INTERACTION TEMPLATES FOR MULTI-ROBOT SYSTEMS

A Thesis

by

JAMES DONALD MOTES

Submitted to the Office of Graduate and Professional Studies of
Texas A&M University
in partial fulfillment of the requirements for the degree of

MASTER OF SCIENCE

Chair of Committee,	Nancy M. Amato
Committee Members,	Dylan Shell
	Suman Chakravorty
Head of Department,	Dilma Da Silva

December 2019

Major Subject: Computer Science

ABSTRACT

This work describes a framework for multi-robot problems that require or utilize interactions between robots. Solutions consider interactions on a motion planning level to determine the feasibility and cost of the multi-robot team solution. Modeling these problems with current integrated task and motion planning (TMP) approaches typically requires reasoning about the possible interactions and checking many of the possible robot combinations when searching for a solution.

We present a multi-robot planning method called Interaction Templates (ITs) which moves certain types of robot interactions from the task planner to the motion planner. ITs model interactions between a set of robots with a small roadmap. This roadmap is then tiled into the environment and connected to the robots' individual roadmaps. The resulting combined roadmap allows interactions to be considered by the motion planner. We apply ITs to homogeneous and heterogeneous robot teams under both required and optional cooperation scenarios which previously required a task planning method. We show improved performance over a current TMP planning approach.

CONTRIBUTORS AND FUNDING SOURCES

Contributors

This work was supported by a thesis committee consisting of Nancy M. Amato [advisor] as well as Dylan Shell of the Department of Computer Science and Engineering and Suman Chakravorty of the Department of Aerospace Engineering.

Much of the supporting code base used was written by various members of Parasol laboratories. Dr. Shawna Thomas provided tremendous theoretical and analytical help every step of the way. Read Sandström provided mentorship and guidance, support for any code issues, and helped ensure the code base used was up to date and ready to be integrated for every purpose for which it was needed. Will Adams and Tobi Ogunyale helped with the initial development of the work.

All other work conducted for the thesis was completed by the student independently.

Funding Sources

Graduate study was supported by a Engineering Graduate Merit Fellowship from the College of Engineering at Texas A&M University and research funding from Professor Nancy M. Amato.

TABLE OF CONTENTS

Page

LIST OF FIGURES

FIGURE Page

LIST OF TABLES

TABLE Page

1. INTRODUCTION

Multi-robot systems (MRS) often provide significant advantages over utilizing a single robot. Heterogeneous systems have the ability to perform tasks requiring different hardware, and homogeneous systems can increase throughput by executing tasks in parallel. However, detailed planning must be conducted for the system to sufficiently utilize each robot. [1]

One important factor of multi-robot planning is the interaction of robots for task hand-off, or the passing of a task from one robot to another. This can occur because a specific task either requires multiple disjoint robot types or can be executed more efficiently with multiple robots of the same capability. For instance, a payload transport problem involves moving a payload from some source to a destination, such as a team of trucks transporting boxes from a shipping center to a series of destinations. 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. A payload transport problem defines a set of motion tasks that move each box from a start (the shipping center) to a goal 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 [2, 3, 4]. This can potentially involve a large number of motion feasibility tests as various combinations of robots are tried for satisfying a task. In a payload transport problem, a TMP method would have to determine which manipulator picks up the box, which truck receives the payload, that the robots should interact, where the truck delivers the box, etc.. Another approach assumes that the interaction(s) can be specified in terms of time-parameterized inter-robot distances [5]. This requires 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

[1] Part of this section is reprinted with permission from, [1].

pre-processing step and are subsequently tiled into the robots' 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 a payload transport problem, single roadmaps for loading and unloading the payload can be used for every terminal at the shipping centers and destination terminals, respectively. This framework moves the responsibility for interaction planning from the 'task' layer to the 'motion' layer and allows interaction planning without a task layer.

Discrete planning work has investigated these types of problems [6]. However, these methods assume a state graph exists to model the transition from one robot type to another. ITs can be viewed as a method for constructing a state graph that describes the entire MRS in a discrete-planning fashion. To our knowledge, this has not been done in a way that incorporates the interactions between robots.

We demonstrate the method on different simulated problems spanning required and optional cooperation scenarios with either a heterogeneous teams of mobile robots or a homogeneous team of manipulators. We show improved performance over FFRob [2], a current TMP method.

1.1 Contributions

1. A method for solving multi-robot motion tasks utilizing inter-robot interactions
2. Automatic placement methods for ITs.
   (a) A method to find interaction locations along disjoint workspaces boundaries.
   (b) A method to find interaction locations inside of overlapping workspaces.

Some of the results in this work have been accepted for publication in Robotics and Automation Letters (RA-L) [1] and are used with permission. This paper presented the general framework for our method. This work also included verification and comparison to other existing methods. These items are © 2019 IEEE. Note that some excerpts from that work have been used in this thesis verbatim.

1.2 Outline

This thesis will first highlight related work, in motion planning, task planning, and their integration as well as multi-robot systems. Next the IT method will be presented, providing details of the approach and the algorithms developed to support it. Then a theoretical analysis of the method will be discussed identifying the theoretical improvement over existing TMP methods. Finally, results and the overall outcomes of this research will be discussed.

2. RELATED WORK

This chapter discusses work that has been done by others related to the field of motion planning, task planning, and their integration as well as multi-robot systems. Broadly speaking, this includes sampling based motion planning, various components of task planning, different approaches to their integration as well as the FFRob method in addition to different approaches to planning for multi-robot systems. [1]

2.1 Motion Planning

A *configuration* is a complete specification of a robot's *degrees of freedom* (DOFs). Each DOF corresponds to a robot parameter (e.g., position, orientation, joint angles, etc.). The set of all possible configurations is called the *configuration space* (C_{space}) [7]. The validity of a configuration can be determined relatively efficiently by performing a collision check with the obstacles and other robots in the *workspace*, which is the physical 3-*d* world in which the robots operate. 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*. Multi-robot motion planning specifies a start and goal configuration for each robot, and additionally requires that the robots do not collide.

In response to the complexity of motion planning [8], 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 [9], 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. Valid transitions between configurations are represented as edges in the roadmap. A path between configurations can thus be extracted by

[1] Part of this section is reprinted with permission from, [1].

searching the roadmap with a shortest-path search.

2.2 Task Planning

Task planning requires computing subsequent actions that transition a robot from an initial state to a desired goal state. When considering a MRS, task planning is often comprised of task decomposition and task allocation [11]. In this model, the decomposition creates subtasks that can be executed by one robot in order to simplify the task allocation process.

2.2.1 Task decomposition

Task decomposition breaks down an abstract task into subtasks based on factors like environment and robot-types [12]. Each subtask is executable by a single robot and defines the requirements of individual robots to achieve the complete task. One approach is to use a centralized coordinator to decompose a task into subtasks [13, 14, 15]. Our IT method employs this strategy.

2.2.2 Task Allocation

Task allocation assigns a set of tasks to a set of robots 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 robot, assuming each robot 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 robots as well as the utility cost of carrying out the tasks [11].

The most commonly utilized method for task allocation is the auction system [16, 17, 18], which we employ in this work. Our system utilizes a centralized coordinator as auctioneer.

2.3 Integrated Task and Motion Planning

General task planners deal with discrete sets of actions and cannot reason about geometric motions. Motion planners 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 can exist within a MRS. It integrates the high-level decisions of task planning with the low-level geometric constraints of motion planning [19, 20].

Studies on this integration can be split into two groups: integration done at search level and integration done at representation level [3]. There are several methods that utilize a forward-search task planner to build the task plan while checking feasibility with a motion planner at each step [21, 22, 23, 24, 25, 2]. This allows the task planner to focus the motion plan search process. Meanwhile, with integration at the representational level, methods such as [26, 27, 28, 4] use external motion planners to check the feasibility of primitive actions by finding trajectories for the actions. When an action is found 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 [26, 27] while others only conduct some of their feasibility checks with the integration [28, 4].

In this work, we compare our IT method with FFRob [2], a method with search level integration. FFRob searches the state space with a Fast Forward planner to find helpful actions for the next step in the action sequence. The resulting states are then evaluated with the same heuristic and the highest scoring action is chosen to move forward. This process is repeated iteratively until the goal state is reached. FFRob constructs a roadmap for the robot to enable validation of actions considered in the forward search that the authors refer to as a *reachability graph*. We adapt this concept to the multi-robot problem considered in this paper.

2.4 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 requires team members to use distributed algorithms to coordinate amongst themselves. In our method, centralized coordination is only necessary for the task planning portion. The task allocation can be implemented with either approach.

2.4.1 Centralized Coordination

Centralized coordination provides the advantage of a single agent being able to plan utilizing a global awareness of the system and is often reliable [29]. The utilization of a centralized group controller is similar to the purpose it has in [13], 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.

2.4.2 Decentralized Coordination

Decentralized methods, such as [30, 29], are more responsive to uncertainty. Robots are able to modify their response as they further explore their environment and failure of a single robot does not necessarily compromise the integrity of the system.

3. PROBLEM STATEMENT

Given a homogeneous or heterogeneous team of robots $R = R_1 \cup R_2 \cup ... \cup R_K$, where R_i is the set of robots of type i, and a set of possible interactions I, we wish to solve a problem comprised of motion tasks M. A motion task $m \in M$ consists of moving one or more objects (which may be passive objects, robots, or intangibles such as information) between locations in some environment. The start and goal state s_0, s_g describe the required transitions of these objects. A successful team plan is comprised of individual robot motions that move the system between s_0 and s_g, possibly using interactions $I_{ij} \in I$ between robots of type i and j. [1]

The interactions considered in this paper model the handing over of a task. This enables the system to pass the task from one robot $r_1 \in R_i$ to another $r_2 \in R_j$. These interactions could be a manipulator loading/unloading boxes from a truck. A solution for each $m \in M$ needs to consider the cost of the elements of I and utilize them whenever necessary (i.e. loading/unloading the boxes) or beneficial for execution utility (i.e. using multiple trucks in a relay).

3.1 Required Cooperation

As described by [6, 31], there exist some problems that require cooperation or coordination amongst the robots in the system. Formally, any problem that is solvable by multiple robots but not by one is considered to be a required cooperation problem. These problems cannot be solved without using at least one element in I.

3.2 Optional Cooperation

There exist problems that require only a single robot to execute but can also be performed by multiple robots. These are optional cooperation problems and can be solved without using any elements in I.

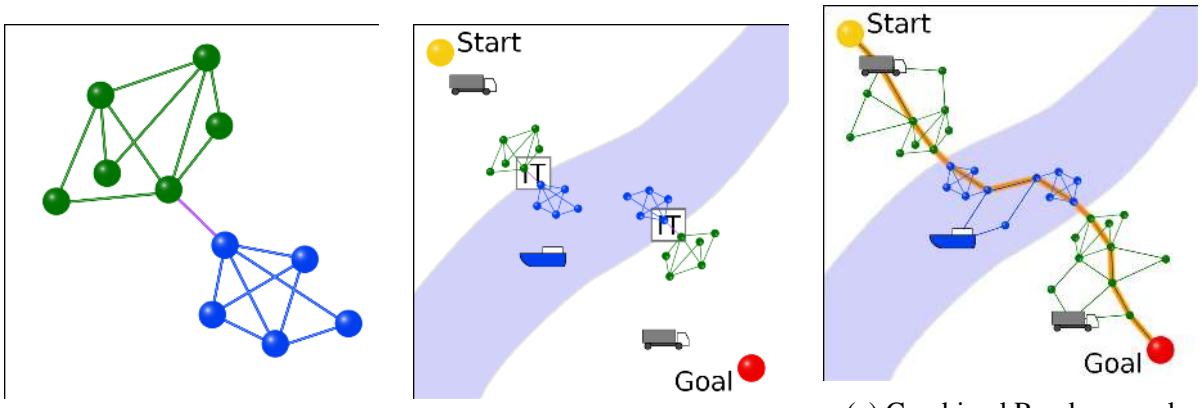
In either setting, there are often numerous choices for which robots should interact and where the interaction should occur. The value of these choices can be quantified by a utility function

[1] Part of this section is reprinted with permission from, [1].

that defines the quality of a task solution based on some desired metric (e.g., energy consumption, shortest time, personal reward, etc.). An ideal multi-robot planning system would employ cooperation whenever doing so improves the value of the defined utility function.

The state-of-the-art solution to this problem requires the use of an integrated TMP method that decomposes the task into individual robot movements which require motion plans and interactions between robots. In this paper, we present an alternative method that solves this problem using PRM methods with no explicit task layer.

An *Interaction Template* (IT) models a possible interaction between a pair of robots. It is composed of a pair of small roadmaps which are built up around a specific pair of configurations that define the relative position of an interaction. These configurations are connected with an *interaction edge*, which represents a possible task flow instead of a robot state change (Fig. 4.1a). 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. In a payload transport problem, this could represent the loading/unloading of a box as the 'task' would pass from the manipulator to the truck or vice versa. [1]



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

Figure 4.1: 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. Reprinted with permission from, [1].

These templates are generated in a vacuum and must be transformed into locations in the en-

[1]This chapter contains previously published material from [1].

vironment where interactions have the potential to be useful (Fig. 4.1b). In the payload transport problem, they could be placed at the loading docks. A method is presented later for determining possible locations for interactions between robots with both overlapping and disjoint workspaces (Alg. 2, Alg. 3).

Robot-type refers to a subset of homogeneous robots in a MRS. We define *robot-type roadmap* as a roadmap connecting all the ITs of the same robot-type.

The IT method aims to employ interactions in required and optional cooperation problems by solving a motion planning problem. This is done by generating a *combined roadmap* composed of all robot-type roadmaps connected via interaction edges in ITs. The combined roadmap contains roadmaps for each of the robot-types in the MRS and edges representing a way for the task flow to move between robot-type roadmaps via an interaction (Fig. 4.1c).

The problems can then be solved by performing a motion planning query across the combined roadmap. This provides an advantage over TMP methods that must consider the action space as the combined roadmap scales linearly with the number of robot types, however, the action space increases at a rate of $|R|^2$ as the number of robots increase.

The path generated across the combined roadmap represents the task flow for the MRS including necessary robot motions and interactions (orange path in Fig. 4.1c). Any time the path crosses between robot-type roadmaps by traversing an interaction edge, an interaction between robot-types must occur. The path can also traverse an interaction edge between configurations of the same robot-type modeling an interaction between homogeneous robots.

4.1 Interaction Template Construction

ITs model task hand-off interactions for a pair of robots. The initial step of the method is to build the ITs (Alg. 1). These handoff interactions consist of passing of a task between robots. An IT is defined as a set of small roadmaps, one for each robot involved in an interaction that encodes the relative motions needed to execute the interaction. These roadmaps are constructed via a motion plan between an arbitrary start and its interaction goal while avoiding contact with the other robots at their respective goals. This is computed with any graph-based motion planner

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

1: **procedure** CREATEINTERACTIONTEMPLATES(Set of robot-types A, query pairs Q, interaction cost W, graph-based motion planner P)
2: create blank interaction template T
3: **for all** robot-type, query pair $(a_i, q_i) \in Q$ **do**
4: **for all** robot-type, query pair $(a_j, q_j) \in Q$ **do**
5: **if** $i = j$ **then**
6: **continue**
7: **end if**
8: Configure a_j at q_j.goal
9: **end for**
10: $r_i, f_i \leftarrow$Solve motion plan for q_i for a_i with P
11: T.*append*(r_i, f_i)
12: **end for**
13: **for all** $f_i \in T$ **do**
14: **for all** $f_j \in T$, $j > i$ **do**
15: Add edge in T with weight W, between f_i and f_j
16: **end for**
17: **end for**
18: **return** T
19: **end procedure**

On line 8, r_i is the solution roadmap and f_i is the final configuration of the path found.

(such as PRM or its derivatives) in an empty (obstacle-free) environment.

The goal configurations in the roadmaps are connected with an edge modeling the cost of the hand-off (Fig. 4.1a). The weight of this interaction edge represents the cost of performing the interaction relative the metric used for the roadmaps (e.g. distance, energy, etc.). This is set by the user and can be used to bias the system towards utilizing an interaction if the cost of traversing the edge is less than a robot state transition between the configurations would be. Rather than a direct value, the weight represents the desired cost relative a normal robot state transition. In our implementation, the metric is time, and we set this cost to 0 to bias the system into performing interactions.

Algorithm 2 For disjoint workspaces discretize the boundary into segments and attempts to place the IT on each segment.

```
1: procedure FINDITLOCATIONS(IT i, Boundary b_1, Boundary b_2, Granularity g, Sampling Attempts N)
2:     Set of location transformations L ←
3:     b ←MINPERIMETER(b_1, b_2)
4:     segments ← discretize b into g segments
5:     for all s ∈ segments do
6:         for k ← 0 to N do
7:             Transformation t
8:             t.translation ← random point p ∈ s
9:             set t.rotation so that i's interaction edge is orthogonal to s with each robot on the correct side
10:            if ISVALID(i, t) then
11:                L ← L ∪ t
12:                break
13:            end if
14:        end for
15:    end for
16:    return L
17: end procedure
```

4.2 Interaction Template Transformation

There are two scenarios to consider when finding possible locations for interactions between robots: disjoint and overlapping workspaces. Here we present a method for both.

4.2.1 Disjoint Workspaces

One condition that can make the interaction of robots necessary for a MRS is the need to traverse disjoint workspaces (i.e., to move through regions of workspace that are not accessible to all of the robots). Placing an IT at the border of disjoint workspaces allows the task flow to cross such boundaries.

Disjoint workspaces for two robots implies that the configurations attached to the interaction edge must lie in their separate, corresponding workspaces. In order to simplify the search space for an IT location, the boundary of one of the workspaces in question is discretized (Alg. 2). A randomly sampled point along each discritized segment is checked for IT placement by validating

the interaction configurations on either side of the border. The number of segments and sampling attempts are parameters set by the user.

4.2.2 Overlapping Workspaces

Placing an IT in the overlapping workspace of robots in the MRS allows the IT planning method to identify useful interactions such as an optional hand-off between two robots of the same type or exchanges between different capabilities of robots (e.g. a manipulator unloading a truck). Regions of workspace where the obstacle topology is changing are often useful locations for interactions. Placing templates here enables the leveraging of potential interactions near significant workspace features.

These regions can be identified with a topological skeleton of the workspace, such as a medial axis [32]. The edges of this structure represent continuous regions of workspace, while the vertices represent junctions of such continuous regions. The skeleton vertices are thus reasonable positions for template placement (Alg. 3). We align the interaction edge so that it is roughly parallel to a pair of incident edges, which allows the flow to naturally follow the topology of free space. An additional optimization is to prune the skeleton relative to the task start and goal as described in [33].

After discovering possible interaction locations, we tile the ITs into the environment by transforming the IT roadmaps to each desired location (Fig. 4.1b), similar to the method in [34]. 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.

4.3 Construct Robot Type and Combined Roadmaps

The method constructs an initial robot-type roadmap for each robot-type in the MRS. These roadmaps connect all of the transformed IT roadmaps for the given robot-type. They are constructed via a coordinator agent, which employs a PRM algorithm to connect the IT roadmaps for each robot-type (Alg. 4). These are distributed to all of the robots of the corresponding robot-type which use them as initial solutions to generate their individual paths.

Algorithm 3 For overlapping workspaces, ITs are placed on skeleton nodes, denoting changes in the workspace topology.

procedure FINDITLOCATIONS(IT i, Boundary b_1, Boundary b_2, State *start*, State *goal*)
    Set of location transformations $L \leftarrow$
    $b \leftarrow b_1 \cap b_2$ ▷ Find the overlapping workspace.
    $S \leftarrow$ SKELETONIZE(b) ▷ Create a skeleton.
    PRUNE(s, $start$, $goal$)
    for all vertex $v \in S$ **do**
        Transformation t
        t.*translation* $\leftarrow v$
        set t.*rotation* so that the interaction edge is roughly parallel to a pair of incident edges of v
        if ISVALID(i, t) **then**
            $L \leftarrow L \cup t$
        end if
    end for
    return L
end procedure

The combined roadmap is an aggregate roadmap representing the entire MRS. It is composed of the transformed ITs and the robot-type roadmaps. The interaction edges in each IT connect the robot-type roadmaps to each other forming one combined roadmap for the system (Fig. 4.1c). The combined roadmap is implicitly constructed with the transformation of the ITs and the construction of the robot-type roadmaps. With the ITs providing locations for possible interactions, solving a motion planning task across the combined roadmap provides a way to evaluate robot interactions by only searching the roadmap without a task planning layer.

4.4 Task Planning

A motion satisfying the task presented to the MRS 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 robot type which can occupy these locations and added to their respective robot-type roadmaps. These robot start and end configurations are connected to the abstract task start and end nodes by zero weight edges. This enables a single motion plan query of the combined roadmap between the task start and end node, which

Algorithm 4 Create the robot-type roadmaps from set of transformed ITs I

```
1: procedure CREATEROBOTTYPEROADMAPS(Set of transformed ITs I, set of robot-types A,
   lazy graph-based motion planner L, task start and goal S, G)
2:     Set of robot-type Roadmaps R ←
3:     for all robot-type i ∈ A do
4:         initialize robot-type roadmap r_i
5:         for all t_1 ∈ I_ij do
6:             plan for r_i from t_1 to S with L
7:             for all t_2 ∈ I_ji do
8:                 plan for r_i from t_1 to t_2 with L
9:             end for
10:            for all t_2 ∈ I_ji do
11:                plan for r_i from t_2 to G with L
12:            end for
13:        end for
14:        R.append(r_i)
15:    end for
16:    return R
17: end procedure
```

naturally encodes the needed robot types at each step.

The plan from the task start and end models its flow through the environment and across the robot-types (Fig.4.1c). This plan amounts to a set of subtasks for the MRS.

4.5 Subtask Assignment

The configurations along the combined roadmap path indicate which robot-type is performing the task at that particular step. Task decomposition occurs at the points along the path where an interaction edge is taken indicating the task has been passed off to another robot via an interaction. As the entire task flow is planned across valid roadmaps, no invalid subtask will be generated.

The resulting subtasks are ordered by time priority and assigned using a task assignment algorithm. Many task assignment algorithms can be chosen. An auction system was used in our implementation. Each subtask is auctioned off to the robot of the specified type that can start the subtask.

The auction can be easily distributed, and while the rest of the approach utilizes a central

coordinator, this can be relaxed by delegating the creation of the robot-type roadmaps to robots of the corresponding type, and selecting a leader to generate the task plan and split it into subtasks.

5. THEORETIC ANALYSIS

The theoretical analysis of the IT method can be divide into the pre-processing stage and the planning stage. The set T denotes the set of all templates where T_{ij} is a template for an interaction between robots of type i and type j (Fig. 4.1a). We will use s to denote the cost of a simple motion plan in a small completely obstacle free environment and p to denote a nontrivial motion plan in all other scenarios. [1]

5.1 Pre-Processing

This first stage consists of constructing the templates, finding interaction locations, and robot-type roadmap construction. Each template is generated in a small obstacle free environment by computing two motion plans to generate both the roadmap and a path x representing the interaction sequence for both robots. The cost is $\mathcal{O}(s)$ for every $T_{ij} \in T$.

The set of all workspace regions is W, where W_i is the set of regions for robot-type i as each robot-type may have several regions. In Fig. 4.1b, W_{truck} consists of two regions, one on either side of the river. Interactions can take place across disjoint workspaces or within overlapping workspaces. The subset $T_{dis} \subseteq T$ denotes all templates that can occur across disjoint workspaces (e.g. a land and water robot interacting along the shoreline or a dock), and $T_{over} \subseteq T$ denotes all templates that can occur within overlapping workspaces (e.g. a manipulator loading a truck).

5.1.1 Disjoint Workspaces

Finding interaction locations in disjoint workspace consists of examining the border of every pair of disjoint workspace regions from the sets W_i, W_j for each $T_{ij} \in T_{dis}$. This border is discretized with a granularity g such that it consists of g segments. In Fig. 4.1b, the border of the river (i.e. the border between the disjoint land and water workspaces) has a granularity of $g = 2$, so that each side of the river becomes a segment, and one template is placed on either side.

Validating an interaction location requires transforming the template to the possible location

[1] Part of this section is reprinted with permission from, [1].

and performing collision checks along each path x representing the interaction sequences for the robots involved. As this cost can vary depending on the robots and the path length, we will denote the cost of collision checking a path $c(x)$. Thus the cost of validating an interaction location is $\mathcal{O}(c(x))$. The total cost for finding disjoint interaction locations is:

$$\mathcal{O}(c(x)g \sum_{T_{ij} \in T_{dis}} |W_i||W_j|) \tag{5.1}$$

5.1.2 Overlapping Workspaces

Finding interaction locations for overlapping workspaces requires intersecting pairs of workspace regions from the sets W_i, W_j for each $T_{ij} \in T_{over}$. We denote the cost of computing one of these intersections with n. The intersecting region is then skeletonized to create *skel* with a given skeletonization method of cost k. For each edge $e \in$ *skel* if the length is greater than some threshold we insert new vertices along the edge such that no edge has a length longer than the threshold. This results in a total number of vertices:

$$|V| = |V \in \textit{skel}| + \sum_{e \in \textit{skel}} \lfloor \frac{e.\textit{length}}{\textit{threshold}} \rfloor \tag{5.2}$$

Each $v \in$ *skel* is validated as an interaction location resulting in a cost of finding the overlapping workspace instances of:

$$\mathcal{O}((c(x)|V| + k + n) \sum_{T_{ij} \in T_{over}} |W_i||W_j|) \tag{5.3}$$

The set of instances of ITs is denoted I where I_{ij} is a an instance of a template T_{ij}. The templates in Fig. 4.1b are instances of the template in Fig. 4.1a. The interaction placement methods produce the set of disjoint instances I_{disj} and the set of overlapping instances I_{over} respectively.

To construct each robot-type i roadmap, each IT instance I_{ji} is connected to every instance I_{ij}. These connections are made by performing a motion plan and expanding the roadmap. The

cost of these nontrivial motion plans is denoted p as previously mentioned. The cost of connecting these instances for all robot types then is $\mathcal{O}(p \sum_i \sum_j |I_{ji}||I_{ij}|)$. An attempt is made to connect the task start and the task goal to all instances I_{ij} and I_{ji} respectively with costs $\mathcal{O}(p \sum_i |I_{ij}|)$ and $\mathcal{O}(p \sum_i |I_{ji}|)$. This results in a total cost for constructing the combined roadmap of:

$$\mathcal{O}(p \sum_i \sum_j |I_{ji}||I_{ij}|) \tag{5.4}$$

5.2 Plan Extraction

The planning stage consists of querying the combined roadmap, decomposing the discovered path, and assigning the subtasks to the individual robots. Let V be the set of vertices in the combined roadmap and E be the set of edges. A task plan via a Dijkstra search takes $\mathcal{O}(|E| + |V|\log|V|)$.

Let the length of the path generated be L. The task decomposition is a scan over the path to detect when it crosses over an IT and splitting it up into subtasks at these points. The cost of the scan is the length of the path $\mathcal{O}(L)$.

Each of these subtasks has a robot-type i and is auctioned off amongst members of R_i. Let the set of subtasks for each robot-type be U_i. Each bid consists of a query over a robot-type roadmap for the subtask with cost denoted by q. The total cost of the auction process then is $\mathcal{O}(q \sum_i |U_i||R_i|)$.

6. EXPERIMENTS AND RESULTS

We demonstrate the method for required and optional cooperation payload transport problems with 30 runs each. We employ our IT method and the TMP method FFRob to evaluate the performance. Both require an initial roadmap to approximate $\mathcal{C}_{free}$ for the MRS to evaluate feasible movements. In our IT method, this is referred to as the *combined roadmap* and in FFRob as the *reachability graph*. In this problem, these are equivalent and thus the preprocessing cost for both methods is the same. [1]

We provide FFRob with three actions to choose from: `Move`(*robot, start, end*), `StartTask`(*robot, task, location*), and `Interact`(*robot*$_1$*, robot*$_2$*, location*). These actions were available to all robots in the team and considered for a set of locations comprised of the robot starting positions, the IT locations, and the task start and goal location.

Both methods are given the transformed ITs and robot-type roadmaps. This allows FFRob to check the conditions for any action requiring a motion with only a graph search, and provides both methods the same knowledge about the connectivity of the workspace and possible interaction locations. We use time as the evaluation metric for both methods.

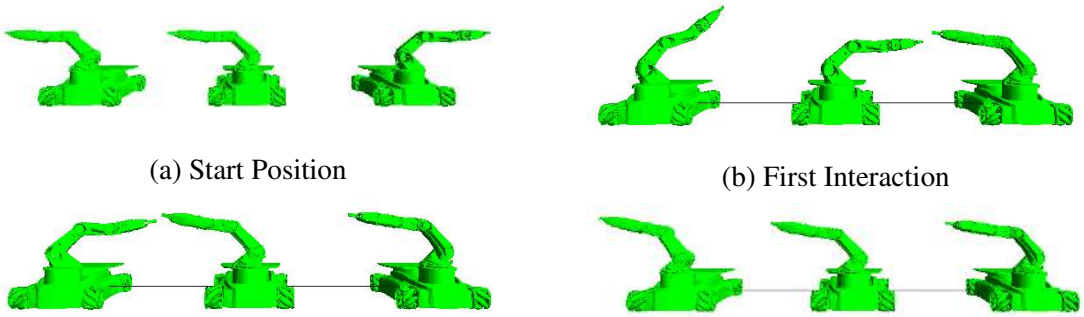
For many of the experiments we create heterogeneous robot types by splitting the robots into designated teams of 'boat robots' and 'truck robots.' The boat robots are capable of moving across regions of the environment consisting of water. The truck robots are capable of moving across land regions. These regions are represented by geometric boundaries and denote the disjoint workspaces. The border of these boundaries represents a shoreline in these experiments, and an IT along this boundary represents the passing of the task between a land robot and a water robot.

[1] Part of this section is reprinted with permission from, [1].

6.1 Required Cooperation

6.1.1 Manipulator Relay

In this experiment three simulated fixed base Kuka YouBots are arranged in a row. One manipulator can reach only the start, and another manipulator can reach only the goal. These two cannot reach each other, so an additional manipulator is placed in between them. The task is to move an object from the start on one side of the manipulator relay to the goal on the other side constituting a homogeneous required cooperation payload transport problem. Our IT method and FFRob find identical plans as there is only one solution: the first manipulator starts the task, passes the object over to the middle manipulator which passes the object on to the final manipulator (Fig. 6.1).



(a) Start Position

(b) First Interaction

(c) Second Interaction

(d) Final Position

Figure 6.1: The fixed base Kuka YouBots form a relay line to pass the payload from the right to the left. a) Only the right most manipulator can reach the start. b) The first interaction occurs between the rightmost and middle manipulators to pass the payload along the relay. c) The second interaction passes the payload to the leftmost manipulator which is able to reach the goal location. d) The leftmost manipulator then moves the payload to the final location. Reprinted with permission from, [1].

The pre-processing cost is the same for both methods, as they are building the same combined roadmap (Table 6.1).

The simulation time for both methods is roughly equivalent as they generate the same plan (Table. 6.2). The IT method finds the plan significantly faster as it only queries the combined

	Build IT		Place IT		Manipulator Roadmap	
	Time(s)	Std Dev(s)	Time(s)	Std Dev(s)	Time(s)	Std Dev(s)
Manipulator Relay	41.85	0.80	0.001	0.00	84.56	1.02

Table 6.1: Pre-processing times are shown for the manipulator relay. Reprinted with permission from, [1].

	Manipulator Relay			
	Planning		Execution	
	Time(s)	Std Dev(s)	Time(s)	Std Dev(s)
IT	3.21	0.13	103.7	0.4
FFRob	18.57	0.36	103.8	0.46

Table 6.2: Planning and execution times are shown for the manipulator relay. Reprinted with permission from, [1].

roadmap once and then auctions off the individual subtasks while FFRob qureies the robot-type roadmaps for each action considered as it searches the action space.

6.1.2 Rivers

To create required cooperation between the boat and truck robots, we created an environment in which the land is bisected by a river (Fig. 4.1c). The task is to move the payload from one side of the river to the other. We then extended this environment for five rivers to compare how the methods scale. Both river experiments constitute a heterogenous required cooperation payload transport problem.

The five river experiment MRS consists of 5 boat robots and 6 truck robots with a boat robot in each river and a truck robot in each land region. The time to construct the robot-type roadmaps increases as the number of ITs to connect increases with the number of rivers and granularity for IT placement (Fig. **??**). The single river experiment uses a granularity of 2, and the five river experiment uses a granularity of 4. The increase in granularity is due to an increase in the length of the river shoreline.

The pre-processing is again the same, as they both compute the same combined roadmap. As

	Build IT		Place IT		Boat-Type Roadmap		Truck-Type Roadmap	
	Time(s)	Std Dev(s)	Time(s)	Std Dev(s)	Time(s)	Std Dev(s)	Time(s)	Std Dev(s)
Single River	0.09	0.01	0.21	0.02	0.85	0.14	2.13	0.70
Five Rivers	0.04	0.00	0.48	0.02	53.65	4.76	79.66	16.55

Table 6.3: The pre-processing time is shown for both the single river and five rivers experiments. Reprinted with permission from, [1].

	Single River				Five Rivers			
	Planning		Execution		Planning		Execution	
	Time(s)	Std Dev(s)	Time(s)	Std Dev(s)	Time(s)	Std Dev(s)	Time(s)	Std Dev(s)
IT	1.01	0.06	52.07	3.67	29.56	2.14	159.06	13.25
FFRob	4.36	1.02	50.62	3.31	1839.55	221.11	165.39	12.51

Table 6.4: The planning and execution times are shown for both methods for both the single and five rivers experiment. The execution time is roughly equivalent, however, the planning time is drastically improved in the IT method, especially once the experiment is scaled to a larger MRS. Reprinted with permission from, [1].

seen in Table 6.3, the cost increases as the number of rivers increase as the number of ITs per river is doubled resulting in 10 times the number of ITs to connect.

The execution time is similar for both methods in either experiment, as there is a required sequence of interactions to complete the task (Table 6.4). The choice of which ITs to use accounts for the variation in execution time. The improvement in planning time for the IT method is significant for both experiments. In the single river experiment the planning time is roughly $\frac{1}{4}$ the planning time of FFRob. In the five river experiment this number becomes $\frac{1}{60}$ with FFRob averaging 30 minutes to make a team plan while the IT method took 30 seconds.

This improvement is a result of not searching the action space for MRS problems. As the number of robots increases, the action space increases at a rate of $|R|^2$ when considering multi-robot interactions. The combined roadmap grows linearly with the number of robot types, and planning through the action space needs to consider at least the robot-type roadmaps to validate movement actions.

6.2 Optional Cooperation

We again use the boat and truck robots to build the MRS. However, instead of a river dividing the environment, the water region is a lake that occupies a central region (Fig. 6.2). One boat robot sits in the lake, and one truck robot starts on either side of the lake. This creates a heterogenous optional cooperation payload transport problem and provides the planning methods the option to choose to complete the task using only a single robot or to utilize multiple robots with an interaction at either end of the lake.

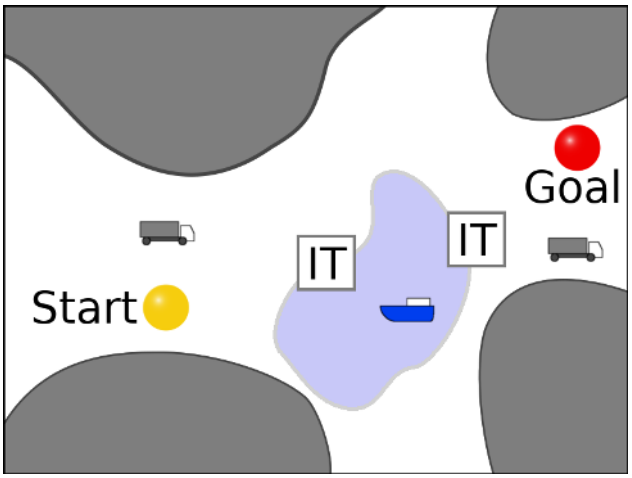


Figure 6.2: The small lake environment allows an efficient plan to be made either using or not using interactions as found by the planning method. Reprinted with permission from, [1].

We perform two experiments with this setup. In both cases the task can be solved with or without the interactions. The first has a smaller lake that lies slightly offset from a direct path between the start and goal location (Fig. 6.2). This creates a situation where utilizing a single robot can provide the most efficient solution. The second has a much larger lake that significantly increases the cost of a land-only solution, causing the interaction solution to be more efficient.

Constructing the robot-type roadmaps for the large lake scenario took longer with a larger environment and increased IT placement granularity for the longer shoreline (Table 6.5).

	Build IT		Place IT		Boat-Type Roadmap		Truck-Type Roadmap	
	Time(s)	Std Dev(s)	Time(s)	Std Dev(s)	Time(s)	Std Dev(s)	Time(s)	Std Dev(s)
Small Lake	0.09	0.01	0.04	0.00	0.30	0.06	1.57	0.12
Large Lake	0.09	0.01	0.07	0.01	1.77	0.22	3.46	0.47

Table 6.5: The pre-processing time is shown for both the single river and five rivers experiments. Reprinted with permission from, [1].

	Small Lake				Large Lake			
	Planning		Execution		Planning		Execution	
	Time(s)	Std Dev(s)	Time(s)	Std Dev(s)	Time(s)	Std Dev(s)	Time(s)	Std Dev(s)
IT	1.02	0.37	36.78	3.53	1.56	0.09	40.26	5.45
FFRob	3.15	0.53	35.27	2.82	10.16	2.57	63.54	2.15

Table 6.6: The planning and execution times are shown for both methods for both the small and large lake experiments. Reprinted with permission from, [1].

In the smaller lake scenario, the IT method chose to perform interactions 45.5% of the time and FFRob chose to never perform an interaction. In the larger lake scenario, the IT method chose to perform an interaction 91.5% of the time while FFRob again chose to never perform an interaction. In the instances where the IT method chose not to perform an interaction, the sampled locations for the ITs were at the far end of the lake relative the start and goal such that the path in the combined roadmap was shorter without interacting.

The IT method utilizes interaction templates whenever they result in the shortest path across the combined roadmap. FFRob's heuristic stops searching the action space once it determines that a set of actions can achieve the goal state. This reduces the amount of the state space that must be explored to find a solution, but ignores plans that use more actions even if the summed cost of executing those actions is lower (Table. 6.6). As either truck robot can move the object from the start to the goal by following the same action sequence, FFRob does not consider the longer action path that requires performing interactions despite it resulting in a more efficient plan. In our implementation this is evaluated by time, but as the IT method considers the edge weights in the combined roadmap, this can be any metric represented with edges (e.g. distance, energy).

7. CONCLUSION

A method for solving motion planning tasks for MRS utilizing interactions between robots has been described and analyzed. This method uses ITs to capture the motions for a multi-robot interaction in a roadmap and then transforms this information into the environment. These ITs are connected to represent possible task flow for the MRS. In MRS problems, the action space increases at a rate of $|R|^2$ when interactions between robots are considered. As a result, integrating the interaction information into a combined roadmap for all robot-types leads to a faster planning time and discovers solutions that some TMP methods miss in an effort to optimize the search through action space. [1]

Future work includes extending the framework to handle interactions involving three or more robots, time constraints, moving frames, and auxiliary interactions which do not represent a task flow (e.g. refueling).

[1]This chapter contains previously published material from [1].

REFERENCES

[1] J. Motes, R. Sandström, W. Adams, T. Ogunyale, S. Thomas, and N. M. Amato, "Interaction templates for multi-robot systems," *IEEE Robot. and Automat. Letters (RA-L)*, June 2019.

[2] C. R. Garrett, T. Lozano-PÃl'rez, 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.

[3] 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.

[4] 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 *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)*, pp. 4575–4581, IEEE, 2011.

[5] 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.

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

[7] 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.

[8] 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.

[9] 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.

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

[11] 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.

[12] 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.

[13] 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.

[14] 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 *Proc. IEEE Int. Conf. Intel. Rob. Syst. (IROS)*, October 2000.

[15] 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.

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

[17] M. B. Dias, *TraderBots: A New Paradigm for Robust and Efficient Multirobot Coordination in Dynamic Environments*. PhD thesis, Carnegie Mellon University, 2004.

[18] B. P. Gerkey and M. J. Mataric, "Sold!: Auction methods for multirobot coordination," in *IEEE Trans. Robot. Automat.*, October 2002.

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

[20] S. Cambon, R. Alami, and F. Gravot, "A hybrid approach to intricate motion, manipulation and task planning," in *Int. J. Robot. Res.*, January 2009.

[21] F. Gravot, S. Cambon, and R. Alami, "asymov: a planner that deals with intricate symbolic and geometric problems," in *Proc. Int. Symp. on Robotics Research (ISRR)*, pp. 100–110, Springer, 2005.

[22] 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.

[23] L. P. Kaelbling and T. Lozano-Pérez, "Hierarchical task and motion planning in the now," in *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)*, pp. 1470–1477, IEEE, 2011.

[24] E. Plaku and G. D. Hager, "Sampling-based motion and symbolic action planning with geometric and differential constraints," in *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)*, pp. 5002–5008, IEEE, 2010.

[25] 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.

[26] 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.

[27] 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.

[28] 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.

[29] 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.

[30] 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.

[31] 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.

[32] F. Chin, J. Snoeyink, and C.-A. Wang, "Finding the medial axis of a simple polygon in linear time," in *Proc. 6th Annu. Internat. Sympos. Algorithms Comput.*, vol. 1004 of *Lecture Notes Comput. Sci.*, pp. 382–391, Springer-Verlag, 1995.

[33] J. Denny, R. Sandstrom, A. Bregger, and N. M. Amato, "Dynamic region-biased exploring random trees," in *Proc. Int. Workshop on Algorithmic Foundations of Robotics (WAFR)*, (San Francisco, CA), December 2016.

[34] 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.