prm planner to compare centralized and decoupled planning for multi-robot systems," in *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)*, vol. 2, pp. 2112–2119, 2002.

[29] M. Roth, R. Simmons, and M. Veloso, "Decentralized communication strategies for coordinated multi-agent policies," in *Multi-Robot Systems. From Swarms to Intelligent Automata Volume III*, pp. 93–105, Springer, 2005.

[30] J.-S. Liu and K. P. Sycara, "Multiagent coordination in tightly coupled task scheduling," in *Readings in Agents* (M. N. Huhns and M. P. Singh, eds.), pp. 164–171, San Francisco, CA, USA: Morgan Kaufmann Publishers Inc., 1998.

[31] L. Han and N. M. Amato, "A kinematics-based probabilistic roadmap method for closed chain systems," in *New Directions in Algorithmic and Computational Robotics*, pp. 233–246, Boston, MA: A. K. Peters, 2001. Book contains the proceedings of the International Workshop on the Algorithmic Foundations of Robotics (WAFR), Hanover, NH, 2000.

[32] J. Hoffmann and B. Nebel, "The ff planning system: Fast plan generation through heuristic search," *Journal of Artificial Intelligence Research*, vol. 14, pp. 253–302, 2001.