11-2022

Multi-Robot Symbolic Task and Motion Planning Leveraging Human Trust Models: Theory and Applications

Huanfei Zheng
huanfez@g.clemson.edu

MULTI-ROBOT SYMBOLIC TASK AND MOTION PLANNING LEVERAGING HUMAN TRUST MODELS: THEORY AND APPLICATIONS

A Dissertation
Presented to
the Graduate School of
Clemson University

In Partial Fulfillment
of the Requirements for the Degree
Doctor of Philosophy
Mechanical Engineering

---

by
Huanfei Zheng
December 2022

Accepted by:
Dr. Yue Wang, Committee Chair
Dr. John Wagner
Dr. Ardalan Vahidi
Dr. Yongqiang Wang

Abstract

Multi-robot systems (MRS) can accomplish more complex tasks with two or more robots and have produced a broad set of applications. The presence of a human operator in an MRS can guarantee the safety of the task performing, but the human operators can be subject to heavier stress and cognitive workload in collaboration with the MRS than the single robot. It is significant for the MRS to have the provable-correct task and motion planning solution for a complex task. That can reduce the human workload during supervising the task and improve the reliability of human-MRS collaboration.

This dissertation relies on formal verification to provide the provable-correct solution for the robotic system. One of the challenges in task and motion planning under temporal logic task specifications is developing computationally efficient MRS frameworks. The dissertation first presents an automaton-based task and motion planning framework for MRS to satisfy finite words of linear temporal logic (LTL) task specifications in parallel and concurrently.

Furthermore, the dissertation develops a computational trust model to improve the human-MRS collaboration for a motion task. Notably, the current works commonly underemphasize the environmental attributes when investigating the impacting factors of human trust in robots. Our computational trust model builds a linear state-space (LSS) equation to capture the influence of environment attributes on human trust in an MRS. A Bayesian optimization based experimental design (BOED) is proposed to sequentially learn the human-MRS trust model parameters in a data-efficient way.

Finally, the dissertation shapes a reward function for the human-MRS collaborated complex task by referring to the above LTL task specification and computational trust model. A Bayesian active reinforcement learning (RL) algorithm is used to concurrently learn the shaped reward function and explore the most trustworthy task and motion planning solution.

Acknowledgments

I wish to express my gratitude to my advisor Dr. Yue Wang who believed in me and supported me both financially and academically so I can reach this point as a PhD student.

I would also like to thank my committee members, Dr. John Wagner, Dr. Ardalan Vahidi, and Dr. Yongqiang Wang for providing valuable insights during my studies as a PhD student, for serving as my committee members, and for their constructive comments and suggestions.

Table of Contents

Title Page . . . . . **i**

Abstract . . . . . **ii**

Acknowledgments . . . . . **iii**

List of Tables . . . . . **vi**

List of Figures . . . . . **vii**

1 Introduction **1**
1.1 Research Motivation and Background 1
1.2 Contributions 3
1.3 Outline of Dissertation 6

2 Human-Robot Trust Integrated Task Allocation and Symbolic Motion Planning **7**
2.1 Introduction 7
2.2 Problem Setup 9
2.3 Task Allocation and Symbolic Motion Planning 11
2.4 Human-Robot Trust Model of Task Allocation 14
2.5 Simulation 20
2.6 Conclusion 22

3 Multi-Robot Task and Motion Planning under Temporal Logic Specifications **24**
3.1 Introduction 24
3.2 Preliminaries and Problem Setup 27
3.3 Related Work 34
3.4 Parallel Decomposition of Task Specification 36
3.5 Parallel Decomposition based Robot Assignment and Task Planning without Robot Transition System 40
3.6 Parallel Decomposition based Robot Assignment and Task Planning with Robot Transition System 43
3.7 Case Study: Symbolic Task and Motion Planning for MRS Manufacturing Task 56
3.8 Conclusion 65

4 Bayesian-based Trust Model for Human Multi-Robot Collaborative Motion Tasks **66**
4.1 Introduction 66
4.2 Preliminaries and Problem Setup 68
4.3 Related Work 71
4.4 Time-series Trust Model of Human Multi-Robot Collaboration 73
4.5 Bayesian Optimization Based Experimental Design 82

4.6 Case Study: Human Multi-Robot Collaboration for Bounding Overwatch Task 86
4.7 Conclusion 96

5 Bayesian Active Reinforcement Learning for Human Multi-robot Collaboration 98
5.1 Introduction 98
5.2 Preliminaries and Problem Setup 101
5.3 Related work 106
5.4 Reward Shaping for Human-MRS Collaboration 107
5.5 Bayesian Active Reinforcement learning 112
5.6 Case Study: Bayesian Active Reinforcement Learning for Human Multi-Robot Bounding Overwatch 121
5.7 Conclusion 127

6 Conclusions and Future Work 128
6.1 Conclusions 128
6.2 Limitations and Future Work 129

Appendices 131
A Equation Derivation 132

Bibliography 140

List of Tables

Table 2.1 Maximum trust of each robot before task reallocation. 22

Table 3.1 Robot's mobility between each station $\mathtt{i}$ and its designated station set $\Gamma_{\mathtt{i}}$ 58
Table 3.2 Robot capable states and the corresponding average time costs 58
Table 3.3 Scalability, computation and execution comparison between the parallel decomposition based task planning framework and centralized task planning. 64

List of Figures

Figure 2.1 Human-robot trust integrated task allocation and motion planning framework. $\{a,\ b\},\ \{c\}$ are the action sets of parallel subtask automata. $T_1, \cdots, T_i, \cdots, T_I$ are the respective trust value of robots. Ψ is the synthesized task allocation automaton. 10

Figure 2.2 A dynamic Bayesian network (DBN) based model for dynamic, quantitative, and probabilistic trust estimates. 15

Figure 2.3 Abstracted workspace and paths of motion planning where "cross" marks the obstacle and "star" marks the allocated task for each robot. 20

Figure 2.4 Parallel subtask automata G_1, G_2 and G_3. 21

Figure 2.5 Evolution of trust distribution of each robot. Black curves are the maximum values of trust distribution. 22

Figure 3.1 Flowchart of the proposed automaton based task decomposition and planning framework for the MRS. 26

Figure 3.2 (a) The DFA of the task specification "robots in the MRS achieve task B directly; or they first achieve task A and next achieve task B", (b) the DFA of the task specification "the MRS achieves tasks A and B in any sequence", (c) parallel decomposition of DFA G_{φ_2}, and (d) possible satisfaction processes of the two DFAs. DFA G_{φ_1} can be satisfied by "Plan 1"; DFA G_{φ_2} can be satisfied by either "Plan 1" or "Plan 2". . . . 29

Figure 3.3 The converted automata DFA of LTL task specifications φ_1 and φ_2. Note the DFA here is obtained by removing the transitions that infinitely visit the final states of the converted DBA from software "spot". 30

Figure 3.4 Triangulated 2D workspace. The obstacle polygons are over-estimation of the actual size of the obstacles. 32

Figure 3.5 (a) The global task DFA G_g with event set E_g, (b) the event-equivalent words composed automaton $\check{G}_1$ with event set $\check{E}_1 = \{\pi_e, \pi_a, \pi_b, \pi_c\}$, and (c) the event-equivalent words composed automaton $\check{G}_2$ with the event set $\check{E}_2 = \{\pi_e, \pi_a, \pi_b, \pi_d\}$. 37

Figure 3.6 Automaton $\check{G}_2$ has event set $\check{E}_2 = \{\pi_e, \pi_a, \pi_b, \pi_d\}$, and the set of event set pairs $\mathbb{E}_2 = \{(\{\pi_e, \pi_a\}, \{\pi_e, \pi_b, \pi_d\}),\ (\{\pi_e, \pi_b\}, \{\pi_e, \pi_a, \pi_d\}),\ (\{\pi_e, \pi_d\}, \{\pi_e, \pi_a, \pi_b\})\}$, (a) the projections of $\check{G}_2$ to the event set pairs $(\{\pi_e, \pi_b\}, \{\pi_e, \pi_a, \pi_d\})$ and the composition result of the projections; (b) the projections of $\check{G}_2$ to the event set pairs $(\{\pi_e, \pi_b\}, \{\pi_e, \pi_a, \pi_d\})$ and the composition result of the projections. 39

Figure 3.7 Illustration of two example robot-task assignments in the paper [103]. Assignment example 1 shows the assignment of the "white" robot to switch across the two corresponding subtask automata and collaborate with the "yellow" robot, and the "black" robot to switch across two subtask automata and collaborate with the "green" and "blue" robots. Assignment example 2 shows the assignment of the "white" and "red" robots as one subgroup and the "black" and "blue" robots as another subgroup. Both example assignments assume every robot can transit among states arbitrarily. The validity of the assignments can be pretty different if each of the robots has state transition restrictions (as the case considered in this paper and illustrated in the grey shaded box in the left bottom corner). For example, in assignment example 2, assume the "yellow" robot is assigned to the subtask automaton labeled with $\pi_0\pi_2\pi_1$. The robot can not accomplish the assigned subtask automaton given the state transition constraints $s_0 \to s_1 \to s_2$, i.e., the task plan $s_0 \to s_2 \to s_1$ is not viable for the "yellow" robot. 44
Figure 3.8 (a) Robot transition systems $TE_1,\ TE_2$, and TE_3, (b) subtask automaton G_1 and SPA $\Psi_1^{s,1}$ ($\Psi_1^{s,2}$ and $\Psi_1^{s,3}$) with respect to robot assignment $\{r_1\}$ ($\{r_2\}$ and $\{r_3\}$), (c) subtask automaton G_2, parallel composition $TE_1 \| TE_3$, and SPA Ψ_2^s (in minimized size) synthesized from $TE_1 \| TE_3$ and G_2. 47
Figure 3.9 (a) Robot transition systems $TE_4,\ TE_5$, and TE_6, (b) subtask automaton G_3 and G_4, and (c) parallel SPA Ψ^c (in minimized size) synthesized with $(TE_4 \| TE_5 \| TE_6) \times (G_3 \| G_4)$. The robot index corresponding to each composite state is $\langle r_4, r_5, r_6 \rangle$ in the parallel composition of transition systems. 50
Figure 3.10 Example typologies of robot assignment. 53
Figure 3.11 (a) Task plan implemented in a sequential manner, (b) task plan achieved in parallel at steps $\langle 2, 2 \rangle$ and $\langle 3, 2 \rangle$ by implementing α_b and α_c simultaneously. The steps and arrow length do not reflect the actual time. 55
Figure 3.12 Triangulated 2D workspace. The obstacle polygons are over-estimation of the actual size of the obstacles. 58
Figure 3.13 Transition system of each robot based on its capabilities and the abstracted environment. Each state $s_n^{\mathtt{i}}$ is labeled with $\pi_{\mathtt{i}}$, $\mathtt{i} = \epsilon, \mathtt{A}, \cdots, \mathtt{G}$. To make better visualization, simplify the graph here by saying each idle state s_ϵ can transit to all the other normal working states inside the dotted circle and vice versa. 59
Figure 3.14 The paths of MRS in the working environment for completing the manufacturing tasks. 61

Figure 4.1 Architecture of the Bayesian optimization based trust model for human-MRS collaborative motion task. The left figure describes the overall procedure of the sequential experiment of BOED. The right figure presents the detailed processes of estimating the trust model parameters' posterior distribution and obtaining the preferable path in trial s. 68
Figure 4.2 Trust-based motion planning in human-MRS collaboration 69
Figure 4.3 The causality graph of the computational trust model in human-MRS collaboration. . 71
Figure 4.4 The MRS navigation process, where $T1$ is the human-operated subteam of robots, and $T2$ is the autonomous subteam. Each subteam moves forward in a line formation, and the two subteams move abreast in a column formation. Each square represents a cell region for the robot teams to perform bounding overwatch. The red sectors denote that the robots are observing the bounding team to cover them in case of any dangerous or uncertainty situations. The gray lines label the footprints of the robots. . 86
Figure 4.5 (Left) A global view of the geological LiDAR information generated human-MRS bounding overwatch environment. Trees are simplified as triangular prisms. (Right) A local view of human-MRS bounding overwatch in Gazebo simulator. Robot r_0 is the human-operated robot. Robots $r_1,\ r_2$ and r_3 are the autonomous robots. 88

Figure 4.6 (a) - (e) present five discrete paths ρ_1 - ρ_5. The human-MRS performs bounding overwatch between each two discrete cells according to Fig. 4.4. The white patches are the top view of the environmental obstacles. Trees are scattered in the environment though not visible in appearance. The cells with cross are the risky regions where robots have high potential to collide with obstacles or turn over due to low traversability. The circle labeled cells are the risky regions where the human operator may lose track of the autonomous team members due to low visibility. 88
Figure 4.7 Human-MRS interaction interface. The human operator provides trust through clicking the sliders (left panel). The front camera view of each robot is shown (middle panel). Traversability and visibility for each robot (right panel) are plotted in real-time for human reference. 90
Figure 4.8 A Bayesian update of the posterior probability density function (PDF) of model parameter $\boldsymbol{\beta}$ under the BOED at trials 1, 5, 10, 15, and 20. The credible interval of the posterior distribution gets more and more condensed as the number of trials increase. The vertical dot-dash lines represent the location of the ground truth values of the simulated human's trust model parameters $\boldsymbol{\beta}$. The mean of the posterior distribution also approximates to the ground truth. 91
Figure 4.9 The information of the sixth trial of a participant: traversability, visibility, predicted trust, predicted trust change (curve "prediction") as well as the human provided actual trust change of robots (curve "observation"). 93
Figure 4.10 (Top) BIC of CTM0 and CTM; (Bottom) BIC difference value between CTM0 and CTM of every participant. BIC difference larger than 0 (labeled "+") indicates that the CTM fits better than the CTM0 for the participant. BIC difference smaller than 0 (labeled "−") indicates that the CTM0 fits better than the CTM. 94
Figure 4.11 Comparison of C1 and C2 with the 32 participants regarding the number of collisions, frequency of losing contact between robot subteams, time of completing the task, workload, usability, and situational awareness. 95

Figure 5.1 A graph representation of the labeled MDP for a robot. 99
Figure 5.2 A graph representation of the labeled MDP for a robot. 102
Figure 5.3 A graph representation of the converted DFA for LTL$_f$ $\varphi = \Box\neg woody \wedge \Diamond rocky$. . . 104
Figure 5.4 A graph representation of the product-MDP. Node $\langle s_0, 0\rangle$ is the initial state of the product-MDP. Nodes $\langle s_0, 1\rangle$, $\langle s_1, 1\rangle$ and $\langle s_2, 1\rangle$ are the accepting states. The state transition probabilities between the solid arrows are unknown, while the state transition probability between any two unconnected nodes is 0. 110
Figure 5.5 Scenario 1: (a) Top view of scenario. (b) Traversability map of the scenario. (c) Visibility map of the scenario. 123
Figure 5.6 The optimal trajectory pool of Bayesian active RL and the selected optimal trajectory (red) of each query strategy. The right most is the ground truth reward function under the simulated human agent. The arrows depict the optimal policy after 20 episodes. . 124
Figure 5.7 Scenario 2: (a) Top view of scenario. (b) Traversability map of the scenario. (c) Visibility map of the scenario. 125
Figure 5.8 Scenario 3: (a) Top view of scenario. (b) Traversability map of the scenario. (c) Visibility map of the scenario. 125
Figure 5.9 Trust-based Bayesian active RL results in scenario 2: (top) update of optimal trajectory pool under LTL$_f$ "$\Diamond dest$". (bottom) update of optimal trajectory pool under LTL$_f$ $\Box\neg peak \wedge (\Diamond gully \wedge \bigcirc\Diamond dest)$. 126
Figure 5.10 Trust-based Bayesian active RL results in scenario 3: (top) update of optimal trajectory pool under LTL$_f$ "$\Diamond dest$". (bottom) update of optimal trajectory pool under LTL$_f$ $\Box\neg peak \wedge (\Diamond int \wedge \bigcirc\Diamond dest)$. 126

Chapter 1

Introduction

1.1 Research Motivation and Background

Multi-robot system (MRS) has achieved a broad of applications in areas such as the warehouse robots, manufacturing manipulators, and vehicle platooning. These application of MRS has significantly increased the level of autonomy and improved the task efficiency. Despite advances in MRS task performing, the robot coordination and the possibility of unintended emergent behaviors, etc. remain to be potential risks that cause the safety issues of MRS task performing.

Temporal logic task [12] can specify the logic sequence of many "go-to-goal" tasks along a timeline under a task performing scenario, while a simple "go-to-goal" task only requires a robotic system to generate the trajectory to the goal position. Recently, temporal logic task specifications become increasingly popular in describing the time property of robot task performing. Many works have achieved the satisfaction of temporal logic tasks for MRS. However, the task assignment and plan to satisfy the temporal task specification is not singleton nor obvious to identify. Therefore, the generic problem of temporal logic described task is guaranteeing the provable-correctness of the robot-task assignment and planning.

The most recent work commonly relies on model checking techniques to obtain the task plan for temporal logic task. The general idea of model checking [25, 20, 19, 88, 86, 87, 34, 35, 77, 78] based task and motion planning is that robotic system first obtains the the counter-example in its state transition that violates the negation form of the temporal task; then the counter-example can be the robot's task plan satisfying the temporal task. The model checking is effective in guaranteeing the satisfaction of a robotic system for a task specification which usually contains the reachability, safety and liveness property of robot task performing.

However, the model checking has the state space explosion problem during the generation of the correct task plan. Then, the focus of the problem becomes developing a computational efficiently framework for robot to accomplish the task planning [12, 91].

On the other hand, human supervision is still necessary to ensure safe and efficient operations in uncertain, dynamic, or noisy environments where robot sensing and perception may not be fully reliable. The reason is humans excel at high-level decision-making in such environments and can help autonomous robots achieve better performance while keeping design costs low. However, human error is also a main cause of machine malfunctions, and human performance degrades when overloaded. When designing autonomous robotic systems, it is therefore important to consider factors related to human-robot interaction (HRI).

Extant HRI solutions highly specialized and focused on human-machine interface (HMI) design [13]. The design process for robotic systems in high-level decision-making and coordination, is still largely one of trial and error. The process often lacks quantitative models and real-time analytic approaches that could be used to provide safety and performance guarantees. Furthermore, the problem that a single human interacts with multiple autonomous robots is especially challenging due to the problem size, the need for robot coordination, the possibility of unintended emergent behaviors, etc.

This dissertation considers trust as metric to improve the HRI performance, where trust is defined as "the attitude that an agent will help achieve an individual's goals in a situation characterized by uncertainty and vulnerability". Humans respond socially to robots, establishing a level of trust to manage workload not possible with mere human endeavor. The informed trust is an assessment of when and how much autonomy should be employed, and when to intervene. Then, humans can either gain or lose trust in robots based on the progress of the task [22]. Finally, trust can be a dynamic feature of HRI that heavily affects a human's acceptance and hence use of a robot.

In addition, the task planning of robotic system requires the reward or cost so that system can generate the optimal task plan. Consideration of trust in HRI as a reward or cost can guarantee the human workload to be kept within acceptable bounds. It is especially important for the supervisory control of multiple robots. The originate model checking only concentrates on the counter-example that does not violate the temporal logic task specification. Therefore, a trust based task and motion planning framework is significant in improving the usability of the temporal logic task in robot task performing.

1.2 Contributions

1.2.1 Trust-based Task Assignment for MRS Symbolic Motion Planning

This chapter presents a human-robot trust integrated task allocation and motion planning framework for MRS in performing a set of tasks concurrently. A set of task specifications in parallel are conjuncted with MRS to synthesize a task allocation automaton. Each transition of the task allocation automaton is associated with the total trust value of human in corresponding robots. Here, the human-robot trust model is constructed with a dynamic Bayesian network (DBN) by considering individual robot performance, safety coefficient, human cognitive workload and overall evaluation of task allocation. Hence, a task allocation path with maximum encoded human-robot trust can be searched based on the current trust value of each robot in the task allocation automaton. Symbolic motion planning (SMP) is implemented for each robot after they obtain the sequence of actions. The task allocation path can be intermittently updated with this DBN based trust model. The overall strategy is demonstrated by a simulation with 5 robots and 3 parallel subtask automata.

The contribution of the chapter is two-fold. First, we synthesize an automatic task (re)allocation framework that can generate solutions with maximum human-robot trust for the system. It enables real-time updating of task allocation of robots in a human-like way. Furthermore, we construct a dynamic Bayesian network (DBN) based human-robot trust model. This model will evaluate the robot performance, safety, human cognitive workload, and the task (re)allocation framework in a system wide trust setting.

1.2.2 Symbolic Task and Motion Planning of MRS

Chapter 3 presents an automaton-based task and motion planning framework for MRS to satisfy finite words of linear temporal logic (LTL) task specifications in parallel and concurrently. A parallel decomposition algorithm is developed to iteratively decompose a global task specification into a set of smaller subtask automata. Robots are assigned to the smallest task component in each subtask automaton. The capability transition system of the assigned robots and these subtask automata synthesize a corresponding set of subtask planning automata (SPA), each of which is either an independent satisfaction of an individual subtask automaton or a concurrent satisfaction of multiple subtask automata. The overall robot assignments and SPA can guarantee the MRS to satisfy all the subtask automata. Each SPA can generate a minimal cost task plan by taking into account the costs of multi-robot tasking. The robots then plan motions to execute the tasks

associated with the minimal cost task plans. The proposed framework is demonstrated with a multi-robot experiment for manufacturing tasks in a lab setting. Extensive numerical simulations are also performed to evaluate the scalability, computational complexity, and execution efficiency of the proposed framework and show its advantages over the centralized task and motion planning strategy.

The main theoretical contributions of the chapter are summarized as follows:

1. A global task specification is parallel decomposed into a unique set of parallel executable subtask specifications. The decomposition process considers the generation of variable event sets that can make the task specification parallel decomposable, which is missing in the extant decomposition frameworks [45, 20, 77, 78].

2. Assuming robots with overlapping capabilities, the robot assignment to each subtask specification is determined by considering the level of parallelism of multi-robot task and motion planning. The parallel decomposition based task assignment enables the MRS to satisfy the atomic tasks of task specification in parallel. It is more efficient in executing the task plan than the centralized task and motion planning approach, which assigns robots to perform the atomic tasks consecutively.

3. A set of subtask planning automata (SPA) is synthesized from the subtask automata and corresponding robot state transition systems after the parallel decomposition based robot assignment. These SPA can guarantee viable switches of robots between different parallel subtask specifications. The optimal task planning solution can be obtained from these SPA by taking into account the cost of task execution. The parallel decomposition based SPA have much smaller size of state spaces and require less computation to obtain the task planning solution compared with the centralized framework.

1.2.3 Computational Trust model for MRS

In a human multi-robot collaborative task, an appropriate level of human trust in the MRS can release the human operator's stress and cognitive workload, thus improving collaborative task performance. Chapter 4 develops a computational trust model to improve the human-MRS collaboration for a motion task under an offroad environment. The computational trust model describes the decision-making of the MRS with a time series linear relation between human's trust in MRS and the MRS situational awareness, such as traversability and line of sight. Bayesian statistics are first used to learn the computational trust model parameters with the corresponding human-MRS collaborative task trials. Since human-MRS collaboration trials are cost-expensive, we perform Bayesian optimization based sequential trials to iteratively collect the

robot situational awareness information and update the learned trust model parameters. The path for the robots to travel through in each iteration is determined to be the up-to-date optimal path. Finally, human subjective tests are conducted for a human-MRS collaborative bounding overwatch task in the ROS Gazebo simulator. The test results demonstrate that (1) human's trust in a leading robot has a significant influence on the trust in its succeeding robot in the MRS bounding overwatch task, (2) the Bayesian optimization strategy can improve the cost performance of human-MRS trials. The main contribution of Chapter 4 can be summarized as follows,

1. We assume that the human's trust in each robot of MRS is a continuous-valued time series data. A generalized linear state space (LSS) model is developed to capture the inter-robot trust influence for the human-MRS collaboration.

2. We use Bayesian inference to estimate the parameters of the LSS model. The Bayesian inference method estimates the posterior probability distribution of the trust model parameters based on the observations' likelihood and a prior belief of the model parameters.

3. We rely on Bayesian optimization strategy to obtain the optimal path for the human-MRS collaborated motion task. It reduces the unnecessary trials on the paths that may not be valuable to observe the robot behaviors.

4. We deploy multiple ground robots in the ROS Gazebo simulator to perform a bounding overwatch task to demonstrate the effectiveness of the computational trust model for the human-MRS collaborative motion task.

1.2.4 Trust-based Reward Function Learning and Optimal Trajectory Exploration

Chapter 5 develops trust-based Bayesian active reinforcement learning (RL) framework for a human multi-robot collaborative system to accomplish an offroad motion task. On the basis of Chapter 4, Chapter 5 first captures the human trust dynamics evolution in the motion task with a computational human-MRS trust model, which can encode the human's trust in the robots as a reward function of the labeled MDP of human-MRS. Then, it utilizes LTL formulae to encode the human's task requirements for MRS, such as the motion reachability and safety, in the offroad environment. The LTL formulae plus the labeled MDP of the robots' motion behaviors synthesize a product-MDP for the human-MRS, which guarantees the provably safe behaviors of human-MRS in the task performing. Next, different query strategies of the Bayesian active RL

are developed for human-MRS to simultaneously learn the trust-based reward function and find the optimal trajectory. A case study on human-MRS collaborative offroad motion task illustrates the effectiveness of the proposed algorithm. The contribution of this chapter is as follows:

1. We shape and learn a trust-based reward function for trajectory planning of the human-MRS collaborative offroad motion tasks.

2. We synthesize the product-MDP to provide the provably correct state-actions for the human-MRS to satisfy the LTL task specification. It is used to shaped an LTL_f-based reward function for human-MRS trajectory planning which can meet the human's task requirement for robot reachability and safety in the task environment.

3. We integrate the human trust-based reward function and LTL_f-based reward function into a Bayesian active RL algorithm, which can explore the most trustworthy trajectory for human-MRS to travel along and annotate data.

4. We continue the case study of human-MRS collaborative bounding overwatch. It verifies the advantages of our proposed active RL framework in satisfying the human requirement on the MRS offroad motion task and improving the RL reliability for the human-MRS collaborative task.

1.3 Outline of Dissertation

The remaining chapters of this dissertation are organized as follows. Chapter 2 presents a framework for trust-based task assignment for MRS to accomplish a temporal logic task. Chapter 3 presents a parallel decomposition framework for multi-Robot task and motion planning under temporal logic specifications. Chapter 4 investigates a generalized computational trust model for human-MRS collaboration under an offroad environment. Chapter 5 proposes a human trust-based Bayesian active RL framework to learn the human's trust model and find the optimal policy to accomplish temporal logic task. Chapter 6 discusses the conclusions of this dissertation.

Chapter 2

Human-Robot Trust Integrated Task Allocation and Symbolic Motion Planning

2.1 Introduction

Symbolic motion planning (SMP) solves complex motion planning problems for robots using linear temporal logic (LTL), languages and automata theory [12]. It enables the automatic control of a robot or teams of robots from high level with different task specifications. However, computationally efficient frameworks are often needed to deal with the increasing complexity of task specifications and multi-robot system (MRS). Many centralized and decentralized frameworks have been developed to deal with the "state-space explosion" problems. A compositional multi-robot motion planning framework in [76] uses precomputed motion primitives for robots and employs a satisfiability modulo theory solver to synthesize robot trajectories. Event-based synchronization approach is proposed in [86] to address interdependencies among robots in motion planning. In [34, 35], a bottom-up strategy is proposed where each robot is assigned with a local task and inter-robot dependence is achieved through cooperative motion and task planning. In [25, 20], a top-down framework is presented for the automatic deployment of a robotic team from a specification by giving each robot the capabilities to serve the cooperation requirements. Supervisor synthesis with compositional verification techniques is utilized to guarantee robot performance in [23], where a given team mission

is decomposed into individual tasks.

The above multi-robot motion planning frameworks are however restricted either in scalability in terms of the size of robot teams, or in complexity of tasks due to the inter-dependencies among robots. The robot-task pairs are given as fixed in dealing with the global task specifications. That is, these works do not consider the task allocation problem. In this chapter, we will first establish a multi-robot multi-task task allocation framework to guarantee the reachability of tasks and optimal assignment of robots. The motion planning of each robots is implemented sequentially based on the task allocation results. Reallocations can also be triggered in this automatic process to deal with the uncertainties of the motion planning in the dynamic environment.

Under our proposed framework, SMP can therefore ensure automatic and scalable solutions for MRS motion planning. Moreover, human supervision may facilitate the efficiency and safety of MRS in a dynamic and uncertain environment because humans excel in complex decision-making and robot's performance in such scenarios is not usually not satisfactory. Various designs and analyses related with human-robot team co-operation have been conducted to improve human's situation awareness of robot. The work in [49] focuses on the development and evaluation of complex socio-technical system for human-robot teaming in Urban Search and Rescue with a user-centric design methodology, which ranges from modeling situation awareness, human robot interaction (HRI), flexible planning, and cognitive system design. In [79], an empirical analysis of human teamwork is conducted to investigate the ways teammates incorporate coordination behaviors, including both verbal and nonverbal cues, into their action planning. Measurable Shared Mental Models (SMMs) are developed in [64] to promote an effective human-robot teaming by observing a team of expert human workers prior to task execution, and then robots executing an interactive planning and cross-training process with a human co-worker to iteratively refine and converge the team model. The framework of discrete-time stochastic hybrid systems is utilized in [90] to model human-in-the-loop cyber-physical systems with discrete choices, and pose the question of expected outcome in terms of a stochastic reachability problem. The paper [18] constructs a POMDP based trust model for human to a single robot to improve performance of the joint human-robot system.

In this work, we will develop a human-robot trust integrated task allocation and motion planning framework for MRS in order to enable a human-like automatic decision-making process for multi-robot tasking. The contribution of the chapter is two-fold. First, we synthesize an automatic task (re)allocation framework that can generate solutions with maximum human-robot trust for the system. It enables real-time updating of task allocation of robots in a human-like way. Furthermore, we construct a dynamic Bayesian

network (DBN) based human-robot trust model. This model will evaluate the robot performance, safety, human cognitive workload, and the task (re)allocation framework in a system wide trust setting.

The organization of the rest of the chapter is as follows. Section 2.2 provides the problem setup with a schematic of the human supervised MRS. Section 2.3 describes the human-robot trust associated task allocation framework and symbolic motion planning of each robot for a set of parallel subtasks. Section 2.4 details the construction of the DBN based human-trust model, and integrates the human-robot trust evaluation into task (re)allocation and motion planning framework. A simulation in Section 2.5 demonstrates the viability and effectiveness of the proposed framework and Section 2.6 concludes this chapter.

2.2 Problem Setup

The schematic of human-robot trust integrated task allocation and SMP is shown in Figure 2.1. Initially, a task allocation automaton is synthesized for a task requirement that concurrently implementable subtasks are to be performed by a set of heterogeneous robots. The subtasks are described by automata and each robot can perform multiple actions in the subtasks. A task allocation path is generated with the maximum accumulated trust of robots from the task allocation automaton. Local action and motion specifications of each robot are mapped from the maximal trust encoded task allocation path so that each robot can execute the motions and actions sequentially. The SMP will also deal with the obstacle collision avoidance in the discrete environment. All these performance will be evaluated to contribute to the calculation of the computational human-robot trust model.

On the other hand, human is allowed to participate into this task allocation and motion planning process in order to improve system performance and reliability. A system-wide human-robot trust model is constructed based on the MRS task allocation by considering robot performance on task performing, safety evaluation on malfunctioning, human cognitive workload of supervision and inter-robot influence from task allocation. The trust of human in each robot will be updated with the progress of task performing. The system-wide trust model will increase or decrease the trust of robots involved with task allocation to construct the interdependence relationship among robots. Once an action is completed by a robot, human will be inquired for the task reallocation based on his/her trust in the current robot. Finally, the parallel subtasks will be completed with a maximum trust encoded task allocation solution and motion planning paths by intermittently updating the task allocation under human supervision.

We summarize the human-robot trust integrated multi-robot task allocation and motion planning

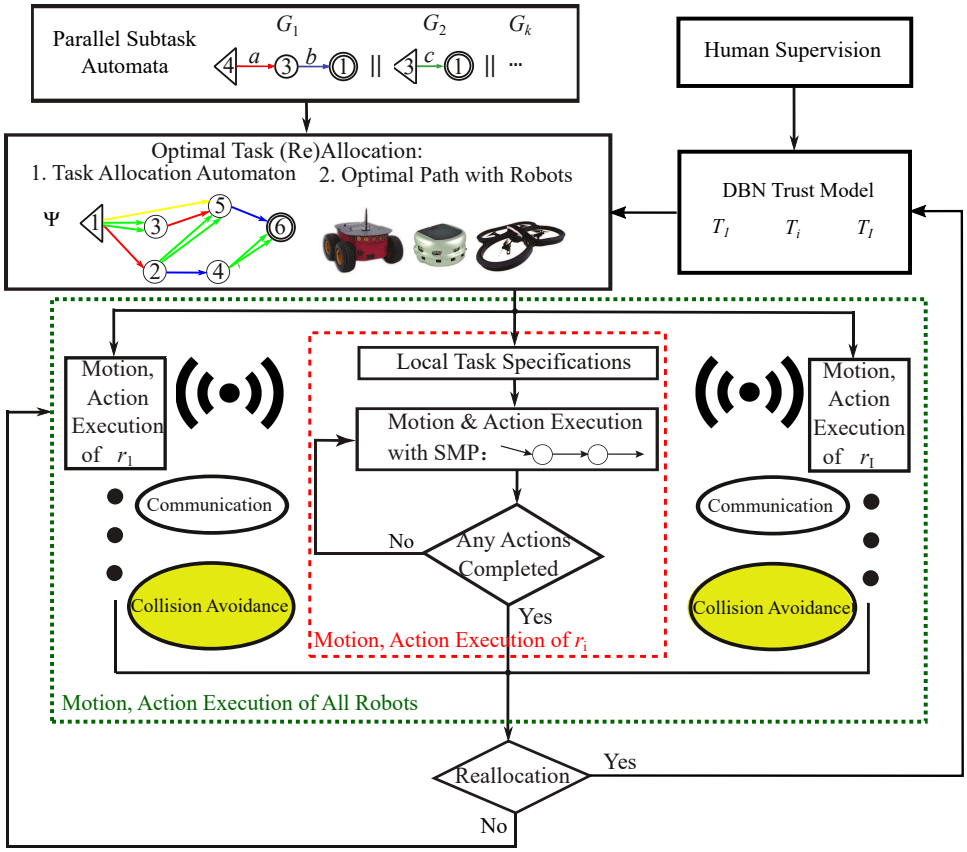


Figure 2.1: Human-robot trust integrated task allocation and motion planning framework. $\{a,\ b\}$, $\{c\}$ are the action sets of parallel subtask automata. $T_1, \cdots, T_i, \cdots, T_I$ are the respective trust value of robots. Ψ is the synthesized task allocation automaton.

with the following assumption and problem.

Assumption 1. Each heterogeneous robot $r_i \in \mathcal{R}$ is associated with an action set Er_i describing its capabilities in performing tasks. Assume that $\cup_{i=1}^{I} Er_i \supseteq E_g$, i.e. all the subtasks can be collaboratively performed by the MRS. Each robot may be able to perform multiple actions, but it is assumed that a robot can only perform one action at a time.

Problem of Interest 1. Given a set of parallel subtasks, described with automata $\{G_k, k = 1, \cdots, K\}$, each is associated with an action set E_k that robots need to perform, and all actions are scattered in a dynamic environment, design a task (re)allocation framework such that these subtasks can be completed by I heterogeneous robots $r_i \in \mathcal{R}$ with corresponding capability action set Er_i, where $i \in \{1, \cdots, I\}$; In the meantime, a human-robot trust model is integrated into the MRS to enable trust-based task reallocation such that human-like decision-making can be deployed for multi-robot multi-task allocation.

2.3 Task Allocation and Symbolic Motion Planning

The subtasks of MRS can be described with a set of automata $\{G_k, k = 1, \cdots, K\}$. These subtasks satisfy a parallel relationship with $\|_{k=1}^{K} G_k \cong ((G_1\| \cdots G_k)\| \cdots \|G_{K-1})\|G_K$, which represent a set of subtask automata G_ks that can be concurrently dealt with using a team of robots. We will construct the task allocation automaton with trust associated transitions for these parallel subtasks and robots so that a maximal trust encoded task allocation solution can be generated for the MRS.

2.3.1 Human-Robot Trust Associated Task Allocation

In order to find a maximum trust encoded task allocation solution, we synthesize a task allocation automaton Ψ by taking into account both robot capabilities and subtask automata. A path from task allocation automaton gives a task allocation solution regarding what actions to be allocated to what robots. To explain the process of generating a maximum trust encoded path from the task allocation automaton, we introduce the following definitions.

Definition 1 (Minimal Suffix Set of Language). Given an automaton G with an action set E, the minimal suffix set of language $L(G)$ is denoted as $\mathcal{L} = \{\ell \in E^* : \ell \text{ is the suffix of } \min(s),\ s \in L(G)\}$, where $\min(s)$ is one of the minimal length paths in $L(G)$ and E^* is the Kleene-closure of E.

Definition 2 (Implementable Action Set). Given a task automaton G with an action set E, an action $e \in E$ is said to be implementable for a robot r with a capability action set Er if the following two conditions are satisfied: 1) $e \in Er$, which means the action e can be performed by robot r, and 2) $e \in \{\ell(0)\}$, i.e. action e is ready to be performed at the states of G that match ℓ, where $\ell(0)$ is the first element of ℓ. Hence, an implementable action set of automaton G for robot r is denoted as $IA = \{(r, e) : e \in Er \cap \{\ell(0)\}\}$.

Accordingly, the implementable action set of automaton G_k for all robots in the set $\mathcal{R}$ can be given as $IG_k = \cup_{i=1}^{I} IA_{i,k}$, where $IA_{i,k} = \{(r_i, e_k) : e_k \in Er_i \cap \{\ell_k(0)\}\}$.

Definition 3 (Multi-action Set). The set $Act_\psi = \{act_\psi = (\omega_1, \cdots, \omega_k, \cdots, \omega_K) : \omega_k = (r_i, \hat{e}_k), r_i \in \mathcal{R}, \hat{e}_k \in Er_i \cap \{\ell_k(0)\} \cup \{\epsilon\}\}$ defines a multi-action set, where the single-action $\omega_i = (r_i, \hat{e}_k)$, $\hat{e}_k \neq \epsilon$ defines an action in automaton G_k performed by a robot in $r_i \in \mathcal{R}$; $\omega_k = (r_i, \epsilon)$ means no action in G_k is assigned to robot r_i. A multi-action $act_\psi \in Act_\psi$ holds if and only if it is (1) effective, i.e. $\exists \omega_k \notin \{(r_i, \epsilon), i = 1, \cdots, I\}$, (2) unique, i.e. $\forall \omega_k, \omega_{k'} \notin \{(r_i, \epsilon), i = 1, \cdots, I\}$, if $k \neq k'$, then $r_i \neq r_{i'}$. For simplicity of notation, we further denote $\omega_k = (r_i, \epsilon)$ as $\mathcal{E}$.

Remark 1. The multi-action set combines multiple implementable actions and guarantees not only the state transition of each subtask automaton but also the mutual exclusion of subtask automata for robots.

Finally, we can define the task allocation automaton for multiple robots to perform multiple tasks as follows.

Definition 4 (Task Allocation Automaton). The task allocation automaton Ψ describes the assignment of robots to the actions in subtask automata. It is given by a tuple $\Psi = (X_\psi, Act_\psi, \delta_\psi, \chi_0, W_\psi)$ where

1. $X_\psi = \mathcal{L}_1 \times \cdots \mathcal{L}_k \times \cdots \mathcal{L}_K$ is the composite state set of task allocation including a set of composite states $\chi = (\ell_1, \cdots, \ell_k, \cdots, \ell_K)$ for the parallel processes $\|_{k=1}^{K} G_k$, $\ell_k \in \mathcal{L}_k$, where $\mathcal{L}_k$ is the minimal suffix set of $L(G_k)$,

2. the multi-action set $Act_\psi \subseteq \prod_{k=1}^{K}(IG_k \cup \{\mathcal{E}\})$ (Def. 3) includes a finite set of actions $act_\psi \in Act_\psi$ that the heterogeneous robots group $\mathcal{R}$ can perform,

3. the transition relation $\delta_\psi(\chi, act_\psi)$ is a process $\chi \xrightarrow{act_\psi} \chi'$, which can be detailed as $(\ell_1 \xrightarrow{\omega_1} \ell_1', \cdots, \ell_K \xrightarrow{\omega_K} \ell_K')$,

4. χ_0 is the initial state of task allocation,

5. $W_\psi : Act_\psi \to \mathbb{R}$ is the set of accumulated trust of all robots associated with the completion of each action, $W_\psi(act_\psi) = \sum_{i=1}^{I} T_i(t)$, where $T_i(t)$ is the trust of human in a single robot r_i at time t.

A finite path $\mathbb{S}_\Psi = act_\psi^{(0)} \cdots act_\psi^{(\tau)} \cdots act_\psi^{(\mathbf{T})}$ with $act_\psi^{(\tau)} \in Act_\psi$ presents a task allocation solution for the parallel processes $\|_{k=1}^{K} G_k$ with robots r_i, $i = 1, \cdots, I$.

An initial task allocation $\mathbb{S}_\psi$ with maximum accumulated trust of all robots $\max(\sum_{\tau=0}^{\mathbf{T}} W_\psi)$ from initial state χ_0 to final state χ_F can be generated by searching the task allocation automaton Ψ. The maximum trust encoded path presents the optimal assignment of robots for all actions in a human-like decision-making pattern, since the associated trust values in task allocation automaton are evaluated with the impact factors in Section 2.4, such as robot performance, safety, human cognitive workload, and system wide trust evaluated task allocation. The accept state is reached when all subtask automata are reduced to be empty and all its actions are completed. The parallel process based task allocation is conducted among heterogeneous robots without subtask inter-dependency.

2.3.2 Symbolic Motion Planning

The maximum trust encoded task allocation path may provide a task performing sequence for each robot in a human-like decision-making process, but it is also necessary to consider how to deal with the reachability of all these actions in the dynamic environment. For SMP with a team of robots, paths satisfying the global specification can be generated by model checking, which encodes each robot a single path in the abstracted workspace.

In this work, we assume that the task environment is not known a prior, which is also an important prerequisite for robot performance estimation on obstacle avoidance to be discussed in Section 2.4. The paths of motion planning for robots are intermittently replanned upon the information they get through exploring the area. The mapping of $\mathbb{S}_\psi$ into each r_i from initial step to step $\mathbf{T}$ will give each robot a task allocation path $s_{\psi,i} = \omega_{k_0}^{(0)} \cdots \omega_{k_\tau}^{(\tau)} \cdots \omega_{k_\mathbf{T}}^{(\mathbf{T})}$, where $k_\tau \in \{1, \cdots, K\}$ is an index of subtask automaton.

Denote $s_{\pi,i} = \pi_{k_0}^{(0)} \cdots \pi_{k_\tau}^{(\tau)} \cdots \pi_{k_\mathbf{T}}^{(\mathbf{T})}$ as the corresponding sequence of motion specifications of r_i. $\pi_{k_\tau}^{(\tau)}$ describes the reachability of $\omega_{k_\tau}^{(\tau)}$ with "a robot r_i will go to the position of action $e_k \in E_k$ if and only if it finds that the previous action $\omega_{k_\tau}^{(\tau-1)}$ has been completed". Thus, it requires each $\pi_{k_\tau}^{(\tau)}$ to be conducted before $\omega_{k_\tau}^{(\tau)}$. The motion specification can guarantee the actions to be conducted in a logic sequence by robots in the decentralized multi-robot motion planning, and every robot obtains the information about completion of action $\omega_{k_\tau}^{(\tau-1)}$ by traveling to the location itself.

Definition 5 (**Product Automaton of Robot).** Given an automaton $A = (X, E, f, x_0, X_m)$ and a robot transition system $TS = \big(Q, \delta, q_{init}, \pi, L_q, W_q\big)$. We define the product automaton $\mathcal{P} = TS \times A = (\hat{X}, E, \hat{f}, \hat{x}_0, \hat{X}_m, W_q)$, where

1. $\hat{X} = Q \times X$ is the state set,
2. E is the event set for the transitions,
3. $\hat{f}(\hat{x}, e') = \hat{x}'$ is the transition relation with $\hat{x} = (q, x),\ \hat{x}' = (q', x')$, $q \to q',\ f(x, e') = x'$, where $q, q' \in Q$ and $x, x' \in X$,
4. $\hat{x}_0 = (q_{init}, x_0)$ is the initial state, $\hat{X}_m = Q \times X_m$ is the final state set,
5. W_q is the cost set for the transitions in δ.

The transition system of robot r_i is abstracted as $TS_{i,k}^{(\tau)}$ for each action $e_k \in E_k$ it is going to perform at step τ in the discrete space (e.g. see Figure 2.3). $\mathcal{A}_{k,i}^{(\tau)}$ is an automaton representation of the

motion specification $\pi_{k_\tau}^{(\tau)}$ of robot r_i regarding subtask automaton G_k. The model checking $TS_{i,k}^{(\tau)} \times \mathcal{A}_{k,i}^{(\tau)}$ can provide a motion planning path $\sigma_{i,k}^{(\tau)}$ satisfying a motion specification $\pi_{k_\tau}^{(\tau)}$.

Collision avoidance can be dealt within the transition system through a reactive approach. Each robot r_i detects its abstracted surrounding area and stores the detected obstacles in the obstacle set $Obs_i^{(t)}$. These robots regard the obstacles as inaccessible states in its transition system $TS_{i,k}^{(\tau)}$.

In addition, neighboring robots in communication ranges are required to exchange the information of respective obstacles and next states. The transition system is updated after the robot completes the current allocated task or detects new obstacles.

2.4 Human-Robot Trust Model of Task Allocation

2.4.1 DBN based Human-Robot Trust Model

The human-robot trust model is developed to improve the task allocation and motion planning of MRS to be similar as human decision-making. The trust evaluation of human in each robot is involved with robot performance, risk of occurrence of malfunctioning, and human cognitive workload. The robot performance evaluation is dependent on the amount of tasks completed and the success of obstacle collision avoidance. Risks are defined as the occurrence of malfunctioning situations such as robots are unable to move or perform tasks due to low battery level. The cognitive workload is related with the complexity of surrounding environment, such as the amount of surrounded obstacles of each robot, as well as the amount of robots that human has to supervise after a task reallocation. These are all the possible factors that may influence human's interaction with multi-robot task allocation and motion planning. Hence, it is favorable for the MRS to have a human-robot trust model to integrate all these factors in order to enable human-like decision-making for task allocation.

Besides the above influence from MRS, human and environment, we also consider system-wide trust based influence (either positive or negative) into human-robot trust evaluation regarding task reallocation. That is, robots assigned with an action in the task reallocation will be given an opposite trust evaluation with other robots that have a common implementable action but are not selected for this action. Such influence on MRS will construct a system-wide trust inter-relationship among robots. Finally, we will utilize a DBN based human-robot trust model to assist MRS in task (re)allocation and motion planning. Human will be intermittently inquired whether to allow a task reallocation with this model.

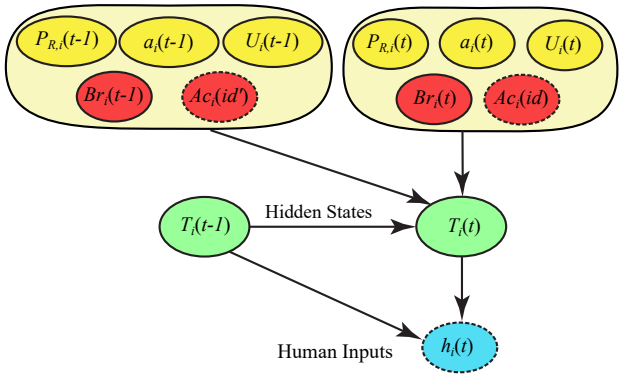


Figure 2.2: A dynamic Bayesian network (DBN) based model for dynamic, quantitative, and probabilistic trust estimates.

The DBN trust model[1] is shown in Figure 2.2. Based on the DBN model, we denote the belief update of trust $T_i(t)$ for robot r_i as

$$\begin{aligned} bel(T_i(t)) = &\text{Prob}(T_i(t)|P_{R,i}(1:t), a_i(1:t), U_i(1:t), \\ &Br_i(1:t), Ac_i(1:t), h_i(1:t), T_i(0)), \end{aligned} \tag{2.1}$$

where $\left[P_{R,i}, a_i, U_i, Br_i, Ac_i\right]^T$ are the impacting factors of the hidden trust state, denoted by Ω_i, and h_i is the observed evidence. To be more specific, $P_{R,i}(\cdot)$ is the accumulated performance evaluation of robot r_i, $a_i(\cdot) \in [0,1]$ is the safety coefficient of risk evaluation, $U_i(\cdot)$ is the human cognitive workload due to the obstacle-crowded environment, $Br_i(\cdot)$ is the human cognitive workload on supervising the MRS (e.g., monitoring multiple robots decided by the task allocation), $Ac_i(\cdot)$ is the extra positive or negative influence on the robots after human accepting of the task (re)allocation. $h_i(\cdot)$ is the human intervention on whether to allow a task reallocation for the MRS. Note that $Br_i(\cdot)$, $Ac_i(\cdot)$ and $h_i(\cdot)$ are only intermittently updated when a task reallocation occurs.

$$\Omega_i(t) = \begin{cases} \left[P_{R,i}(t), a_i(t), U_i(t), Br_i(t)\right]^T, \text{ if no reallocation} \\ \left[P_{R,i}(t), a_i(t), U_i(t), Br_i(t), Ac_i(id)\right]^T, \text{ otherwise} \end{cases}, \tag{2.2}$$

A forward algorithm is utilized by applying the principle of dynamic programming to avoid incurring

[1]The DBN human-robot model in this chapter deals with the human input with respect to each robot individually. In our future work, we will determine the human input based on the trust of all robots involved in task allocation.

exponential computation time due to the increase of t. Eqn. (2.1) can be calculated as

$$bel(T_i(t)) = \frac{\int \overline{bel}(T_i(t), T_i(t-1))\mathrm{d}T_i(t-1)}{\int\int \overline{bel}(T_i(t), T_i(t-1))\mathrm{d}T_i(t-1)\mathrm{d}T_i(t)}, \tag{2.3}$$

where

$$\begin{aligned}\overline{bel}(T_i(t), T_i(t-1)) = &\text{Prob}(h_i(t)|T_i(t), T_i(t-1))\cdot \\ &\text{Prob}(T_i(t)|T_i(t-1), \Omega_i(t), \Omega_i(t-1)) \cdot bel(T_i(t-1)),\end{aligned} \tag{2.4}$$

To obtain the belief update of each robot trust, Prob$(h_i(t)|T_i(t), T_i(t-1))$ and Prob$(T_i(t)|T_i(t-1), \Omega_i(t), \Omega_i(t-1))$ are respectively calculated with different distribution models as shown in the upcoming paragraphs.

The term Prob$(h_i(t)|T_i(t), T_i(t-1))$ is the conditional probability of human intervention given the current and prior trust, which can follow a similar sigmoid distribution as in [98]. Therefore, the conditional probability distribution (CPD) of human intervention based on trust can be modeled as follows

$$\text{Prob}(h_i(t) = 1|T_i(t), T_i(t-1)) = \frac{1}{1 + exp(-\alpha_1 T_i(t) + \alpha_2 T_i(t-1))}, \tag{2.5}$$

where α_1 and α_2 are positive weights and this CPD indicates higher willingness for human to allow a task reallocation when human-robot trust value is higher.

The CPD of human trust in robot r_i at time t can be constructed based on the previous trust value, robot performance, risk coefficient, human cognitive workload, and task allocation evaluation. It is expressed as a Gaussian distribution with mean value $\bar{T}_i(t)$ and variance $\rho_i(t)$,

$$\begin{aligned}&\text{Prob}(T_i(t)|T_i(t-1), \Omega_i(t), \Omega_i(t-1)) = \mathcal{N}(T_i(t); \bar{T}_i(t), \rho_i(t)), \\ &\bar{T}_i(t) = A \cdot \bar{T}_i(t-1) + B_1 \cdot a_i(t) \cdot P_{R,i}(t) - B_2 \cdot a_i(t-1) \cdot P_{R,i}(t- \\ &\quad 1) + C_1 \cdot U_i(t) - C_2 \cdot U_i(t-1) + D_1 \cdot Br_i(t) - D_2 \cdot Br_i(t \\ &\quad -1) + E_1 \cdot Ac_i(id) - E_2 \cdot Ac_i(id'),\end{aligned} \tag{2.6}$$

where $\bar{T}_i(t) \in (0, 1)$ represents the mean value of human trust in robot r_i at time t, and $\rho_i(t)$ reflects the variance in each individual's trust update. Each parameter is evaluated with a function of respective influence factors in task allocation and motion planning. The coefficients $A, B_1, B_2, C_1, C_2, D_1, D_2, E_1, E_2$

are determined by data collected from human subject tests [73].

In our scenario, the accumulated performance evaluation $P_{R,i}(t)$ is modeled as a function of rewards on robot for its completion of actions as well as the avoidance of obstacles,

$$P_{R,i}(t) = P_{R,i}(t-1) + w(r_i, \hat{e}_k, t) + \beta_i(t) \cdot w(o_i^{(t)}), \tag{2.7}$$

where $P_{R,i}(t-1)$ is the performance of robot at $t-1$, $w(r_i, \hat{e}_k, t) \in \{0, 1\}$ is the reward on robot r_i for completing an action $\hat{e}_k \in E_k$, $w(o_i^{(t)}) \in \{0, 1\}$ is the reward on robot for avoiding a detected obstacle $o_k^{(t)}$ at t, $\beta_i(t)$ is the number of obstacles the robot can avoid by re-planning path at t. The safety coefficient $a_i(t)$ is introduced to evaluate the potential of a single robot in completing all the capable actions in E_{r_i}. Here, the risk of malfunction refers to the possibility of low battery level of the robot, which may constrain it to perform more actions and thus need other robots to substitute it for the remaining uncompleted actions. The safety coefficient $a_i(t)$ is constructed as

$$a_i(t) = \begin{cases} 1, & r_i \text{ is in normal state} \\ \frac{1}{|E_{r_i}|}, & r_i \text{ is in low battery state} \end{cases}. \tag{2.8}$$

This implies the system tends to trust the robot to complete all its capable actions if it has enough electric capacity. On the other hand, if the battery level is low, the robot is assigned to at most complete the current allocated action.

The human cognitive workload is a result of interaction with the complex environment and multiple robots. For the environmental complexity resulted workload, it is constructed as

$$U_i(t) = 1 - \gamma(t)^{S_{o,i}(t)+1}, \tag{2.9}$$

where $S_{o,i}(t)$ is the number of obstacles within sensing range of robot, and $\gamma(t)$ is the utilization ratio [85, 91]. The human cognitive workload resulted from supervising robots always exists but is only updated after a task reallocation is implemented. It is estimated by the amount of robots that human can deal with as well as the actual activated robots in the supervision of a MRS. It is intermittently updated based on the idth task

reallocation as

$$Br_i(t) = \begin{cases} 1 - I_{act}(id)/I_{\max}, & \text{if } r_i \text{ is activated} \\ 1, & \text{otherwise} \end{cases}, \tag{2.10}$$

where $I_{act}(id)$ is the actual amount of activated robots in this task reallocation, and $I_{\max}$ is the maximal number of robots that human feels comfortable in supervising the MRS. Each robot will be updated the same workload with the system-wide trust theory if it is activated in the task (re)allocation, while $Br_i(t) = 1$ if the robot is not activated at all.

The extra positive or negative influence of task reallocation on each robot also works according to the system-wide trust theory after human accepts a task reallocation solution. Recall that each robot has a task reallocation path $s_{\psi,i} = \omega_{k_0}^{(0)} \cdots \omega_{k_\tau}^{(\tau)} \cdots \omega_{k_{\mathbf{T}}}^{(\mathbf{T})}$. Opposite influences can be enforced on the following situation: (1) the action in path $s_{\psi,i}$ is an implementable action of r_i, and (2) the implementable action of r_i is not selected in the current task allocation. As a result, an extra influence of idth task reallocation on each robot is constructed as

$$\begin{aligned} Ac_i(id) &= \sum_{\tau=0}^{\mathbf{T}} ac_i(\tau), \\ ac_i(\tau) &= \begin{cases} \mu/I, & \text{if } \omega_{k_\tau}^{(\tau)} = IA_{i,k}(\tau) \\ \overline{\mu}/I, & \text{if } \omega_{k_\tau}^{(\tau)} \neq IA_{i,k}(\tau) \end{cases}, \end{aligned} \tag{2.11}$$

where $ac_i(\tau)$ is the positive or negative influence of each action in task reallocation path on robot r_i, $\mu > 0$ and $\overline{\mu} < 0$ are the influence coefficients. If the action in task reallocation path is an implementable action of r_i, a positive influence will be added for this robot, which implies a trust increase of this robot in the current task reallocation; A negative influence will be associated to a robot by decreasing the trust of the robot if the implementable action of the robot is not selected in the current task allocation.

Remark 2. The network parameters for the DBN such as α_1, α_2 in Equation (2.5) can be learned by the well-known expectation maximization (EM) algorithm [59] off-line during the training session and hence will not affect the functionality of the system and the user experience in real-time operation. Besides, a separate and personalized trust model should be trained based on each user's experience since the model strongly depends on individual human intervention $h_i(\cdot)$ as well as the impacting factors $\left[P_{R,i}(\cdot), a_i(\cdot), U_i(\cdot), E_{H,i}(\cdot), Ac_i(\cdot)\right]$.

2.4.2 Human-Robot Trust based Real-time Interactive Task Reallocation and Motion Re-planning

Human trust in robot can be updated at each time step t or intermittently after a task reallocation of MRS. Consequently, the previous maximum trust encoded task allocation and motion planning solution need to be updated.

The reallocation request is triggered after an action is completed by a robot in the MRS and the human-robot trust is higher enough. Human works as a supervisor and will be inquired if he/she would like to have a task reallocation. The system will reallocate the actions to robots and re-plan the motion path of individual robot if human allows to have a task reallocation. The task reallocation will be implemented on these uncompleted actions, i.e. a task allocation automaton is re-synthesized with the unperformed actions constituted state set. As a result, a new maximum trust encoded path is generated from this automaton for the remaining task. In the mean time, the human trust in each activated robot will be changed with $Ac_i(id)$ from a system-wide trust perspective. However, if human refuses the task reallocation, the MRS will continue the previous task allocation and motion planning path. On the other hand, the human-robot trust model will be continuously updated for robot performance, safety coefficient, and human cognitive workload estimations while the robot is exploring in the work space. Algorithm 1 describes the complete process of human-robot trust model based interactive task allocation and motion planning. The process is iterated until all actions are completed.

Algorithm 1 Human-robot trust integrated task (re)allocation and SMP

```
Initial task allocation S_{ψ,0}
Update influence Ac_i(0), trust T_i(t) for r_i ∈ I
while Exist unperformed actions do
    if An action completed then
        Update P_{R,i}(t), a_i(t), U_i(t), Br_i(t), T_i(t)
        if Allow reallocation then
            Task reallocation S_{ψ,τ}
            Update Ac_i(id), T_i(t)
            Motion planning σ_i for all r_i
        end if
    end if
    Execute motions and actions
    Update P_{R,i}(t), a_i(t), U_i(t), T_i(t)
end while
```

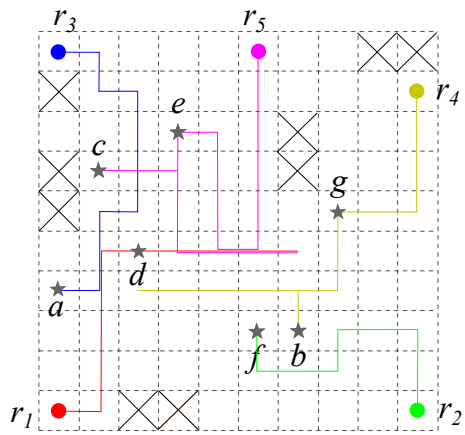


Figure 2.3: Abstracted workspace and paths of motion planning where "cross" marks the obstacle and "star" marks the allocated task for each robot.

2.5 Simulation

2.5.1 System Configuration and Task Specification

The workspace of MRS is abstracted as a 10×10 grid environment occupied with obstacles and task stations as shown in Figure 2.3. Each task station is associated with an action that needs a robot to perform. In the SMP, we assume the motion primitives of each robot are the abstracted from one grid to its adjacent four grids (north, east, south and west). A robot is also assumed to be unable to enter into grids that are partially or totally taken by obstacles. The linear quadratic regulator (LQR) is utilized to control the motion of robots between two grids.

3 parallel subtasks are to be allocated for MRS, each of the subtask is described by an automaton, see Figure 2.4. The languages of the 3 subtask automata are $L(G_1) = \{abc, acb\}$, $L(G_2) = \{de\}$, and $L(G_3) = \{f, gf\}$. A team of 5 robots $r_i,\ i = 1, \cdots, 5$ are assigned to perform the actions in the 3 parallel subtasks. Each robot is associated with its capability: $Er_1 = \{a, c, d\}$, $Er_2 = \{b, e, f\}$, $Er_3 = \{a, f, g\}$, $Er_4 = \{b, d, g\}$, and $Er_5 = \{c, e\}$. In addition, we assume omni-directional sensors and set as two-grid length. The communication radius is set as the same length.

2.5.2 Results

The final motion paths of robots are shown in Figure 2.3. The corresponding trust change of each robot is shown in Figure 2.5. The initial generated task allocation path is $\mathbb{S}_\psi = ((r_3, a), (r_1, d), (r_4, g))^{(0)}((r_4, b),$

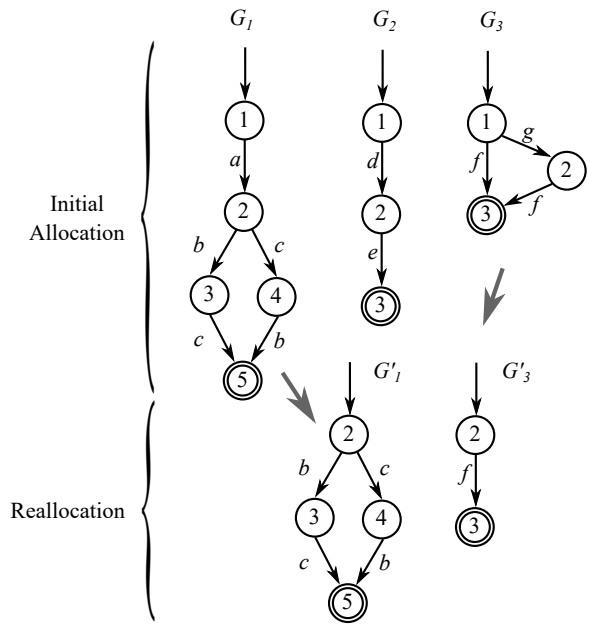
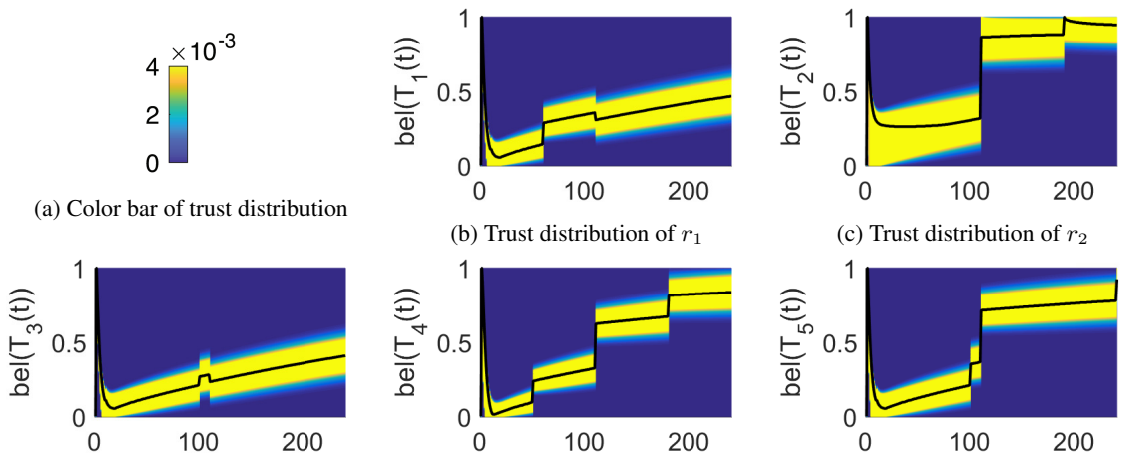


Figure 2.4: Parallel subtask automata G_1, G_2 and G_3.

$(r_5, e), (r_3, f))^{(1)}((r_1, c), \mathcal{E}, \mathcal{E})^{(2)}$. The task allocation mappings into each robot are $s_{\psi,1} = (r_1, d)^{(0)}(r_1, c)^{(2)}$, $s_{\psi,3} = (r_3, a)^{(0)}(r_3, f)^{(1)}$, $s_{\psi,4} = (r_4, g)^{(0)}(r_4, b)^{(1)}$ and $s_{\psi,5} = (r_5, e)^{(1)}$. As a result, the trust of each robot will be updated regarding the positive or negative influence of task allocation on each robot.

Actions (r_4, g), (r_1, d), (r_5, e), (r_3, a) are first sequentially completed with reference to the current task allocation solution and motion specification. Each robot verifies the completion state of actions that need to be performed before their current allocated actions in the subtask automaton. The results are demonstrated in Figure 2.3, where robot r_4, r_5 and r_3 first go to the neighboring positions of a, d, and g respectively (i.e. within the robot's sensing range) to detect the completion states before they perform the current allocated actions. The action a is almost completed by robot r_3 at the same time with (r_5, e). The rewards in performance evaluation of each robot are updated immediately after they completed each assigned actions, and the cognitive workload is also updated during the robot exploration. A reallocation inquiry is triggered after robot r_5 completes its only assigned action e. The reallocation is synthesized for the remaining subtasks $L(G_1) = \{c\}$ and $L(G_3) = \{f\}$, while action b is still performed by r_4 considering the robot's previous effort. The reallocation moment is at 110 time step, and trust change of each robot is demonstrated in the trust distribution in Figure 2.5. The maximum trust value of each robot before the reallocation are shown in Table 2.1.

The robots that enable the task allocation associated with the highest accumulated trust are selected



(a) Color bar of trust distribution

(b) Trust distribution of r_1

(c) Trust distribution of r_2

(d) Trust distribution of r_3

(e) Trust distribution of r_4

(f) Trust distribution of r_5

Figure 2.5: Evolution of trust distribution of each robot. Black curves are the maximum values of trust distribution.

Table 2.1: Maximum trust of each robot before task reallocation.

robot	1	2	3	4	5
Max trust	0.3566	0.3167	0.2818	0.3267	0.3666

to perform the remaining actions. According to the maximum trust value table above, robot r_2 and r_5 are selected respectively to perform action f and c rather than r_3 and r_1. The updated task allocation path is $\mathbb{S}'_\psi = ((r_3, a)(r_1, d)(r_4, g))^{(0)}((r_4, b), (r_5, e), (r_2, f))^{(1)}((r_5, c), \mathcal{E}, \mathcal{E})^{(2)}$. The newly assigned task allocation mapping into each robot are $s_{\psi,2} = (r_2, f)^{(1)}$, $s_{\psi,4} = (r_4, b)^{(1)}$ and $s_{\psi,5} = (r_5, c)^{(2)}$. Since human accepted the reallocation, the positive or negative trust influence on each robot get updated. Eventually, the MRS completes the remaining actions with this updated solution.

2.6 Conclusion

This chapter presents a human supervised task allocation and motion planning framework for MRS to perform multiple parallel subtasks in a human-like decision making manner. These subtasks are described by automata and conjuncted with MRS to synthesize a task allocation automaton. Transitions of task allocation automaton are associated with the estimations of robot performance and human cognitive workload. They are combined with a DBN human-robot trust model and a maximal trust encoded task allocation path

can be found. This path reflects the maximum trust of human in task assignment of MRS. Symbolic motion planning (SMP) is implemented for each robot after the task allocation. The task reallocation is triggered after an action being completed with human permission. The above process is demonstrated to be effective for MRS task allocation by a simulation with 5 robots and 3 parallel subtasks.

Chapter 3

Multi-Robot Task and Motion Planning under Temporal Logic Specifications

3.1 Introduction

In general, robot task planning generates a sequence of actions that robots need to implement to satisfy the task-performing goals; and motion planning identifies the trajectories associated with the task plan for the robots. In formal verification of robotic system, the high-level goals of robot task performing can usually be expressed with the temporal logic syntax and formulae. Model checking theory can obtain the task or motion plan satisfying the high-level goals in a discretized workspace [12, 81]. The model checking based task and motion planning accommodates robot constraints and environment complexity with provably correct solutions for temporal logic task. However, the model checking based task and motion planning usually has the state-space explosion problem, especially for multi-robot systems (MRS) [12]. Therefore, one of the challenges in task and motion planning under temporal logic task specifications is to develop computationally efficient frameworks for MRS.

Many relevant frameworks have been developed for MRS to deal with complex task specifications that have linear time property, such as linear temporal logic (LTL), computation tree logic (CTL), and regular expression (RE). These frameworks can be generalized into the centralized and decentralized regarding their computational framework.

The most straightforward task and motion planning strategy for MRS under temporal logic speci-

fications is the centralized computational frameworks [76, 30, 42, 43, 89, 44]. The centralized framework takes all the robots as a whole during the task planing. It composes all the robot transition systems of the MRS and constructs a concurrent transition system for the MRS. Then, the framework directly synchronizes the converted automaton of an LTL task specification with the concurrent transition system of the MRS. The centralized framework is restricted in scalability and only suitable for medium-sized robot teams or specific types of tasks.

The decentralized framework synchronizes each robot transition system of the MRS with the robot's corresponding capable task pieces in the task specification. The framework does not require the composition of all the robots' transition system during the formal verification, which reduces the computation significantly. Instead, the framework requires that either the task pieces are well assigned to each robot or the task specification can be decomposed into task pieces with respect to each robot. There are different decomposition strategies developed in the recent years [25, 20, 19, 88, 86, 87, 34, 35, 77, 78]. However, each of the strategies is only durable for a specific type of task specification.

This chapter develops a decentralized top-down task and motion planning framework for MRS, which assigns robots to atomic tasks of task specification, reduces the computation of the robot task and motion planning in a decentralized fashion, and enables these robots to work in parallel and concurrently with a high level of parallelism. Fig. 3.1 illustrates the proposed automaton based parallel task decomposition, assignment, and motion planning framework for MRS. The framework first obtains a global task automaton of an LTL described task specification (see step 1). The global task automaton is then parallel decomposed into a set of subtask automata utilizing an automaton based iterative parallel decomposition algorithm (step 2). Each generated decomposition component, i.e. subtask automaton, is assigned to a subgroup of robots that are capable of satisfying the event set of the subtask automaton. These subtask automata and the robot capability transition systems of the assigned robots are combined to synthesize a set of SPAs for task planning (step 3). Each SPA can generate the lowest cost task plan by further considering the task performing cost[1] (step 4). These robots can get their corresponding initial task plans (step 5). Thus, the hybrid local motion planner can generate motion trajectories for each robot based on their task plans (step 6 & 7). A dynamic task redecomposition and replanning process (steps 2 - 8) is triggered intermittently for neighboring robots that can coordinate with each other. This process continues until the MRS accomplishes all the task plans.

The main theoretical contributions of the chapter are summarized as follows:

[1]The cost depends on the task objective and can be in different forms, such as the estimated time or energy consumption to complete a task

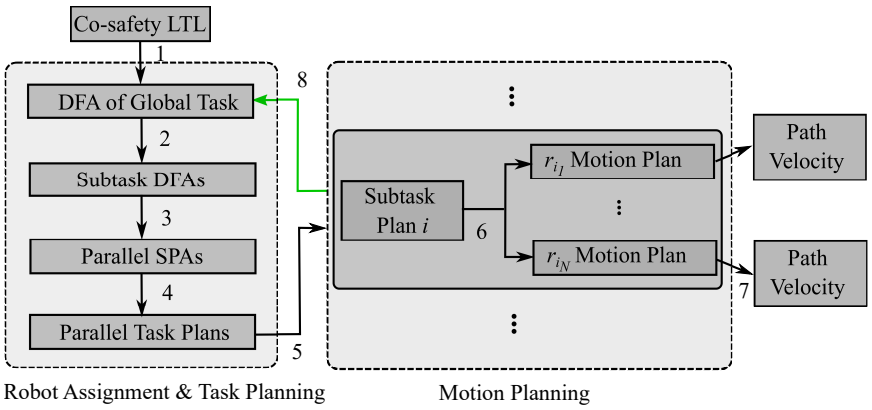


Figure 3.1: Flowchart of the proposed automaton based task decomposition and planning framework for the MRS.

1. A global task specification is parallel decomposed into a unique set of parallel executable subtask specifications. The decomposition process considers the generation of variable event sets that can make the task specification parallel decomposable.

2. Assuming robots with overlapping capabilities, the robot assignment to each subtask specification is determined by considering the level of parallelism of multi-robot task and motion planning. The parallel decomposition based task assignment enables the MRS to satisfy the atomic tasks of task specification in parallel. It is more efficient in executing the task plan than the centralized task and motion planning approach, which assigns robots to perform the atomic tasks consecutively.

3. A set of subtask planning automata (SPA) is synthesized from the subtask automata and corresponding robot state transition systems after the parallel decomposition based robot assignment. These SPA can guarantee viable switches of robots between different parallel subtask specifications. The optimal task planning solution can be obtained from these SPA by taking into account the cost of task execution. The parallel decomposition based SPA have much smaller size of state spaces and require less computation to obtain the task planning solution compared with the centralized framework.

The remaining parts of the chapter are as follows. Chapter 3.2 provides the preliminaries and the problem setup. Chapter 3.3 presents the up-to-date related work of multi-robot task and motion planning under temporal logic task. Chapter 3.4 details the top-down automaton-based task decomposition algorithm. Chapter 3.5 and 3.6 explains the robot assignment and task planning process with the decomposition results as well as the parallel and concurrent execution of the task planning results. Chapter 3.7 presents a set of MRS experiment and simulations to demonstrate the viability, scalability, and computational efficiency of the

overall framework. Chapter 3.8 summarizes the work and results in this chapter.

3.2 Preliminaries and Problem Setup

3.2.1 Preliminaries

Definition 6 (Finite Automaton [10]). A finite automaton, denoted by G, is a tuple $G := (X, E, f, x_0, X_F)$, where X is the set of states; E is the finite set of events associated with G; $f : X \times E \to X$ is the transition function: $f(x, e) = x'$ means that there is a transition labeled by event e from state x to x'; x_0 is the initial state; and $X_F \subseteq X$ is the set of final states. A deterministic finite automaton (DFA) requires each transition to satisfy$\left|f(x, e)\right| \leq 1$. In comparison, a nondeterministic finite automaton (NFA) has a transition$\left|f(x, e)\right| > 1$.

An event $e \in E$ is a "single event" if it only contains an atomic task and needs one robot to complete (e.g., a robot goes to a goal, or picks up an object), or a "cooperative event" if it is a conjunction of atomic tasks and needs multiple robots to be completed simultaneously (e.g., one robot holds the door and another robot goes through the door).

An automaton G will be used to describe the temporal property of the atomic tasks in the event set. A word of G, denoted by $\rho := e^{(1)} \cdots e^{(\tau)} \cdots e^{(\mathrm{T})}$, is a sequence of events satisfying $f(x_0, \rho) \in X_F$, where $e^{(\tau)} \in E$. The induced states of word ρ in G, i.e., $x^{(0)} \cdots x^{(\tau)} \cdots x^{(\mathrm{T})}$, is the corresponding accepted run of word ρ. The language generated by an automaton G is $L(G) := \{\rho \in E^* | f(x_0, \rho) \in X_F\}$, which can be described concisely with regular expression operations, such as the concatenation $(e_1 \cdot e_2)$, union $(e_1 + e_2)$, and Kleene star $({e_1}^*)$, with $e_1,\ e_2 \in E$.

Definition 7 (Projection and Inverse Projection of Automaton [10]). Denote the projection of word $\rho \in E^*$ to an event set E_b as $P_b(\rho)$. The projection operation can be defined with the following rules: (1) $P_b(\varepsilon) := \varepsilon$, ε is the empty event or word, (2) $P_b(e) := e$ if $e \in E_b$, $P_b(e) := \varepsilon$ if $e \in E \setminus E_b$, (3) $P_b(\rho e) := P_b(\rho)P_b(e)$ for $\rho \in E^*, e \in E$. The projection of automaton G with the event set E into a smaller event set E_b, i.e., $P_b(G)$, $E_b \subseteq E$, can be applied over languages of automaton: $P_b(L(G)) := \{\rho_b \in {E_b}^* | \rho \in L(G),\ \rho_b = P_b(\rho)\}$; Conversely, the inverse projection of automaton G_b with a smaller event set E_b into the event set $E \supset E_b$ can be described by $P^{-1}(L(G_b)) := \{\rho \in E^* | P_b(\rho) \in L(G_b)\}$.

Definition 8 (Parallel Composition of Automata [10]). The parallel composition of $G_1 := (X_1, E_1, f_1, x_{0,1}, X_{F,1})$ and $G_2 := (X_2, E_2, f_2, x_{0,2}, X_{F,2})$ is the automaton $G_1 \parallel G_2 := (X_1 \times X_2, E_1 \cup E_2, f_{\parallel}, (x_{0,1}, x_{0,2}), X_{F,1} \times$

$X_{F,2}$), where $f_{\|}((x_1, x_2), e) :=$

$$\begin{cases} (f_1(x_1, e), f_2(x_2, e)) & \text{if } \exists f_1(x_1, e) \wedge \exists f_2(x_2, e), e \in E_1 \cap E_2 \\ (f_1(x_1, e), x_2) & \text{if } \exists f_1(x_1, e), e \in E_1, e \notin E_2 \\ (x_1, f_2(x_2, e)) & \text{if } \exists f_2(x_2, e), e \in E_2, e \notin E_1 \\ \text{undefined} & \text{otherwise} \end{cases}$$

The language resulting from a parallel composition can be characterized as: $L(G_1 \parallel G_2) := P^{-1}(L(G_1)) \cap P^{-1}(L(G_2))$.

Definition 9 (**Automaton Parallel Decomposition [45]).** An automaton G with the event set E is said to be *strictly decomposable* with respect to a set of P_i if $\|_{i=1}^{I} P_i(G) \cong G$. Here, P_i is a projection from G into local event sets E_i, $i = 1, \cdots, I$, $E = \cup_{i=1}^{I} E_i$, and $\|_{i=1}^{I} P_i(G) = (((P_1(G) \parallel P_2(G)) \parallel P_3(G)) \parallel \ldots) \parallel P_I(G)$. In addition, a *strictly decomposable* G only allows the cooperative events in the initial or last transition of G.

The bisimulation relationship ($\cong$) here can be reduced to check the language equivalence between the parallel compositions of G_is and the automaton G, i.e., $L(\|_{i=1}^{I} P_i(G)) \equiv L(G)$ if G is a DFA. Given an atomic task set $\mathbb{AP}$ including all the atomic tasks π that an MRS needs to satisfy, one can then use a DFA G to describe the desired task specification of an MRS, where the automaton event set is $E \subseteq 2^{\mathbb{AP}}$. The following example illustrates the concept of automaton based parallel decomposition of a task specification for an MRS.

Example 1. Given two tasks A and B (e.g., go-to-goal tasks), Fig. 3.2 shows two DFAs based on the two tasks, where the atomic task π_A denotes "a robot achieves task A" and π_B describes "a robot achieves task B". Then, the DFA G_{φ_1} in Fig. 3.2(a) describes that "robots in the MRS achieve task B directly; or they first achieve task A and next achieve task B". The DFA G_{φ_2} in Fig. 3.2(b) says "the MRS achieves tasks A and B in any sequence". The language of the two automata are $L(G_{\varphi_1}) = \{\pi_A\pi_B,\ \pi_B\}$, and $L(G_{\varphi_2}) = \{\pi_A\pi_B,\ \pi_B\pi_A\}$.

For the DFA G_{φ_2} in Fig. 3.2(b), the projection of the automaton into two event subsets $E_1 = \{\pi_A\}$ and $E_2 = \{\pi_B\}$ are $L(P_1(G_{\varphi_2})) = \{\rho_1 \mid \rho_1 = P_1(\rho),\ \rho \in L(G_{\varphi_2})\} = \{\pi_A\}$ and $L(P_2(G_{\varphi_2})) = \{\rho_2 \mid \rho_2 = P_2(\rho),\ \rho \in L(G_{\varphi_2})\} = \{\pi_B\}$, respectively. The two projections of the automaton are shown as $G_1 =$

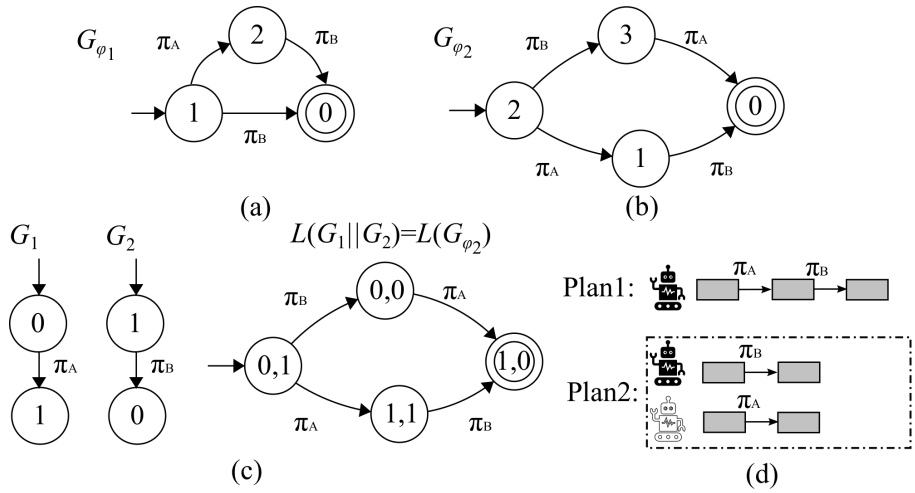


Figure 3.2: (a) The DFA of the task specification "robots in the MRS achieve task B directly; or they first achieve task A and next achieve task B", (b) the DFA of the task specification "the MRS achieves tasks A and B in any sequence", (c) parallel decomposition of DFA G_{φ_2}, and (d) possible satisfaction processes of the two DFAs. DFA G_{φ_1} can be satisfied by "Plan 1"; DFA G_{φ_2} can be satisfied by either "Plan 1" or "Plan 2".

$P_1(G_{\varphi_2})$ and $G_2 = P_2(G_{\varphi_2})$ in Fig. 3.2(c) left. According to Def. 8, the state transitions of parallel composition of the two projections, i.e., $G_1 \| G_2$, are $\{f((0,1),\pi_A) = (1,1), f((0,1),\pi_B) = (0,0), f((1,1),\pi_B) = (1,0), f((0,0),\pi_A) = (1,0)\}$ (see Fig. 3.2(c) right). It satisfies $L(P_1(G_{\varphi_2}) \| P_2(G_{\varphi_2})) \equiv L(G_{\varphi_2})$ according to Def. 9. Therefore, G_{φ_2} can be parallel decomposed into two automata whose languages are $L(G_1) = L(P_1(G_{\varphi_2}))$ and $L(G_2) = L(P_2(G_{\varphi_2}))$. As a result, the words $\rho_1 = \pi_A$ and $\rho_2 = \pi_B$, i.e., the two go-to-goal tasks, can be satisfied consecutively by one robot (Plan 1 in Fig. 3.2(d)) or in parallel by two independent robots (Plan 2 in Fig. 3.2(d)).

The projections of DFA G_{φ_1} into two event subsets $E_1 = \{\pi_A\}$ and $E_2 = \{\pi_B\}$ are the same with that of G_{φ_2}, i.e., $L(P_1(G_{\varphi_1})) = \{\pi_A\}$ and $L(P_2(G_{\varphi_1})) = \{\pi_B\}$. However, the state transitions of parallel composition of the two projections do not satisfy the language equivalence relation with G_{φ_1}, i.e., $L(P_1(G_{\varphi_1}) \| P_2(G_{\varphi_1})) \not\equiv L(G_{\varphi_1})$, according to Def. 9. Therefore, the task specification DFA G_{φ_1} can only be achieved sequentially as "Plan 1" but not in parallel as "Plan 2" because it does not satisfy parallel decomposability. •

One can also describe the linear time property of an MRS in an intuitive and mathematically precise expression, i.e., LTL.

Definition 10 (LTL Specification [10]). An LTL formula φ is formed from atomic propositions, propositional logic operators, and temporal operators according to the grammar $\varphi ::= \text{true} \mid \pi \mid \neg\varphi \mid \varphi_1 \vee \varphi_2 \mid$

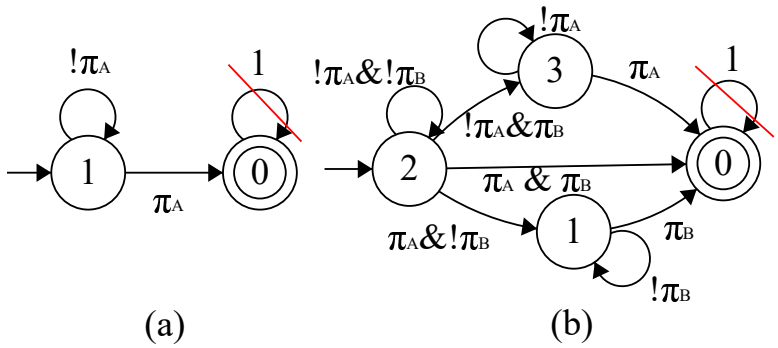


Figure 3.3: The converted automata DFA of LTL task specifications φ_1 and φ_2. Note the DFA here is obtained by removing the transitions that infinitely visit the final states of the converted DBA from software "spot".

$\bigcirc\varphi \mid \varphi_1\ U\ \varphi_2$, where π is an atomic proposition, $\neg$ (negation) and $\vee$ (disjunction) are Boolean operators, and $\bigcirc$ (next) and U (until) are temporal operators. More expressive operators can be constructed from the above operators, such as, conjunction: $\varphi_1 \wedge \varphi_2 = \neg(\neg\varphi_1 \vee \neg\varphi_2)$, eventually: $\Diamond\varphi = \text{true}\ U\ \varphi$, and always: $\Box\varphi = \neg\Diamond\neg\varphi$.

A co-safety LTL formula, which is a subset of LTL formulae only occurring the $\bigcirc$, $\Diamond$ and U temporal operators, can be converted to a deterministic finite automaton[2](DFA). The chapter deals with the finite words of a task specification in the MRS parallel satisfaction process and focuses on the co-safety LTL.

Example 2. Based on the two atomic tasks in Example 1, one can describe different co-safety LTL specifications. Here, let us consider two example LTL specifications: $\varphi_1 = \Diamond\pi_A$: "finally a robot in the MRS achieves the go-to-goal task A" and $\varphi_2 = \Diamond\pi_A \wedge \Diamond\pi_B$: "finally the MRS achieve the go-to-goal tasks A and B in any sequence". Each LTL formula can be converted to a DFA as shown in Fig. 3.3. •

3.2.2 Problem Setup

Assume a group of heterogeneous robots $\mathcal{R}$ has a set of atomic tasks $\mathbb{AP}$ to accomplish, but their assignment relation is unknown. The atomic tasks in $\mathbb{AP}$ are subject to the linear time property and formulate an LTL task specification φ according to Def. 10. The capability of each robot can be described by a finite labeled capability transition system TE_n according to the definition of the finite transition system in [10]. The process of checking if there are viable transitions in the robot capability transition system to satisfy the task specification is a verification problem.

[2]A co-safety LTL specification can be converted into a DFA by first using open source toolboxes "spot": `https://spot.lrde.epita.fr/app/` to derive a deterministic Büchi automaton (DBA) and further removing the suffix satisfying LTL for infinite times.

Definition 11 (Finite Labeled Capability Transition System). Given an indexed robot $r_n \in \mathcal{R}$, the task execution capability of robot r_n in a specific environment can be abstracted as the robot capability transition system

$$TE_n \coloneqq \left(S_n, A_n, \delta_n, s_{0,n}, AP_n, \mathcal{L}_n, W_n\right), \tag{3.1}$$

where the state set S_n contains all the abstracted task performing states of robot r_n; A_n is the action set of the robot; $\delta_n(s, \alpha) = s'$ describes the transition of robot from task performing state $s \in S_n$ to $s' \in S_n$ by executing an action $\alpha \in A_n$; $s_{0,n} \in S_n$ is the initial task performing state; AP_n is the set of atomic propositions related with task performing states; $\mathcal{L}_n : S_n \to 2^{AP_n}$ labels the robot states with the propositions derived from AP_n; and $W_n : S_n \times A_n \to \mathbb{R}^+$ weighs the cost of robot r_n at state $s_n \in S_n$ with action $\alpha \in A_n$.

The label relation $\mathcal{L}_n$ from state set S_n to proposition set 2^{AP_n} represents the tasks that each robot r_n is capable of performing. Each robot r_n achieves the atomic task $\pi \in \mathbb{AP}$ if it has a task performing state $s_n \in S_n$ satisfies π, i.e., $\mathcal{L}_n(s_n) \ni \pi$. Here, each abstracted task performing state describes that a robot performs a general task, such as "picking up an object" or "unloading objects".

Definition 12 (Parallel Composition of Capability Transition Systems [10]). Given two finite labeled capability transition systems of robots $TE_1 \coloneqq \left(S_1, A_1, \delta_1, s_{0,1}, AP_1, \mathcal{L}_1, W_1\right)$ and $TE_2 \coloneqq \left(S_2, A_2, \delta_2, s_{0,2}, AP_2, \mathcal{L}_2, W_2\right)$ according to Def. 11, their parallel composition is a transition system $TE_1 \| TE_2 \coloneqq \left(S_1 \times S_2, A_1 \cup A_2, \delta_{1,2}, \langle s_{0,1}, s_{0,2}\rangle, AP_1 \cup AP_2, \mathcal{L}_{1,2}, W_1 \times W_2\right)$, where the transition relation $\delta_{1,2}$ can be specified as $\delta_{1,2}(\langle s_1, s_2\rangle, \alpha) \coloneqq \langle s_1, s_2\rangle' \coloneqq$

$$\begin{cases} \langle s_1', s_2\rangle & \text{if } \alpha \in A_1 \setminus A^c \wedge \delta_1(s_1, \alpha) = s_1', \\ \langle s_1, s_2'\rangle & \text{if } \alpha \in A_2 \setminus A^c \wedge \delta_2(s_2, \alpha) = s_2', \\ \langle s_1', s_2'\rangle & \text{if } \alpha \in A^c \wedge \delta_n(s_n, \alpha) = s_n',\ n = 1, 2, \end{cases}$$

where $A^c \coloneqq A_1 \cap A_2$ denotes the set of cooperative actions that need the simultaneous transitions of robots r_1 and r_2's capability states; and the labeled relation $\mathcal{L}_{1,2} : S_1 \times S_2 \to 2^{AP_1 \cup AP_2}$ labels each state $\langle s_1, s_2\rangle$ through $\mathcal{L}_{1,2}(\langle s_1, s_2\rangle) \coloneqq \mathcal{L}_1(s_1) \cup \mathcal{L}_2(s_2)$.

Now assume the MRS has N robots, the parallel composition of capability transition systems $TE_n,\ n = 1, \cdots, N$ can be $\|_{n=1}^{N} TE_n \coloneqq ((TE_1 \| TE_2) \| \cdots) \| TE_N$ according to Def. 12. The paral-

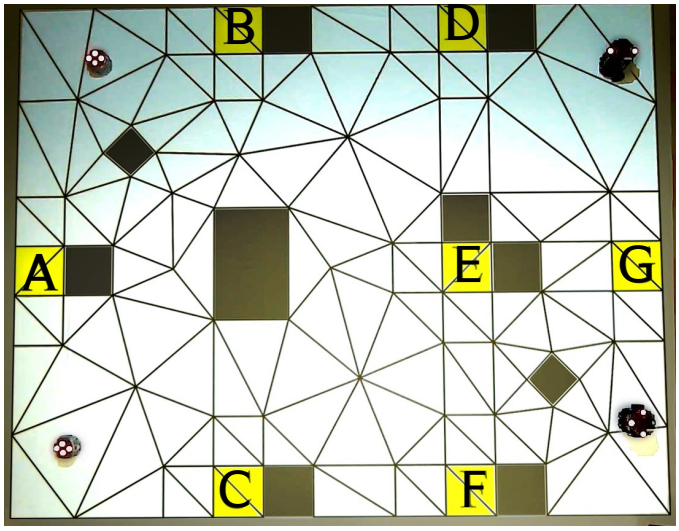


Figure 3.4: Triangulated 2D workspace. The obstacle polygons are over-estimation of the actual size of the obstacles.

lel composition $\|_{n=1}^{N} TE_n$ describes all the states of the MRS and their viable transitions. A sequence of task performing states of $\|_{n=1}^{N} TE_n$ is a path $\langle s_1^{(0)}, \cdots, s_N^{(0)} \rangle \cdots \langle s_1^{(\tau)}, \cdots, s_N^{(\tau)} \rangle \cdots \langle s_1^{(\mathrm{T})}, \cdots, s_N^{(\mathrm{T})} \rangle$. Denote the state transition operator as $\delta_{\mathcal{R}}$ and $\delta_{\mathcal{R}}(\langle s_1^{(\tau)}, \cdots, s_N^{(\tau)} \rangle, \alpha) = \langle s_1^{(\tau+1)}, \cdots, s_N^{(\tau+1)} \rangle$. Denote the label relation of $\|_{n=1}^{N} TE_n$ as $\mathcal{L}_{\mathcal{R}}$, the path is said to satisfy the task specification φ if the corresponding trace $\mathcal{L}_{\mathcal{R}}(\langle s_1^{(0)}, \cdots, s_N^{(0)} \rangle) \cdots \mathcal{L}_{\mathcal{R}}(\langle s_1^{(\tau)}, \cdots, s_N^{(\tau)} \rangle) \cdots \mathcal{L}_{\mathcal{R}}(\langle s_1^{(\mathrm{T})}, \cdots, s_N^{(\mathrm{T})} \rangle)$ has an accepted run of states in the converted automaton G_φ of φ. In other words, the MRS can work sequentially according to the path to satisfy the task specification. The path is also called the task plan.

The task plan requires the corresponding robots r_n to travel to the designated locations of their assigned atomic tasks. The hybrid local motion planner in [56] is utilized to generate the trajectory between each two designated locations of interest. The overall process is summarized as follows. A finite labeled motion transition system TM_n is first introduced based on the discrete workspace (e.g., see Fig. 3.4) for any robot r_n.

Definition 13 (Finite Labeled Motion Transition System). Given an indexed robot $r_n \in \mathcal{R}$, with a state set C_n containing all cells of a discretized workspace, the robot motion can be constructed as a finite labeled transition system

$$TM_n := \left(C_n, \delta_n^c, c_{0,n}, AP_n^c, \mathcal{L}_n^c, W_n^c\right), \tag{3.2}$$

where $\delta_n^c(c) = c'$ describes the robot's motion transition from cell $c \in C_n$ to $c' \in C_n$; $c_{0,n} \in C_n$ is the

initial cell that the robot is located at; AP_n^c is the set of atomic propositions related with robot motion states; $\mathcal{L}_n^c : C_n \to 2^{AP_n^c}$ labels each cell regarding whether it satisfies the propositions derived from AP_n^c; and $W_n^c : \delta_n^c \to \mathbb{R}^+$ is the cost set of each motion. A path of TM_n is a sequence of states $\sigma_n := c^{(0)}c^{(1)} \cdots c^{(t)} \cdots$, where $c^{(t)} \in C_n$ is the state at step $t \geq 0$ and $\delta_n^c(c^{(t)}) = c^{(t+1)}$.

Each designated location is labeled with propositions in $2^{AP_n^c}$. These propositions can be utilized to encode safety or liveness specifications, such as avoidance or reachability of a cell. A model checker, such as NuSMV, can generate a counter-example that violates the negation of safety or liveness specifications in TM_n. Then, each robot can take the counter-example as a discrete path σ_n to the designated location of $s_n^{(\tau)}$ while satisfying the reachability of the desired cell and collision avoidance with obstacles in a cell. Finally, a hybrid controller generates the trajectory that steers each robot from one discrete cell to an adjacent cell in the discrete path σ_n.

Generally, the MRS can accomplish the atomic tasks of task specification according to the centralized strategy using $\|_{n=1}^N TE_n := ((TE_1 \| TE_2) \| \cdots) \| TE_N$. Nevertheless, according to Def. 9 and Example 1, the task specification φ can be achieved in parallel by the MRS if φ satisfies the parallel decomposability. The MRS can be divided into different subgroups of robots and work in parallel for each decomposed subtask specification. The level of parallelism of MRS can be quantified with the number of independently and simultaneously achieved subtask automata. Parallelism will reduce not only the complexity of the multi-robot multi-task process but also the computation of generating the task plan. However, the MRS may lack enough robots to work in parallel for these multiple subtask specifications. Then, it may result in concurrent task performing situations that a subgroup of robots has to work for the composition of several subtask specifications. Besides, different robots in the MRS can have overlapping capabilities and are capable of performing the same tasks; and the tasks that all the provided robots are capable of performing can cover the given task specification, i.e., $2^{\cup_{n=1}^N AP_n} \supseteq 2^{\mathbb{AP}}$. In such cases, robot assignments for atomic tasks will lead to different levels of parallelism for the MRS. Therefore, the MRS needs a generalizable robot assignment and task planning framework that can guarantee the high level of parallelism. Under the above multi-robot multi-tasking configuration, the problem of interest can be formulated as follows.

Problem of Interest 2. Given a set of heterogeneous robots $\mathcal{R}$ without any preassigned tasks and a task specification φ composed with a set of atomic propositions $\mathbb{AP}$, design a decentralized top-down task and motion planning framework for the MRS such that

1. the task specification φ can be decomposed into smaller tasks, and

2. the robots $\mathcal{R}$ can accomplish the tasks in parallel and concurrently without violating the task specification φ.

3.3 Related Work

Although centralized framework of MRS task and motion planning has the high computation, several sampling-based algorithm has been developed to reduce the computation. Kantaros and Zavlanos seek to improve the computation speed of the centralized framework by incrementally building "trees" that can approximate the state-space and transitions of the synchronization results, i.e., the product automaton, and using a sampling-based approach to search for the optimal solution [42, 43]. Other relevant works for improving the computational efficiency of obtaining the optimal solution using sampling-based searching strategies can be found in [89, 44]. In this chapter, it instead focus on improving the computational efficiency of task and motion planning for MRS under temporal logic specifications by reducing the state-space of the generated product automaton, which can be achieved in a decentralized framework. The above sampling-based searching strategy may be applied over the proposed decentralized framework to further improve the computational efficiency, which is however outside the scope.

Some extant decentralized computational frameworks first synchronize each robot's transition system with the converted automaton of its assigned task; then guarantee the combination of all the synchronization results to satisfy the desired global property of MRS. A few computationally efficient decentralized task and motion planning frameworks have been developed to deal with the LTL or RE task specifications [25, 20, 19, 88, 86, 87, 34, 35, 77, 78].

In a top-down framework, the global task specification is decomposed into subtasks under some prescribed conditions for each robot to accomplish, such as the trace-closedness property in [25, 20, 19, 88] and the parallel decomposability property in [45, 23]. The satisfaction of global task specification is ensured by each robot accomplishing its subtask. These top-down frameworks have a predefined and fixed assignment relation between robots and subtasks such that these robots can satisfy the desired task specifications in a decentralized form. In comparison, bottom-up strategies design local control rules for each robot and coordinate among robots so that the prescribed global property can be collaboratively guaranteed [86, 87, 34, 35, 77, 78]. These bottom-up task planning frameworks are developed for specifications featured by loosely coupled subtasks. The complexity of connections between subtasks affects the inter-dependencies among robots.

In the decentralized computational frameworks, by default, the task specifications are achieved in a sequential manner. For example, in the LTL navigation tasks of ([20, 78]), the frameworks generally synthesize a single chain of optimal task plan satisfying the task specification. The MRS accomplishes the mission at each step of the task plan by assigning a single robot or a collaborated subgroup of robots to reach the designated locations. However, by having multiple robots working simultaneously and independently, the tasks can be completed faster [32]. A process executing multiple tasks on multiple robots simultaneously (in parallel) is called *parallelism* [38]. In the above navigation example, the *parallelism* can be that the task plan is split into multiple chains of independent task sub-plans without shared robots. The associated robots of each task sub-plan navigate to their respective destinations independently from other sub-plans. Parallelism is usually accompanied by *concurrency*, where a robot or a group of robots accomplishes multiple tasks through interleaved executions [38]. In the above navigation example, the *concurrency* can be that there are shared robots preventing the task plan from being split into the independent task sub-plans. The associated robots of each task sub-plan can not navigate to their respective destinations independently from other sub-plans, but can accomplish all the missions of sub-plans in an interleaved manner. Nevertheless, very few works on task and motion planning for MRS under temporal logic specifications consider this problem.

Furthermore, it is critical to consider the assignment of robots to task specifications in the task planning framework. Different robot assignments can lead to various decentralized task planning solutions with the corresponding computational complexity. In addition, the robot-task assignments that lead to the higher level of parallelism and concurrency will significantly reduce the coordination difficulty and speed up the multi-robot multi-task process. However, existing top-down frameworks give a predefined assignment relation between each robot and atomic task, i.e., the smallest task component in a task specification, while the majority of the bottom-up frameworks assign each robot to a specific local task specification. These decentralized frameworks can achieve low computational complexity if and only if the MRS works under their predefined robot assignment. They fail to consider the situation where the MRS can have different robot-task assignment relations if robots have partially overlapping capabilities and are capable of accomplishing the same subset of atomic tasks [33, 104, 83, 47]. Only recent works [78, 11] start to consider the overlapping capability and allow robots to switch between different subtasks in satisfying the task specification, which however still begin with a predetermined assignment relationship between robots and task specifications.

3.4 Parallel Decomposition of Task Specification

Assume a global task automaton $G_g := \big(X_g, E_g, f_g, x_{0,g}, X_{F,g}\big)$ according to Def. 6 and the event set is $E_g \subseteq 2^{\mathbb{AP}}$. The MRS can work in parallel for a global task specification, if the automaton G_g of the task specification satisfies the parallel decomposability, i.e., $L(G_g) \equiv L(P_1(G_g)\| \cdots P_i(G_g)\| \cdots P_I(G_g))$ with event sets $\{E_i, i := 1, \cdots, I\}$, and $\bigcup_{i=1}^{I} E_i = E_g$ according to Def. 9. Additionally, it is possible for an automaton G_g to generate a set of parallel decomposition components $P_i(G_g)$ if it contains a subautomaton that strictly satisfies the parallel decomposability, i.e., $L(G_g) \supseteq L(\|_{i=1}^{I} P_i(G_g))$. Therefore, one can enhance the generality of the parallel decomposition algorithm by finding the subset decomposable language from $L(G_g)$.

Definition 14 (Event-equivalent Words Composed Automaton). Given an automaton G, the words $\rho \in L(G)$ and $\rho' \in L(G)$ are called the event-equivalent words of the automaton G iff (1) the event sets composing ρ and ρ' are the same, and (2) the occurrence frequencies of every event in ρ and ρ' are equal. If all the words $\rho \in L(G)$ are event-equivalent with the event set $\check{E}$, denote the automaton as $\check{G}$.

Let $\{\check{G}_l,\ l \in \mathbb{Z}\}$ be a collection of event-equivalent words composed subautomata of G_g, where all the words of each subautomaton $\check{G}_l$ are event-equivalent with respect to event set $\check{E}_l$ and $L(\check{G}_l) \subseteq L(G_g)$. The automaton G_g is also parallel decomposable if it has a subautomaton $\check{G}_l$ whose language satisfies the parallel decomposability.

Example 3. A global task DFA G_g is shown in Fig. 3.5 (a). The event-equivalent words composed two automata can be extracted with event sets $\check{E}_1 = \{\pi_e, \pi_a, \pi_b, \pi_c\}$ and $\check{E}_2 = \{\pi_e, \pi_a, \pi_b, \pi_d\}$. The paths of each subautomaton have the same event set. The details are shown in Fig. 3.5 (b) and (c). •

To simplify the notation, ignore the subscript of $\check{G}_l$ and $\check{E}_l$ in the rest of this section. The iterative parallel decomposition can be summarized with the below process. Denote the single event set of automaton $\check{G}$ as $\check{E}^s$, the cooperative event set as $\check{E}^c$, and $\check{E}^c \cup \check{E}^s = \check{E}$. The power set of all the single events is $2^{\check{E}^s}$, which collects all the subsets of $\check{E}^s$. Denote single event set $E_j^s \in 2^{\check{E}^s},\ j = 1, \cdots, J$, as an arbitrary subset of $\check{E}^s$. Then, subset $\check{E}^s \setminus E_j^s$ collects the remaining single events in $\check{E}^s$. One can divide the event set $\check{E}$ of automaton $\check{G}$ into a pair of events $E_j := E_j^s \cup \check{E}^c, E_{\bar{j}} := (\check{E}^s \setminus E_j^s) \cup \check{E}^c$, and the event set pair $(E_j, E_{\bar{j}})$ satisfies $E_j \cup E_{\bar{j}} = \check{E}$. Then, all the event set pairs can be described as

$$\mathbb{E} := \{(E_j, E_{\bar{j}}) | E_j^s \in 2^{\check{E}^s}, E_j := E_j^s \cup \check{E}^c, E_{\bar{j}} := (\check{E}^s \setminus E_j^s) \cup \check{E}^c\}. \tag{3.3}$$

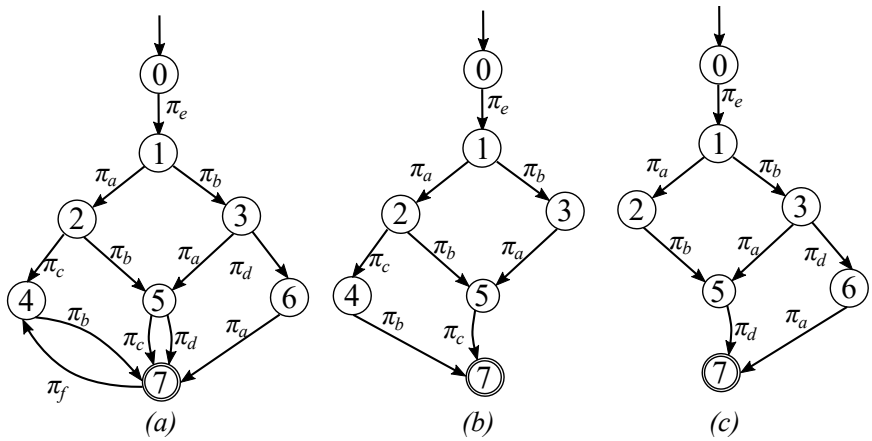


Figure 3.5: (a) The global task DFA G_g with event set E_g, (b) the event-equivalent words composed automaton $\check{G}_1$ with event set $\check{E}_1 = \{\pi_e, \pi_a, \pi_b, \pi_c\}$, and (c) the event-equivalent words composed automaton $\check{G}_2$ with the event set $\check{E}_2 = \{\pi_e, \pi_a, \pi_b, \pi_d\}$.

The parallel decomposition verification is implemented for all event set pairs $(E_j, E_{\bar{j}})$ queued by the size of E_j (i.e., $|E_1| \leq \cdots |E_j| \leq \cdots |E_J|$) until it finds the pair of projections $(P_j(\check{G}), P_{\bar{j}}(\check{G}))$ that satisfies $P_j(\check{G}) \| P_{\bar{j}}(\check{G}) \cong \check{G}$ for an arbitrary j. Here, $P_j(\check{G})$ and $P_{\bar{j}}(\check{G})$ are the corresponding projections of $\check{G}$ into event set pairs $(E_j, E_{\bar{j}})$, and are taken as the decomposition components $G_1 = P_j(\check{G})$ and $G_{\bar{1}} = P_{\bar{j}}(\check{G})$ in the first decomposition iteration. If none of the event set pairs in $\mathbb{E}$ can satisfy the decomposability property, the automaton $\check{G}$ is indecomposable. Otherwise, a further decomposability verification is implemented on $G_{\bar{1}}$ to obtain new components. The above processes are repeatedly implemented on the decomposition component $G_{\bar{i}}$ in the $i+1$th iteration until it cannot find any new decomposition components. As a result, each parallel decomposition result of $\check{G}$ contains a maximum amount of indecomposable subtask automata, i.e., each subtask automaton is the smallest decomposition component. It can guarantee the MRS performs tasks with a maximum amount of sub-processes, and hence achieves the global task with the highest level of parallelism. Summarize the above iterative decomposition process in Alg. 2.

Example 4. Take the event-equivalent words composed automata in Example 3(c) as an example, one can synthesize all the event set pairs $\mathbb{E}_2 = \{(\{\pi_e, \pi_a\}, \{\pi_e, \pi_b, \pi_d\}), \ (\{\pi_e, \pi_b\}, \{\pi_e, \pi_a, \pi_d\}), \ (\{\pi_e, \pi_d\}, \{\pi_e, \pi_a, \pi_b\})\}$. Project the automaton $\check{G}_2$ (Fig. 3.5(c)) into every event set pair in $\mathbb{E}_2$, and verify the language equivalence of the composition of the projections to $\check{G}_2$. Fig. 3.6(a) shows the projections of $\check{G}_1$ to $(\{\pi_e, \pi_a\}, \{\pi_e, \pi_b, \pi_d\})$ and the composition result of the projections. The result is not language equivalent to $\check{G}_2$. In comparison, Fig. 3.6(b) shows the projections of $\check{G}_2$ to $(\{\pi_e, \pi_b\}, \{\pi_e, \pi_a, \pi_d\})$ and the composition result of the projections is language

Algorithm 2 Iterative Parallel Decomposition of Task

Input: Initial task automaton $\check{G}$
Output: Decomposition results: subtask automata G_is

```
function TASKDECOM(Ǧ)
    𝔼 ← Eqn. (3.3), i ← 1, j ← 1
    while |Ě| > 1 do                                   ▷ Size of Ě
        for (E_j, E_j̄) ∈ 𝔼 do                          ▷ E_j ∪ E_j̄ = Ě, try all the event set pairs
            if L(P_j(Ǧ)‖P_j̄(Ǧ)) ≡ L(Ǧ) then            ▷ Language equivalence
                Ǧ ≅ P_j(Ǧ)‖P_j̄(Ǧ), G_i ← P_j(Ǧ), i ← i + 1   ▷ Decomposable
                Ǧ ← P_j̄(Ǧ), Ě ← E_j̄, 𝔼 ← Eqn. (3.3), j ← 1   ▷ Prepare for the next decomposition
                break
            else
                Ǧ ≇ P_j(Ǧ)‖P_j̄(Ǧ), j ← j + 1              ▷ Indecomposable
            end if
        end for
        J ← |𝔼|                                        ▷ Amount of all the event set pairs
        if j = J + 1 and Ǧ ≇ P_J(Ǧ)‖P_J̄(Ǧ) then        ▷ Stop after trying all event set pairs
            break
        end if
    end while
    I ← i, G_I ← Ǧ
return G_1, ··· , G_{I−1}, G_I                         ▷ Ǧ ≅ G_1‖ ··· G_{I−1}‖G_I
end function
```

equivalent to $\check{G}_2$. As a result, the DFA $\check{G}_2$ can be initially parallel decomposed into the two projections $\{L(G_1) = \{\pi_e\pi_b\},\ L(G_{\bar{1}}) = \{\pi_e\pi_a\pi_d\}\}$ in Fig. 3.6 (b).

In the second iteration of parallel decomposition, the remaining (larger size) projection $L(G_{\bar{1}}) = \{\pi_e\pi_a\pi_d\}$ has the event set pairs $\mathbb{E}_2 = \{(\{\pi_e, \pi_a\}, \{\pi_e, \pi_d\})\}$. The projection of automaton $G_{\bar{1}}$ into the only event set pair $(\{\pi_e, \pi_a\}, \{\pi_e, \pi_d\})$ generates two components $L(P_1(G_{\bar{1}})) = \{\pi_e\pi_a\}$ and $L(P_2(G_{\bar{1}})) = \{\pi_e\pi_d\}$. It is obvious that the composition of the two projections is not language equivalent to the automaton $G_{\bar{1}}$, i.e., $P_1(G_{\bar{1}})\|P_2(G_{\bar{1}}) \not\equiv G_{\bar{1}}$. Hence, the automaton $G_{\bar{1}}$ can not be decomposed into any smaller automata according to the parallel decomposability definition and can be denoted as G_2. Finally, the parallel decomposition results are: $\{L(G_1) = \{\pi_e\pi_b\},\ L(G_2) = \{\pi_e\pi_a\pi_d\}\}$ for automaton in Fig. 3.5(c). •

Theorem 1. *The iterative parallel decomposition of a task automaton $\check{G}_l$ with respect to its event set pairs $\mathbb{E}_l$ (generated in Eqn. (3.3)) guarantees a unique set of smallest parallel decomposition components, i.e., the subtask automata.*

Proof In the iterative decomposition process, the queuing of the event set pairs in $\mathbb{E}_l$ guarantees the

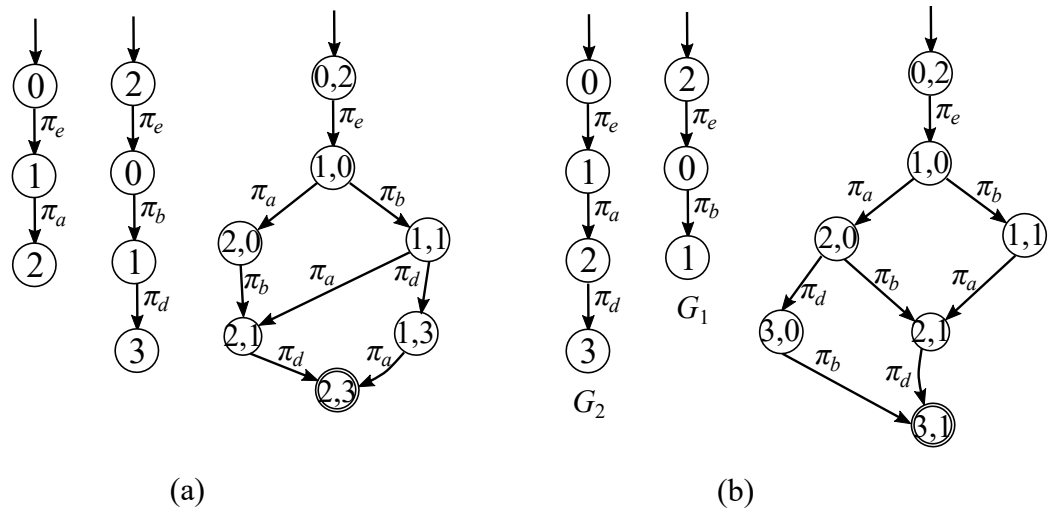


Figure 3.6: Automaton $\check{G}_2$ has event set $\check{E}_2 = \{\pi_e, \pi_a, \pi_b, \pi_d\}$, and the set of event set pairs $\mathbb{E}_2 = \{(\{\pi_e, \pi_a\}, \{\pi_e, \pi_b, \pi_d\}),\ (\{\pi_e, \pi_b\}, \{\pi_e, \pi_a, \pi_d\}),\ (\{\pi_e, \pi_d\}, \{\pi_e, \pi_a, \pi_b\})\}$, (a) the projections of $\check{G}_2$ to the event set pairs $(\{\pi_e, \pi_b\}, \{\pi_e, \pi_a, \pi_d\})$ and the composition result of the projections; (b) the projections of $\check{G}_2$ to the event set pairs $(\{\pi_e, \pi_b\}, \{\pi_e, \pi_a, \pi_d\})$ and the composition result of the projections.

projection G_i to be the smallest indecomposable subtask automaton in each decomposition. It is because a smaller event set can be verified before its event set E_i for the decomposability verification, if G_i can be further decomposed.

Uniqueness of the decomposition components: Assume there are two different sets of smallest decomposition components for $\check{G}_l$, i.e., $\{G_1, \cdots, G_I, G'_1, \cdots, G'_{I'}\}$ with event sets $\{E_1, \cdots, E_I, E'_1, \cdots, E'_{I'}\}$ and $\{G_1, \cdots, G_I, G''_1, \cdots, G''_{I''}\}$ with event sets $\{E_1, \cdots, E_I, E''_1, \cdots, E''_{I''}\}$. Here, $\{G_i, i = 1, \cdots, I\}$ represent possible common decomposition components, which can also be empty without loss of generality. For all the remaining smallest decomposition components $G'_{i'}, i' = 1, \cdots, I'$, given an event set $E'_{i'} \subset \cup_{i'' \in \mathbb{I}} E''_{i''}$ with $\mathbb{I} \subseteq \{1, \cdots, I''\}$, it follows that $L(G'_{i'}) = L(P_{i'}(\|_{i'' \in \mathbb{I}} G''_{i''}))$ and vice versa for all $G''_{i''}$. One can also obtain the event set of the projection $P_{i'}(\|_{i'' \in \mathbb{I}} G''_{i''})$ as $\bigcup_{i'' \in \mathbb{I}} (E_{i''} \cap E_{i'})$, and conclude that $L(P_{i'}(\|_{i'' \in \mathbb{I}} G''_{i''})) = L(\|_{i'' \in \mathbb{I}} P_{i'}(G''_{i''}))$. It is because each $P_{i'}(G''_{i''})$ substitutes transitions featured by events in $E_{i''} \setminus E_{i'}$ with ϵ, and it does not affect the parallel property. Consequently, $G'_{i'}$ can still satisfy the decomposability, which is contradicted with the assumption about smallest decomposition component. Therefore, the set of smallest decomposition components is unique. □

In the overall decomposition process, enumerate all the event subsets and identify those subsets that make the parallel decomposition viable. In comparison, the works in [45, 78, 77] conduct the decomposition of a task specification only concerning the given capability set of each robot. Next, denote the subautomaton

$\check{G}_{l'}$ as the decomposable automaton in $\{\check{G}_l,\ l \in \mathbb{Z}\}$ because not all the event-equivalent words composed automata are decomposable. The corresponding event set of subautomaton $\check{G}_{l'}$ can be denoted as the decomposable event set $\check{E}_{l'}, l' \in \mathbb{Z}$.

Proposition 2. *A global task automaton G_g is said to be loosely decomposable, if there exists a subautomaton $\check{G}_{l'}$ with $L(\check{G}_{l'}) \subseteq L(G_g)$, which is event-equivalent with the event set $\check{E}_{l'}$ and satisfies the equivalence relation $\check{G}_{l'} \cong \|_{i=1}^{I_{l'}} P_i(\check{G}_{l'})$ according to Thm. 1. Denote the collection of all the decomposable event sets as $\mathbb{E}_g := \{\check{E}_{l'}, l' \in \mathbb{Z}\}$ and the collection of all decomposable subautomata of G_g as $\{\check{G}_{l'}, l' \in \mathbb{Z}\}$.*

Proof The automaton G_g can be decomposed in different ways, if $L(G_g)$ contains the strict decomposable language $L(\check{G}_{l'})$ with respect to different decomposable event sets $\check{E}_{l'}$ according to Thm. 1. Any $L(\check{G}_{l'}) \subseteq L(G_g)$ with $\check{E}_{l'} \in \mathbb{E}_g$ can satisfy the global task specification, and the words $L(G_g) \setminus L(\check{G}_{l'})$ do not affect this satisfaction. Therefore, one can take $\{P_i(\check{G}_{l'}), i = 1, \cdots, I_{l'}\}$ as the parallel decomposition components satisfying the global task specification. □

3.5 Parallel Decomposition based Robot Assignment and Task Planning without Robot Transition System

This section presents an automatic robot and task assignment framework to synthesize an optimal solution from the sets of subtasks $\{\{P_i(G \downarrow_{\Sigma_{id}}), i \in \mathbb{Z}\}, \Sigma_{id} \in \Sigma_D\}$.

3.5.1 Parallel Task Allocation Automaton of MRS

Definition 15 (Task Allocation Automaton). Given a task automaton $G = (X, E, f, x_0, X_m)$ and a robot set $\mathcal{R}$, a task allocation automaton $\Psi = (X_\psi, Act_\psi, \delta_\psi, x_{\psi,0}, X_{\psi,m}, W_\psi)$ can be synthesized to describe the assignment of robots to the events in the task automaton, where

1. $X_\psi = X$ is the task completion state set, $x_{\psi,0} \in X_\psi$ is the initial state, $X_{\psi,m} \subseteq X_\psi$ is a final state set;

2. $Act_\psi = \{(R, e) | R \subseteq \mathcal{R}, e \in \cup_{r \in R} E_r \bigcap E\}$ is the set that describes a subset R of robots performing the event e of the task automaton, where e can be a single or a cooperative event, and $\cup_{r \in R} E_r \bigcap E$ ensures the robots in set R are capable of performing the events in G;

3. $\delta_\psi(x_\psi, act_\psi) = x'_\psi$ iff $f(x_\psi, e) = x'_\psi$ with $x_\psi,\ x'_\psi \in X_\psi$;

4. $W_\psi : Act_\psi \to \mathbb{R}$ is the cost set with $W_\psi(act_\psi)$ estimating the total cost of robot set R for event $e \in E$.

The above structure can be used to synthesize a global task allocation automaton for the MRS to perform tasks in a consecutive way, which does not consider the task concurrency. On the other hand, we have obtained a set of decompositions $P_i(G \downarrow_{\Sigma_{id}})$ that can work in parallel. To add the concurrency of MRS tasking, we incorporate the concurrent execution $\|_{i\in\mathbb{Z}} P_i(G \downarrow_{\Sigma_{id}})$ to synthesize a *parallel* task allocation automaton, which contains all the concurrent task performing by robot set. We start with a parallel execution of two of such subtask allocation automata as follows.

Definition 16 (**Parallel Execution of Task Allocation Automata).** Given two subtask allocation automata $\Psi_1 = (X_{\psi,1}, Act_{\psi,1}, \delta_{\psi,1}, x^1_{\psi,0}, X^1_{\psi,m}, W_{\psi,1})$ and $\Psi_2 = (X_{\psi,2}, Act_{\psi,2}, \delta_{\psi,2}, x^2_{\psi,0}, X^2_{\psi,m}, W_{\psi,2})$, a parallel execution of Ψ_1 and Ψ_2 is defined as $\Psi_1 \oplus \Psi_2 = (X_{\psi,1} \times X_{\psi,2}, Act^{1,2}_\psi, \delta^{1,2}_\psi, (x^1_{\psi,0}, x^2_{\psi,0}), X^1_{\psi,m} \times X^2_{\psi,m}, W^{1,2}_\psi)$, where

1. $Act^{1,2}_\psi = Act_{\psi,1} \bigcup Act_{\psi,2} \bigcup Act^{1\oplus 2}_\psi$, where $Act^{1\oplus 2}_\psi = \{(act^{1\oplus 2}_{\psi,1}, act^{1\oplus 2}_{\psi,2}) | act^{1\oplus 2}_{\psi,1} = (R_1, e_1) \in Act_{\psi,1}, act^{1\oplus 2}_{\psi,2} = (R_2, e_2) \in Act_{\psi,2}, R_1 \cap R_2 = \emptyset\}$ is the set that describes two subsets of robots R_1 and R_2 parallel executing events e_1 and e_2;

2. $\delta^{1,2}_\psi((x_{\psi,1}, x_{\psi,2}), act^{1,2}_\psi) =$

$$\begin{cases} (x'_{\psi,1}, x'_{\psi,2}), & \text{if } act^{1,2}_\psi \in Act^{1\cup 2}_\psi, \\ (x'_{\psi,1}, x_{\psi,2}), & \text{if } act^{1,2}_\psi \in Act_{\psi,1} \setminus Act^{1\cup 2}_{\psi,1}, \\ (x_{\psi,1}, x'_{\psi,2}), & \text{if } act^{1,2}_\psi \in Act_{\psi,2} \setminus Act^{1\cup 2}_{\psi,2}; \end{cases}$$

   where $Act^{1\cup 2}_\psi = Act^{1\oplus 2}_\psi \bigcup (Act_{\psi,1} \cap Act_{\psi,2})$. $Act^{1\cup 2}_{\psi,1}$ and $Act^{1\cup 2}_{\psi,2}$ are the respective subsets of $Act^{1\cup 2}_\psi$ associated with Ψ_1 and Ψ_2;

3. $W^{1,2}_\psi : Act^{1,2}_\psi \to \mathbb{R}^+$ is the set denoting the costs of robots in R_1 and R_2 for executing events e_1 and e_2, where $w_{\psi,1} \in W_{\psi,1}$ and $w_{\psi,2} \in W_{\psi,2}$.

Remark 3. Note that in the set $Act^{1\cup 2}_\psi$, if $act^{1,2}_\psi \in Act_{\psi,1} \cap Act_{\psi,2}$, it corresponds to a robot set $R \subseteq R_1 \cap R_2$ performing a cooperative event $e \in E_1 \cap E_2$, where R_1 and R_2 are the respective robots associated with Ψ_1 and Ψ_2. If $act^{1,2}_\psi \in Act^{1\oplus 2}_\psi$, it describes two sets of robots R_1 and R_2 performing their corresponding events $e_1 \in E_1$ and $e_2 \in E_2$ in parallel with $R_1 \cap R_2 = \emptyset$. Furthermore, not all events $e_1 \in E_1$ and $e_2 \in E_2$ can be parallel executed in cases $R_1 \cap R_2 \neq \emptyset$, i.e. a robot cannot perform different tasks at the same time. •

For a set of parallel subtask allocation automata $\{\Psi_i, i = 1, \cdots, I\}$, we can obtain the parallel execution of these Ψ_i with $\bigoplus_{i=1}^{I} \Psi_i = ((\Psi_1 \oplus \Psi_2) \oplus \cdots) \oplus \Psi_I$. We call this automaton the parallel task allocation automaton, and denote it as $\bigoplus_{i=1}^{I} \Psi_i = (X_\oplus, Act_\oplus, \delta_\oplus, x_{\oplus,0}, X_{\oplus,m}, W_\oplus)$. The *parallel* task allocation automaton not only explicitly presents the multi-robot parallel task performing, but also reduces the states of task allocation.

A finite path $\mathtt{S}_\oplus = act_\oplus^{(0)} \cdots act_\oplus^{(\tau)} \cdots act_\oplus^{(\mathbf{T})}$ presents a task allocation solution to $\bigoplus_{i=1}^{I} \Psi_i$, where $act_\oplus^{(\tau)} = (act_{\psi,1}^{(\tau)}, \cdots, act_{\psi,I}^{(\tau)}) \in Act_\oplus$ denotes different sets of robots R_i performing their respective event e_i in parallel. Note that an $act_{\psi,i}^{(\tau)}$ may be empty, denoted by ϵ, in an $act_\psi^{\oplus}$ since not all transitions of the parallel task allocation automaton have I compositions. Denote $\eta(act_\oplus^\tau)$ as the actual number of parallel executions of subtask allocation automata in $act_\oplus^\tau$. We can lower the costs of the task allocation solutions with higher concurrency by imposing a discount to the corresponding transitions of $\bigoplus_{i=1}^{I} \Psi_i$ with $\alpha^{\eta(act_\oplus^\tau)-1} W_\oplus(act_\oplus^\tau)$, where $\alpha \in [0, 1]$ is a concurrency weight.

3.5.2 Optimal Parallel Task Allocation Solution for MRS

We next aim to obtain an optimal task allocation solution for the MRS, which involves the costs of (i) concurrency for multi-robot multi-task processing, (ii) task execution considering robot heterogeneity, and (iii) robot motion considering traveling distance and energy consumption. A minimization of these three costs contributes to a global optimization of our task allocation. Since $\bigoplus_{i=1}^{I} \Psi_i$ already takes into account the costs of concurrency and task execution in each transition, the next step is to add the motion cost and search for the path with minimal accumulated cost along the transitions of $\bigoplus_{i=1}^{I} \Psi_i$.

The costs of motion of each transition in $\bigoplus_{i=1}^{I} \Psi_i$ are directly related to the reachability of each event in $act_{\psi,i} \in Act_{\psi,i}$ for robot set R_i. We denote the reachability as a motion specification $Mot_{\psi,i}^{(\tau)}$, which is conducted before each $act_{\psi,i}^{(\tau)}$ in $\mathtt{S}_\oplus$. Each specification $Mot_{\psi,i}^{(\tau)}$ can be described in the LTL form as $\neg L_q^{-1}(act_{\psi,i}^{(\tau)})\ U\ act_{\psi,i}^{(\tau-1)}, L_q^{-1} : \pi_i \rightarrow q$, which means that: "At step τ, each robot $r_i \in R_i$ will go to the position of its implementable action $(r_i, \pi_i) \models act_{\psi,i}^{(\tau)}$ iff the previous $act_{\psi,i}^{(\tau-1)}$ in the same subtask automaton has been completed". The estimated costs of each robot's motion specification is propositional to the distance of planned paths. The transition system of robot r_k for $Mot_{\psi,i}^{(\tau)}$ in a discrete space can be described as a transition system of robot path planning according to Eqn. (13) at step τ corresponding to subtask automaton G_i, denoted as $TS_{i,k}^{(\tau)}$. The model checking with LTL motion specification $Mot_{\psi,i}^{(\tau)}$ and transition system $TS_{i,k}^{(\tau)}$ can generate a motion planning path $\sigma_{i,k}^{(\tau)} = q_{i,k}^{(\tau,0)} \cdots q_{i,k}^{(\tau,t)} \cdots$ satisfying $Mot_{\psi,i}^{(\tau)}$ with the lowest motion cost $\min(W_{q,k}(\sigma_{i,k}^{(\tau)}))$.

We use the Dijkstra algorithm to search the optimal task allocation solution from the parallel task allocation automaton $\bigoplus_{i=1}^{I} \Psi_i$. Note that the costs of motion at each transition are varied based on the previous task allocations during the Dijkstra searching process; hence motion costs can only be computed during this phase. In addition, recall that we may have different sets of decompositions $\{\{P_i(G \downarrow_{\Sigma_{id}}), i \in \mathbb{Z}\}, \Sigma_{id} \in \Sigma_D\}$. Each $\{P_i(G \downarrow_{\Sigma_{id}}), i \in \mathbb{Z}\}$ can synthesize an suboptimal task allocation solution $\mathtt{S}_{\oplus,id}$ with respect to Σ_{id}. We can finally compare these suboptimal solutions and select the one with the minimal costs, i.e. $\mathtt{S}^*_{\oplus},\ \Sigma^*_{id} \in \Sigma_D$.

Given an optimal task allocation $\mathtt{S}^*_{\oplus}$, the optimal sequence of task allocation and motion specifications is then given as

$$\mathtt{MS}^*_{\oplus} = Mot^{(0)}_{\oplus} act^{(0)}_{\oplus} \cdots Mot^{(\tau)}_{\oplus} act^{(\tau)}_{\oplus} \cdots Mot^{(\mathbf{T})}_{\oplus} act^{(\mathbf{T})}_{\oplus}. \tag{3.4}$$

3.6 Parallel Decomposition based Robot Assignment and Task Planning with Robot Transition System

The work in the previous section dealt with a set of subtasks in parallel by assuming the robot states can transit arbitrarily. The work first assigned robots to the automaton corresponding to each subtask specification. Then the framework composed all the robot-automaton assignment results and sought for the optimal task planning solution. Fig. 3.7 shows the concept of the preliminary work with two different robot-automaton assignment examples, each of which has a level of parallelism. However, a practical task execution process needs viable robot state transitions for the satisfaction of a task specification (see the left bottom section of Fig. 3.7). It is a verification problem and can become more challenging in situations where insufficient robot resources are shared among multiple parallel subtask specifications. The reason is one robot may not be able to switch across the different subtask specifications. That means a limited number of robots will result in conflicts among the parallel task performing processes, which leads to a *deadlock*. The parallel decomposition-based SPA in this section can resolve the above issue and provide a deadlock-free solution.

Given a task automaton G_g and a group of robots with capability transition systems $TE_n,\ n = 1, \cdots, N$, a robot assignment and task planning problem considers which robots shall be assigned to which atomic tasks so that these robots can satisfy the global task specification. The robot assignment and task planning solution can be obtained with model checkers, such as SPIN and NuSMV, by verifying the global automaton G_g with the parallel composition of transition systems $\|_{n=1}^{N} TE_n$. However, the synthesized

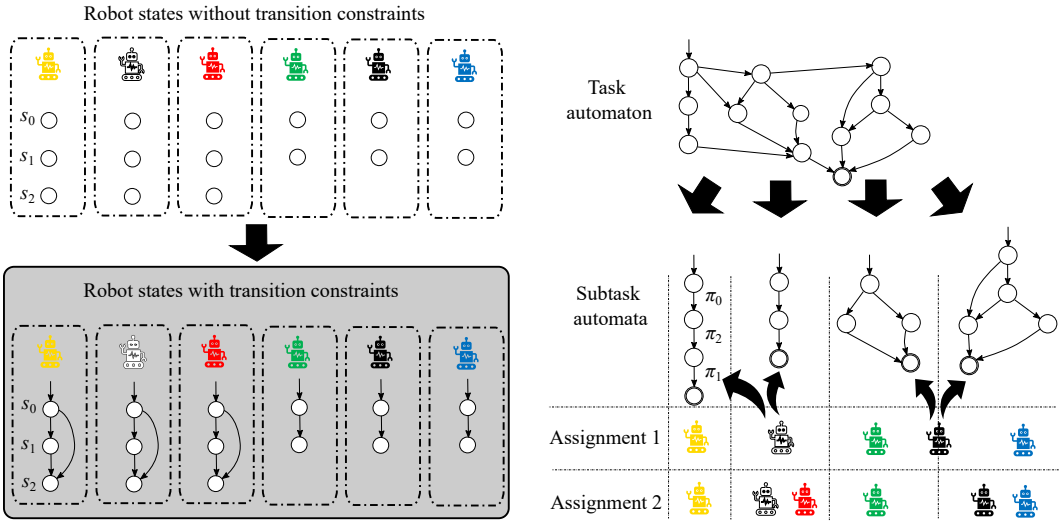


Figure 3.7: Illustration of two example robot-task assignments in the paper [103]. Assignment example 1 shows the assignment of the "white" robot to switch across the two corresponding subtask automata and collaborate with the "yellow" robot, and the "black" robot to switch across two subtask automata and collaborate with the "green" and "blue" robots. Assignment example 2 shows the assignment of the "white" and "red" robots as one subgroup and the "black" and "blue" robots as another subgroup. Both example assignments assume every robot can transit among states arbitrarily. The validity of the assignments can be pretty different if each of the robots has state transition restrictions (as the case considered in this paper and illustrated in the grey shaded box in the left bottom corner). For example, in assignment example 2, assume the "yellow" robot is assigned to the subtask automaton labeled with $\pi_0\pi_2\pi_1$. The robot can not accomplish the assigned subtask automaton given the state transition constraints $s_0 \rightarrow s_1 \rightarrow s_2$, i.e., the task plan $s_0 \rightarrow s_2 \rightarrow s_1$ is not viable for the "yellow" robot.

solution merely describes a set of atomic tasks consecutively satisfied by the MRS and is not able to achieve parallel execution of the task specification.

To enable a parallel execution process, divide the MRS into different subsets of robots and synthesize an SPA Ψ_i with each subset of robots and its allocated subtask automaton $P_i(\check{G}_{l'}),\ i = 1, \cdots, I_{l'},\ \check{E}_{l'} \in \mathbb{E}_g$. Then, an optimal task plan can be searched in each SPA by considering the accumulated costs of performing atomic tasks. In comparison to the previous section, this section generate the task plan through further verifying the corresponding robot capability transition systems with each decomposition component $P_i(\check{G}_{l'})$ rather than directly associating the robots with the events of the subtask automaton. The process will guarantee feasible transitions of robot states between different atomic tasks in the generated task plan. This section also achieve the concurrent satisfaction of multiple subtask automata $P_i(\check{G}_{l'})$, which deals with these subtask automata in an interleaved manner so that deadlock-free solutions can be obtained by the MRS to satisfy the task specifications. For the sake of notation simplicity, this section use G_i to represent the subtask automata $P_i(\check{G}_{l'})$ for the discussion of the SPA in the rest of this section. All the definitions and theorem in this section apply directly to the decomposition components $P_i(\check{G}_{l'}),\ i = 1, \cdots, I_{l'},\ \check{E}_{l'} \in \mathbb{E}_g$ of the general global task automaton G_g.

3.6.1 Individual Subtask Planning Automaton

Definition 17 (Robot Assignment to Task Automaton). Given a set of robots $\mathcal{R}_i$ and a task automaton G_i with event set E_i, define $\mathcal{R}_i$ as a *robot assignment of* G_i if (1) each robot $r_n \in \mathcal{R}_i$ is capable of some events of the automaton G_i, i.e., $2^{AP_n} \cap E_i \neq \emptyset$; and (2) the robot set $\mathcal{R}_i$ is capable of all the events of automaton G_i, i.e., $2^{\cup_{r_n \in \mathcal{R}_i} AP_n} \supseteq E_i$. The satisfaction of $\mathcal{R}_i$ to G_i can be described as a robot-automaton pair $(\mathcal{R}_i, G_i)$. Denote $\mathcal{R}_{\min,i} \in 2^{\mathcal{R}_i}$ as the minimum robot assignment of G_i if $\mathcal{R}_{\min,i}$ has the smallest amount of robots satisfying $2^{\cup_{r_n \in \mathcal{R}_{\min,i}} AP_n} \supseteq E_i$. Denote $\mathcal{R}_{\max,i} \in 2^{\mathcal{R}_i}$ as the maximum robot assignment if it contains all the robots within $\mathcal{R}$ that can be utilized to achieve G_i.

Denote each of the robot-automaton pair of the parallel decomposition as $(\mathcal{R}_i, G_i)$, where $\mathcal{R}_i := \{r_{i_1}, \cdots, r_{i_n}, \cdots, r_{i_N}\}$, the parallel composition of capability transition systems $TE_n,\ n = i_1, \cdots, i_N$ can be generated with reference to Def. 12 as $\|_{n=i_1}^{i_N} TE_n := ((TE_{i_1} \| TE_{i_2}) \| \cdots) \| TE_{i_N}$. Each subtask automaton G_i allows only one event to be achieved in each of its state transition, i.e., the robot set $\mathcal{R}_i$ needs to work sequentially for each event in the subtask automaton G_i. Then, it needs to verify whether the robot assignment $\mathcal{R}_i$ with the parallel composition $\|_{n=i_1}^{i_N} TE_n$ satisfies the subtask automaton G_i sequentially. The

verification results will form a product automaton containing all the robot-task assignments by taking into account the transition constraints of events in the subtask automaton G_i. The resulting automaton can be referred to the following definition.

Definition 18 (Subtask Planning Automaton (SPA)). Given an indecomposable task automaton $G := (X, E, f, x_0, X_F)$ and a parallel composition of transition systems $\|_{n=1}^{N} TE_n := (\prod_{n=1}^{N} S_n, \bigcup_{n=1}^{N} A_n, \delta_{\mathcal{R}}, \langle s_{0,1}, \cdots, s_{0,N}\rangle, \cup_{n=1}^{N} AP_n, \mathcal{L}_{\mathcal{R}}, \prod_{n=1}^{N} W_n)$, an SPA $\Psi^s := (X_\psi^s, Act_\psi^s, \delta_\psi^s, x_{\psi,0}^s, X_{\psi,F}^s, AP_\psi^s, \mathcal{L}_\psi^s, W_\psi^s)$ can be synthesized from the operation $\|_{n=1}^{N} TE_n \times G$ to describe the assignment of robots to the events in the task automaton, where

1. $X_\psi^s := \prod_{n=1}^{N} S_n \times X$ is the state set containing all the composite states composed by states of robot capability and states of the task automaton;

2. $Act_\psi^s := \cup_{n=1}^{N} Act_{\psi,n} \bigcup Act_{\psi,c}$ is the action set, where $Act_{\psi,n} := \{\langle \alpha, \mathcal{R}^\alpha\rangle | \alpha \in A_n, \mathcal{R}^\alpha = r_n\}$ and $Act_{\psi,c} := \{\langle \alpha, \mathcal{R}^\alpha\rangle | \alpha \in A^c, \mathcal{R}^\alpha = \langle r_1, \cdots, r_N\rangle\}$, describes robots $\mathcal{R}^\alpha$ implementing the corresponding actions in their own capability transition systems $TE_1, \cdots, TE_N$;

3. the transition relation $\delta_\psi^s(x_\psi^s, act_\psi^s) := x_\psi^{s'}$ with $x_\psi^s := (\langle s_1, \cdots, s_N\rangle, x)$, $act_\psi^s := \langle \alpha, \mathcal{R}^\alpha\rangle \in Act_\psi^s$ and $x_\psi^{s'} := (\langle s_1, \cdots, s_N\rangle', x')$ iff $f(x, e) = x' \wedge \delta_{\mathcal{R}}(\langle s_1, \cdots, s_N\rangle, \alpha) = \langle s_1, \cdots, s_N\rangle' \wedge \mathcal{L}_{\mathcal{R}}(\langle s_1, \cdots, s_N\rangle') \models e$;

4. the initial state $x_{\psi,0}^s := (\langle s_{0,1}, \cdots, s_{0,N}\rangle, x_1) \in X_\psi^s$ if $\exists x_1 = f(x_0, e) \wedge \mathcal{L}_{\mathcal{R}}(\langle s_{0,1}, \cdots, s_{0,N}\rangle) \models e$.

5. the accepted state set $X_{\psi,F}^s := \{(\langle s_1, \cdots, s_N\rangle, x_F) | \langle s_1, \cdots, s_N\rangle \in \prod_{n=1}^{N} S_n,\ x_F \in X_F\}$;

6. $AP_\psi^s := \cup_{n=1}^{N} AP_n$ is the proposition set containing atomic propositions and logic propositions and $E \subseteq 2^{AP_\psi^s}$;

7. $\mathcal{L}_\psi^s : X_\psi^s \to 2^{AP_\psi^s}$ labels a state x_ψ^s;

8. $W_\psi^s : X_\psi^s \times Act_\psi^s \to \mathbb{R}^+$ is the cost set with $W_\psi^s(x_\psi^s, act_\psi^s)$ estimating the total cost of robot set $r_1, \cdots, r_N$ associated with state $\langle s_1, \cdots, s_N\rangle$ for actions $\alpha_n \in A_n$.

Denote $\boldsymbol{R}_i := \{\mathcal{R}_i | \mathcal{R}_{\min,i} \subseteq \mathcal{R}_i \subseteq \mathcal{R}_{\max,i}\}$ as the collection of all the robot assignments for each subtask automaton $G_i,\ i = 1, \cdots, I$. Each pair $(\mathcal{R}_i, G_i)$ can then be utilized to synthesize an SPA Ψ_i^s based on Def. 18.

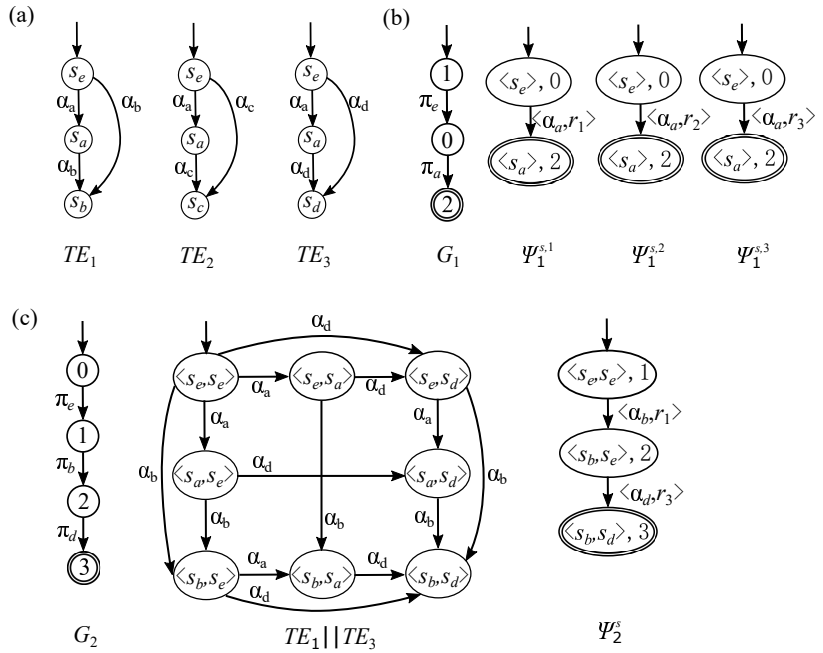


Figure 3.8: (a) Robot transition systems TE_1, TE_2, and TE_3, (b) subtask automaton G_1 and SPA $\Psi_1^{s,1}$ ($\Psi_1^{s,2}$ and $\Psi_1^{s,3}$) with respect to robot assignment $\{r_1\}$ ($\{r_2\}$ and $\{r_3\}$), (c) subtask automaton G_2, parallel composition $TE_1 \| TE_3$, and SPA Ψ_2^s (in minimized size) synthesized from $TE_1 \| TE_3$ and G_2.

Example 5. Given 3 robots r_1, r_2, and r_3, whose transition systems are shown in Fig. 3.8(a). To better relate the atomic tasks to the states of the robot transition systems, abuse the state notation and label the relation for each state as $\mathcal{L}_n(s_k) = \{\pi_k\}$, $k = a, b, c, d, e$, $n = 1, 2, 3$. Take two parallel automata G_1 and G_2 shown in Fig. 3.8(b) and (c) as examples. As a result, subtask automaton G_1 has the robot assignments $\boldsymbol{R}_1 = \{\{r_1\}, \{r_2\}, \{r_3\}\}$, and subtask automaton G_2 has the robot assignments $\boldsymbol{R}_2 = \{\{r_1, r_3\}\}$. According to Def. 18, the transition system TE_1 (TE_2, TE_3) and the subtask automaton G_1 generate the SPA $\Psi_1^{s,1}$ ($\Psi_1^{s,2}, \Psi_1^{s,3}$), respectively (shown in Fig. 3.8(b)). The parallel composition of transition systems $TE_1 \| TE_3$ and the subtask automaton G_2 synthesize the SPA Ψ_2^s (shown in Fig. 3.8(c)). •

Remark 4. Note that there are three robot assignment solutions for the two parallel automata in Example 5, i.e., Solution 1: robot assignment $\{r_1\}$ to automaton G_1, robot assignment $\{r_1, r_3\}$ to automaton G_2; Solution 2: robot assignment $\{r_2\}$ to automaton G_1, robot assignment $\{r_1, r_3\}$ to automaton G_2; and Solution 3: robot assignment $\{r_3\}$ to automaton G_1, robot assignment $\{r_1, r_3\}$ to automaton G_2. In Solution 1, both automata G_1 and G_2 will need robot r_1 in synthesizing their SPA. A coordination of robot r_1 between automata G_1 and G_2 is necessary to guarantee the deadlock-free of task performing with Solution 1. The same situation happens for Solution 3 with robot r_3. These situations will be discussed in Sec. 3.6.2.

In addition, it is possible that an SPA has no effective path to satisfy the subtask automaton G_i even if the maximum robot assignment $\mathcal{R}_{\max,i}$ is provided with $2^{\cup_{r_n \in \mathcal{R}_{\max,i}} AP_n} \supseteq E_i$. In this case, the current set of provided robots $\mathcal{R}_i$ is not capable of achieving the subtask automata $G_i, i = 1, \cdots, I$. A set of competent robots will need to be supplemented to guarantee the existence of a path for a synthesized SPA. •

3.6.2 Concurrent Subtask Planning Automaton of MRS

The individual SPA in the previous section would satisfy the global automaton G_g if the MRS satisfies the two conditions: (1) there exists robot assignment $\mathcal{R}_i$ for each G_i; and (2) each robot assignment $\mathcal{R}_i$ works independently and exclusively for the corresponding subtask automaton G_i (e.g., the robot assignments $\{r_2\}$ for $\Psi_1^{s,2}$ and $\{r_1, r_3\}$ for Ψ_2^s, as shown in Example 5). However, there may exist situations where robots need to work for multiple subtask automata and switch among these subtask automata (e.g., the robot assignments $\{r_1\}$ for SPA $\Psi_1^{s,1}$ and $\{r_1, r_3\}$ for Ψ_2^s in Example 5). One can assign the robot r_1 first to work for SPA $\Psi_1^{s,1}$ and then for SPA Ψ_2^s, or vice versa. An alternative is to assign robot r_1 to work concurrently for SPA $\Psi_1^{s,1}$ and Ψ_2^s, which can significantly improve the task performing efficiency. Nevertheless, the latter needs to guarantee a *deadlock-free* process, i.e., there exist viable switches for a robot to transit between states across these subtask automata. Therefore, a concurrent deadlock-free SPA is synthesized to guarantee valid switches of robots across these subtask automata. In addition, this chapter aim to achieve less frequent switches across the subtask automata so that the MRS can achieve more subtask automata in parallel. To realize these, define the topology of robot assignment for subtask automaton as follows.

Definition 19 (Topology of Robot Assignment)**.** Given a set of robot-automaton pairs $(\mathcal{R}_i, G_i), i = 1, \cdots, I$, describing the automaton G_i and their associated robot assignments $\mathcal{R}_i$, define the topology of robot assignment with a graph $\mathcal{G} := (\mathcal{V}, \mathcal{E})$, where the vertex set is $\mathcal{V} := \{(\mathcal{R}_i, G_i) | i = 1, \cdots, I\}$, and the edge set is $\mathcal{E} := \{(v, v') | v = (\mathcal{R}_i, G_i) \in \mathcal{V}, v' = (\mathcal{R}_{i'}, G_{i'}) \in \mathcal{V}, i \neq i', \mathcal{R}_i \cap \mathcal{R}_{i'} \neq \emptyset\}$, where each edge describes that two automata G_i and $G_{i'}$ require the same robots in $\mathcal{R}_i \cap \mathcal{R}_{i'}$.

According to the topology of robot assignment $\mathcal{G}$, the existence of an edge between any two nodes, i.e., any two robot-automaton pairs, means the two corresponding automata require the same robots. That implies the two automata can not be achieved in parallel due to the insufficient robots. Therefore, $\mathcal{G}$ can describe the level of parallelism of subtask automata with its isolated subgraphs. Denote an isolated subgraph as $\mathcal{G}_y,\ y \in \mathbb{Z}$. Every isolated subgraph $\mathcal{G}_y,\ y \in \mathbb{Z}$ of $\mathcal{G}$ represents an independent subset of subtask automata and robot assignments that need to work concurrently. Next, identify the subtask automata belonging to a

same isolated subgraph as the concurrent automaton set.

Definition 20 (**Concurrent Automaton Set).** Given a topology of robot assignment $\mathcal{G}$ that is formed with task automata G_i, $i = 1, \cdots, I$ and their associated robot assignments $\mathcal{R}_i$, assume it has Y numbers of isolated subgraphs $\mathcal{G}_y := (\mathcal{V}_y, \mathcal{E}_y)$, $y = 1, \cdots, Y$. Define the automaton set $\mathbb{G}_y := \{G_i | (\mathcal{R}_i, G_i) \in \mathcal{V}_y,\ i \in \{1, \cdots, \tilde{I}_y\}\}$ as the concurrent automaton set, and $\mathbb{R}_y := \{\mathcal{R}_i | (\mathcal{R}_i, G_i) \in \mathcal{V}_y,\ i \in \{1, \cdots, \tilde{I}_y\}\}$ as the associated robot assignment set.

Based on the above definition, the set of subtask automata $\{G_i, i = 1, \cdots, I\}$ with robot assignments $\{\mathcal{R}_i,\ i = 1, \cdots, I\}$ can be divided into a set of concurrent automaton sets $\{\mathbb{G}_y, y = 1, \cdots, Y\}$. The corresponding subgroup of robots in $\mathbb{R}_y$ needs to work concurrently for all the subtask automata in $\mathbb{G}_y$ rather than in parallel. That is, each concurrent automaton set $\mathbb{G}_y$ identifies the subtask automata that need robots to switch across them in order to complete the tasks. A concurrent SPA can be synthesized for a set of concurrent task automata with their corresponding robot assignments.

Definition 21 (**Concurrent Subtask Planning Automaton).** Given a set of concurrent task automata $\mathbb{G}_y = \{G_1, \cdots, G_{\tilde{I}_y}\}$ with robot set $\mathbb{R}_y = \{\mathcal{R}_1, \cdots, \mathcal{R}_{\tilde{I}_y}\}$, $y \in \{1, \cdots, Y\}$ according to Def. 20 and each $G_i := (X_i, E_i, f_i, x_{0,i}, X_{F,i})$, a concurrent SPA can be synthesized as $\Psi_y^c := (X_\psi^c, Act_\psi^c, \delta_\psi^c, x_{\psi,0}^c, X_{\psi,F}^c, AP_\psi^c, \mathcal{L}_\psi^c, W_\psi^c)$ with the operation $\|_{r_n \in \mathcal{R}^c} TE_n \times \|_{i=1}^{\tilde{I}_y} G_i$ according to Def. 18, where $\mathcal{R}^c = \cup_{i=1}^{\tilde{I}_y} \mathcal{R}_i$.

Remark 5. The concurrent SPA deals with the parallel composition of concurrent automata rather than each individual subtask automaton in comparison to the individual SPA. In other words, whether the SPA is concurrent or individual depends on if it is synthesized from multiple subtask automata or a single subtask automaton. The synthesis process avoids the potential deadlocks that can be caused by the simultaneous requests for the intersected robots from different concurrent automata. The concurrent SPA verifies feasible transitions of the robots across subtask automata. •

Example 6. Given 3 robots r_4, r_5 and r_6, whose transition systems are shown in Fig. 3.9(a), where the labeling relation for each state is $\mathcal{L}_n(s_k) = \{\pi_k\}$, $k = a, b, c, d, f, h$. The parallel decomposition results $\{G_3, G_4\}$ of a task automaton are shown in Fig. 3.9(b). A robot assignment $\{r_4, r_5\}$ is for the decomposition component G_3, and $\{r_4, r_6\}$ is for G_4. Both G_3 and G_4 need robot r_4, so they are concurrent subtask automata, and $\{\{r_4, r_5\}, \{r_4, r_6\}\}$ is the robot set of $\{G_3, G_4\}$. The concurrent SPA Ψ^c is synthesized from $\|_{n=4}^{6} TE_n \times \|_{i=3}^{4} G_i$ according to Def. 21, as shown in Fig. 3.9(c). •

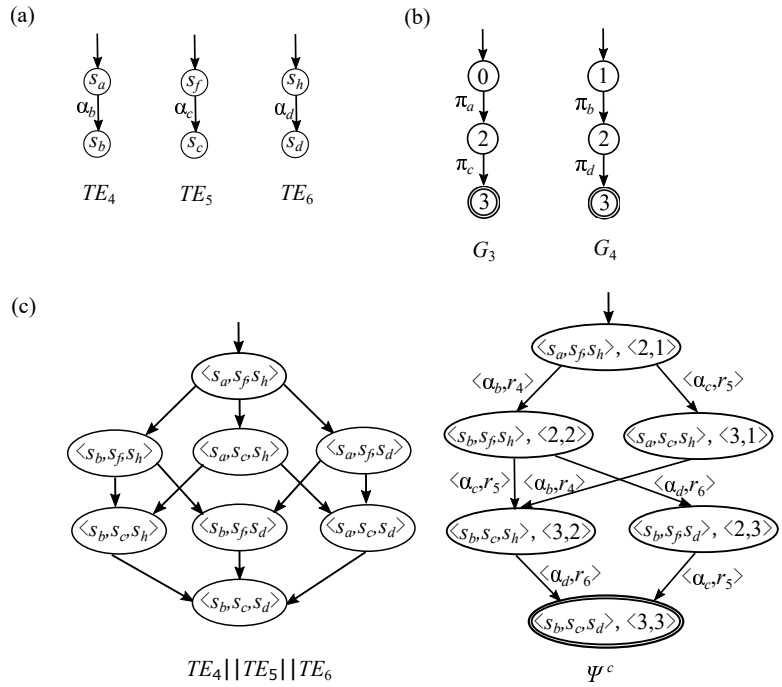


Figure 3.9: (a) Robot transition systems TE_4, TE_5, and TE_6, (b) subtask automaton G_3 and G_4, and (c) parallel SPA Ψ^c (in minimized size) synthesized with $(TE_4 \| TE_5 \| TE_6) \times (G_3 \| G_4)$. The robot index corresponding to each composite state is $\langle r_4, r_5, r_6 \rangle$ in the parallel composition of transition systems.

3.6.3 Parallelism and Computational Complexity of Parallel Task Planning

According to Secs. 3.6.1 and 3.6.2, individual and concurrent SPA can be synthesized for the subtask automaton set $\{G_i, i = 1, \cdots, I\}$. Each subtask automaton here can be assigned to different robot subsets, i.e., different robot assignments $\mathcal{R}_i$ for each subtask automaton G_i. Then, multiple choices of concurrent automaton sets $\{\mathbb{G}_y, y = 1, \cdots, Y\}$ can be generated, which leads to different sets of corresponding individual and concurrent SPA according to Defs. 18 and 21. The cardinality of $\{\mathbb{G}_y, y = 1, \cdots, Y\}$ is Y and can be described as the level of parallelism of a task planning solution. A larger Y brings a higher parallelism for the MRS task performing.

This chapter aim to obtain a task planning solution with the highest level of parallelism for the multi-robot multi-tasking process given a limited amount of robots. That is, allowing the maximum groups of robots to work in parallel while avoiding deadlocks. In the previous section, this purpose is achieved by composing all the task planning results and searching for the optimal solution in the composed results. It however has a higher computational requirement with an increasing number of parallel decomposition components. In this section, a new strategy is proposed to avoid the high computation while increasing the parallelism of MRS

task planning.

Denote $\Psi_y,\ y = 1, \cdots, Y$ as the generalized description of all the individual (Ψ^s) and concurrent (Ψ^c) SPA. The aim in this section is to synthesize the maximum amount of SPAs Ψ_y with the given robots $\mathcal{R}$. Denote $\prod_{i=1}^{I} \boldsymbol{R}_i$ as the collection of combinations of all robot assignments for subtask automata G_i. Each combination of robot assignments $(\mathcal{R}_1, \cdots, \mathcal{R}_I) \in \prod_{i=1}^{I} \boldsymbol{R}_i$ and corresponding subtask automata $(G_1, \cdots, G_I)$ will result in a topology of robot assignment, which has a level of parallelism quantified by the number of its isolated subgraphs. Select the optimal robot assignment tuple[3] $(\mathcal{R}_1^*, \cdots, \mathcal{R}_I^*) \in \prod_{i=1}^{I} \boldsymbol{R}_i$ that forms the maximum amount of isolated subgraphs $\mathcal{G}_y^* := (\mathcal{V}_y^*, \mathcal{E}_y^*),\ y = 1, \cdots, Y$ in generating the topology of robot assignment. The optimal robot assignment tuple yields the highest level of parallelism. The included concurrent automaton sets $\mathbb{G}_y^*$ and robot sets $\mathbb{R}_y^*$ can synthesize the corresponding individual and concurrent SPA Ψ_y.

Theorem 3. *Given a set of parallel task automata $\{G_i, i = 1, \cdots, I\}$ and each G_i has a set of different choices of robot assignments $\mathcal{R}_i$. A group of individual and concurrent SPA Ψ_y can be synthesized from the robot assignment tuple $(\mathcal{R}_1^*, \cdots, \mathcal{R}_I^*)$, which guarantees the highest level of parallelism in completing the tasks. Furthermore, these individual and concurrent SPA Ψ_y also reduce the computational complexity compared to direct synthesis of the task planning, i.e., $\|_{n=1}^{N} TE_n \times \|_{i=1}^{I} G_i$.*

Proof Completeness: The parallel subtask planning framework groups Y numbers of concurrent automaton sets $\mathbb{G}_y, y = 1, \cdots, Y$. Each $\mathbb{G}_y$ has $\tilde{I}_y$ numbers of subtask automata. As a result, this framework entails a set of parallel processes, each of which can be described by an SPA $\|_{n=1}^{|\mathbb{R}_y|} TE_n \times \|_{i=1}^{\tilde{I}_y} G_i,\ y = 1, \cdots, Y$. Parallel compose these SPA together and obtain the result $\|_{y=1}^{Y} (\|_{n=1}^{|\mathbb{R}_y|} TE_n \times \|_{i=1}^{\tilde{I}_y} G_i)$, which presents all the task planning solutions of the framework. Here, all these subtask automata $G_i, i = 1, \cdots, \tilde{I}_y,\ y = 1, \cdots, Y$ are in parallel, which is equivalent to the global task automaton $\|_{i=1}^{I} G_i$, i.e., $\|_{y=1}^{Y} \|_{i=1}^{\tilde{I}_y} G_i = \|_{i=1}^{I} G_i$. The composition of provided robot transition systems, $TE_n,\ n = 1, \cdots, |\mathbb{R}_y|,\ y = 1, \cdots, Y$, is the same with that of the direct centralized synthesis of global task planning, i.e., $\|_{y=1}^{Y} \|_{n=1}^{|\mathbb{R}_y|} TE_n = \|_{n=1}^{N} TE_n$. Then, the composition result of parallel subtask planning framework is the same as the result of the direct centralized framework, i.e., $\|_{n=1}^{N} TE_n \times \|_{i=1}^{I} G_i$. Therefore, the parallel subtask planning framework can guarantee the completeness of task planning for $G_i,\ i = 1, \cdots, I$.

Parallelism: According to Definitions 18 - 21, the associated automata and robots of each isolated subgraph $\mathcal{G}_y$ form a concurrent (or individual) SPA Ψ_y that can work in parallel with other SPA. The number

[3]Note that the optimal robot assignment has to be an effective assignment, which means the assignment can synthesize a non-empty SPA and provide a task planning solution. In the actual application, the system with a robot assignment may not output a viable task planning solution. Such an assignment will not be considered as an effective one. More details can be found in the experiment section.

of isolated subgraphs that are formed in synthesizing the SPA Ψ_y quantifies the parallelism of task planning. The optimal robot assignment tuple $(\mathcal{R}_1^*, \cdots, \mathcal{R}_I^*) \in \prod_{i=1}^{I} \boldsymbol{R}_i$ generates the maximum amount of isolated subgraphs $\mathcal{G}_y^* := (\mathcal{V}_y^*, \mathcal{E}_y^*),\ y = 1, \cdots, Y$. Therefore, the optimal robot assignment tuple guarantees the highest level of parallelism in completing the tasks.

Computational Complexity: The direct synthesis of task planning, i.e., $\|_{n=1}^{N} TE_n \times \|_{i=1}^{I} G_i$, will have the computation with $\boldsymbol{V}_d := \prod_{n=1}^{N} |S_n| \times \prod_{i=1}^{I} |X_i|$ states and $\boldsymbol{V}_d \times \boldsymbol{V}_d$ edges. In comparison, the parallel subtask planning, i.e., the individual and concurrent SPA Ψ_y, will have the computation with $\boldsymbol{V}_p := \sum_{y=1}^{Y} (\prod_{n=1}^{|\mathbb{R}_y|} |S_n| \times \prod_{i=1}^{\tilde{I}_y} |X_i|)$ states and $\boldsymbol{V}_p \times \boldsymbol{V}_p$ edges. Because $\mathbb{R}_y$, $y = 1, \cdots, Y$ are independent and $|\mathbb{R}_y| \leq N$, then $\prod_{n=1}^{|\mathbb{R}_y|} |S_n| \times \prod_{i=1}^{\tilde{I}_y} |X_i| \leq \prod_{n=1}^{N} |S_n| \times \prod_{i=1}^{\tilde{I}_y} |X_i|$. As a result, $\boldsymbol{V}_p \leq \prod_{n=1}^{N} |S_n| \times \sum_{y=1}^{Y} \prod_{i=1}^{\tilde{I}_y} |X_i| \leq \boldsymbol{V}_d$. In the worst case, the whole group of robots works concurrently for all subtask automata G_i, i.e., $y = Y = 1$, $\mathbb{R}_y \equiv \mathcal{R}$ and $\boldsymbol{V}_p = \boldsymbol{V}_d$. Except this special case, the parallel subtask planning framework requires less computation compared to the direct synthesis process. □

Example 7. Consider the 6 robots $r_1 - r_6$ with their respective capability transition systems as shown in Examples 5 and 6, as well as the associated subtask automata $G_1 - G_4$. Correspondingly, there are different topologies of robot assignment. Take two of the typologies as examples, as shown in Fig. 3.10. Robots belonging to the same subgraph of automata need to coordinate with each other in a decentralized manner to sequentially perform the atomic tasks, while robots in different subgraphs can work in parallel. Denote the state size of the transition system of each robot r_k as $|S_k|$. As a result, the computation complexity of topology 1 is $\boldsymbol{V}_{p,1} = |X_1||X_2||S_1||S_3| + |X_3||S_2||S_5| + |X_4||S_4||S_6|$, while computation complexity of topology 2 is $\boldsymbol{V}_{p,2} = |X_1||S_2| + |X_2||S_1||S_3| + |X_3||X_4||S_4||S_5||S_6|$. On the other hand, the centralized task planning framework takes all the robots as a whole and performs product composition of transition systems with composition of automata. It has computation complexity of $\boldsymbol{V}_d = \prod_{i=1}^{4} |X_i| \prod_{k=1}^{6} |S_k|$. One can easily see that $\boldsymbol{V}_{p,1} < \boldsymbol{V}_d$ and $\boldsymbol{V}_{p,2} < \boldsymbol{V}_d$.

3.6.4 Parallel and Concurrent Execution of MRS Task Planning

According to Prop. 2, a global task automaton G_g can be *loosely decomposable* if it has a subautomaton $\check{G}_{l'}$ with event set $\check{E}_{l'} \subseteq E_g$ satisfying the parallel decomposability. Based on Thm. 3, each resultant set of decomposition components $\{P_i(\check{G}_{l'}), i = 1, \cdots, I_{l'}\}$ can be used to generate its corresponding set of

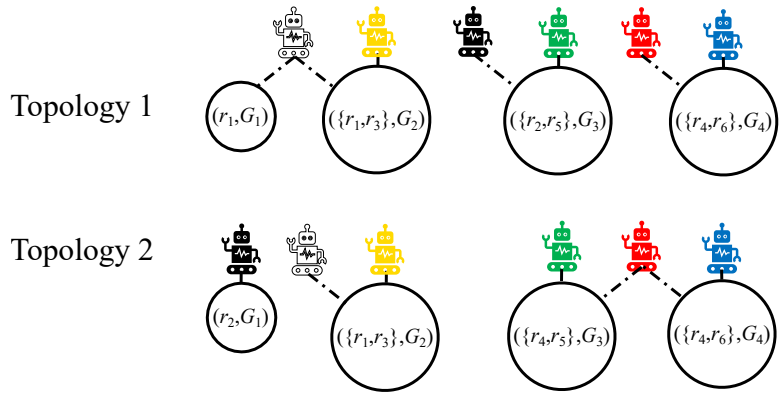


Figure 3.10: Example typologies of robot assignment.

SPA $\{\Psi_y,\ y = 1, \cdots, Y_{l'}\}$. A path (task plan) from each SPA can be as follows Ψ_y

$$\mathbb{X}_y := x_\psi^{(0)} \xrightarrow{act_\psi^{(0)}} \cdots x_\psi^{(\tau)} \xrightarrow{act_\psi^{(\tau)}} \cdots x_\psi^{(\mathtt{T})} \xrightarrow{act_\psi^{(\mathtt{T})}} x_\psi^{(\mathtt{T}+1)}, \tag{3.5}$$

where $x_\psi^{(\tau)} := (\langle s_1^{(\tau)}, \cdots, s_n^{(\tau)}\rangle, x^{(\tau)})$ and $act_\psi^{(\tau)} := \langle \alpha, \mathcal{R}^\alpha\rangle^{(\tau)}$. Denote the labeling of $\mathbb{X}_y$ as $\mathcal{L}_\psi(\mathbb{X}_y) = \mathcal{L}_\psi(x_\psi^{(0)}) \cdots \mathcal{L}_\psi(x_\psi^{(\tau)}) \cdots \mathcal{L}_\psi(x_\psi^{(\mathtt{T})})\mathcal{L}_\psi(x_\psi^{(\mathtt{T}+1)})$, which satisfies the parallel automata (or single automaton) of $\mathbb{G}_y$.

The task plan from each SPA is independent with each other. Therefore, the cost of a task plan can be described as

$$\mathbb{J}(\mathbb{X}_1, \cdots, \mathbb{X}_y, \cdots, \mathbb{X}_Y) = \sum_{y=1}^{Y} \mathbb{J}(\mathbb{X}_y), \tag{3.6}$$

where $\mathbb{J}(\mathbb{X}_y)$ is the cost function of each task plan $\mathbb{X}_y$ from the SPA Ψ_y. Assume the cost of each state and action pair, i.e., $W_\psi(x_\psi^{(\tau)}, act_\psi^{(\tau)})$ is independent from past states and actions. Hence, $\mathbb{J}(\mathbb{X}_y) = \sum_{\tau=0}^{\mathtt{T}} W_\psi(x_\psi^{(\tau)}, act_\psi^{(\tau)})$.

Based on the above process, the minimal cost task plans $\mathbb{X}_y^*$ can be searched in each set of SPA $\Psi_y, y = 1, \cdots, Y_{l'}$. The corresponding minimal cost task planning paths $\mathbb{S}_n^*$ can be obtained for each robot r_n. If the global task automaton G_g has multiple decomposable event-equivalent words composed automata $\check{G}_{l'}$, $l' \in \mathbb{Z}$, the subautomaton $\check{G}_{l'}$ that can generate the lowest cost task plan is selected as the initially synthesized minimal cost solution. Denote the corresponding results of the task plan as $\{\mathbb{X}_y^{**}, y = 1, \cdots, Y_{l^*}\}$, $\check{E}_{l^*} \in \mathbb{E}_g$. Alg. 3 shows the pseudo code for generating the above minimal cost task plan solution given different sets of automaton described subtasks.

Furthermore, the concurrency of implementing the minimal cost task plan can be improved. See the

Algorithm 3 Synthesis of Parallel Task and Motion Plan

Input: $\{\{P_i(\check{G}_{l'}), i = 1, \cdots, I_{l'}\}, \check{E}_{l'} \in \mathbb{E}_g\}, \{TE_n, r_n \in \mathcal{R}\}$ ▷ Prop. 2
Output: Task plan $\{\mathbb{X}_y^{**}, y = 1, \cdots, Y_{l^*}\}$ and $\check{E}_{l^*} \in \mathbb{E}_g$

function OPTIMALPATH($\{\{P_i(\check{G}_{l'}), i = 1, \cdots, I_{l'}\}, \check{E}_{l'} \in \mathbb{E}_g\}, \{TE_n, r_n \in \mathcal{R}\}$)
  for $\check{E}_{l'} \in \mathbb{E}_g$ **do** ▷ Different event-equivalent words composed automaton $\check{G}_{l'}$
    Obtain $\boldsymbol{R}_i$ for $P_i(\check{G}_{l'}),\ i = 1, \cdots, I_{l'}$ ▷ All robot assignments
    for $(\mathcal{R}_1, \cdots, \mathcal{R}_{I_{l'}}) \in \prod_{i=1}^{I_{l'}} \boldsymbol{R}_i$ **do**
      Generate $\mathcal{G}_y, y = 1, \cdots, Y_{l'}$ with all $(\mathcal{R}_i, P_i(\check{G}_{l'}))$ ▷ Topology of robot assignment
      $\{\mathbb{G}_y,\ \mathbb{R}_y,\ y = 1, \cdots, Y_{l'}\} \Leftarrow \mathcal{G}_y, y = 1, \cdots, Y_{l'}$ ▷ Concurrent automaton sets
    end for
    Select $(\mathcal{R}_1^*, \cdots, \mathcal{R}_{I_{l'}}^*)$ and $\{\mathbb{G}_y^*,\ \mathbb{R}_y^*,\ y = 1, \cdots, Y_{l'}\}$ ▷ Optimal robot assignment
    for $y = 1, \cdots, Y_{l'}$ **do**
      $\Psi_y \Leftarrow \mathbb{G}_y^*,\ \mathbb{R}_y^*$ ▷ Individual/Concurrent SPA
      Dijkstra search Ψ_y for $\mathbb{X}_y^*$
    end for
    Obtain $\{\mathbb{X}_y^*, y = 1, \cdots, Y_{l'}\}$ ▷ Optimal task plans for $\check{G}_{l'}$
  end for
  Optimal paths $\{\mathbb{X}_y^{**}, y = 1, \cdots, Y_{l^*}\}, \check{E}_{l^*} \in \mathbb{E}_g$ ▷ Optimal task plans for G_g
end function

following motivation example.

Example 8. Take a path $(\langle s_a, s_f, s_h\rangle, \langle 2, 1\rangle) \xrightarrow{\langle \alpha_b, r_4\rangle} (\langle s_b, s_f, s_h\rangle, \langle 2, 2\rangle) \xrightarrow{\langle \alpha_c, r_5\rangle} (\langle s_b, s_c, s_h\rangle, \langle 3, 2\rangle) \xrightarrow{\langle \alpha_d, r_6\rangle} (\langle s_b, s_c, s_d\rangle, \langle 3, 3\rangle)$ from the concurrent SPA Ψ^c in Fig. 3.9 as an example. Assume the MRS is currently at state $(\langle s_a, s_f, s_h\rangle, \langle 2, 1\rangle)$. Fig. 3.11(a) shows the remaining task plan of the path executed in a sequential manner. In the figure, each robot has its own progress bar, but the task performing states of the three robots need to be synchronous at each step.

Nevertheless, the transition of the red robot r_4 to step $\langle 2, 2\rangle$ can be in parallel with the transition of the green robot r_5 to step $\langle 3, 2\rangle$. The reason is the parallel composition $\mathcal{L}_\psi^c(\langle s_b, s_f, s_h\rangle, \langle 2, 2\rangle) \| \mathcal{L}_\psi^c(\langle s_b, s_c, s_h\rangle, \langle 3, 2\rangle) \supset \{\pi_b\pi_c, \pi_c\pi_b\}$ does not violate the parallel composition of the corresponding task automata $G_3 \| G_4$ of the concurrent SPA Ψ^c. Therefore, the red robot r_4 can work simultaneously with green robot r_5 at steps $\langle 2, 2\rangle$ and $\langle 3, 2\rangle$, which is shown in Fig. 3.11(b). The above concurrent execution process is an enhancement of the sequential task plan. •

Example 8 shows that different robots $r_n \in \mathcal{R}_i \in \mathbb{R}_y^*$ and $r_{n'} \in \mathcal{R}_{i'} \in \mathbb{R}_y^*$ belonging to the same subgraph's robot assignment $\mathbb{R}_y^*$ may work in parallel, where $r_n \notin \mathcal{R}_{i'}$, $r_{n'} \notin \mathcal{R}_i$. Originally, these two robots r_n and $r_{n'}$ have to work sequentially according to the task plan in the SPA Ψ_y. However, a section of events in $G_i \| G_{i'}$ may be satisfied by the two robots in parallel because the simultaneous transition of their states does not conflict with $G_i \| G_{i'}$. Here, $G_i \in \mathbb{G}_y^*$ and $G_{i'} \in \mathbb{G}_y^*$ are the corresponding two automata of

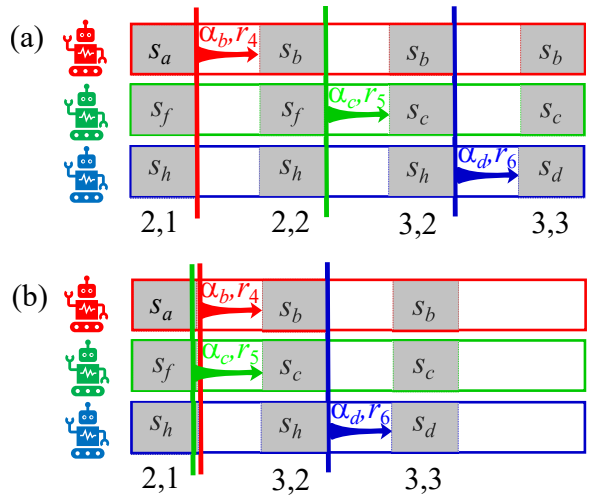


Figure 3.11: (a) Task plan implemented in a sequential manner, (b) task plan achieved in parallel at steps $\langle 2, 2\rangle$ and $\langle 3, 2\rangle$ by implementing α_b and α_c simultaneously. The steps and arrow length do not reflect the actual time.

the SPA Ψ_y. The parallel satisfaction for the section of events will further reduce the coordination difficulty and speed up the multi-robot multi-task process when executing the task plan in SPA Ψ_y in comparison with the centralized task planning framework.

Corollary 1. *Given a task plan* $\mathbb{X}_y := x_\psi^{(0)} \xrightarrow{act_\psi^{(0)}} \cdots x_\psi^{(\tau)} \xrightarrow{act_\psi^{(\tau)}} x_\psi^{(\tau+1)} \cdots x_\psi^{(\mathtt{T})} \xrightarrow{act_\psi^{(\mathtt{T})}} x_\psi^{(\mathtt{T}+1)}$ *from an SPA* Ψ_y*, a section of the task plan starting from a step* τ *(state* $x_\psi^{(\tau)}$*) to a step* $\tau + \Delta$ *(state* $x_\psi^{(\tau+\Delta)}$*) can be executed concurrently if it satisfies (1) the composition results* $\mathcal{L}_\psi(x_\psi^{(\tau)})\| \cdots \|\mathcal{L}_\psi(x_\psi^{(\tau+\Delta)})$ *do not violate the composition of the task automata* $\mathbb{G}_y$ *of* Ψ_y*, and (2) the actions* $act_\psi^{(\tau)}, \cdots, act_\psi^{(\tau+\Delta-1)}$ *can transit simultaneously in* $x_\psi^{(\tau+\Delta)}$*. As a result, the original* $\mathtt{T}$ *steps task plan can be described as a concurrent plan in* $\mathtt{T} - \Delta$ *steps as* $\tilde{\mathbb{X}}_y := x_\psi^{(0)} \xrightarrow{act_\psi^{(0)}} \cdots x_\psi^{(\tau)} \xrightarrow{act_\psi^{(\tau)}, \cdots, act_\psi^{(\tau+\Delta-1)}} x_\psi^{(\tau+\Delta)} \xrightarrow{act_\psi^{(\tau+\Delta)}} \cdots x_\psi^{(\mathtt{T})} \xrightarrow{act_\psi^{(\mathtt{T})}} x_\psi^{(\mathtt{T}+1)}$.

Proof For a section Δ of the path $\mathbb{X}_y$, i.e., $x_\psi^{(\tau)}, \cdots, x_\psi^{(\tau+\Delta)}$, the corresponding sequence of events are $\mathcal{L}_\psi(x_\psi^{(\tau)}), \cdots, \mathcal{L}_\psi(x_\psi^{(\tau+\Delta)})$. The sequence of events can be achieved in parallel if it satisfies that (1) the transition results of concurrent implementation on actions $act_\psi^{(\tau)}, \cdots, act_\psi^{(\tau+\Delta-1)}$ do not violate the subtask automata of Ψ_y, which means all the transitions of parallel composition $\mathcal{L}_\psi(x_\psi^{(\tau)})\| \cdots \|\mathcal{L}_\psi(x_\psi^{(\tau+\Delta)})$ are contained in the composition of the task automata of concurrent SPA Ψ_y (or in the single task automaton of Ψ_y if Ψ_y is an individual SPA); (2) none of the assigned robots at different steps are the same robots so that they can satisfy the corresponding events $\mathcal{L}_\psi(x_\psi^{(\tau)}), \cdots, \mathcal{L}_\psi(x_\psi^{(\tau+\Delta)})$ simultaneously. It implies that actions $act_\psi^{(\tau)}$ and $act_\psi^{(\tau+\Delta-1)}$ can transit simultaneously. Then, the assigned robots in $\mathbb{X}_y$ of this SPA can work concurrently and transit from state $x_\psi^{(\tau)}$ to $x_\psi^{(\tau+\Delta)}$, which can reduce the $\mathtt{T}$ steps sequential task plan $\mathbb{X}_y$ into

$\mathtt{T} - \Delta$ steps. $\square$

Accordingly, the subtask plan $\mathbb{X}_y^{**}$ may have a concurrently executable plan $\tilde{\mathbb{X}}_y^{**}$. The task performing process associated with $\mathbb{X}_y^{**}$ can have higher level of parallelism from step τ to $\tau + \Delta$, but does not violate each concurrent SPA. As a result, the efficiency of multi-robot task performing is improved. The execution cost of the task plan solution does not change either. The generated minimal cost task plan $\mathbb{X}_y^{**}, y = 1, \cdots, Y_{l^*}$, requires the corresponding robots r_n to travel to the designated locations of their assigned atomic tasks in $\mathbb{S}_n^{**}$. The hybrid local motion planner in [56] is utilized to generate the trajectory using the motion transition system in Def. 13.

Remark 6. The initial parallel decomposition, robot assignment and task planning process can generate the task planning solution with the highest level of parallelism and the minimal task performing cost for the MRS. The task planning solution enables each set of robots in the robot assignment set $\mathbb{G}_y^*$ to work independently for the assigned tasks with the minimal cost. Nevertheless, some of the robots assigned with the initial tasks may complete their tasks earlier during the task execution process. They can replace or share tasks with other robots that still have tasks to be completed. These remaining unperformed tasks are reassigned to the robots with no currently assigned task to expedite the task performing process and improve the concurrency of task planning. In addition, the remaining subtask automata may also become decomposable. A redecomposition for the subtask automaton may find new decomposition components to improve the task concurrency and reduce cost. Hence, a task redecomposition and reallocation is enabled among neighboring robots. Thus, a dynamically updated optimal task is obtained in each phase of the parallel decomposition, robot assignment, and task planning. •

3.7 Case Study: Symbolic Task and Motion Planning for MRS Manufacturing Task

A multi-robot experiment for complex manufacturing tasks is designed to demonstrate the MRS parallel task and motion planning framework. Consider robot manufacturing tasks under temporal logic constraints in manufacturing plants where frequent changes of operation sequence are necessary to accommodate for customized products. Heterogeneous robots are utilized to achieve the manufacturing task specifications. More specifically, the parallel decomposition technique decomposes the complex manufacturing tasks into multiple subtasks so that the global manufacturing task can be performed in parallel instead of sequentially.

The robot assignment and task planning process (Alg. 3) synthesizes a task planning solution with the highest level of parallelism for these subtasks. The local motion planner generates robot trajectories that satisfy the liveness and safety specifications, such as reach of the task stations and obstacle avoidance as mentioned in Def. 13.

The Robotic Operating System (ROS) is utilized to build an architecture for the above task and motion planning framework and implement the continuous trajectories of MRS. Task specification redecomposition, replanning and robot reassignment are additionally demonstrated in the overall task performing process. The task planning and performing process is also simulated with varying complexities of task specifications and numbers of robots; then is compared with the centralized task planning strategy to show its advantage in reducing the state size of the SPA, computational time of generating the task plan, level of parallelism, and actual execution time.

3.7.1 Experimental Setup

Consider a manufacturing environment composed of 7 stations labeled as $\mathtt{A} - \mathtt{G}$. Station $\mathtt{A}$ is for providing raw parts so that mobile robots can load parts there and deliver them to the other stations; Station $\mathtt{B}$ is configured with milling machines which can perform common operations such as thickness tapering or hole drilling; Station $\mathtt{C}$ is configured with grinding machines that can perform deburring and finishing operations; Stations $\mathtt{D} - \mathtt{F}$ are the assembling stations; Station $\mathtt{G}$ is for collecting and packing processed products from other stations and may require multiple manufacturing robots to cooperate simultaneously. The 2D view of the manufacturing environment is shown in Fig. 3.4. It is discretized with triangles[4] for the local motion planning. The dark areas represent inaccessible cells that contain static obstacles. Each robot can travel from one triangle cell to its adjacent cells, but is not allowed to enter into cells that are partially or totally taken by obstacles. This allows the local motion planner to generate trajectories satisfying the liveness and safety specifications.

Two Khepera robots r_1, r_2 and two Turtlebot3 robots r_3, r_4 are provided to assist the manufacturing tasks at each station. Each mobile robot can either provide spare parts for the specific stations $\mathtt{A} - \mathtt{C}$ or perform auxiliary operations at stations $\mathtt{D} - \mathtt{G}$. Turtlebot3 is equipped with a LiDAR and can achieve better navigation. Hence, Turtlebot3 is assigned to move among all the stations $\mathtt{A} - \mathtt{G}$, while restricting Khepera among the machining stations $\mathtt{B}, \mathtt{C}$ and related stations $\mathtt{A}, \mathtt{G}$. In addition, the motion of each robot is restricted

[4]The offline triangulation for the working environment can be achieved with the package in the link: `https://www.cs.cmu.edu/~quake/triangle.html`

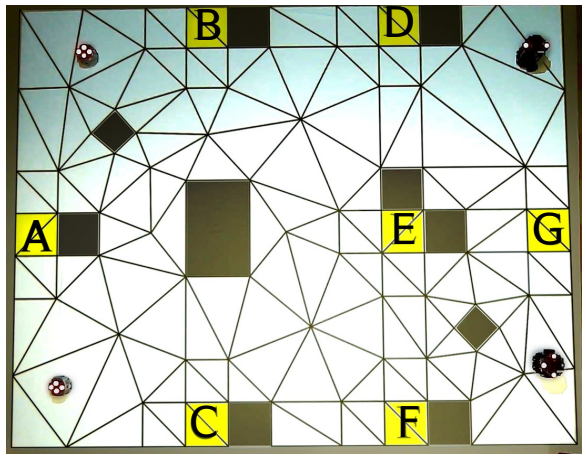


Figure 3.12: Triangulated 2D workspace. The obstacle polygons are over-estimation of the actual size of the obstacles.

between a station $\mathtt{i} \in \{\mathtt{A}, \cdots, \mathtt{G}\}$ and its designated stations $\mathtt{j} \in \Gamma_{\mathtt{i}}$, which can lower the planning complexity of the delivery process. Here, the designated station set $\Gamma_{\mathtt{i}}$ of a station $\mathtt{i}$ is listed in Table 3.1.

i	A	B	C	D	E	F	G
$\Gamma_{\mathtt{i}}$	B, C	A, C, D, E, F	A, B, D, E, F	B, C, E, G	B, C, D, F, G	B, C, E, G	D, E, F, G

Table 3.1: Robot's mobility between each station $\mathtt{i}$ and its designated station set $\Gamma_{\mathtt{i}}$

Then, each task performing state $s_n^{\mathtt{i}}$ abstracts that robot r_n provides parts or performs auxiliary operations at station $\mathtt{i}$. Each state $s_n^{\mathtt{i}}$ can be labeled with a set of atomic propositions $\pi_{\mathtt{i}}$ by $\mathcal{L}_n(s_n^{\mathtt{i}}) = \{\pi_{\mathtt{i}}\}$, where $\pi_{\mathtt{i}}$ corresponds to the satisfaction of the defined manufacturing tasks at each station. An *idle* state s_n^{ϵ} is added for each robot and its labeling is empty, i.e., $\mathcal{L}_n(s_n^{\epsilon}) = \emptyset$. Furthermore, the atomic proposition $\pi_{\mathtt{G}}$ is a cooperative event depending on the type of the packaging at station $\mathtt{G}$. All the other atomic propositions are single events. Finally, the discrete states $s_n^{\mathtt{i}}$ can be used to construct the transition systems TE_n of each robot r_n according to Def. 11, as shown in Fig. 3.13. The action $\alpha^{\mathtt{i}}$ triggers a state of r_n transiting to state $s_n^{\mathtt{i}}$, $\mathtt{i} := \mathtt{A}, \cdots, \mathtt{G}$. The capable states and corresponding costs of each robot r_n are listed in Table 3.2, where "-" means that the robot does not have the capability. The cost of each state $s_n^{\mathtt{i}}$ here is the estimated average time units that each mobile robot takes to complete an auxiliary operation task at the station $\mathtt{i}$.

	$s_{\mathtt{A}}$	$s_{\mathtt{B}}$	$s_{\mathtt{C}}$	$s_{\mathtt{D}}$	$s_{\mathtt{E}}$	$s_{\mathtt{F}}$	$s_{\mathtt{G}}$	s_{ϵ}
r_1/r_2	5.0	10.0	10.0	-	-	-	3.0	1.0
r_3/r_4	7.0	12.0	12.0	12.0	12.0	12.0	5.0	1.0

Table 3.2: Robot capable states and the corresponding average time costs

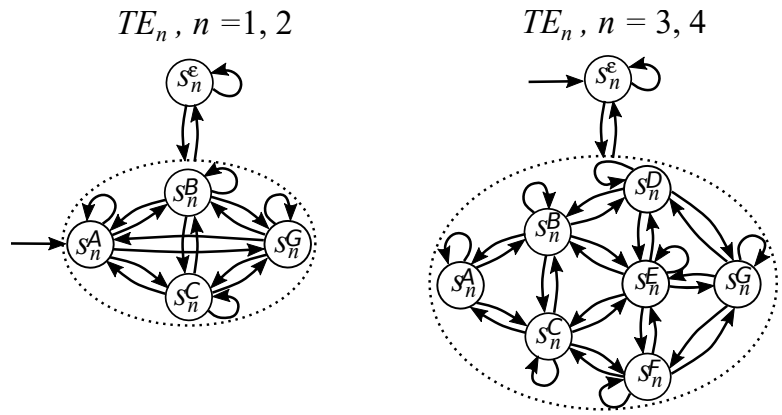


Figure 3.13: Transition system of each robot based on its capabilities and the abstracted environment. Each state $s_n^{\mathtt{i}}$ is labeled with $\pi_{\mathtt{i}}$, $\mathtt{i} = \epsilon, \mathtt{A}, \cdots, \mathtt{G}$. To make better visualization, simplify the graph here by saying each idle state s_ϵ can transit to all the other normal working states inside the dotted circle and vice versa.

3.7.2 Task Specification and Planning Results

(Task specification 1) Consider a complex task specification for the manufacturing process, which requires that (1) the system first provides some raw parts at station A, then finally performs assembling operations with these parts at station F; (2) the system mills the parts at station B, then finally assembles the parts at station D and also finally assembles them at station E; (3) the system needs to finally satisfy each specification of (1) and (2). The above task specifications can be described by an LTL formula $\varphi_1 = \Diamond(\pi_{\mathtt{B}} \wedge \bigcirc(\Diamond\pi_{\mathtt{D}} \wedge \Diamond\pi_{\mathtt{E}})) \wedge \Diamond(\pi_{\mathtt{A}} \wedge \bigcirc\Diamond\pi_{\mathtt{F}})$. Derive the DFA of this LTL task specification, and extract an event-equivalent words composed task automaton $\check{G}_{g_1}$, whose atomic propositions $\mathbb{AP}_{g_1} = \{\pi_{\mathtt{A}}, \pi_{\mathtt{B}}, \pi_{\mathtt{D}}, \pi_{\mathtt{E}}, \pi_{\mathtt{F}}\}$. Let $\check{E}_{g_1} = 2^{\mathbb{AP}_{g_1}}$. The automaton $\check{G}_{g_1}$ can be parallel decomposed into two subtask automaton G_1 and G_2, whose language can be described as $L(G_1) = (\check{E}_{g_1})^*\pi_{\mathtt{A}}(\check{E}_{g_1})^*\pi_{\mathtt{F}}(\check{E}_{g_1})^*$ and $L(G_2) = (\check{E}_{g_1})^*\pi_{\mathtt{B}}(\check{E}_{g_1})^*(\pi_{\mathtt{E}}(\check{E}_{g_1})^*\pi_{\mathtt{D}} + \pi_{\mathtt{D}}(\check{E}_{g_1})^*\pi_{\mathtt{E}})(\check{E}_{g_1})^*$.

Robots $r_1 - r_4$ can satisfy the above two subtasks through an appropriate robot assignment. The robot assignment set $\{\{r_3\}, \{r_4\}, \{r_1, r_3\}, \{r_1, r_4\}, \{r_2, r_3\}, \{r_2, r_4\}, \{r_3, r_4\}, \{r_1, r_3, r_4\}, \{r_2, r_3, r_4\}, \{r_1, r_2, r_3, r_4\}\}$[5] includes all the robot assignments of subtask automaton G_1, while $\{\{r_3\}, \{r_4\}, \{r_3, r_4\}, \{r_1, r_3, r_4\}, \{r_2, r_3, r_4\}, \{r_1, r_2, r_3, r_4\}\}$ presents all the robot assignments of subtask automaton G_2. The robot assignment tuple $(\{r_2, r_3\}, \{r_4\})$ can provide the robot-automaton assignments for MRS to satisfy the task specification with the highest level of parallelism. Then, the robot assignment $\{r_2, r_3\}$ and the subtask automaton G_1 generate an SPA, which has the minimal cost task plan $\mathbb{X}_1^{**} := (\langle s_2^{\mathtt{A}}, s_3^{\epsilon}\rangle, 2) \xrightarrow{\langle \alpha_{\mathtt{F}}, r_3\rangle} (\langle s_2^{\mathtt{A}}, s_3^{\mathtt{F}}\rangle, 1)$. The task plan $\mathbb{X}_1^{**}$ is implemented as follows: robot r_2 first reaches the state $s_2^{\mathtt{A}}$ to initialize

[5]The robot assignments $\{r_3\}$, $\{r_4\}$ output an empty task planning solution. Hence, they will not be the effective assignments.

the SPA process, and r_3 transits to state $s_3^{\mathtt{F}}$ by executing $\alpha_{\mathtt{F}}$. In parallel with the above process, the robot assignment $\{r_4\}$ and the subtask automaton G_2 synthesize an SPA that has the minimal cost task plan $\mathbb{X}_2^{**} := (s_4^{\mathtt{B}}, 3) \xrightarrow{\langle \alpha_{\mathtt{D}}, r_4 \rangle} (s_4^{\mathtt{D}}, 4) \xrightarrow{\langle \alpha_{\mathtt{E}}, r_4 \rangle} (s_4^{\mathtt{E}}, 2)$. It can be seen that robot r_4 satisfies the task plan $\mathbb{X}_2^{**}$ by working in parallel with r_2 and r_3.

The language of the remaining subtask automata become $L(G_1) = (\check{E}_{g_1})^* \pi_{\mathtt{F}} (\check{E}_{g_1})^*$ and $L(G_2) = (\check{E}_{g_1})^* (\pi_{\mathtt{E}} (\check{E}_{g_1})^* \pi_{\mathtt{D}} + \pi_{\mathtt{D}} (\check{E}_{g_1})^* \pi_{\mathtt{E}}) (\check{E}_{g_1})^*$ after robots complete the atomic tasks $\pi_{\mathtt{A}}$ and $\pi_{\mathtt{B}}$. In addition, the redecomposition and replanning request is triggered after each time an atomic task is completed. A successful redecomposition and replanning happens after robots complete atomic tasks $\pi_{\mathtt{A}}$ and $\pi_{\mathtt{B}}$. The remaining task specification G_2 can be redecomposed into two automata $L(G_3) = (\check{E}_{g_1})^* \pi_{\mathtt{D}} (\check{E}_{g_1})^*$ and $L(G_4) = (\check{E}_{g_1})^* \pi_{\mathtt{E}} (\check{E}_{g_1})^*$. At that moment, r_3 implements its action at station F and communicates with r_4. The minimal cost solution is updated as robot r_3 performs the new task plan $\mathbb{X}_1^{**} = (s_3^{\mathtt{F}}, \langle 1, 1 \rangle) \xrightarrow{\langle \alpha_{\mathtt{E}}, r_3 \rangle} (s_3^{\mathtt{E}}, \langle 1, 0 \rangle)$ and r_4 performs the new task plan $\mathbb{X}_2^{**} = (s_4^{\mathtt{D}}, 1)$. The MRS completes the remaining tasks with the updated task planning solution.

Note that the above process only concerns the satisfaction of the task specifications with the task plan. To accomplish the task plan, each robot still need to achieve the physical transition from one station to another in the triangulated environment based on the motion transition system and hybrid controller. The final paths of all the robots are shown in Fig. 3.14 (a).

(Task Specification 2) Given a global task specification for the whole manufacturing process: (1) the system first obtains raw parts from station A, and next performs the milling operation for them at station B. Grinding operation at station C can be either repetitively performed after milling operation or not performed at all. Then, assembling operations are encoded in a complex form, which is (2) finally assembling products at each station of D, E and F in any sequence and then pack them at station G. The corresponding LTL specification is $\varphi_2 = \pi_{\mathtt{A}} \wedge \bigcirc(\pi_{\mathtt{B}} \wedge \bigcirc(\pi_{\mathtt{C}} U (\Diamond(\pi_{\mathtt{D}} \wedge \bigcirc \pi_{\mathtt{G}}) \wedge \Diamond(\pi_{\mathtt{E}} \wedge \bigcirc \pi_{\mathtt{G}}) \wedge \Diamond(\pi_{\mathtt{F}} \wedge \bigcirc \pi_{\mathtt{G}}))))$. The converted automaton $\check{G}_{g_2}$ of this task specification can be obtained and its atomic propositions $\mathbb{AP}_{g_2} = \{\pi_{\mathtt{A}}, \pi_{\mathtt{B}}, \pi_{\mathtt{C}}, \pi_{\mathtt{D}}, \pi_{\mathtt{E}}, \pi_{\mathtt{F}}, \pi_{\mathtt{G}}\}$. Here, the atomic proposition $\pi_{\mathtt{G}}$ is a cooperative event that needs multiple robots to coordinate with each other simultaneously to pack finished products at station G. Let $\check{E}_{g_2} = 2^{\mathbb{AP}_{g_2}}$. All the subautomata that can be extracted from $\check{G}_{g_2}$ and satisfy event-equivalent words according to Def. 14 are initially not parallel decomposable. Thus, this global task automaton needs to be dealt with as a whole. A set of *robot assignments* $\{\{r_3\}, \{r_4\}, \{r_1, r_3\}, \{r_1, r_4\}, \{r_2, r_3\}, \{r_2, r_4\}, \{r_3, r_4\}, \{r_1, r_3, r_4\}, \{r_2, r_3, r_4\}, \{r_1, r_2, r_3, r_4\}\}$ can be generated to achieve the indecomposable task subautomaton according to Def. 17. The optimal robot assignment $\{r_2, r_3\}$ is selected to synthesize the SPA for the task subautomaton. The minimal cost

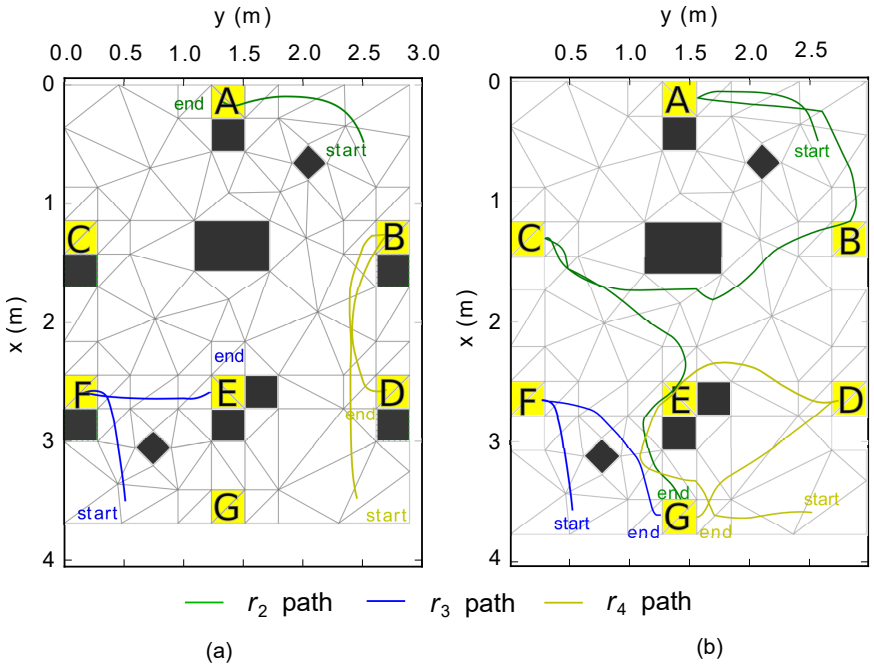


Figure 3.14: The paths of MRS in the working environment for completing the manufacturing tasks.

task plan $\mathbb{X}^{**} = (\langle s_2^{\mathtt{A}}, s_3^{\epsilon}\rangle, 0) \xrightarrow{\langle \alpha_{\mathtt{B}}, r_2\rangle} (\langle s_2^{\mathtt{B}}, s_3^{\epsilon}\rangle, 5) \xrightarrow{\langle \alpha_{\mathtt{C}}, r_2\rangle} (\langle s_2^{\mathtt{C}}, s_3^{\epsilon}\rangle, 11) \xrightarrow{\langle \alpha_{\mathtt{F}}, r_3\rangle} (\langle s_2^{\mathtt{C}}, s_3^{\mathtt{F}}\rangle, 1) \xrightarrow{\langle \alpha_{\mathtt{E}}, r_3\rangle} (\langle s_2^{\mathtt{C}}, s_3^{\mathtt{E}}\rangle, 6) \xrightarrow{\langle \alpha_{\mathtt{D}}, r_3\rangle)} (\langle s_2^{\mathtt{C}}, s_3^{\mathtt{D}}\rangle, 3) \xrightarrow{\langle \alpha_{\mathtt{G}}, r_2, r_3\rangle} (\langle s_2^{\mathtt{G}}, s_3^{\mathtt{G}}\rangle, 2)$ is searched in the SPA.

A redecomposition request for the remaining task specification is sent when each time r_2 or r_3 completes one of its atomic tasks. After robot r_3 enters into state $s_2^{\mathtt{C}}$, three new subtask automata are obtained by decomposing the automaton of remaining task specification $\varphi_2' = \Diamond(\pi_{\mathtt{D}} \wedge \bigcirc \pi_{\mathtt{G}}) \wedge \Diamond(\pi_{\mathtt{E}} \wedge \bigcirc \pi_{\mathtt{G}}) \wedge \Diamond(\pi_{\mathtt{F}} \wedge \bigcirc \pi_{\mathtt{G}})$. The corresponding languages of the three new subtask automata are $L(G_1) = (\check{E}_{g_2})^* \pi_{\mathtt{D}} (\check{E}_{g_2})^* \pi_{\mathtt{G}} (\check{E}_{g_2})^*$, $L(G_2) = (\check{E}_{g_2})^* \pi_{\mathtt{E}} (\check{E}_{g_2})^* \pi_{\mathtt{G}} (\check{E}_{g_2})^*$, and $L(G_3) = (\check{E}_{g_2})^* \pi_{\mathtt{F}} (\check{E}_{g_2})^* \pi_{\mathtt{G}} (\check{E}_{g_2})^*$. Robots r_1 and r_4 are the robots without any assigned tasks and robot r_4 has the capability of assisting r_3 for its assigned tasks. Then, the SPA is synthesized from the parallel composition $TE_3 \| TE_4$ and the unperformed tasks of r_3, i.e., $P_3(\mathcal{L}_{\psi}^c(\mathbb{X}^{**}))$. The generated SPA is a single path automaton and gives the reassigned task plan for r_3 and r_4. The parallel execution format of reassigned task plan $\tilde{\mathbb{X}}^{**}$ includes $(\langle s_2^{\mathtt{C}}, s_3^{\mathtt{F}}\rangle, 1) \xrightarrow{\langle \alpha_{\mathtt{G}}, r_2, r_3\rangle} (\langle s_2^{\mathtt{G}}, s_3^{\mathtt{G}}\rangle, 2)$ and $(s_4^{\mathtt{E}}, 6) \xrightarrow{\langle \alpha_{\mathtt{D}}, r_4\rangle} (s_4^{\mathtt{D}}, 3) \xrightarrow{\langle \alpha_{\mathtt{G}}, r_4\rangle} (s_4^{\mathtt{G}}, 2)$ according to Cor. 1. Each robot then achieves the generated task plans with the paths shown in Fig. 3.14 (b).

3.7.3 Scalability and Computation Evaluation

This section evaluate the scalability and computational complexity of the proposed framework with respect to the number of atomic tasks and robots. A computer with Intel Core i5 2.3 GHz processor and 8 GB RAM is used to run the algorithm. The results are listed in Table 3.3. The basic task specification is $\varphi_z = \Diamond(\pi^z_{\mathtt{B}} \wedge \bigcirc(\pi^z_{\mathtt{B}}\ U\ \pi^z_{\mathtt{E}})) \wedge \Diamond(\pi^z_{\mathtt{C}} \wedge \bigcirc(\pi^z_{\mathtt{C}}\ U\ \pi^z_{\mathtt{F}}))$, $z \in \mathbb{Z}$, which is synthesized from four atomic tasks in a similar workspace as shown in Fig. 3.4. Increase the complexity of task specification through integrating more φ_z into the LTL formula. The corresponding task performing environment is a workspace composed of multiple subspaces, each of which is the same as the one shown in Fig. 3.4. Thus, a general form for the task specification can be described as $\varphi_{gz} = \bigwedge_{z=1}^{Z} \varphi_z$, $Z \in \mathbb{Z}$, where Z is the total number of subspaces and corresponds to Column 1 in Table 3.3. The converted automaton and decomposition results of the LTL are shown in Column 2 of Table 3.3. A varied numbers of robots are also provided for each task specification, which corresponds to Column 3 of Table 3.3. The transition system of each robot is the same as TE_3 in Fig. 3.13 and all the provided robots in each row are assigned with atomic tasks. The corresponding robot assignment is shown in Column 4. The provided robots and their configurations can demonstrate not only the scalability of the task planning framework regarding the robot amount, but also the influence of robot assignment on the computational complexity and task execution efficiency. The subtask automata are assigned with corresponding robots for the task planning process. The corresponding computational complexity and task execution efficiency are represented with the resulting state size of the SPA (see Column 5), average runtime of generating the task plan (Column 6), level of parallelism (Column 7), and the average time required to complete the tasks (Column 8). Compare the above four results (Columns 5 - 8) of the framework with those of the centralized task planning (values in the bracket of Columns 5 - 8), which directly synchronizes MRS transition system $\|_{n=1}^{N} TE_n$ with the global task automaton.

The evaluation results show that the state size of SPA and runtime of generating task plan increase exponentially with respect to the increasing robot amount and complexity of task specification in the centralized framework. “N/A” means that the runtime is too long to be available. The task planning framework has to additionally consider the runtime for the parallel decomposition of task specification. Even though the computation of parallel decomposition based task planning needs the additional computational cost, it grows at a much slower speed compared to the centralized one. In addition, the decentralized SMP framework has the lowest computation for each task specification only under the particular robot assignments (see the highlighted rows with bold fonts in Table 3.3), which provides the minimum amount of robots for each

decomposition and result in the highest level of parallelism. Robot assignment with too few robots (see the rows before the highlighted ones in each task specification case in Table 3.3) can result in more concurrent SPA in the proposed framework, which approaches to the centralized strategy. In the worst-case scenario that only one robot is assigned to the task specification, the framework becomes the centralized fashion. Robot assignment with too many robots can result in unnecessarily large-size SPA (see the rows after the highlighted ones in each task specification case in Table 3.3). In conclusion, the proposed parallel decomposition and robot assignment strategy can help to reduce the computation of the task planning.

The parallel decomposition based task planning framework also improves the level of parallelism. Robot assignment with more robots can result in a higher level of parallelism in each task specification case (see the rows at and before the highlighted ones in each task specification case in Table 3.3). That is because more robots can work simultaneously and satisfy the subtask automata in parallel. However, the level of parallelism at most equals to the number of decomposition components (see the rows after the highlighted ones in each task specification case in Table 3.3). The corresponding average time of completing the task specification is reduced as well compared with the centralized strategy. Their changes are consistent with the level of parallelism. Therefore, the parallel decomposition based robot assignment and task planning framework can greatly improve the task performing efficiency.

There are several existing representative MRS decentralized task planning frameworks [20, 45, 78], as mentioned in the introduction. Both the parallel decomposition based task planning framework and these existing frameworks aim to reduce the computation in the MRS task planning by decomposing the task specification into smaller pieces. However, every framework requires the temporal logic task specification to satisfy its defined property before applying the decentralized computation for the task planning. For example, the parallel decomposition based task planning framework requires the task specification to satisfy the parallel decomposability; and work in [20] requires the trace-closedness property of a task specification under its provided robot configuration. A task specification can hardly satisfy both of the required properties. The situation applies to other frameworks. That implies every task planning framework commonly can not deal with any other task planning frameworks' task specifications. Hence, it is difficult to quantitatively compare these frameworks and conclude that one strategy provides more advantages than others. Nevertheless, the results of the experiments and comparison in Table 3.3 show the uniqueness of the work by presenting the task specifications satisfying the parallel decomposability that others may not achieve.

Z	Automaton	Robot #	Robot Assignment	SPA State size	Average runtime	Parallelism	Tasking time
1	State size: 9, (G_1: 3 states, G_2: 3 states)	1	$(r_1, \{G_1, G_2\})$	72	74 ms	1	113s
		2	$(r_1, G_1), (r_2, G_2)$	**48 [576]**	**72 ms [173 ms]**	**2 [1]**	**54s [107s]**
		4	$(\{r_1, r_2\}, G_1)$, $(\{r_3, r_4\}, G_2)$	384 [36864]	152 ms [1.85 s]	2 [1]	56s [110s]
2	State size: 81, (G_i: 3 states, $i = 1, \cdots, 4$)	2	$(r_1, \{G_1, G_2\})$, $(r_2, \{G_3, G_4\})$	144 [5184]	431 ms [3.27 s]	2 [1]	112s [220s]
		4	(r_n, G_i)**,** $n = i$	**96 [331776]**	**416 ms [1950.018 s]**	**4 [1]**	**54s [208s]**
		8	$(\{r_n, r_{n'}\}, G_i)$, $n = 2i - 1, n' = 2i$	768 [1.36×10^9]	580 ms [N/A]	4 [1]	56s [214s]
3	State size: 729, (G_i 3 states, $i = 1, \cdots, 6$)	2	$(r_1, \{G_1, G_2, G_3\})$, $(r_2, \{G_4, G_5, G_6\})$	432 [46656]	6.973 s [211 s]	2 [1]	N/A
		4	$(r_1, G_1), (r_2, G_2)$, $(r_3, \{G_3, G_4\})$, $(r_4, \{G_5, G_6\})$	192 [2.99×10^6]	6.764 s [N/A]	4 [1]	N/A
		6	(r_n, G_i)**,** $n = i$	**144** [1.91×10^8]	**6.751 s** [N/A]	**6** [1]	**N/A**
		8	(r_n, G_i), $n = i \leq 4$ $(\{r_5, r_6\}, G_5)$, $(\{r_7, r_8\}, G_6)$	480 [1.22×10^{10}]	6.828 s [N/A]	6 [1]	N/A
		12	$(\{r_n, r_{n'}\}, G_i)$, $n = 2i - 1, n' = 2i$	1152 [5.0×10^{13}]	6.923 s [N/A]	6 [1]	N/A
4	State size: 6561, (G_i: 3 states, $i = 1, \cdots, 8$)	2	$(r_1, \{G_1, \cdots, G_4\})$, $(r_2, \{G_5, \cdots, G_8\})$	1296 [419904]	1906.107 s [4.65 hr]	2 [1]	N/A
		4	$(r_1, \{G_1, G_2\})$, $(r_2, \{G_3, G_4\})$, $(r_3, \{G_5, G_6\})$, $(r_4, \{G_7, G_8\})$	288 [2.69×10^7]	1905.675 s [N/A]	4 [1]	N/A
		8	(r_n, G_i)**,** $n = i$	**192** [1.10×10^{11}]	**1905.658 s [N/A]**	**8 [1]**	**N/A**
		12	(r_n, G_i), $n = i \leq 4$ $(\{r_{n'}, r_{n''}\}, G_{i'})$, $i' = 5, \cdots, 8$, $n' = 2i' - 1$, $n'' = 2i'$	864 [4.51×10^{14}]	1905.801 s [N/A]	8 [1]	N/A
		16	$(\{r_n, r_{n'}\}, G_i)$, $n = 2i - 1, n' = 2i$	1536 [1.85×10^{18}]	1905.864 s [N/A]	8 [1]	N/A

Table 3.3: Scalability, computation and execution comparison between the parallel decomposition based task planning framework and centralized task planning.

3.8 Conclusion

This chapter presented a top-down framework for the parallel task and motion planning of MRS to achieve a global task specification with automaton theories. We first introduced an iterative parallel decomposition algorithm and its enhanced version to decompose the global specification. The decomposition components were a unique set of smallest parallel subtask automaton and each component was assigned a set of heterogeneous robots. A maximum amount of individual and concurrent SPA were then synthesized with these subtask automata and the capability transition systems of the assigned robots. Each SPA provided a minimal cost task plan for the MRS and all the task plans were executed in parallel. The task planning process provided higher level of parallelism task plans for MRS compared with the centralized approach that directly synthesizes task plans with the global task automaton and robot transition systems. The parallel task planning process was also proved to be more computationally efficient compared to the centralized approach. Furthermore, dynamic concurrent execution was performed for the task plan from each parallel SPA in order to improve the concurrency of the task performing process.

Chapter 4

Bayesian-based Trust Model for Human Multi-Robot Collaborative Motion Tasks

4.1 Introduction

In human-robot collaborative task performing, human's trust in robot describes human's willingness to collaborate with the robot at the risks of robot reliability in uncertain situations [26, 39, 91]. Trust is an important determinant of the human's acceptance of robotic system performance, given that the robot is of integrity and has good intent in collaborating with the human. A robot's gaining an appropriate level of trust from a human can reduce the human stress and cognitive workload during the collaboration [26, 3].

Despite the recent surge in human-robot trust research, many works remain at a descriptive level [39, 21]. These works analyze impacting factors of human trust in robotic systems with different experimental designs in the corresponding scenarios. It is not enough to achieve the planning and control of robot behaviors through trust analysis. Quantification of trust can help explain the trustworthiness of robot behaviors and guide the robot to gain more trust from the operator during human-robot collaboration. However, the human's trust in a robot is often latent and complex to quantify [26]. It is challenging to construct a computational model to capture the temporal nature of human trust in robots. Hence, it is significant for a human-robot collaborative system to quantify the trust with a computational model.

This chapter focuses on the interpretability of causality between trust and its impacting factors in a human's decision-making process. We assume that the human's trust in each robot is a continuous-valued

time series data. Then, we build a linear state space (LSS) model to capture the human's trust in all the robots during a human-MRS collaborated motion task. The human-MRS collaborated motion task is deployed in an offroad environment, where a group of ground robots accomplishes the motion task and is subject to the influence of the environment's characteristics. Compared with the time series trust models in [74, 75], the LSS model takes into account both the impacting factors of the environment and the uncertainties of the trust evaluation. The LSS model also considers the unobservable property of human trust during the collaboration process. It takes the human's feedback as the observations of human trust instead of directly taking it as the trust value. In addition, we provide a generalized state-space equation in our LSS model to capture the inter-robot trust influence for the human-MRS collaboration compared with [98, 91, 101, 55]. We take the LSS trust model that does not consider human's trust influence among robots as a baseline. Then, we compare the model that considers the inter-robot trust influence with the baseline regarding the prediction accuracy.

We use Bayesian inference to estimate the parameters of the LSS model. The Bayesian inference method estimates the posterior probability distribution of the trust model parameters based on the observations' likelihood and a prior belief of the model parameters. The obtained posterior probability distribution of the trust model parameters can then be taken as the model parameters' prior belief as more observations become available. The iterated updating of the posterior distribution of the trust model parameters brings convenient computation and requires less data under the LSS model. In addition, it is challenging to estimate the latent variables in the LSS model. In this chapter, we utilize Kalman filter and smoother with a forward filtering and backward sampling (FFBS) algorithm to sample the unobserved trust values.

We also aim to obtain the optimal path for the human-MRS collaborated motion task based on the estimated LSS model parameters. Traditionally, people first design a standard sequential experiment to obtain the LSS model parameters and then predict the optimal path with the trained LSS model. The procedure of the standard experiment typically has a predetermined sequence of trials on a set of paths of environment. This chapter relies on the Bayesian optimization strategy to design the experiment (see Fig. 4.1). That is we use the up-to-date estimated LSS trust model parameters to plan the optimal path and allocate this path for the human-MRS to perform the subsequent trial; the estimation of the LSS model and planning for the optimal path are sequentially iterated within the environment. The Bayesian optimization strategy is built on the Bayesian inference based trust model estimation. It reduces the unnecessary trials on the paths that may not be valuable to observe the robot behaviors. The strategy can improve the cost performance, such as operator's perceived workload, usability, and situational awareness, of deploying the human-MRS trials and generate the optimal path more cost-effectively than the standard experimental design.

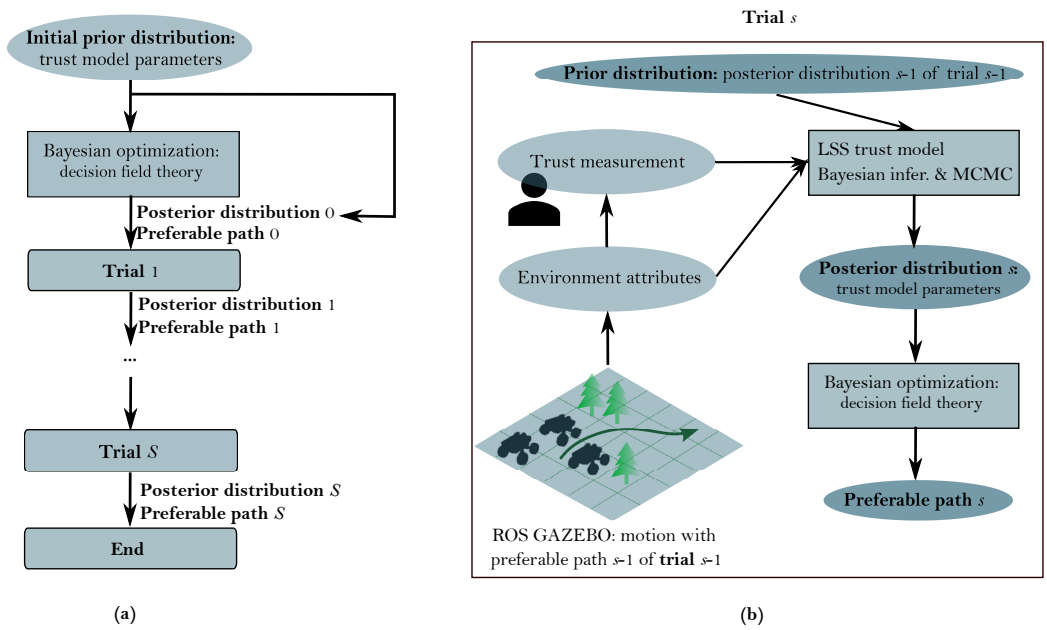


Figure 4.1: Architecture of the Bayesian optimization based trust model for human-MRS collaborative motion task. The left figure describes the overall procedure of the sequential experiment of BOED. The right figure presents the detailed processes of estimating the trust model parameters' posterior distribution and obtaining the preferable path in trial s.

The organization of the rest of the chapter is as follows. Chapter 4.2 provides the preliminaries and problem setup. Chapter 4.3 introduces our computational trust model for human-MRS collaboration with Bayesian inference and MCMC. Chapter 4.3 discusses the exploration of the preferable path with the BOED. Chapter 4.5 demonstrates the overall strategy with a case study on human-MRS bounding overwatch tasks in offroad environments. Chapter 4.6 analyzes the experimental results of the case study. Chapter 4.7 concludes the work.

4.2 Preliminaries and Problem Setup

4.2.1 Human's trust in robots and MRS motion planning

We give the formal definition of human's trust in ground robot under a human-MRS collaboration scenario according to [39]. That is, trust describes a human's willingness to accept ground robot-produced paths and motion behaviors so that the human can assign tasks to the robots and utilize the benefit of robotic systems. The impacting factors that can affect human's trust in a ground robotic system can be generally concluded into: (1) the ground robot characteristics and capability, (2) the terrain environment and human-

MRS teaming formalism, (3) human operator psychological, physical states and attributes.

This chapter aims to estimate the human's trust in a ground MRS with the above impacting factors and explore for the optimal path for the MRS in an iterated process. We present a general process to achieve the objective as follows, (1) the system learns the human decision-making mechanism, i.e., human's trust model, based on the related impacting factors and operator's interaction history with the MRS; (2) the system computes the trustworthiness of all the paths with the human's trust model and the motion planner generates the probabilistic optimal path for the human-MRS to perform the motion task in the environment; (3) the system keeps updating the human's trust model and generating the optimal path for the MRS to navigate. The workflow of the above process is summarized in Fig. 4.2.

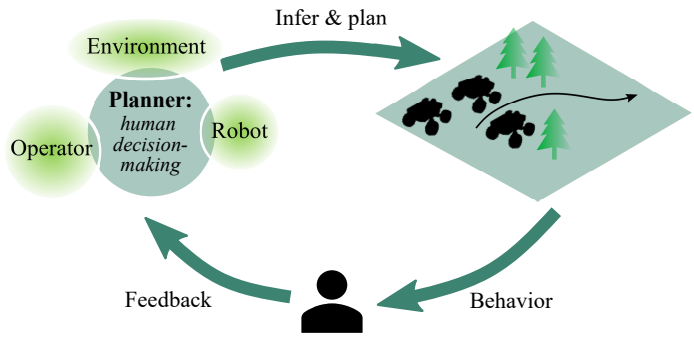


Figure 4.2: Trust-based motion planning in human-MRS collaboration

Among all the different impacting factors of trust, we focus on the influence of the offroad terrain environment on the human's trust in the ground robotic system. We reply on the robots' perception of terrain environment attributes, such as slope, visibility, etc., to describe the influence. Then, robots' situational awareness, such as traversability and line of sight, can be the two representative perceptions of environment attributes. Here, the traversability describes the capability of a robot to reside over a terrain region under an admissible state wherein it can enter given its current state. This capability can be quantified with the kinematic constraints of the vehicle model in a terrain. Works [58, 9, 99, 68] extract traversability as a cost with the geological information of terrain and utilize the cost to perform motion planning for the unmanned vehicles. The line of sight describes the unobstructed vision from an observer to the target in the motion. One can estimate the line of sight with the observer's viewshed which is the visible geographic area with a sensor from a specific location. Works [51, 7, 97] conduct the motion planning for unmanned surface vehicle with the viewshed information. However, a path with good traversability may not be characterized with a superior line of sight in the motion process and vice versa. It is necessary to generate a path that can take care of both traversability and line of sight so that the robots can achieve the best performance with their inherent

mobility and sensing capability. Therefore, a computational trust model can be meaningful in including the traversability and line of sight information in the human-MRS collaboration.

4.2.2 Problem Setup

Every form of coordination among robot members has a specific motion mechanism and the associated decision-making mechanism will be distinct. This chapter exemplifies the human's trust in an MRS coordination scenario where two subteams of robots navigating abreast in an offroad environment. The robot members of each subteam coordinate in a line formation as usually adapted in infantry platoon, convoy, etc.[1] In addition, one subteam that contains all the autonomous robots always moves ahead of the other human-operated subteam and inspects the environment to avoid the potential risks, as shown in Fig. 4.4. The human operator controls the manned subteam following the autonomous one. Therefore, it is crucial to estimate the trustworthiness of the autonomous subteam's behaviors and protection in the above navigation.

Next, we illustrate the trust causality of MRS in the motion process. We assume that the human operator's psychological and physical states keep stable, and the MRS' mechanical characteristics and sensor functions are also consistent during the task performing. Then, the robots' situational awareness in the environment will be the main factors influencing the human's trust if the above human-MRS teaming formalism keeps constant. Assume the autonomous subteam has I robots $r_i,\ i = 1, \cdots, I$. Denote the human operator's trust in each robot r_i at a time step k to be x_i^k, and the situational awareness of each robot at time step k to be $\mathbf{z}_i^k = [z_{i,1}^k, \cdots, z_{i,m}^k, \cdots, z_{i,M}^k]^\top$, where time step $k = 1, \cdots, K$ and each attribute $z_{i,m}^k,\ m = 1, \cdots, M$ can be robot r_i's traversability, or line of sight and so on in the terrain. According to the previous section, we consider the human's trust x_i^k in each robot r_i to be affected by its situational awareness $\mathbf{z}_i^k$. Furthermore, because each robot $r_i,\ 2 \leq i \leq I$ follows a preceding robot r_{i-1}, the human's trust x_i^k is additionally affected by the trust x_{i-1}^k of its preceding robot. Finally, the human's trust x_i^k also has a temporal effect, i.e., humans may make decisions based on their memories of the previous trust x_i^{k-1}. Therefore, we set the trust x_i^k to be affected by the previous trust x_i^{k-1}. A visual description of the causality between the trust x_{i-1}^k and situational awareness $\mathbf{z}_i^k$ is shown in Fig. 4.3. Ideally, we can measure the human's trust x_i^k in every robot r_i and then infer the trust model parameter. However, it is often challenging to conduct an accurate and robust measurement for human trust by referring to the human psychological and physical state with any sensor. Therefore, we consider the actual trust value x_i^k to be a latent variable (hidden state) and do not observe it directly. We develop an human-computer interface for the human operator to provide the trust change

[1] https://www.presby.edu/doc/military/FM3-21-8.pdf

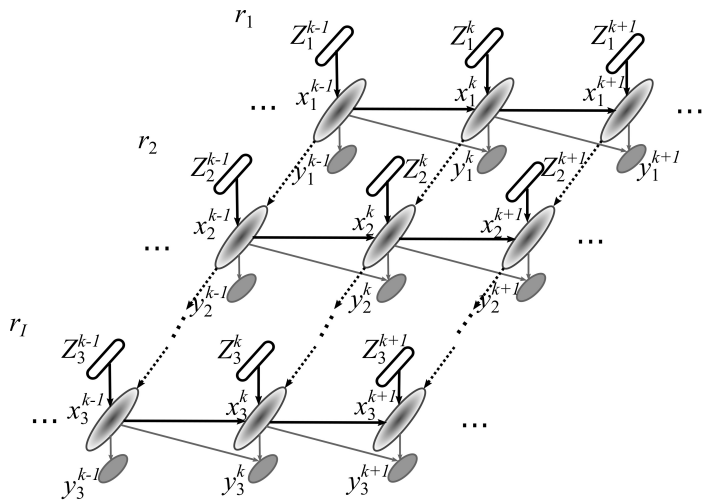


Figure 4.3: The causality graph of the computational trust model in human-MRS collaboration.

$y_i^k = x_i^k - x_i^{k-1}$ at the moment of time step k and take the trust change as the observation for the system.

Then, we can derive computational model between trust and robot situational awareness based on the above causal relationship.

Finally, we can formulate our problem as follows,

Problem of Interest 3. Given a go-to-goal human-MRS collaborative motion task, (1) design an HRI system to dynamically learn the trust-based decision-making mechanism that the human set in mind by referring to the causality in Fig. 4.3; (2) find the optimal path for the human-MRS, which approaches the human-like decision-making and gains the highest level of trust from the human operator.

4.3 Related Work

There have been several computational trust models developed recently in human single-robot collaboration [98, 74, 18, 75, 3, 2, 8]. According to the structure of the trust models, they can be generally categorized into the discrete-valued partially observable Markov decision process (POMDP) models and continuous-valued time series models.

The discrete-valued POMDP models generally formulate the state evolution of trust as a Markov chain. Chen et al. in [18] propose a POMDP to improve the performance of a table-clearing task in a human-manipulator collaborated team. The work approximates human's trust value with the HRI history; and composes the environment state with the human's trust value to be the state of the POMDP. The performance-

centric model uses the estimated trust value to decide the upcoming actions of the human-manipulator. In the ground vehicle scenario, works [3, 2] integrate a human's trust and workload to be a POMDP state and use the POMDP to capture the dynamic change of human's trust in reconnaissance missions. The model aims to trade off the operator workload and the transparency, i.e., the amount of information provided to the human. Overall, the discrete-valued POMDP models use a large amount of data to capture the dynamics of the trust, but they lack the interpretability regarding the influence of impacting factors on the trust change.

The continuous-valued time series models commonly describe a human's trust evolution with a linear equation and an auto-regression term captures the influence of the previous trust state on the current trust state. Sadrfaridpour et al. in [74] develop a first-order Autoregressive Moving Average (ARMA) model to capture the trust variation in their human-manipulator collaboration scenario. The human trust model is formulated as a constraint function of optimal control during the manipulator's pickup task. Saeidi et al. in [75] utilize the weighted human and robot performance to build a linear trust model, where the robot performance is captured with an Auto-regression with extra inputs (ARX) model. The hierarchical time series model reduces the human operator's workload in the teleoperated UVA task by using trust as a metric to allocate human autonomy tasks. Xu et al. in [98] developed an Online Probabilistic Trust Inference Model (OPTIMo) to capture the causality between robot performance and human's trust in a human-UAV supervisory collaborative task. This time series model obtains the maximum likelihood estimate of trust model parameters based on a 2-step temporal Bayesian network. Azevedo et al. in [8] utilize a linear time invariant state space model to describe the evolution of operator's trust with respect to the autonomous driving systems' sensor behaviors. Their trust estimation framework can successfully compute the trust based on the interactions between the drivers and autonomous driving systems. These continuous-valued time series models have good interpretability and can be integrated well into the control of robot behaviors. However, these performance-centric trust models fail to capture the influence from environmental attributes in inferring the human's trust. More impacting factors, such as environment and human characteristics, remain to be investigated for the human-robot collaboration.

Furthermore, multi-robot systems (MRS) can accomplish more complex tasks with two or more robots and have produced a broad set of applications. The presence of a human operator in an MRS can guarantee the safety of the task performing and is necessary for many scenarios [62, 91, 63, 55]. In this circumstance, human operators can be subject to heavier stress and cognitive workload in collaboration with the MRS than in the single robot scenario. However, there are seldom computational trust models developed for the human-MRS collaborative task except [62, 63]. Nam et al. in [62, 63] develop a human-

robotic swarm trust model for a search mission. The trust model composes human's trust in a swarm and swarm physical characteristics as a state of Markov decision process (MDP) and captures the dynamics of the states with inverse reinforcement learning. Though the model can evaluate the human's trust in the swarm, the direct causality between trust and swarm characteristics remains unknown. The provided assistance for the human-swarm environment exploration lacks interpretability regarding the underlying human decision-making mechanism. In addition, though the discrete-valued MDP models can accurately predict the human's trust, they tend to overfit the human decision-making process's model parameters.

Our previous works [101, 91] investigate human's trust in every robot of an MRS with a linear time series model for a temporal logic described motion planning task. We build a Dynamic Bayesian Network to capture the causality of human trust in the MRS motion task. Similarly, Mahani et al. in [55] develop an Input-Output Hidden Markov Model (IOHMM) to describe the trust evolution of multi-UVA rescue task. The work utilizes expectation maximization (EM) algorithm to estimate the parameters of the discrete-valued time series model. The above trust models can capture human trust dynamics in all the individual robots of MRS. They also have good interpretability regarding the causality of trust. However, there could exist trust influence between each two robot members in the MRS. These computational trust models of MRS do not consider the explicit trust causality between different robots.

4.4 Time-series Trust Model of Human Multi-Robot Collaboration

According to the summarized cognitive models in [16], the human's decision-making under risks and uncertainty mainly depends on their attention to different attributes of events. A weighted combination of an event's attributes can be used as the utilities of a human's decision-making, where each weight represents the allocated attention. Then, we can use an LSS model to combine the MRS situational awareness and interpret the trust-based decision-making in the human-MRS collaborative motion tasks.

On the basis of the causality in Fig. 4.3, we use a linear state-space equation to describe the general relationship between the pairs among MRS situational awareness $\mathbf{Z}_{1:I}^k = \left[\mathbf{z}_1^k,\ \cdots,\ \mathbf{z}_I^k\right]^\top$, human trust $\mathbf{x}_{1:I}^k = \left[x_1^k,\ \cdots,\ x_I^k\right]^\top$ and human feedback $\mathbf{y}_{1:I}^k = \left[y_1^k,\ \cdots,\ y_I^k\right]^\top$. The state space equations are as

follows,

$$\mathbf{x}_{1:I}^{k} = \boldsymbol{B}_0 \mathbf{x}_{1:I}^{k-1} + \sum_{m=1}^{M} \boldsymbol{B}_m \mathbf{z}_{1:I,m}^{k} + \mathbf{b} + \boldsymbol{\epsilon}_w^k, \tag{4.1}$$

$$\mathbf{y}_{1:I}^{k} = \mathbf{x}_{1:I}^{k} - \mathbf{x}_{1:I}^{k-1} + \boldsymbol{\epsilon}_v^k, \tag{4.2}$$

where the $I \times I$ coefficient matrix $\boldsymbol{B}_0$ is the autoregression term and discounts the previous trust $\mathbf{x}_{1:I}^{k-1}$. It captures the temporal nature of the human trust. Each of the $I \times I$ coefficient matrices $\boldsymbol{B}_m,\ m = 1, \cdots, M$ is the dynamic feature term and quantifies the weight of robots' mth column attribute $\mathbf{z}_{1:I,m}^{k} = [z_{1,m}^{k}, \cdots, z_{I,m}^{k}]^\top$ in situational awareness matrix $\mathbf{Z}_{1:I}^{k}$. The constant vector $\mathbf{b}$ describes the unchanging bias of human's trust in the MRS. The residue $\boldsymbol{\epsilon}_w^k$ is a zero-mean process noise and follows a multivariate normal distribution, i.e., $\boldsymbol{\epsilon}_w^k \sim N(0,\ \Delta_w)$. The residue $\boldsymbol{\epsilon}_v^k$ is a zero-mean observation noise $\boldsymbol{\epsilon}_v^k \sim N(0,\ \Delta_v)$ during the measurement of the trust change $\mathbf{y}_{1:I}^{k}$.

In the human-MRS collaboration, we expect to analyze the trust influence of the preceding robot on the succeeding robot, i.e., the causality labeled with dotted arrows in Fig. 4.3. In Subsec. 4.4.1, we introduce trust model without considering the influence of human's trust in each preceding robot r_{i-1} on that of its succeeding robot r_i. In Subsec. 4.4.3, we capture the influence of human's trust in each preceding robot r_{i-1} on that of its succeeding robot r_i under a line formation of MRS.

4.4.1 Computational trust model without inter-robot trust influence

For the sake of illustration simplicity, we first introduce the linear dynamic trust model without the influences from each preceding robot r_{i-1} on its succeeding robot r_i. We quantify the relation between trust $x_i^k \in \mathbf{x}_{1:I}^{k}$ and the situational awareness $z_{i,m}^{k} \in \mathbf{z}_{1:I,m}^{k}$ of individual robot r_i at time step k. The linear dynamic model for any individual robot r_i becomes

$$x_i^k = \beta_0 x_i^{k-1} + \sum_{m=1}^{M} \beta_m z_{i,m}^{k} + b + \epsilon_{w,i}^{k}, \tag{4.3}$$

where coefficient β_0 is the weight for the previous trust x_i^{k-1}, coefficients $\beta_1, \cdots, \beta_M$ are the weights for the situational awareness $z_{i,m}^{k},\ m = 1, \cdots, M$, the term b is the constant bias, and the zero-mean residue $\epsilon_{w,i}^{k} \sim N(0,\ \delta_w^2)$. Given all the above information, we can have the trust x_i^k following a normal distribution,

i.e.,

$$x_i^k \mid x_i^{k-1},\ \mathbf{z}_i^k,\ \boldsymbol{\beta},\ \delta_w^2 \sim N(\boldsymbol{\beta}^\top \tilde{\mathbf{z}}_i^k,\ \delta_w^2), \tag{4.4}$$

where trust model coefficients $\boldsymbol{\beta} = [\beta_0,\ \beta_1,\ \cdots,\ \beta_M,\ b]^\top$, and vector $\tilde{\mathbf{z}}_i^k = \left[x_i^{k-1},\ z_{i,1}^k,\ \cdots,\ z_{i,M}^k,\ 1\right]^\top$. Similarly, the observation, i.e., the trust change $y_i^k \in \mathbf{y}_{1:I}^k$ of human operator, is

$$y_i^k = x_i^k - x_i^{k-1} + \epsilon_{v,i}^k, \tag{4.5}$$

where the zero-mean residue $\epsilon_{v,i}^k \sim N(0,\ \delta_v^2)$. Then, we can have trust change y_i^k following a normal distribution

$$y_i^k \mid x_i^k,\ x_i^{k-1},\ \delta_v^2 \sim N(x_i^k - x_i^{k-1},\ \delta_v^2). \tag{4.6}$$

We assume that all the robots will subject to a same trust evaluation process from an operator. The details of model parameters $\boldsymbol{\theta} = (\boldsymbol{B}_0,\ \boldsymbol{B}_1,\ \cdots,\ \boldsymbol{B}_M,\ \mathbf{b},\ \Delta_w,\ \Delta_v)$ in Eqns. (1) and (2) become

$$\begin{aligned}
\boldsymbol{B}_m &= \textbf{diag}(\beta_m, \cdots, \beta_m)_{I\times I}, \\
\mathbf{b} &= [b, \cdots, b]_{I\times 1}^\top, \\
\Delta_w &= \textbf{diag}(\delta_w^2, \cdots, \delta_w^2)_{I\times I}, \\
\Delta_v &= \textbf{diag}(\delta_v^2, \cdots, \delta_v^2)_{I\times I}.
\end{aligned}$$

Hence, the trust model parameters can be simplified to be $\boldsymbol{\theta} = (\boldsymbol{\beta},\ \delta_w^2,\ \delta_v^2)$.

To simplify the notations, we can conclude the vectors $\tilde{\mathbf{z}}_1^k, \cdots, \tilde{\mathbf{z}}_I^k$ to be the matrix $\tilde{\mathbf{Z}}_{1:I}^k = [\tilde{\mathbf{z}}_1^k, \cdots, \tilde{\mathbf{z}}_I^k]_{I\times(M+2)}^\top$. Furthermore, we denote each of the K time steps data as follows

$$\begin{aligned}
\mathbb{Z}_{1:I}^{1:K} &= [\mathbf{Z}_{1:I}^1, \mathbf{Z}_{1:I}^2, \cdots, \mathbf{Z}_{1:I}^K]_{K\times I\times M}, \\
\tilde{\mathbb{Z}}_{1:I}^{1:K} &= [\tilde{\mathbf{Z}}_{1:I}^1, \tilde{\mathbf{Z}}_{1:I}^2, \cdots, \tilde{\mathbf{Z}}_{1:I}^K]_{K\times I\times(M+2)}, \\
\mathbf{X}_{1:I}^{1:K} &= [\mathbf{x}_{1:I}^1,\ \mathbf{x}_{1:I}^2,\ \cdots,\ \mathbf{x}_{1:I}^K]_{I\times K}, \\
\mathbf{Y}_{1:I}^{1:K} &= [\mathbf{y}_{1:I}^1,\ \mathbf{y}_{1:I}^2,\ \cdots,\ \mathbf{y}_{1:I}^K]_{I\times K}.
\end{aligned}$$

We use the Bayesian inference to estimate the computational trust model parameters $\boldsymbol{\theta}$ in Eqns. (4.3) and (4.5) (or equivalently in Eqns. (4.1) and (4.2)). Bayesian inference infers the model parameters $\boldsymbol{\theta}$ by combining the likelihood of observing the trust change $\mathbf{Y}_{1:I}^{1:K}$, i.e., $\Pr(\mathbf{Y}_{1:I}^{1:K}, \mathbf{X}_{1:I}^{1:K} \mid \mathbb{Z}_{1:I}^{1:K}, \boldsymbol{\theta})$, with the prior distribution

$\pi_0(\boldsymbol{\theta})$. Given the prior distribution of $\boldsymbol{\theta}$ as $\pi_0(\boldsymbol{\theta})$ and the trust distribution information of Eqns. (4.4) and (4.6), the posterior distribution[2] of trust model parameters $\boldsymbol{\theta}$ is

$$\begin{aligned}
&\pi(\boldsymbol{\theta} \mid \mathbf{Y}_{1:I}^{1:K}, \mathbf{X}_{1:I}^{1:K}, \mathbb{Z}_{1:I}^{1:K}) \\
&\propto \Pr(\mathbf{Y}_{1:I}^{1:K} \mid \mathbf{X}_{1:I}^{1:K}, \mathbb{Z}_{1:I}^{1:K}, \boldsymbol{\theta}) \Pr(\mathbf{X}_{1:I}^{1:K} \mid \mathbb{Z}_{1:I}^{1:K}, \boldsymbol{\theta}) \pi_0(\boldsymbol{\theta}) \\
&= (2\pi\delta_v^2)^{-\frac{IK}{2}} \exp\left(-\frac{\sum_{k=1}^{K}\sum_{i=1}^{I}(y_i^k - (x_i^k - x_i^{k-1}))^2}{2\delta_v^2} \right) \\
&\quad (2\pi\delta_w^2)^{-\frac{IK}{2}} \exp\left(-\frac{\sum_{k=1}^{K}\sum_{i=1}^{I}(x_i^k - \boldsymbol{\beta}^\top \tilde{\mathbf{z}}_i^k)^2}{2\delta_w^2} \right) \pi_0(\boldsymbol{\theta}).
\end{aligned} \tag{4.7}$$

4.4.2 Bayesian inference of computational trust model parameters

It is often impossible to obtain the analytical solution from Eqn. (4.7) for the hyperparameters of LSS model in Eqns. (4.1) and (4.2). Markov chain Monte Carlo (MCMC) sampling is a sequential sampling approach and commonly used to obtain the approximated values of the posterior distribution's hyperparameters. Denote $(\mathbf{X}_{1:I}^{1:K})^{(l-1)}$ and $\boldsymbol{\theta}^{(l-1)}$ to be the sampled value at $l-1$ iteration of MCMC. A general iteration of the MCMC sampling for the model with latent variables can be summarized with the following two steps [17]:

- given values of $(\mathbf{X}_{1:I}^{1:K})^{(l-1)}$, sample trust model parameters $\boldsymbol{\theta}^{(l)}$ from their posterior distribution $\pi(\boldsymbol{\theta} \mid \mathbf{Y}_{1:I}^{1:K}, (\mathbf{X}_{1:I}^{1:K})^{(l-1)}, \mathbb{Z}_{1:I}^{1:K})$ in Eqn. (7);
- given values of $\boldsymbol{\theta}^{(l)}$, sample latent variable $(\mathbf{X}_{1:I}^{1:K})^{(l)}$ from the distribution function $\Pr(\mathbf{X}_{1:I}^{1:K} \mid \mathbf{Y}_{1:I}^{1:K}, \mathbb{Z}_{1:I}^{1:K}, \boldsymbol{\theta}^{(l-1)})$ in Eqn. (4.10).

MCMC can sequentially sample a set of value $\mathbf{D} = \{(\boldsymbol{\theta}^{(0)},\ (\mathbf{X}_{1:I}^{1:K})^{(0)}),\ \cdots,\ (\boldsymbol{\theta}^{(l)},\ (\mathbf{X}_{1:I}^{1:K})^{(l)}),\ \cdots\},\ l = 0, 1, \cdots$, to approach to the actual distribution of trust model parameters $\boldsymbol{\theta}$ if $l \to \infty$. The approximation is supportive because the likelihood function without latent variable $\mathbf{X}_{1:I}^{1:K}$ satisfies $\Pr(\mathbf{Y}_{1:I}^{1:K} \mid \boldsymbol{\theta}, \mathbb{Z}_{1:I}^{1:K}) \approx \sum_{(\mathbf{X}_{1:I}^{1:K})^{(l)} \in \mathbf{D}} \Pr(\mathbf{Y}_{1:I}^{1:K} \mid (\mathbf{X}_{1:I}^{1:K})^{(l)},\ \mathbb{Z}_{1:I}^{1:K}, \boldsymbol{\theta}) \Pr((\mathbf{X}_{1:I}^{1:K})^{(l)} \mid \mathbb{Z}_{1:I}^{1:K}, \boldsymbol{\theta})$ under the sampled data set $\mathbf{D}$.

More specifically, we use the Gibbs sampler of MCMC to sample the model parameters $\boldsymbol{\theta}$ in a stepwise manner, i.e., sampling each parameter in $\boldsymbol{\theta}$: $\boldsymbol{\beta},\ \delta_w^2,\ \delta_v^2$ separately based on its conditional posterior

[2]The full process of deriving the posterior distribution of $\boldsymbol{\theta}$ is shown in Appendix A.1 Eqns. (1) - (9)

distribution[3]. The derived posterior distributions of all the model parameters are

$$\boldsymbol{\beta} \mid \mathbf{Y}_{1:I}^{1:K}, \mathbf{X}_{1:I}^{1:K}, \mathbb{Z}_{1:I}^{1:K}, \delta_w^2, \delta_v^2 \sim N(\mathbf{E},\ \mathbf{V}),$$
$$\delta_w^2 \mid \mathbf{Y}_{1:I}^{1:K}, \mathbf{X}_{1:I}^{1:K}, \mathbb{Z}_{1:I}^{1:K}, \boldsymbol{\beta}, \delta_v^2 \sim IG\left(a_K,\ b_K\right),$$
$$\delta_v^2 \mid \mathbf{Y}_{1:I}^{1:K}, \mathbf{X}_{1:I}^{1:K}, \mathbb{Z}_{1:I}^{1:K}, \boldsymbol{\beta}, \delta_w^2 \sim IG\left(c_K,\ d_K\right),$$

where $\mathbf{E},\ \mathbf{V},\ a_K,\ b_K,\ c_K,\ d_K$ are the hyperparameters of the corresponding posterior distributions. The details of deriving the above posterior distributions and their hyperparameters are shown in Appendix A.1 Eqns. (10)-(15).

We can achieve the sampling of model parameters $\boldsymbol{\theta}^{(l)} = \left(\boldsymbol{\beta}^{(l)},\ (\delta_w^2)^{(l)},\ (\delta_v^2)^{(l)}\right)$ in each iteration of MCMC by referring to the above conditional posterior distribution. Then, it comes to the sampling of $\mathbf{X}_{1:I}^{1:K}$ conditional on $\boldsymbol{\theta}^{(l)}$. We utilize the forward filtering backward sampling (FFBS) in [84] to sample all the latent variables $\mathbf{x}_{1:I}^k$. Rearrange the Eqn. (1) and (2) to be the canonical form as follows,

$$\begin{bmatrix} \mathbf{x}_{1:I}^k \\ \mathbf{x}_{1:I}^{k-1} \end{bmatrix} = \tilde{\mathbf{B}}_0 \begin{bmatrix} \mathbf{x}_{1:I}^{k-1} \\ x_{1:I}^{k-2} \end{bmatrix} + \tilde{\mathbf{B}}_1 \begin{bmatrix} \mathbf{z}_{1:I,1}^k \\ \vdots \\ \mathbf{z}_{1:I,M}^k \\ \mathbf{1}_{I\times 1} \end{bmatrix} + \begin{bmatrix} \boldsymbol{\epsilon}_w^k \\ \mathbf{0}_{I\times 1} \end{bmatrix}, \tag{4.8}$$

$$\mathbf{y}_{1:I}^k = \begin{bmatrix} \mathbf{1}_{I\times I} & -\mathbf{1}_{I\times I} \end{bmatrix} \begin{bmatrix} \mathbf{x}_{1:I}^k \\ \mathbf{x}_{1:I}^{k-1} \end{bmatrix} + \boldsymbol{\epsilon}_v^k, \tag{4.9}$$

where matrix $\tilde{\mathbf{B}}_0 = \begin{bmatrix} \boldsymbol{B}_0 & \mathbf{0}_{I\times I} \\ \mathbf{1}_{I\times I} & \mathbf{0}_{I\times I} \end{bmatrix}$, matrix $\tilde{\mathbf{B}}_1 = \begin{bmatrix} \boldsymbol{B}_1 & \cdots & \boldsymbol{B}_M & \mathbf{b}_{I\times I} \\ \mathbf{0}_{I\times I} & \cdots & \mathbf{0}_{I\times I} & \mathbf{0}_{I\times I} \end{bmatrix}$, $\mathbf{1}_{I\times I}$ is the $I \times I$ identity matrix, $\mathbf{0}_{I\times I}$ is the $I \times I$ zero matrix, $\mathbf{1}_{I\times 1}$ is the vector with all elements 1, and $\mathbf{0}_{I\times 1}$ is the vector with all elements 0.

Denote $\tilde{\mathbf{x}}_{1:I}^k = [\mathbf{x}_{1:I}^k,\ \mathbf{x}_{1:I}^{k-1}]^\top$. Correspondingly, we can rearrange the K steps of trust value $\mathbf{X}_{1:I}^{1:K} \equiv [\tilde{\mathbf{x}}_{1:I}^1,\ \tilde{\mathbf{x}}_{1:I}^2,\ \cdots,\ \tilde{\mathbf{x}}_{1:I}^K]_{2I\times K}$. The FFBS can estimate the posterior distribution of state $\tilde{\mathbf{x}}_{1:I}^k$ with

[3]Though Metropolis algorithm of MCMC can perform a random-walk sampling for the entire model parameters $\boldsymbol{\theta}$ at once with the probability density functions (4.7); the Gibbs sampling approach is more efficient and can shorten the amount of time of obtaining the converged posterior distribution of model parameters.

the probabilistic function

$$
\begin{aligned}
&\Pr(\mathbf{X}_{1:I}^{1:K} \mid \mathbf{Y}_{1:I}^{1:K}, \mathbb{Z}_{1:I}^{1:K}, \boldsymbol{\theta}) \\
&= \Pr(\mathbf{x}_{1:I}^{0} \mid \tilde{\mathbf{x}}_{1:I}^{1}, \boldsymbol{\theta}) \prod_{k=2}^{K} \Pr(\tilde{\mathbf{x}}_{1:I}^{k-1} \mid \tilde{\mathbf{x}}_{1:I}^{k}, \mathbf{Y}_{1:I}^{1:k-1}, \\
&\quad \mathbb{Z}_{1:I}^{1:k-1}, \boldsymbol{\theta}) \Pr(\tilde{\mathbf{x}}_{1:I}^{K} \mid \mathbf{Y}_{1:I}^{1:K}, \mathbb{Z}_{1:I}^{1:K}, \boldsymbol{\theta}).
\end{aligned} \tag{4.10}
$$

It is obvious that sampling the entire $\mathbf{X}_{1:I}^{1:K}$ at once is impossible because each state $\tilde{\mathbf{x}}_{1:I}^{k-1}$ is dependent on $\tilde{\mathbf{x}}_{1:I}^{k}$. The FFBS first uses Kalman filter to obtain the one-step ahead prediction mean $\tilde{\mathbf{x}}_{1:I}^{k+1|k}$, covariance $P_{1:I}^{k+1|k}$ and the filtering mean $\tilde{\mathbf{x}}_{1:I}^{k|k}$, covariance $P_{1:I}^{k|k}$ for every step's trust $\tilde{\mathbf{x}}_{1:I}^{k}$. Then, the FFBS samples trust value $\tilde{\mathbf{x}}_{1:I}^{k},\ k = K, K-1, \cdots, 1$ backward from the Kalman smoother of Eqns. (4.8) and (4.9). Finally, we can obtain the entire estimated latent variable $\mathbf{X}_{1:I}^{1:K}$. The details of deriving the Kalman filter and smoother are shown in Appendix A.2.

We summarize the FBFS in Alg. 4. The input to the algorithm is the sampled value of model parameters $\boldsymbol{\theta}^{(l)} : \boldsymbol{\beta}^{(l)},\ (\delta_w^2)^{(l)},\ (\delta_v^2)^{(l)}$ at sampling step l, the sampled trust values $(\mathbf{X}_{1:I}^{1:K})^{(l-1)}$ at the previous sampling step $l-1$, and the observed data $\mathbf{Y}_{1:I}^{1:K},\ \mathbb{Z}_{1:I}^{1:K}$. The results of the FFBS are the sampled trust values $(\mathbf{X}_{1:I}^{1:K})^{(l)}$ at current sampling step l. Lines 2-4 present the mean and variance values of the one-step ahead prediction value and filtering result of $(\mathbf{X}_{1:I}^{1:K})^{(l)}$. Lines 5-8 smooth the prediction value and filtering result backward and sample the trust value $(\mathbf{X}_{1:I}^{1:K})^{(l)}$ in the meantime.

Algorithm 4 FFBS for the latent variable $\mathbf{X}_{1:I}^{1:K}$

Input: parameters $\boldsymbol{\theta}^{(l)} : \boldsymbol{\beta}^{(l)}, (\delta_w^2)^{(l)}, (\delta_v^2)^{(l)}$, latent variable $(\mathbf{X}_{1:I}^{1:K})^{(l-1)}$, data $\mathbf{Y}_{1:I}^{1:K},\ \mathbb{Z}_{1:I}^{1:K}$
Output: latent variable $(\mathbf{X}_{1:I}^{1:K})^{(l)}$

```
1: function FFBS(θ^(l), (X_{1:I}^{1:K})^(l-1), Y_{1:I}^{1:K}, Z_{1:I}^{1:K})
2:     for k = 1, ···, K do
3:         x̃_{1:I}^{k+1|k}, P_{1:I}^{k+1|k}, x̃_{1:I}^{k|k}, P_{1:I}^{k|k} ← KalmanFilter
4:     end for
5:     for k = K, ···, 1 do
6:         μ^k, ν^k ← KalmanSmoother
7:         sample (x̃_{1:I}^k)^(l) ~ N(μ^k, ν^k)
8:     end for
9:     return (X_{1:I}^{1:K})^(l) : (x̃_{1:I}^1)^(l), ···, (x̃_{1:I}^K)^(l)
10: end function
```

The step-wise MCMC process is summarized in Alg. 5. The inputs to the algorithm are the initial prior distribution of the model parameter $\pi_0(\boldsymbol{\theta})$ and the observed data $\mathbf{Y}_{1:I}^{1:K},\ \mathbb{Z}_{1:I}^{1:K}$. The results are the converged posterior distribution of model parameter $\pi(\boldsymbol{\theta})$ and distribution of latent trust value $\mathbf{X}_{1:I}^{1:K}$.

The algorithm first samples an initial value $\boldsymbol{\theta}^{(0)}$: $\boldsymbol{\beta}^{(0)}$, $(\delta_w^2)^{(0)}$, $(\delta_v^2)^{(0)}$ for model parameters $\boldsymbol{\theta}$ based on the prior distribution $\pi_0(\boldsymbol{\theta})$ in line 2. Then, it starts the L-step MCMC sampling from line 3. Lines 3-8 rearrange all the input variables $\tilde{\mathbf{z}}_i^k$ of LSS model since the input variables contain the trust value $(x_i^{k-1})^{(l-1)}$ at previous time step $k-1$. Note lines 5 and 6 also present the initial sampled trust value $(\mathbf{X}_{1:I}^{1:K})^{(0)}$.

The algorithm starts to sample the model parameter $\boldsymbol{\beta}^{(l)}$ conditional on the previous sampled parameter $(\delta_w^2)^{(l-1)}$ and the previous sampled trust value $(\mathbf{X}_{1:I}^{1:K})^{(l-1)}$ in lines 9 and 10. In lines 11 and 12, the sampling of model parameter $(\delta_w^2)^{(l)}$ is conditional on the latest sampled parameter $\boldsymbol{\beta}^{(l)}$ and the previous sampled trust value $(\mathbf{X}_{1:I}^{1:K})^{(l-1)}$ which is included in $\tilde{\mathbb{Z}}_{1:I}^{1:K}$. In lines 13 and 14, the sampling of the model parameter $(\delta_v^2)^{(l)}$ is conditional on the previous sampled trust value $(\mathbf{X}_{1:I}^{1:K})^{(l-1)}$.

Finally, the algorithm utilizes the FFBS (Alg. 4) to sample the trust value $(\mathbf{X}_{1:I}^{1:K})^{(l)}$ conditional on the latest sampled model parameters $\boldsymbol{\beta}^{(l)}$, $(\delta_w^2)^{(l)}$, $(\delta_v^2)^{(l)}$.

Algorithm 5 Gibbs sampling for the posterior trust model

Input: hyperparameters of prior $\pi_0(\boldsymbol{\theta}) : (\boldsymbol{\beta}^0,\ \Sigma^0)$, $(a_0,\ b_0)$, $(c_0,\ d_0)$, data: $\mathbf{Y}_{1:I}^{1:K}$, $\mathbb{Z}_{1:I}^{1:K}$
Output: posterior of $\boldsymbol{\beta}$, δ_w^2, δ_v^2 and $\mathbf{X}_{1:I}^{1:K}$

1: **function** MCMC($\pi_0(\boldsymbol{\theta})$, $\mathbf{Y}_{1:I}^{1:K}$, $\mathbb{Z}_{1:I}^{1:K}$)
2: Sample $\boldsymbol{\theta}^{(0)} \sim \pi_0(\boldsymbol{\theta})$, set $\mathbf{x}_{1:I}^0 = \mathbf{0}$
3: **for** $l = 1, \cdots, L$ **do**
4: **if** $l == 1$ **then**
5: Sample $(x_i^k)^{(0)} \sim N(\boldsymbol{\beta}^{(0)\top}\tilde{\mathbf{z}}_i^k, \delta_w^{2(0)})$
6: Obtain $(\mathbf{X}_{1:I}^{1:K})^{(0)}$
7: **end if**
8: $\tilde{\mathbb{Z}}_{1:I}^{1:K} \leftarrow [\tilde{\mathbf{z}}_1^1, \cdots, \tilde{\mathbf{z}}_I^1], \cdots, [\tilde{\mathbf{z}}_1^K, \cdots, \tilde{\mathbf{z}}_I^K]$
9: Update $\mathbf{V}, \mathbf{E}$ based on $\tilde{\mathbb{Z}}_{1:I}^{1:K}$
10: Sample $\boldsymbol{\beta}^{(l)} \sim N(\mathbf{E},\ \mathbf{V})$
11: Update $a_K,\ b_K$ based on $\tilde{\mathbb{Z}}_{1:I}^{1:K}$ and $\boldsymbol{\beta}^{(l)}$
12: Sample $(\delta_w^2)^{(l)} \sim IG(a_K,\ b_K)$
13: Update $c_K,\ d_K$ based on $(\mathbf{X}_{1:I}^{1:K})^{(l-1)}$
14: Sample $(\delta_v^2)^{(l)} \sim IG(c_K,\ d_K)$
15: $(\mathbf{X}_{1:I}^{1:K})^{(l)} \leftarrow$ FFBS($\boldsymbol{\theta}^{(l)}$, $(\mathbf{X}_{1:I}^{1:K})^{(l-1)}$, $\mathbf{Y}_{1:I}^{1:K}$, $\mathbb{Z}_{1:I}^{1:K}$) with $\boldsymbol{\theta}^{(l)} : \boldsymbol{\beta}^{(l)}, (\delta_w^2)^{(l)}, (\delta_v^2)^{(l)}$
16: **end for**
17: **return** $\boldsymbol{\theta}^{(0)}, (\mathbf{X}_{1:I}^{1:K})^{(0)}, \cdots, \boldsymbol{\theta}^{(L)}, (\mathbf{X}_{1:I}^{1:K})^{(L)}$
18: **end function**

4.4.3 Computational trust model with inter-robot trust influence

In this section, we investigate the trust influence of every preceding robot $r_{i-1},\ 2 \leq i \leq I$ on its succeeding robot r_i. We add x_{i-1}^k, i.e., the human's trust in the preceding robot r_{i-1}, into the linear dynamic

model in Eqn. (4.3),

$$x_i^k = \beta_{-1} x_{i-1}^k + \beta_0 x_i^{k-1} + \sum_{m=1}^{M} \beta_m z_{i,m}^k + b + \epsilon_{v,i}^k, \tag{4.11}$$

where parameter β_{-1} is the weight of human's trust in the preceding robot r_{i-1}. Accordingly, we can upgrade the trust to be

$$x_i^k \mid x_{i-1}^k,\ x_i^{k-1},\ \mathbf{z}_i^k,\ \beta_{-1},\ \boldsymbol{\beta},\ \delta_w^2 \sim N(\beta_{-1} x_{i-1}^k + \boldsymbol{\beta}^\top \tilde{\mathbf{z}}_i^k,\ \delta_w^2). \tag{4.12}$$

Note that though the leading robot r_1 in the autonomous subteam does not have a preceding robot, we can still use Eqns. (4.11) and (4.12) to include this robot by setting the variable $x_{i-1}^k = 0$ if $i = 1$. The corresponding linear dynamic model of robot r_1 remains consistent with Eqn (4.3).

In comparison with Eqns. (4.1) and (4.2), the upgraded state space equations of trust model are as follows,

$$\mathbf{x}_{1:I}^k = \check{\mathbf{B}}_0 \mathbf{x}_{1:I}^{k-1} + \sum_{m=1}^{M} \check{\boldsymbol{B}}_m \mathbf{z}_{1:I,m}^k + \check{\mathbf{b}} + \boldsymbol{B}_{-1} \boldsymbol{\epsilon}_w^k, \tag{4.13}$$

$$\mathbf{y}_{1:I}^k = \mathbf{x}_{1:I}^k - \mathbf{x}_{1:I}^{k-1} + \boldsymbol{\epsilon}_v^k, \tag{4.14}$$

where the model parameters composed matrices

$$\check{\boldsymbol{B}}_m = \boldsymbol{B}_{-1} \cdot \boldsymbol{B}_m,$$

$$\check{\mathbf{b}} = \boldsymbol{B}_{-1} \cdot \mathbf{b},$$

$$\boldsymbol{B}_{-1} = \begin{bmatrix} 1 & & & \\ \beta_{-1} & 1 & & \\ & \ddots & \ddots & \\ & & \beta_{-1} & 1 \end{bmatrix}_{I \times I}.$$

The resulting residue $\boldsymbol{B}_{-1}\boldsymbol{\epsilon}_w^k \sim N(0,\ \check{\Delta}_w)$, where

$$\check{\Delta}_w = \begin{bmatrix} \delta_w^2 & \beta_{-1}\delta_w^2 & & \\ \beta_{-1}\delta_w^2 & (\beta_{-1}^2+1)\delta_w^2 & \ddots & \\ & \ddots & \ddots & \beta_{-1}\delta_w^2 \\ & & \beta_{-1}\delta_w^2 & (\beta_{-1}^2+1)\delta_w^2 \end{bmatrix}_{I\times I},$$

and the residue $\boldsymbol{\epsilon}_v^k \sim N(0,\ \Delta_v)$ are the same to the case of Subsec. 4.4.1 that does not consider the inter-robot trust influence.

Next, we conclude the model parameters and variables by denoting

$$\begin{aligned} \check{\boldsymbol{\beta}} &= [\beta_{-1},\ \beta_0,\ \beta_1,\ \cdots,\ \beta_M,\ b]^\top, \\ \check{\mathbf{z}}_i^k &= \left[x_{i-1}^k,\ x_i^{k-1},\ z_{i,1}^k,\ \cdots,\ z_{i,M}^k,\ 1\right]^\top, \\ \check{\mathbf{Z}}_{1:I}^k &= \left[\check{\mathbf{z}}_1^k,\ \cdots,\ \check{\mathbf{z}}_I^k\right]^\top, \\ \check{\mathbb{Z}}_{1:I}^{1:K} &= \left[\check{\mathbf{Z}}_{1:I}^1,\ \cdots,\ \check{\mathbf{Z}}_{1:I}^K\right]^\top. \end{aligned}$$

The corresponding trust model parameters are summarized as $\check{\boldsymbol{\theta}} = (\check{\boldsymbol{\beta}},\ \delta_w^2,\ \delta_v^2)$. We can have the upgraded conditional posterior distribution function of each model parameter as following,

$$\begin{aligned} \check{\boldsymbol{\beta}} \mid \mathbf{Y}_{1:I}^{1:K}, \mathbf{X}_{1:I}^{1:K}, \mathbb{Z}_{1:I}^{1:K}, \check{\delta}_w, \delta_v^2 &\sim N(\check{\mathbf{E}},\ \check{\mathbf{V}}), \\ \delta_w^2 \mid \mathbf{Y}_{1:I}^{1:K}, \mathbf{X}_{1:I}^{1:K}, \mathbb{Z}_{1:I}^{1:K}, \check{\boldsymbol{\beta}}, \delta_v^2 &\sim IG(\check{a}_K,\ \check{b}_K), \end{aligned}$$

where $\check{\mathbf{E}},\ \check{\mathbf{V}}$, $\check{a}_K,\ \check{b}_K$, $\check{c}_K,\ \check{d}_K$ are the hyperparameters of corresponding model parameters' posterior distribution. The details of deriving the above posterior distributions and their hyperparameters are shown in Appendix A.3 Eqns. (16)-(20). The posterior distribution function of variance δ_v^2 is the same as the one in Subsec. 4.4.1.

Then, we rely on the MCMC in Alg. 5 to obtain the posterior distribution of model parameter $\check{\boldsymbol{\theta}}$ and latent variable $\mathbf{X}_{1:I}^{1:K}$. The inputs of the algorithm are the corresponding hyperparameters of prior $\pi_0(\check{\boldsymbol{\theta}})$ and data $\mathbf{Y}_{1:I}^{1:K}$, $\check{\mathbb{Z}}_{1:I}^{1:K}$. The FFBS in Alg. 4 samples the latent variable $\mathbf{X}_{1:I}^{1:K}$ based on the sampled model parameters $\check{\boldsymbol{\theta}}^{(l)}$. In addition, we update the parameters of Kalman filter and smoother in FFBS by rearranging ma-

trices $\tilde{\mathbf{B}}_0$ and $\tilde{\mathbf{B}}_1$ in Eqns. (11) and (12) to be $\tilde{\mathbf{B}}_0 = \begin{bmatrix} \check{\boldsymbol{B}}_0 & \mathbf{0}_{I\times I} \\ \mathbf{1}_{I\times I} & \mathbf{0}_{I\times I} \end{bmatrix}$, $\tilde{\mathbf{B}}_1 = \begin{bmatrix} \check{\boldsymbol{B}}_1 & \cdots & \check{\boldsymbol{B}}_M & b\boldsymbol{B}_{-1} \\ \mathbf{0}_{I\times I} & \cdots & \mathbf{0}_{I\times I} & \mathbf{0}_{I\times I} \end{bmatrix}$, and the residue to be $\boldsymbol{B}_{-1}\boldsymbol{\epsilon}_w^k \sim N(0,\ \check{\Delta}_w)$. Finally, we can obtain the posterior distribution of trust model parameter $\check{\boldsymbol{\theta}}^{(l)}$ and trust value $\mathbf{X}_{1:I}^{1:K}$ for the case of considering inter-robot trust influence.

Remark 7. We can take the Eqns. (4.1) and (4.2) as the generalized description of any linear computational trust model. Each type of MRS formation has a corresponding set of coefficient matrices $\boldsymbol{B}_0,\ \boldsymbol{B}_1, \cdots,\ \boldsymbol{B}_M$ in the state space equations (4.1) and (4.2). We can upgrade the coefficients matrices based on the associated trust causal relationship of the specific MRS formation as shown in sections 4.4.1 and 4.4.3. In addition, Eqns. (4.8) and (4.9) are the generalized canonical form of the state space equations. As a result, we can describe the trust model of any MRS formation with the above equations and derive the posterior distribution of the model parameter with the MCMC and FFBS.

4.5 Bayesian Optimization Based Experimental Design

In this section, we present the experimental design and data collection process for the computational trust model. Denote a K time-step discrete path as $\rho_j,\ j = 1, \cdots, J$. Then, we can obtain human's trust change data $\mathbf{Y}_{1:I}^{1:K}$ and all the associated environment attributes $\mathbb{Z}_{1:I}^{1:K}$ of robots after the human-MRS travels along the path ρ_j.

4.5.1 Bayesian Optimization

The computational trust model in this paper is designed to be personalized. During the experiment data collection phase, we will have each participant interact with the MRS to collect personalized data in a sequence of trials. Each trial consists of multiple rounds of the human participant operating the MRS to travel along a path and collect data. According to the standard approach for experimental design (e.g.,[55]), we can estimate (e.g., Bayesian LSS model and MCMC sampling in this paper) the trust model parameters after the human-MRS travels along all the predetermined paths ρ_j and collects all the associated data. The standard approach commonly uses equal or random assignment (e.g., Latin squares) of paths for the sequential trials [74, 75, 55, 18, 7]. However, many of the paths can be challenging for the human-MRS to travel through. The operator's workload will be heavy and the MRS task performance will be relatively poor with these challenging paths. Hence, the human-MRS in the standard experiment design will inevitably yield large

costs.

On the other hand, Bayesian optimization is a strategy of finding the global maximizer of an unknown objective function (e.g., the latent trust we seek to model in this paper) by sequentially exploring its variables (e.g., paths ρ_j) [80]. The generic process of Bayesian optimization is:

1. a prior distribution (e.g., prior $\pi_0(\boldsymbol{\theta})$) over the objective function initially is proposed based on the preknowledge about the objective function without any data;

2. a new query data point (e.g., path ρ_j with data $\mathbf{Y}_{1:I}^{1:K}, \mathbb{Z}_{1:I}^{1:K}$) is determined based on the current belief of objective function and an exploration strategy (e.g., find the human preferable path);

3. a corresponding posterior distribution (e.g., posterior $\pi(\boldsymbol{\theta})$) describing the updated beliefs about the objective function is obtained via Bayesian inference once data of step 2 is accumulated through observation;

4. repeat steps 2 and 3 for a number of iterations.

Acquisition function of Bayesian optimization can balance the exploration, i.e., trying a variable (e.g., path ρ_j with data $\mathbf{Y}_{1:I}^{1:K}, \mathbb{Z}_{1:I}^{1:K}$) where the objective function has a high uncertainty, and the exploitation, i.e., trying a variable (e.g., path ρ_j' with data $\mathbf{Y}_{1:I}^{1:K}, \mathbb{Z}_{1:I}^{1:K}$) where the objective function can have a high value. There are different acquisition functions, e.g., probability of improvement, expected improvement, upper confidence bound, and Thompson sampling in the Bayesian optimization [80]. Each acquisition function can describe an exploration strategy to find a path ρ to collect data for learning the objective function.

4.5.2 Decision Field Theory based Acquisition Function

We aim to reduce human workload and improve robot task performance through a strategic exploration of the MRS paths for data collection. Instead of directly using the myopic acquisition function, we apply the decision field theory to build the acquisition function and explore path for each of our sequential trials [71]. The decision field theory describes that a decision maker's preference for each option in multialternative choices evolves by comparing among options for their evaluations on respective attributes over time during his/her deliberation process (e.g., uncertainty and risks). For example, a human needs to compare different attributes in selecting a preferable hiking trail, such as trail length, slopes, vegetation situations, potential dangers, and other users' ratings. Initially, the decision maker is less certain about the advantages and disadvantages of different trails. Human preference can change among multiple options with close attributes.

Later, after the decision maker obtains more precise information about the trails through exploration, he/she can combine the information into the previous preference for further comparison of the options. Eventually, the human makes the determined decision before an imposed deadline.

In our work, path exploration based on human preference can potentially increase the human's willingness to collaborate with the autonomous system and reduce human workload. The human's trust x_i^k can reflect his/her willingness to travel along the path. Therefore, we formulate the human preference value for every path ρ_j based on the predicted human trust value. Then, the sequential trials can assign more explorations into these human preferable paths rather than equal or random exploration to all the provided paths. Based on the above analysis, we can quantify human preference value f_s of path ρ_j among multialternative choices $\rho_1, \cdots, \rho_J$ as follows

$$f_s(\rho_j) = \gamma \cdot f_{s-1}(\rho_j) + \Delta x_s(\rho_j), \tag{4.15}$$

$$\Delta x_s(\rho_j) = x_s(\rho_j) - \frac{\sum_{\tilde{j}=1}^{J} x_s(\rho_{\tilde{j}}) - x_s(\rho_j)}{J-1}, \tag{4.16}$$

$$x_s(\rho_j) = \frac{\sum_{k=1}^{K} \sum_{i=1}^{I} \boldsymbol{\beta}^\top \tilde{\mathbf{z}}_i^k}{KI}, \tag{4.17}$$

where $f_s(\rho_j)$ is the human's dynamic preference value of path ρ_j over all the other paths, the coefficient γ determines the memory of the previous preference $f_{s-1}(\rho_j)$ over the time interval, $\boldsymbol{\beta}^\top \tilde{\mathbf{z}}_i^k$ is the predicted trust value of robot r_i at the k-th step while traveling along path ρ_j, $x_s(\rho_j)$ is the predicted trust value of a path ρ_j at the s-th trial, and $\Delta x_s(\rho_j)$ describes the advantages of path ρ_j over all the other paths regarding their predicted trust value at the s-th trial.

Denote the sampled values of the posterior distribution $\pi(\boldsymbol{\theta})$ at the end of each trial as $\boldsymbol{\Theta} = \{\boldsymbol{\theta}^{(0)}, \cdots, \boldsymbol{\theta}^{(L)}\}$. If $f_s(\rho_j) > 0$ conditional on $\boldsymbol{\Theta}$, it means path ρ_j is preferred rather than the other paths at the s-th trial. Then, we integrate the above decision field theory into the BOED by assigning the acquisition function as

$$\alpha(\rho_j \mid \boldsymbol{\Theta}) = \frac{\Pr[f_s(\rho_j) > 0 \mid \boldsymbol{\Theta}]}{\sum_{j=1}^{J} \Pr[f_s(\rho_j) > 0 \mid \boldsymbol{\Theta}]}. \tag{4.18}$$

This function describes the probability of path ρ_j being preferred rather than others under the current belief of human trust model parameters. We sample the most likely preferable path $\rho^* \in \{\rho_1, \cdots, \rho_J\}$ for the human-MRS to collect data at each trial with multinomial distribution $\rho_j \sim \alpha(\rho_j \mid \boldsymbol{\Theta}),\ j = 1, \cdots, J$.

The acquisition function enables the human-MRS to explore the path in a human-like decision-making pattern. Initially, the trust model is less certain due to the limited human-MRS collaboration experience. The acquisition function can hardly distinguish the human's preference to different paths and can only identify the more obviously advantageous paths. After a period of collaboration, more data is obtained and hence the trust evaluation becomes more determined and consistent. As a result, the acquisition function is capable of comparing among the less obviously advantageous paths and finds the preferable one.

We summarize the BOED for data collection and trust model parameters estimation in Alg. 6. The inputs to the algorithm are the initial prior distribution of the model parameter $\pi_0(\boldsymbol{\theta})$, the number of sequential trials S, and all the candidate paths $\rho_j, j = 1, \cdots, J$. The results are the posterior distribution of model parameter $\pi(\boldsymbol{\theta})$ and the ultimate most likely preferable path ρ^*. The algorithm first initializes the data set $\boldsymbol{\Theta}$ with the sampled value from the prior distribution $\pi_0(\boldsymbol{\theta})$ in line 2. Then, the algorithm starts the iterated sequential trials from line 3. Inside each iteration, it estimates the probability of every path to be preferred in lines 4 - 9 and obtains the most possibly preferable path ρ^* in line 10. In line 11, the human-MRS collects the environment attributes $\mathbb{Z}_{1:I}^{1:K}$ and human feedback data $\mathbf{Y}_{1:I}^{1:K}$ while traveling along the path ρ^*. The MCMC sampling (Alg. 2) approximates the updated posterior distribution of model parameter $\boldsymbol{\theta}$ with the sampled data set $\boldsymbol{\Theta}$ in lines 12 and 13.

Algorithm 6 Decision field theory based Bayesian optimization (DFTBO)

Input: prior distribution $\pi_0(\boldsymbol{\theta})$, number of trials S, paths $\rho_j,\ j = 1, \cdots, J$
Output: posterior $\pi(\boldsymbol{\theta})$, path ρ^*

1: **function** DFTBO($\pi_0(\boldsymbol{\theta})$, S, $\{\rho_1, \cdots, \rho_J\}$)
2: $\quad\boldsymbol{\Theta} \leftarrow$ sample $\boldsymbol{\theta} \sim \pi_0(\boldsymbol{\theta}),\ \pi(\boldsymbol{\theta}) \leftarrow \pi_0(\boldsymbol{\theta})$
3: $\quad$**for** $s = 1, \cdots, S$ **do**
4: $\quad\quad$**for** $j = 1, \cdots, J$ **do**
5: $\quad\quad\quad$**for** $\boldsymbol{\theta} \in \boldsymbol{\Theta}$ **do**
6: $\quad\quad\quad\quad f_s(\rho_j) \leftarrow x_s(\rho_j),\ \Delta x_s(\rho_j)$
7: $\quad\quad\quad$**end for**
8: $\quad\quad\quad\alpha(\rho_j \mid \boldsymbol{\Theta}) \leftarrow f_s(\rho_j) \mid \boldsymbol{\Theta}$
9: $\quad\quad$**end for**
10: $\quad\quad\rho^* \leftarrow \rho_j \sim \alpha(\rho_j \mid \boldsymbol{\Theta}), j = 1, \cdots, J$
11: $\quad\quad$Obtain $\mathbb{Z}_{1:I}^{1:K}, \mathbf{Y}_{1:I}^{1:K}$ from path ρ^*
12: $\quad\quad\boldsymbol{\Theta} \leftarrow$ MCMC($\pi(\boldsymbol{\theta})$, $\mathbf{Y}_{1:I}^{1:K}, \mathbb{Z}_{1:I}^{1:K}$)
13: $\quad\quad\pi(\boldsymbol{\theta}) \leftarrow \boldsymbol{\Theta}$
14: $\quad$**end for**
15: $\quad$**return** $\pi(\boldsymbol{\theta}),\ \rho^*$
16: **end function**

4.6 Case Study: Human Multi-Robot Collaboration for Bounding Overwatch Task

In this section, we consider a case study on human multi-robot collaborative bounding overwatch tasks in offroad environment to evaluate the effectiveness of our proposed trust model and BOED. The human-in-the-loop multi-robot bounding overwatch has a complex task performing process. Investigating the trust dynamics of human in the MRS is crucial in reducing workload and increasing task performance.

A human multi-robot collaborative successive bounding overwatch task follows the motion process shown in Fig. 4.4. First, the human operates one subteam of robots ($T1$) to advance under the protection of the other subteam of autonomous robots ($T2$), while $T2$ takes an overwatch posture to protect $T1$ from potential disturbances and adversaries (see the process A1-A2 of Fig. 4.4). Then, $T1$ and $T2$ alternate their advance and overwatch roles to move forward (see B1-B2 of Fig. 4.4). In addition, $T2$ always advances earlier than $T1$ and inspects the environment to avoid the potential risks.

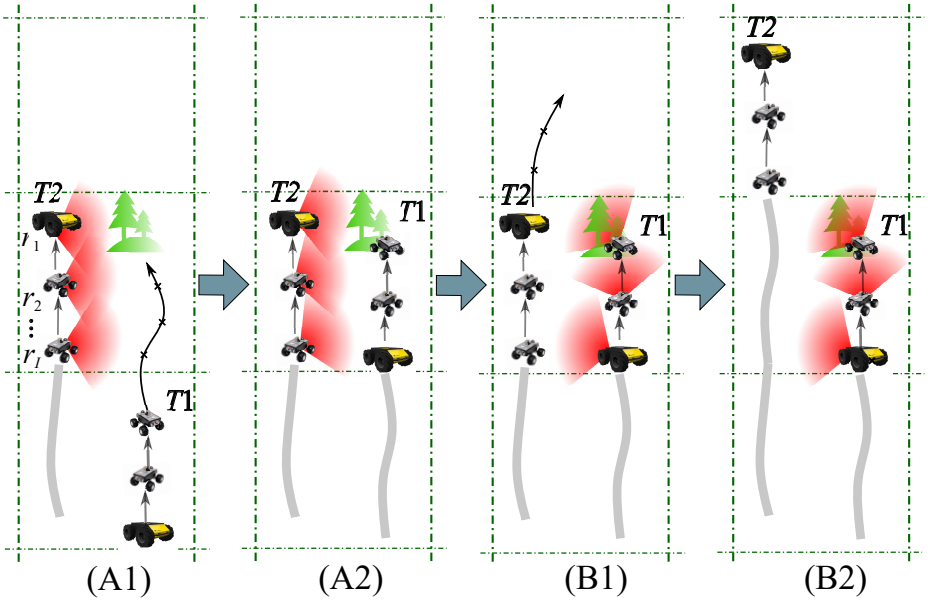


Figure 4.4: The MRS navigation process, where $T1$ is the human-operated subteam of robots, and $T2$ is the autonomous subteam. Each subteam moves forward in a line formation, and the two subteams move abreast in a column formation. Each square represents a cell region for the robot teams to perform bounding overwatch. The red sectors denote that the robots are observing the bounding team to cover them in case of any dangerous or uncertainty situations. The gray lines label the footprints of the robots.

Traversability and LoS are the two mainly referred metrics to conduct safe and reliable motion planning for robots in offroad environment [40, 7, 97, 99, 68]. Traversability describes the capability of a

robot to reside over a terrain region under an admissible state wherein it can enter given its current state. This capability can be quantified with the kinematic constraints of the robot $T2$ in a terrain. The LoS describes the unobstructed vision from an observer to the target in the motion. One can estimate the LoS with the observer's viewshed which is the visible geographic area with a sensor from a specific location. In this case study, we aim to find a path with reference to traversability and LoS such that the robots can achieve the overall best performance in terms of both mobility and sensing. Hence, we consider traversability and LoS of a robot in $T2$ at the specific locations to be the two representative impacting factors of human's trust in this bounding overwatch task. Note that the impacting factors are closely related to the specific task. A task other than bounding overwatch, e.g., reconnaissance, may require other metrics, such as a robot's information collection capability, to be the critical impacting factors. In addition, we assume the human is reliable in performing the bounding overwatch and do not evaluate the robots' trust in human.

In the following sections, we first introduce the experimental setup of the human-MRS collaborative bounding overwatch task in the ROS Gazebo simulator (Sec. 4.6.1). We then use a simulated human to show the capability of the decision field theory based Bayesian optimization in obtaining the computational trust model parameters (Sec. 4.6.2). Finally, we run a set of human subject tests to verify the usability of computational trust model and the benefits of the BOED (Sec. 4.6.3).

4.6.1 Experimental Setup

First, we use the geological LiDAR information of an area ($250m \times 250m$) in Mississippi from the United States Geological Survey (USGS)[4] to generate the offroad environment for the human-MRS. Fig. 4.5 shows the generated environment in the Gazebo simulator. We simplify the human-operated subteam into one human-operated robot r_0 for the ease of simulation. The autonomous robot subteam contains robots r_1, r_2 and r_3. Then, we discretize the workspace into a grid environment from the top view, as shown in Fig. 4.6. Each bounding overwatch process happens between two neighboring cells. Robots can return the traversability and visibility based on their perception in these cells. We provide five verified transversable candidate paths (see paths $\rho_1 - \rho_5$ in Fig. 4.6) with which the human-MRS can accomplish the bounding overwatch task to collect data for the computational trust model.

We estimate the traversability of every autonomous robot by utilizing both the geological digital elevation model (DEM) [58] of the off-road environment and the real-time inertial measurement unit (IMU) data onboard the robot. Here, the DEM contains the height information of the environment's barren ground

[4] https://prd-tnm.s3.amazonaws.com/LidarExplorer/index.html

Figure 4.5: (Left) A global view of the geological LiDAR information generated human-MRS bounding overwatch environment. Trees are simplified as triangular prisms. (Right) A local view of human-MRS bounding overwatch in Gazebo simulator. Robot r_0 is the human-operated robot. Robots r_1, r_2 and r_3 are the autonomous robots.

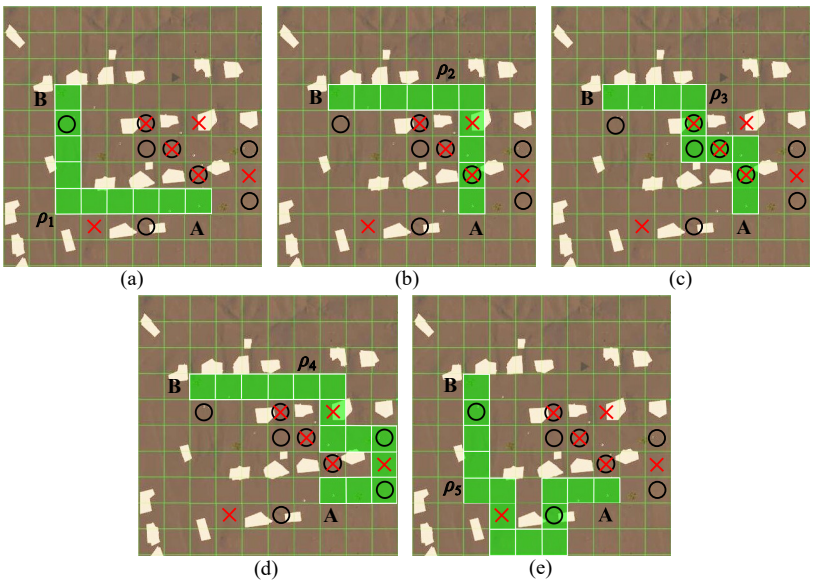


Figure 4.6: (a) - (e) present five discrete paths ρ_1 - ρ_5. The human-MRS performs bounding overwatch between each two discrete cells according to Fig. 4.4. The white patches are the top view of the environmental obstacles. Trees are scattered in the environment though not visible in appearance. The cells with cross are the risky regions where robots have high potential to collide with obstacles or turn over due to low traversability. The circle labeled cells are the risky regions where the human operator may lose track of the autonomous team members due to low visibility.

without trees and buildings. We can derive the barren ground's surface variation information with the DEM in ArcGIS Pro [69]. Then, we estimate the traversability of a robot in a cell with the weighted sum of the surface variation of the cell and the robot's real-time IMU data in that cell. Similarly, we estimate the visibility of an autonomous robot by referring to the geological digital surface model (DSM) [58] of the off-road environment and the robot LiDAR's real-time sensing distance. The DSM captures the environment's natural and artificial features on the surface of the barren terrain, such as the tops of surface obstacles and trees. We can derive the height information of these surface objects by obtaining the difference between DEM and DSM in ArcGIS Pro. The weighted sum of the height of surface objects and the robot Lidar's real-time sensing range at the cell produces the visibility of the robot in the cell.

In the sequential trials of the experiment design, every trial follows the six following steps: (1) A discrete path for the human-MRS is generated according to the decision field theory based acquisition function; (2) the three-robot formed subteam autonomously navigates from the current cell to a temporary destination in the neighboring cell along the selected discrete path. The team then stops temporarily; (3) the human operator provides trust change in each autonomous robots by referring to the recorded traversability and visibility information of autonomous robots (see Fig. 4.7); (4) the human operator manipulates the manned ground robot to bound to the autonomous robots along the same discrete path; (5) meanwhile, the autonomous robots overwatch the surrounding environment; (6) the operator repeats steps (2) - (5) until all the ground robots reach the ultimate destination.

4.6.2 Simulated Human Agent

We assume a simulated human has the known ground truth value of the LSS model parameters $\boldsymbol{\beta}_{\text{true}} = [\beta_{-1,\text{true}},\ \beta_{0,\text{true}},\ \beta_{1,\text{true}},\ \beta_{2,\text{true}},\ b_{\text{true}}]^\top$, $\delta^2_{w,\text{true}}$ and $\delta^2_{v,\text{true}}$. We rely on this simulated human agent to provide trust change value in every autonomous robot.

We provide a non-informative initial prior distribution for the model parameter $\boldsymbol{\beta}$, which has a large variance value $\Sigma^{(0)}$ associated with the randomly assigned mean value $\boldsymbol{\beta}^{(0)}$. Fig. 4.8 (top left) shows the prior distribution of the model parameters[5]. The BOED starts with this prior distribution and predicts the preferable path based on the acquisition function (4.18). Fig. 4.8 shows the Bayesian updating of the model parameter $\boldsymbol{\beta}$'s posterior distribution in one run of 20 sequential trials. The associated preferable path at each trial of that run is as follows, trial 1: ρ_4, trial 2: ρ_1, trial 3: ρ_2, trial 4: ρ_5, trials 5 - 20: ρ_1. The run was

[5]The model parameters $\boldsymbol{\theta}$ also include the residue's variance δ^2_w and δ^2_v. However, we are only interested in the weights $\boldsymbol{\beta}$ of trust model. Hence, in this and next sections, we mainly analyze the parameter $\boldsymbol{\beta}$.

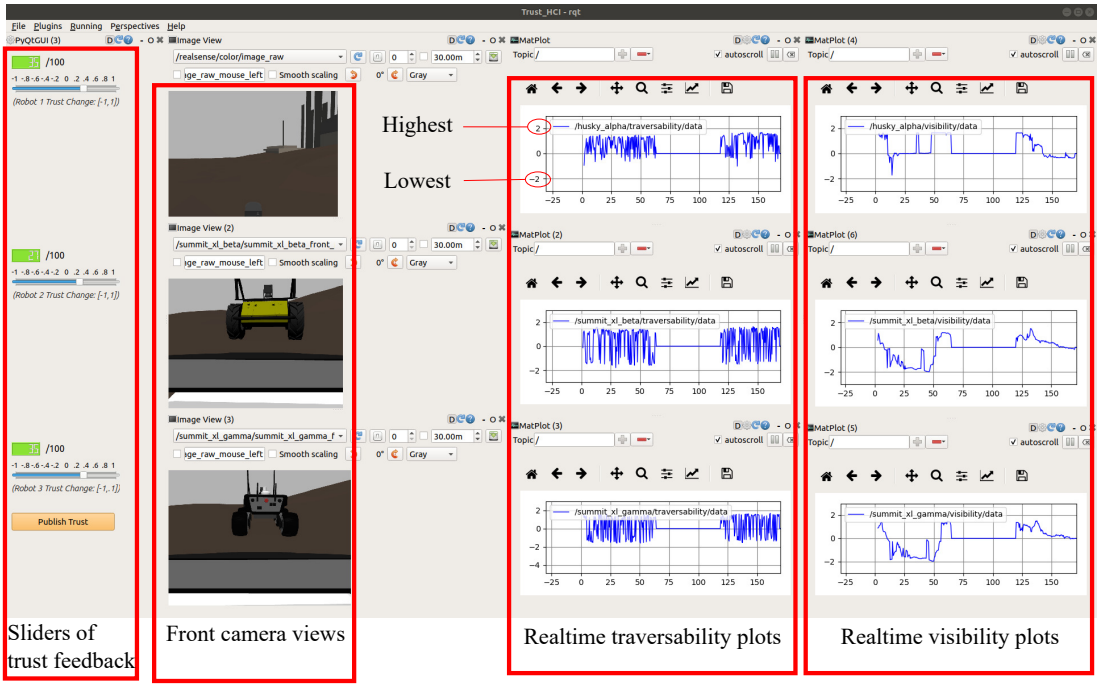


Figure 4.7: Human-MRS interaction interface. The human operator provides trust through clicking the sliders (left panel). The front camera view of each robot is shown (middle panel). Traversability and visibility for each robot (right panel) are plotted in real-time for human reference.

stopped at the 20_{th} trial where we obtained the satisfactory results. As the number of trials increases, the simulated human agent provides more data for the model parameter estimation. The posterior distribution of the trust model parameters gets updated and the resultant credible intervals also become condensed. As a result, the system becomes more and more certain about the ground truth of trust model parameters. We can observe that the posterior distribution of $\boldsymbol{\beta}$ at the 20_{th} trial approximates to the ground truth values. The preferable path ρ_1 at the 20_{th} trial is actually the path with the highest preference value under the utility Eqn. (4.15) and ground truth parameter $\boldsymbol{\beta}_{\text{true}}$.

We perform ten runs of the above BOED to confirm its replicability. At the end of the 20 sequential trials of each run, the posterior distribution of model parameters approximates the ground truth value. The corresponding preferable paths are all finally ρ_1.

The experiment results with simulated human agents demonstrate that the BOED can successfully obtain the computational trust model parameters of the human-MRS in the bounding overwatch motion task. In the case of a simulated human agent with known uncertainty of trust and observation residues, i.e., δ_w^2 and δ_v^2, the BOED presents replicable results of the most likely preferable path and posterior distribution of model parameter.

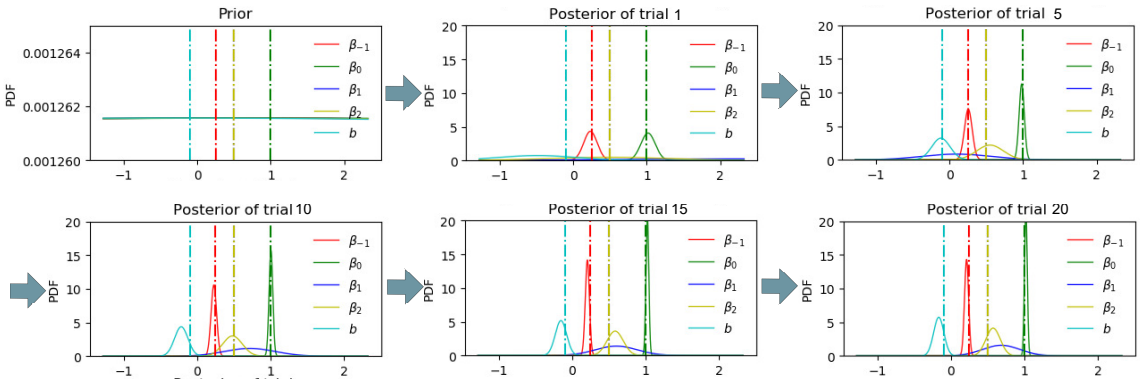


Figure 4.8: A Bayesian update of the posterior probability density function (PDF) of model parameter $\boldsymbol{\beta}$ under the BOED at trials 1, 5, 10, 15, and 20. The credible interval of the posterior distribution gets more and more condensed as the number of trials increase. The vertical dot-dash lines represent the location of the ground truth values of the simulated human's trust model parameters $\boldsymbol{\beta}$. The mean of the posterior distribution also approximates to the ground truth.

4.6.3 Human Subject Test

4.6.3.1 Procedure

We recruited 32 participants (8 females and 24 males with the age ranging from 18 to 32 years old and average age 27) to perform the human-MRS collaborative bounding overwatch task. All participants have experience in driving vehicles or playing 3D computer games. The research was approved by the Clemson University Institutional Review Board (IRB). We randomly divide the participants into two groups. Each group has 16 participants. Each participant in group one performs 6 sequential trials by referring to the procedures in the standard experiment design, while each participant in group two performs 6 sequential trials according to the BOED. Both take the 0.5 hour training and around 1 hour formal operation.

4.6.3.2 Goodness-of-fit of Trust Model

We select the sixteen participants in group one to evaluate the goodness-of-fit of our trust model. Every participant's data from the first five trials is taken as the training data, while data in the sixth trial is taken as the testing data. We evaluate the forecasting accuracy of our proposed time series trust model with the mean absolute scale error (MASE) of the sixteen participant's testing data. The MASE is an average value

of the scaled prediction errors of the output in testing data, i.e.,

$$\text{MASE} = \text{mean}(|q_1^1|, \cdots, |q_i^k|, \cdots, |q_I^K|),$$
$$q_i^k = \frac{\hat{y}_i^k - y_i^k}{\frac{1}{K-k}\sum_{t=k+1}^{K}|y_i^t - y_i^k|},$$

where $\hat{y}_i^k$ is the predicted output, i.e., the predicted human trust change in robot r_i at time step k, and q_i^k is the scaled error[6] of the prediction for robot r_i at time step k. The expectation of our trust model parameters' posterior distribution is also the maximum likelihood estimate, which is used to predict the human's trust change y_i^k and assess the forecasting accuracy. The sixteen participants' MASE values have the mean=0.72<1 with a standard deviation sd=0.16, which suggests that our trust model achieves acceptable accuracy for the general participants.

In addition, Fig. 4.9 presents one of the participant's associated traversability, visibility and the predicted human trust change values along the path ρ_1. The path presents good traversability in all the cells and only the initial position and 9-th (x-axis: 8) cell have relatively low visibility. Overall, the plots show that the estimated model parameters and their predicted trust change values can follow the trend of human provided actual trust change data in the testing data. The trust change of robots are positively correlated with the traversability and visibility according to the curves. However, due to the uncertainties of human self-report of trust change, several locations, such as the 7-th (x-axis: 6), 9-th (x-axis: 8), and 10-th (x-axis: 9) cells of path ρ_1, present higher discrepancies between the predicted values and observations. In addition, the three autonomous robots follow the similar trajectories, so their traversability, visibility and predicted trust, and predicted trust change curves present the similar patterns.

A benchmark computational trust model (CTM0) that does not consider inter-robot causality can be obtained by removing the coefficient β_{-1} and term x_{i-1}^k, i.e., the human's trust value in the preceding robot r_{i-1}, on the basis of Eqns. (4.1), (4.3), and (4.4). We first use the Bayesian information criterion[7](BIC) with the training data to compare our proposed computational trust model (CTM) with CTM0. The BIC is

[6]A naive prediction for time series data is assuming all the future prediction outputs are equal to the current output. The scaled error is the proportion of our prediction error to the naive prediction error. The scaled error smaller than 1 means the forecasting is better than the naive prediction. It is an alternative to the percentage errors.

[7]BIC describes the deviation of the fitted model from the observed data (in the training). The lower BIC value, the better fit of a model. It also penalizes more on the more complicated models.

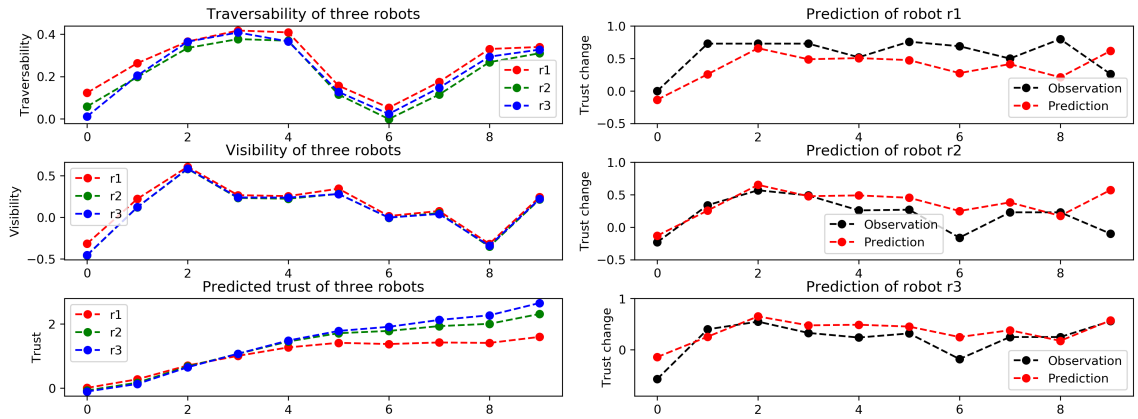


Figure 4.9: The information of the sixth trial of a participant: traversability, visibility, predicted trust, predicted trust change (curve "prediction") as well as the human provided actual trust change of robots (curve "observation").

evaluated as

$$\text{BIC} = (M+5)\log(IK) - 2\log\left(\hat{L}\right),$$
$$\hat{L} = \max\left(\Pr(\mathbf{Y}_{1:I}^{1:K} \mid \mathbf{X}_{1:I}^{1:K}, \delta_v^2)\right),$$

where $\hat{L}$ is the maximized value of the likelihood function, $M+5$ is the total number of model parameters, and IK is the number of observations used in the parameter estimation. We derive the BIC difference, i.e., the BIC of CTM0 minus the BIC of CTM, of every participant. Fig. 4.10 (Top) shows the pair of BIC values of every participant and Fig. 4.10 (bottom) presents the BIC difference value of every participant.

81% of the participants have the BIC difference larger than 0, and 75% of them have the difference larger than 10 (significant difference). To verify whether the CTM makes the difference, we performed a Wilcoxon signed-rank test [57] on the BIC values of the two models with the sixteen participants. The sixteen participants satisfy the minimum sample size requirement of the nonparametric test [60]. We rank the BIC difference and find that the median of BIC difference is smaller than 0 ($w = 18.5,\ p = 0.01$). Therefore, we reject the null hypothesis and conclude that the CTM is a better model than CTM0 in fitting the observed data. The more complicated trust model CTM that additionally captures human's trust value in the preceding robot r_{i-1} makes a difference.

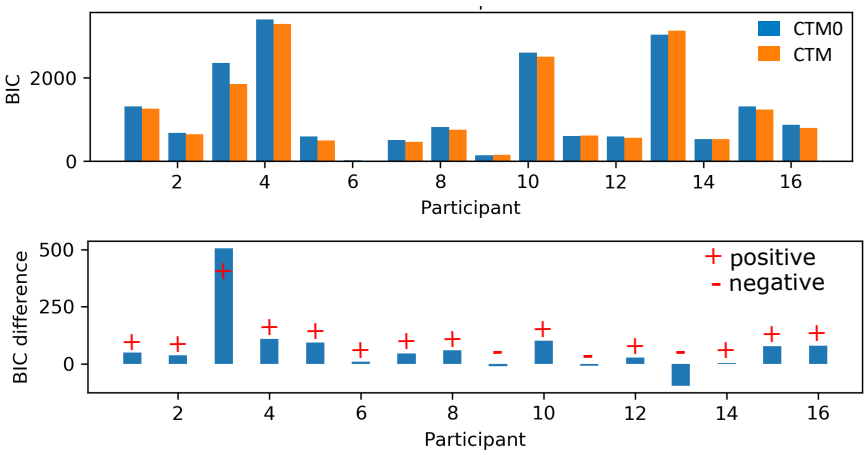


Figure 4.10: (Top) BIC of CTM0 and CTM; (Bottom) BIC difference value between CTM0 and CTM of every participant. BIC difference larger than 0 (labeled "+") indicates that the CTM fits better than the CTM0 for the participant. BIC difference smaller than 0 (labeled "−") indicates that the CTM0 fits better than the CTM.

4.6.3.3 Comparison of Experimental Design Results

We perform the Kruskal–Wallis one-way analysis of variance (ANOVA) [57] to figure out whether it is statistically significant to have C2: *the BOED* compared with C1: *the standard experimental design*. We take the experimental design approach as the independent variable (IV) and select the following indices as the dependent variables (DV):

- number of collisions: the number of collisions that the human-MRS is subject to during the bounding overwatch in the cells labeled with circles in Fig. 4.6;
- frequency of losing contact: the frequency of losing contact that the human-MRS is subject to during the bounding overwatch in the cells labeled with crosses in Fig. 4.6;
- time of completing the task: the time that human-MRS takes to complete the task;
- workload: the human's overall subjective workload measured using the NASA task load index (TLX) [36] at the end of the experiment;
- usability: the subjective satisfaction with the usability of the HCI and autonomous robots. It is measured with the IBM usability satisfaction questionnaire [50] at the end of each experiment;
- situational awareness: the human's subjective ratings based on the demands on attention, supply of attention, and understanding of the situation that are measured with the Situation Awareness Rating

Technique (SART) [28] at the end of the experiment.

Fig. 4.11 presents the comparison results of all the above six DVs for the 32 participants. Fig. 4.11(a) shows a statistically significant difference ($\chi^2 = 11.63$, $p = 0.0006$) between C1 (median= 5) and C2 (median= 2) in the number of collisions. Fig. 4.11(b) shows a statistically significant difference ($\chi^2 = 19.06$, $p = 0.000013$) between C1 (median= 6) and C2 (median= 3) in the frequency of contact loss. Fig. 4.11(c) shows a statistically significant difference ($\chi^2 = 5.22$, $p = 0.022$) between C1 (median= 61.5) and C2 (median= 55.5) in the time of completing the task. Fig. 4.11(d) shows a statistically significant difference ($\chi^2 = 4.09$, $p = 0.043$) between C1 (median= 45.22) and C2 (median= 36.61) in workload. Fig. 4.11(e) shows a statistically significant difference ($\chi^2 = 7.26$, $p = 0.007$) between C1 (median= 61.36) and C2 (median= 81.06) in usability. Fig. 4.11(f) shows a statistically significant difference ($\chi^2 = 5.12$, $p = 0.024$) between C1 (median= 33.33) and C2 (median= 25) in situational awareness.

The overall test results reveal that the BOED reduces the number of collisions, improves the robot subteams' contact, and decreases the time of task performing compared to the standard experimental design.

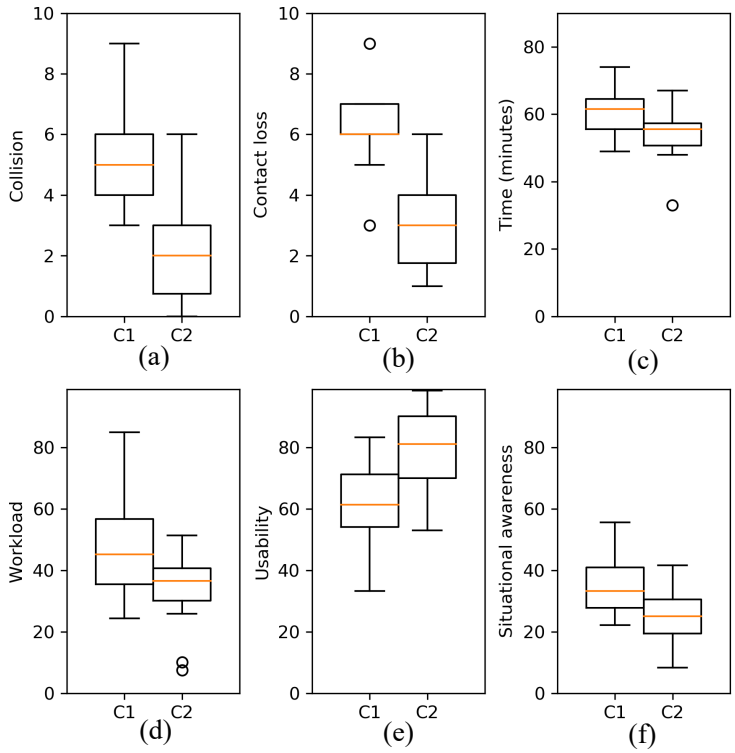


Figure 4.11: Comparison of C1 and C2 with the 32 participants regarding the number of collisions, frequency of losing contact between robot subteams, time of completing the task, workload, usability, and situational awareness.

The BOED also lowers the operator's workload, improves the system's usability, and releases the operator's situational awareness compared to the standard experimental design.

4.6.4 Discussion

In general, a more complex model structure is less interpretable and requires more data to estimate the model parameters. Hence, the less complicated model is preferred in the model selection. The BIC penalizes more on the CTM2 due to its more complicated model structure. Nevertheless, our Wilcoxon signed-rank test with BIC shows that the CTM is better in fitting the observation data of participants. That means the gain from the fitting with CTM is higher than the penalty from the complexity of CTM. It is worthwhile to consider the inter-robot trust causality in building the trust model for the MRS.

Our BOED always tends to determine the preferable paths for each participant to travel through in the sequential trials of the experiment. In the human subject test experiment, we found that the preferable paths for most participants are paths ρ_1 and ρ_5. These two paths are less likely to cause robot turnover and contact loss (see the circle and cross labeled cells in Fig. 4.5) and the associated cells are overall with higher traversability and visibility. This explains why our BOED improves the performance of human-MRS collaboration (fewer collisions, less contact losses and task completion time). In return, travelling with the preferable paths can reduce human workload and increase usability. Last but not least, the BOED has a trade-off between the above five metrics and the situational awareness. The more frequent travel with the preferable path decreases the complexity of the situation and the variability of each participant's subjective experience. As a result, the participant in BOED may become more familiar and relaxed with the task, thus having the relatively lower overall situational awareness in comparison to the standard experimental design.

4.7 Conclusion

This chapter developed two LSS models to capture the quantitative relationship between human trust in MRS and the offroad environmental characteristics, such as traversability and line of sight. One LSS model, i.e., "CTM1", quantifies the causality of trust by assuming that human's trust in each leading robot does not influence the trust in its succeeding robot in the line formation of MRS team; while the other LSS model, i.e., "CTM2", assumes the existence of such a causality between robots. Bayesian inference and MCMC sampling are used to derive the parameters of each computational trust model. In addition, Bayesian optimization based experimental design was applied to collect the data, update the trust model parameters

and obtain the optimal path for MRS motion task.

Chapter 5

Bayesian Active Reinforcement Learning for Human Multi-robot Collaboration

5.1 Introduction

Reinforcement learning (RL) for human-robots collaborative tasks has gained a considerable amount of attention recently [61]. Since the robots work side by side with human partners in a collaborative task, it is critical for the robots to behave in a trustworthy manner and obtain an appropriate level of trust from the human collaborator. Many works achieve the objective by dividing the autonomous robots from the human-robots collaborative system and utilizing RL algorithms to strengthen robots' capability of assisting the human counterpart [100]. However, one of the challenges in the RL for human-robots collaborative tasks is designing a reward function by hand that can precisely translate the human desired objective. A reward function that cannot precisely reflect the human desired robot behaviors may cause the robot to fail the goal of the collaborative task.

Providing the rewards online for all the possible state actions can work but are very expensive and workload heavy. In addition, due to the time limits of human participation in a collaboration experiment, it needs to find the solution to the sequential decision-making problems with a durable amount of exploration of the environment. Therefore, it remains potential to develop RL frameworks that can concretely resolve the

human-centric issues for the human-robots collaborative task.

This chapter develops a trust-based active RL to deal with a human-MRS collaborative offroad motion task under the LTL$_f$ specifications described objective, see Fig. 5.1. The LTL$_f$ specifications capture the system requirements, such as safety and reachability issues of robot offroad motion. Given a labeled MDP that describes the human-MRS motion behaviors in the offroad environment, we first use formal synthesis to generate a product-MDP. The product-MDP specifies the provably correct robot behaviors with its state-action transition function and can guarantee that the robots always explore the offroad environment in a way satisfying the prescribed system requirements. Hence, the formal synthesis can reduce the risks of unsafe robot behaviors during the training loop of the RL. In comparison, the general RL algorithms [65] purely rely on the trials-and-errors strategy to explore the environment and find the optimal policy for a task, which can probably violate the system requirements during the training process.

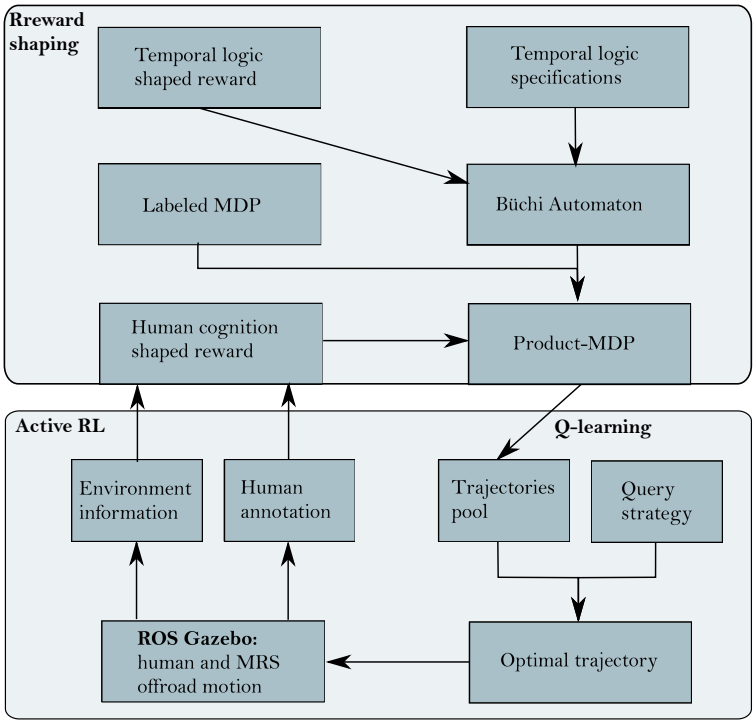


Figure 5.1: A graph representation of the labeled MDP for a robot.

Furthermore, we rely on the human collaborator to shape a personalized reward function for the state-action of the above product-MDP. Here, the personalized reward function is built on the human's trust in the MRS and varies with the human-MRS subjected environmental attributes, which can drive the human-MRS to behave in a human-trusted manner within the state space of the product-MDP. The human intermit-

tently provides trust feedback to the MRS based on the subjected environmental attributes in the training loop of the RL. The posterior probability distribution of the reward function's parameter is updated as more human trust feedback data becomes available in each episode of the RL training loop. The periodic update can gradually decrease the uncertainties of the personalized reward function. Although many HITL-RL works also learn the human-shaped reward function, they are generally trained on the weighted linear function, softmax function, or Gaussian process function [95, 94, 4]. That cannot capture the temporal property of human cognition, i.e., the human's memory of the previous experience could cumulatively influence the human's current decision-making. Our previous investigation [102] indicates that the human's trust feedback is a time-series data and utilizes a linear state space model to capture the human's trust dynamics. Hence, our personalized reward function derived from the human's computational trust can better describe the human desired behaviors than the current start-of-art reward functions. In addition, most reward functions of the PbRL work for the trajectory instead of the state action of an MDP. Although the setup can reduce the human annotation workload, it will underemphasize the decision-making mechanism of the human in each step of our collaborative motion task and reduce the amount of annotated data from the human.

Last but not least, we develop a trust-based sampling strategy for human-MRS to explore the state space of the product-MDP. The trust-based sampling strategy utilizes human decision field theory in multialternative choice to explore the most likely human preferable trajectory and assign it for the human to annotate the human-MRS interactive data, i.e., the observable human's trust change. Many paths in offroad motion are challenging for human-MRS to travel through, and the trust-based sampling strategy can reduce or avoid the visit on those unfavorable trajectories. In comparison, the sampling strategy of the state-of-art active PbRL often focuses on finding the more valuable data objects based on the learning objective [4, 95, 94, 93]. The strategy will find better quality data for the human annotation but cannot take account of the human-MRS collaborative performance and workload in the RL training loop. Compared with these works, our framework considers the cost of the human operational workload in determining the trajectory for the human to annotate data.

The organization of the rest of the chapter is as follows. Chapter 5.2 provides the preliminaries and problem setup. Chapter 5.3 introduces the shaping of human trust-based and LTL_f-based reward functions for human-MRS collaborative motion task. Chapter 5.4 presents the details of Bayesian active RL algorithm with different query strategies. Chapter 5.5 provides the verification and validation results of the Bayesian active RL under the bounding overwatch experiment.

5.2 Preliminaries and Problem Setup

5.2.1 Preliminaries of MDP and RL

Definition 22 (Labeled MDP [37]). Given a robot r with an abstracted state set S, a labeled MDP of robot r in an environment can be constructed as the tuple $\mathcal{M} = (S,\ A,\ P,\ s_0,\ \mathcal{AP},\ \mathcal{L},\ R,\ \gamma)$, where

- A is an action set of the robot;
- $P : S \times A \times S \to [0, 1]$ describes the transition probability of the robot from a state $s \in S$ to a state $s' \in S$ with an action $a \in A$, and $P(s, a, s') = \Pr(s'|s, a)$;
- $s_0 \in S$ is the initial state;
- $\mathcal{AP}$ is a set of atomic propositions;
- $\mathcal{L} : S \to 2^{\mathcal{AP}}$ labels the robot states with the propositions derived from $\mathcal{AP}$;
- $R : S \times A \times S \to \mathbb{R}$ is the reward function;
- γ is the discount factor for the reward.

Compared with a conventional MDP, the labeled MDP also has the initial state s_0, the atomic propositions $\mathcal{AP}$ containing the observable properties of the MDP states, and the labeling function $\mathcal{L}(\cdot)$ mapping every MDP state into its observable properties. In this paper, we mainly use the atomic propositions $\mathcal{AP}$ to describe the environment-relevant properties of robot state in offroad motion, such as "robot is in a region filled with obstacles", "robot is in a state with low visibility", "robot is on the top of a mountain", etc.

Example 9. We provide an exmp of 3-states-composed labeled MDP for a robot that works in offroad environments (see Fig. 5.2). Robot has the probabilistic transitions among states s_0, s_1 and s_2. The actions triggering state transition are a_1 and a_2. We label the probabilities of transition $s_1 \xrightarrow{a_1} s_2$ with value p and transition $s_1 \xrightarrow{a_1} s_1$ with $1.0 - p$. All the other transitions have probabilities of 1.0 and are not labeled in the graph for simplicity. States s_1 and s_2 are associated with observable properties and labeled with atomic positions $ap_1 = \{rocky\}$ and $ap_2 = \{woody\}$, while state s_0 is not labeled.

Definition 23 (Deterministic Policy). A deterministic policy π in the labeled MDP $\mathcal{M}$ is with $\pi : S \to A$.

Definition 24 (Trajectory). A finite trajectory ρ in the labeled MDP $\mathcal{M}$ is a sequence of state-action pairs from time step 0 to K with $\rho = s_0 \xrightarrow{a_1} s_1 \xrightarrow{a_2} \cdots s_K$, where $K \in \mathbb{N}^+$.

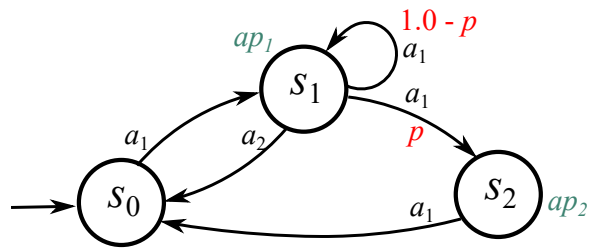


Figure 5.2: A graph representation of the labeled MDP for a robot.

Given a policy π, we can explore a sequence of state-action pairs in the labeled MDP $\mathcal{M}$ and obtain a finite trajectory ρ.

Definition 25 (**Discounted Return).** A discounted return of a trajectory ρ in the labeled MDP $\mathcal{M}$ is denoted as

$$J(\rho) = \sum_{k=0}^{K-1} \gamma^k R(s_k, a_k, s_{k+1}), \tag{5.1}$$

where $R(\cdot)$ is the reward function of the labeled MDP $\mathcal{M}$.

An RL algorithm aims to obtain the optimal policy with the maximum discounted return of an MDP. Similarly, we expect to obtain the optimal policy π^* of the labeled MDP from the initial state s_0 with

$$\pi^* = \underset{\pi}{\operatorname{argmax}}\, \mathbb{E}_\pi[J(\rho)]. \tag{5.2}$$

A state value function describing the expected discounted return of a state $s_t = s \in S$ at a time t under a policy π can be defined as

$$V^\pi(s) = \mathbb{E}_\pi \left[\sum_{k=t}^{K-1} \gamma^{k-t} R(s_k, a_k, s_{k+1}) \mid s_t = s \right].$$

Denote the state-value function of the state $s_t = s$ under the optimal policy π^* as $V^*(s)$

$$V^*(s) = \max_{a \in A} \sum_{s' \in S} \Pr(s'|s,a) \left[R(s,a,s') + \gamma V^*(s') \right], \tag{5.3}$$

and $\pi^*(s) = \operatorname{argmax}_\pi V^*(s)$. A state-action value function describing the expected discounted return from

a state-action (s, a) under a policy π can be defined as

$$Q^{\pi}(s,a) = \mathbb{E}_{\pi}\left[\sum_{k=t}^{K-1} \gamma^{k-t} R(s_k, a_k, s_{k+1}) \mid s_t = s, a_t = a\right].$$

Correspondingly, the state-action value function $Q^*(s, a)$ of a state-action pair (s, a) under the optimal policy π^* is

$$Q^*(s,a) = \sum_{s'} \Pr(s'|s,a)\left[R(s,a,s') + \gamma \max_{a' \in A} Q^*(s',a')\right], \tag{5.4}$$

and $\pi^*(s) = \underset{\pi}{\operatorname{argmax}}\, Q^*(s, a)$.

5.2.2 Preliminaries of LTL$_f$ specification

Definition 26 (LTL$_f$ Specification [10]). An LTL$_f$ formula φ is formed from atomic propositions ($\mathcal{AP}$), propositional logic operators, and temporal operators according to the grammar

$$\varphi ::= \top \mid \bot \mid ap \mid \neg\varphi \mid \varphi_1 \vee \varphi_2 \mid \bigcirc\varphi \mid \varphi_1\, U\, \varphi_2,$$

where $\top$ represents true, $\bot$ denotes false, ap is an atomic proposition, $\neg$ (negation) and $\vee$ (disjunction) are Boolean operators, and $\bigcirc$ (next) and U (until) are temporal operators. More expressive operators can be constructed from the above operators, such as, conjunction: $\varphi_1 \wedge \varphi_2 = \neg(\neg\varphi_1 \vee \neg\varphi_2)$, eventually: $\Diamond\varphi =$ true $U\ \varphi$, and always: $\Box\varphi = \neg\Diamond\neg\varphi$. A finite word w over the alphabet $2^{\mathcal{AP}}$ is defined as a finite sequence $w = l_0 l_1 l_2 l_3 \ldots \in \left(2^{\mathcal{AP}}\right)^*$, where $*$ denotes finite repetition and $l_i \in 2^{\mathcal{AP}}, \forall i \in \mathbb{N}$. The language $\{w \in \left(2^{\mathcal{AP}}\right)^*$ s.t. $w \models \varphi\}$ is defined as the set of words that satisfies the LTL$_f$ formula φ, where $\models$ is the satisfaction relation.

LTL$_f$ formulas are interpreted over finite words. Let $w[i] \in 2^{\mathcal{AP}}$ $(i \geq 0)$ be the i-th point of w. Intuitively, $w[i]$ is the set of propositions that are true at instant i. Additionally, $|w|$ represents the length of w. Given a finite word w and an LTL$_f$ formula φ, we can inductively define when φ is true for w at point i $(0 \leq i < |w|)$, written $w, i \models \varphi$, as follows:

- $w, i \models \top$ and $w, i \not\models \bot$;
- $w, i \models ap$ iff $ap \in w[i]$;
- $w, i \models \neg\varphi$ iff $w, i \not\models \varphi$;

- $w, i \models \varphi_1 \vee \varphi_2$, iff $w, i \models \varphi_1$ or $w, i \models \varphi_2$;

- $w, i \models \bigcirc\varphi$, iff $i + 1 < |w|$ and $w, i + 1 \models \varphi$;

- $w, i \models \varphi_1 U \varphi_2$, iff there exists $i' \in [i, |w|)$ satisfying $w, i' \models \varphi_2$, and for all $i'' \in [i, i')$, they satisfy $w, i'' \models \varphi_1$.

The LTL$_f$ formula can also be converted into a deterministic finite automaton (DFA)[1].

Definition 27 (Deterministic Finite Automaton [10]). A DFA, denoted by $\mathcal{A}$, is a tuple $\mathcal{A} = (Q, \Sigma, \delta, q_0, F)$, where Q is a finite set of states; Σ is an alphabet; $\delta : Q \times \Sigma \to Q$ is a transition function; q_0 is an initial state; and $F \subseteq Q$ is a set of accept states.

Given event $e_\tau \in \Sigma$, automaton state $q_\tau \in Q$, and automaton transition $\delta(q_{\tau-1}, e_\tau) = q_\tau$ under indices $\tau = 1, 2, \cdots$, a run on a word $w = e_1 \cdots e_\tau \cdots$ is a finite sequence of states $q_0 q_1 \cdots q_\tau \cdots$ in the DFA $\mathcal{A}$. The run $q_0 q_1 \cdots q_\tau \cdots$ is accepting if it has a finite amount of indices $\tau \in \mathbb{N}^+$ and $q_\tau \in F$. The language generated by the DFA $\mathcal{A}$ is $L(\mathcal{A}) = \{w \in \Sigma^* \mid w$ brings an accepting run for $\mathcal{A}\}$, where Σ^* denotes the set of all the finite words over Σ. The LTL$_f$ φ and its DFA can describe the task specification of a robotic system (e.g., the labeled MDP in exmp 1) in offroad environments.

Example 10. In this exmp, we provide an instance of an LTL$_f$ formula. Given a task specification requiring that the robot in exmp 1 "always avoids the woody area and eventually reaches the rocky area in the offroad environment", an LTL formula $\varphi = \Box\neg woody \wedge \Diamond rocky$ can be used to encode the temporal logic objective. The LTL$_f$ formula can be converted to a DFA, as shown in Fig. 5.3.

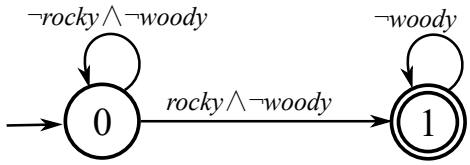


Figure 5.3: A graph representation of the converted DFA for LTL$_f$ $\varphi = \Box\neg woody \wedge \Diamond rocky$.

The finite word $w = (\neg rocky \wedge \neg woody)^*(rocky \wedge \neg woody)(\neg woody)^*$ includes all the finite words of the converted DFA in Fig. 5.3. A trajectory, such as $\rho = s_0 \to s_1 \to s_1$, of the labeled MDP in exmp 1 can satisfy the LTL$_f$ or DFA since the sequence of associated propositions $\emptyset\,\{rocky\}\,\{rocky\}$ does not violate the DFA under the word $(\neg rocky \wedge \neg woody)(rocky \wedge \neg woody)(\neg woody)$.

[1]An LTL$_f$ can be converted to an DFA with the tool "ltl2tgba" on the "Spot" platform: `https://spot.lrde.epita.fr/ltl2tgba.html`.

5.2.3 Problem Setup

In this paper, we consider a human and an MRS collaborative offroad motion task, where humans and MRS depend on each other to explore unknown offroad environments. We aim to find the most reliable MRS behaviors for humans to rely on in the collaborative offroad motion task. It is significant for the MRS to compute and predict human trust in them so that the MRS can plan dependable behaviors for the human during the collaboration. Hence, we will also learn the human trust based cognition pattern in the collaborative offroad motion task.

We assume that the motion of the human-MRS team as a whole can be formulated as a labeled MDP by referring to Def. 22. The labeled MDP has unknown transition dynamics and reward function, which are necessary to guide the human-MRS to explore the optimal policy satisfying the offroad motion task objective. Hence, we will deal with the problem by shaping and learning the reward function for the labeled MDP and exploring the optimal policy concurrently under human assistance.

Human trust in a robot describes a human's willingness to collaborate with the robot at the risk of robot reliability in uncertain situations [26, 39, 91]. In addition, we focus on the offroad environment attributes and their impacts on the robot performance since they are the most prominent factors in robot offroad motion planning [97, 99, 68]. Then, we shape a human trust based reward function of the labeled MDP by referring to the relation between the human trust in MRS and the MRS-perceived offroad environmental attributes. This a human trust based reward function can guide the MRS to behave in a human-trusted manner.

Furthermore, due to the limitations of robot sensors and perception techniques, the MRS can have difficulties in fully recognizing the offroad environment and thus fail the environment exploration task. In that situation, we can refer to the geological information map of an area and identify the critical regions that may fail the robot motion or cause great difficulties for robot motion in the offroad environment. Then, we can use an LTL_f specification φ to encode the safety-related system requirements for the human-MRS. The specification φ can guide the motion planning of the MRS and avoid the unsafe MRS motion behaviors. On the above basis, we can shape an LTL_f based reward function of the labeled MDP by referring to the state evolution process of the LTL_f converted DFA. The LTL_f based reward function can reduce the risks of robot environment exploration.

An RL algorithm with the human trust and LTL_f shaped reward function can explore the trustworthy and safe optimal policy from the labeled MDP $\mathcal{M}$ for the human-MRS. We formulate our problem as follows.

Problem of Interest 4. Given a labeled MDP $\mathcal{M}$ of the human-MRS offroad motion and an LTL_f specifica-

tion φ of offroad motion task, design a human-in-the-loop active RL framework that can

1. shape a reward function for the labeled MDP based on human trust in MRS and LTL_f specification φ to guarantee trustworthy and safe motion behaviors of the human-MRS team,

2. learn the shaped reward function based on human-MRS interaction data and find the optimal policy π^* that maximizes the discounted return of the labeled MDP $\mathcal{M}$.

5.3 Related work

Human-in-the-loop (HITL) RL relies on a human trainer to shape the rewards for the selective state-actions of the MDP instead of providing a stationary reward function in advance or specifying online rewards for all the state-actions [29, 46, 54]. Many state-of-art active RL works have been developed to maximize the rewards for the trajectory in the long term. Those works design different query strategies to select the instances that can provide noiseless and useful features for human or system to annotate the reward. They can reduce the learning costs through exploring the sensitive actions that can most efficiently maximize the discounted sum of rewards [48]. On the basis of active RL, active preference based reinforcement learning (PbRL) directly learns an expert's preferences on data objects instead of a hand-designed reward function [4, 95, 94, 93]. The preference-based feedback is signaled based on pairwise comparison between data objects and indicates the relative instead of absolute utility values of robot performance. Besides the above RL that uses human feedback data to shape the reward function, there are other human centered RL frameworks, such as imitation learning, inverse reinforcement learning [6, 41]. However, they rely on human demonstration to infer the human desired behaviors and beyond the scope of this paper.

The active RL and active PbRL can improve the learning efficiency of RL though obtaining high quality data. However, they do not consider the human's operational workload in the collaboration, where human and robots depend on each other's advantages to deal with a complex task. In such a task scenario, the human workload will be aggregated by the physical burden of collaboration and cognition burden of data annotation. In addition, it also needs to maintain a good level of human's trust in robot since human's trust in robot describes the human's willingness to collaborate with the robot at the risks of robot reliability in uncertain situations [26, 39, 91]. The strategy that only considers satisfying the learning efficiency with high quality data will not be enough to maintain the trust since the robots can frequently fail the task in the trial-and-error of RL. A difficult and high-risk choice in the training loop can result in the human losing

trust in the collaborated robots, which can undermine human's willingness to collaborate with the robots. The current active RL works seldom consider the human robot collaborative task issues and the associated human workload. Last but not least, the state-action exploration strategies in the current active PbRL often rely on simplified human cognition models, i.e., the softmax equations, which cannot fully represent human's decision-making in a complex motion task.

Furthermore, the uncertain and risky choices that robots face with under the RL can cause crashes of the robots during the interactions with the environment. A safe exploration of the robot's state-actions in an RL task is critical to reduce the training costs and increase the trustworthiness of the robots. Many RL works integrate temporal logic described objective into robot task performing and aim to improve the robot safety during the robot's interaction with the environment [96, 72, 53, 15, 14, 67, 92]. Here, temporal logic task specification, such as linear temporal logic (LTL) and signal temporal logic (STL), can describe a desired task objective regarding the time property of a robotic system's behaviors with logic expressions [10, 5]. The specification can impose a safety or liveness requirement for the robot's future behaviors, such as "always avoid a risky area", or "always eventually reach multiple desired locations in sequence". Formal verification or synthesis is conducted to generate the policies satisfying the temporal logic task objective within the MDP of robot behaviors and can guarantee the robots' safe exploration of environment under the RL frameworks.

The above formal synthesis and verification can have an exhaustive check on the state space of the abstracted autonomous robots' behaviors. However, the associated state-actions and temporal logic objective are not precise enough to guarantee the risk-free of robot physical dynamics in the motion. There are seldom formal method considering human decision-making in the task performing except that works [31, 27] model human operator as an MDP in the human-robot collaborative tasks. Their combination of formal method with human can guarantee both the correctness of robotic behaviors and flexibility of the human-MRS collaboration. However, they generally lack the consideration of human's willingness to accept the generated safe strategies. In addition, it is very challenging to build a robust MDP to capture human's psychology state due to the measurement difficulty.

5.4 Reward Shaping for Human-MRS Collaboration

We shape the reward function for the labeled MDP of human-MRS by referring to the human trust in MRS and the LTL_f encoded system requirements. In Sec. 5.4.1, we formulate a trust-based novel reward function by considering the influence of human memory on historical trust and the inter-robot causality of

human trust among MRS. In Sec. 5.4.2, we integrate the trust-based reward function with a reward quantifying the task evolution process of satisfying the LTL$_f$ encoded system requirements φ, which can reduce the safety risks of MRS offroad exploration.

5.4.1 Reward Shaping from Human Trust in MRS

Given human trust $\mathbf{x}_{1:I}^t$ at a time step t, we predict the human's future trust $\hat{\mathbf{x}}_{1:I}^{K|t}$ in the robots $r_1, \cdots, r_I$ over $K-t$ time steps further from t with the following equation

$$\hat{\mathbf{x}}_{1:I}^{K|t} = \mathbf{B}_0 \mathbf{x}_{1:I}^t + \sum_{k=t}^{K-1} \gamma^{k-t} \tilde{\mathbf{B}} \tilde{\mathbf{z}}_{1:I}^{k+1}, \tag{5.5}$$

where matrix $\tilde{\mathbf{B}} = [\mathbf{B}_1, \ \cdots, \ \mathbf{B}_M, \ \mathbf{diag}(\mathbf{b})]_{I \times I(M+1)}$ and the $1 \times I(M+1)$ dimensional vector $\tilde{\mathbf{z}}_{1:I}^{k+1} = [z_{1,1}^{k+1}, \cdots, z_{I,1}^{k+1}, \cdots, z_{1,M}^{k+1}, \cdots, z_{I,M}^{k+1}, 1, \cdots, 1]^\top$ contains the environment attributes that the robots are going to subject to in the future time step k.

Eqn. (5.5) weighs on the future environment attributes in a discounted manner based on the human trust $\mathbf{x}_{1:I}^t$ at time t. We can just estimate the value of $J = \sum_{k=t}^{K-1} \gamma^{k-t} \tilde{\mathbf{B}} \tilde{\mathbf{z}}_{1:I}^{k+1}$ in Eqn. (5.5) and select the highest one if we want to find the trajectory that has the maximum predicted trust. Therefore, we can just estimate the value of J instead of evaluating the $\hat{\mathbf{x}}_{1:I}^{K|t}$.

Then, we can shape a reward function for the human-MRS offroad motion task as follows,

$$R_1(s_k, a_k, s_{k+1}) = \mathbf{1}_{1 \times I} \tilde{\mathbf{B}} \tilde{\mathbf{z}}_{1:I}^{k+1}. \tag{5.6}$$

The resulting discounted return over the $K-t$ time steps horizon is

$$\begin{aligned} J &= \sum_{k=t}^{K-1} \gamma^{k-t} R_1(s_k, a_k, s_{k+1}) \\ &= \sum_{k=t}^{K-1} \gamma^{k-t} \mathbf{1}_{1 \times I} \tilde{\mathbf{B}} \tilde{\mathbf{z}}_{1:I}^{k+1} \\ &= \mathbf{1}_{1 \times I} \hat{\mathbf{x}}_{1:I}^{K|t} - \mathbf{1}_{1 \times I} \mathbf{x}_{1:I}^t. \end{aligned} \tag{5.7}$$

Theorem 4. *We can use the reward function in Eqn. (5.6) and the state value function $V_\pi^*(s_k) = \max_\pi \mathbb{E}_\pi[J]$ to find the optimal policy π^*.*

Proof Given a policy π under the labeled MDP of human-MRS, we can derive the expected sum

value of the predicted trust $\hat{\mathbf{x}}_{1:I}^{K|t}$ over $K-t$ time steps horizon as $\mathbb{E}_\pi[\mathbf{1}_{1\times I}\hat{\mathbf{x}}_{1:I}^{K|t}]$, where $\mathbf{1}_{1\times I}$ is a vector with all elements 1. The corresponding maximum expected sum value of the prediction is $\max_\pi \mathbb{E}_\pi[\mathbf{1}_{1\times I}\hat{\mathbf{x}}_{1:I}^{K|t}]$. The discounted return J satisfies $\operatorname{argmax}_\pi \mathbb{E}_\pi[J] \equiv \operatorname{argmax}_\pi \mathbb{E}_\pi[\mathbf{1}_{1\times I}\hat{\mathbf{x}}_{1:I}^{K|t}]$ since $\mathbb{E}[\mathbf{1}_{1\times I}\mathbf{x}_{1:I}^{t}]$ is already known at time step t and hence a constant. That is to say, the policy maximizing the sum value of the prediction $\mathbb{E}_\pi[\mathbf{1}_{1\times I}\hat{\mathbf{x}}_{1:I}^{K|t}]$ is equivalent to the policy that maximizes the function $\mathbb{E}_\pi[J]$. □

The obtained trajectory ρ^* can provide the highest predicted trust value for the human-MRS. Thus, the trust-based reward function Eqn. (5.6) can drive the human-MRS to behave in a human trusted manner and increase human's willingness to collaborate with the MRS.

Remark 8. Note that the preference model in the state-of-art PbRL doesn't consider the temporal effect of human rating. The human annotation can inevitably have the human memory influence. We separate the temporal effect of human trust from the trust model and utilize the weights of environment attributes to shape the reward function. It provides more accurate reward shaping based on human annotation.

5.4.2 Reward Shaping from LTL$_f$ Encoded System Requirements

In this section, we further shape the reward function of the labeled MDP through integrating the LTL$_f$ specification described system requirements. It is known that a formal synthesis with the labeled MDP in Def. 22 and the DFA in Def. 27 can satisfy the required LTL$_f$ specification in a correct-by-construction approach. Then, given a labeled MDP with unknown transition dynamics and reward function, we can synthesize a product-MDP with the following definition.

Definition 28 (Product-MDP). Given a labeled MDP $\mathcal{M} = (S,\ A,\ P,\ s_0,\ \mathcal{AP},\ \mathcal{L},\ R,\ \gamma)$ and the DFA $\mathcal{A} = (Q,\ \Sigma,\ \delta,\ q_0,\ F)$ of an LTL specification, a product-MDP $\mathcal{G} = \mathcal{M} \times \mathcal{A} = (S_p,\ A_p,\ P_p,\ s_{p,0},\ F_p,\ R_p,\ \gamma)$ can be synthesized, where

- $S_p = S \times Q$ is the state set of the product-MDP;
- $A_p = A$ is the action set of product-MDP;
- $P_p : S_p \times A_p \times S_p \to [0,1]$ describes the transition probability from a state $s_p = \langle s, q\rangle \in S_p$ to another state $s_p' = \langle s', q'\rangle \in S_p$ with an action $a \in A_p$ and it satisfies

$$P_p(s_p, a, s_p') = \begin{cases} P(s,a,s') & \text{if } q \xrightarrow{\mathcal{L}(s')} q', \\ 0 & \text{otherwise;} \end{cases}$$

- $s_{p,0} = \langle s_0, q_1 \rangle$ is the initial state of product-MDP if $\exists q_0 \xrightarrow{\mathcal{L}(s_0)} q_1$;
- $F_p = S \times F$ is the set of accepting states;
- $R_p : S_p \times A_p \times S_p \to \mathbb{R}^+$ is the reward function of product-MDP.

The product-MDP enumerates all the state-action transitions for the composition of the labeled MDP and the DFA. Any state-action (s_p, a_p) or trajectory ρ_p of the product-MDP is provably correct regarding the DFA transitions, i.e., the task evolution process of the LTL$_f$ specification. In other words, the transition probability function $P_p(s_p, a, s'_p) = 0$ ensures that any state-action (s, a) or trajectory ρ in the labeled MDP will not violate the DFA. In the above manner, the product-MDP can guarantee the reliability of the MRS behaviors.

Example 11. Given a labeled MDP with the states in exmp 1, we can synthesize a product-MDP with the DFA in Example 2 according to Def. 28. The result is shown in Fig 5.4. The dashed arrows show labeled transitions that do not exist according to the actual dynamics of the labeled MDP. Initially, we cannot identify the labeled transitions represented by the dashed arrows and consider both solid and dashed situations can exist in the product-MDP. However, the labeled transitions represented by the dashed arrows will not be viable in an RL process since the actual dynamics of the labeled MDP do not support them. Therefore, dashed arrows are removed. As a result, an RL process under the product-MDP will not violate the labeled MDP. In the meantime, the transitions between the unconnected states, such as from state $\langle s_1, 0 \rangle$ to state $\langle s_2, 0 \rangle$, have the probability of 0 since they violate the DFA in Example 2.

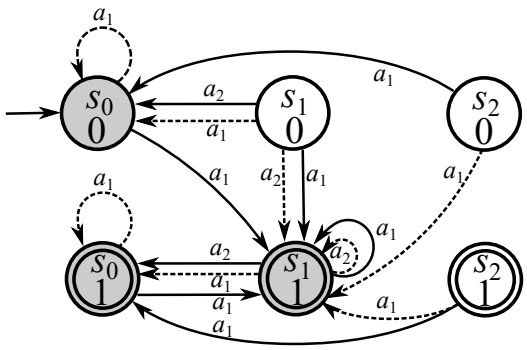


Figure 5.4: A graph representation of the product-MDP. Node $\langle s_0, 0 \rangle$ is the initial state of the product-MDP. Nodes $\langle s_0, 1 \rangle$, $\langle s_1, 1 \rangle$ and $\langle s_2, 1 \rangle$ are the accepting states. The state transition probabilities between the solid arrows are unknown, while the state transition probability between any two unconnected nodes is 0.

The nodes labeled with the gray color in Fig. 5.4 are the reachable states of the product-MDP. Then, the product-MDP can have the trajectories $\langle s_0, 0 \rangle \xrightarrow{a_1} \langle s_1, 1 \rangle \xrightarrow{a_1} \langle s_1, 1 \rangle \xrightarrow{a_1} \langle s_1, 1 \rangle \cdots$ and $\langle s_0, 0 \rangle \xrightarrow{a_1}$

$\langle s_1, 1\rangle \xrightarrow{a_2} \langle s_0, 1\rangle \xrightarrow{a_1} \langle s_1, 1\rangle \xrightarrow{a_2} \langle s_0, 1\rangle \cdots$. The corresponding finite trajectories $\rho_1 = s_0 s_1 s_1 \cdots$ and $\rho_2 = s_0 s_1 s_0 s_1 s_0 s_1 \cdots$ in the labeled MDP can satisfy the DFA or LTL$_f$ specification of Example 2. •

Hence, we shape an LTL$_f$ based reward function based on the synthesized product-MDP. We aim to achieve the following objectives:

1. robots always avoid exploring the path violating the LTL$_f$ specification as much as possible in each episode of the RL;

2. robots always terminate exploring the environment once they reach a state that can satisfy the LTL$_f$ specification in each episode of the RL.

Denote $A(s_p)$ as the available outgoing actions at state s_p, we define a reward function under the condition that $P_p(s_p, a, s'_p) \neq 0$ to be

$$R_2(s_p, a, s'_p) = \begin{cases} c & \text{if } s'_p \in F_p \\ -c & \text{if } s_p \neq s'_p \wedge s_p \in F_p \\ -\infty & \text{if } A(s'_p) = \emptyset \wedge s'_p \notin F_p \\ -\infty & \text{if } \max_{a' \in A(s'_p)} R_2(s'_p, a', s''_p) = -\infty \wedge s'_p \notin F_p \\ 0 & \text{otherwise} \end{cases} \tag{5.8}$$

where c is a constant that is much larger than any value of $R_1(s, a, s')$. More specifically, this reward function can achieve the above two objectives with the following principles of RL trajectory exploration

1. always select an action a if it can make the behavior (s_p, a, s'_p) of human-MRS satisfies $s'_p \in F_p$;

2. always stop at one of the accepting states of the product-MDP. More specifically, the human-MRS avoids sequentially visiting two different accepting states in the product-MDP if the behavior (s_p, a, s'_p) of human-MRS satisfies $s_p \neq s'_p \wedge s_p \in F_p$;

3. always avoid visiting states where the behavior (s_p, a, s'_p) of human-MRS satisfies $\max_{a' \in A(s'_p)} R_2(s'_p, a', s''_p) = -\infty \wedge s'_p \notin F_p$, whose base case is the sink state[2] condition $A(s'_p) = \emptyset \wedge s'_p \notin F_p$;

[2]Sink state is a non-accepting state that does not have outgoing transitions.

The shaped reward function $R_2(s_p, a_p, s'_p)$ can improve the reliability of MRS behaviors in the trajectory planning by directly avoiding the MRS behaviors whose transitions $P_p(s_p, a, s'_p) = 0$. As a complement, the shaped trust-based reward function $R_1(s, a, s')$ can quantitatively label the reward of transitions $P_p(s_p, a, s'_p) \neq 0$ in the product-MDP. Hence, we integrate the LTL$_f$ shaped reward and the human trust shaped reward as reward function

$$R_p(s_p, a, s'_p) = \begin{cases} c & \text{if } s'_p \in F_p \\ -c & \text{if } s_p \neq s'_p \wedge s_p \in F_p \\ -\infty & \text{if } A(s'_p) = \emptyset \wedge s'_p \notin F_p \\ -\infty & \text{if } \max_{a' \in A(s'_p)} R_p(s'_p, a', s''_p) = -\infty \wedge s'_p \notin F_p \\ R_1(s, a, s') & \text{otherwise} \end{cases} \tag{5.9}$$

We present Alg. 7 to simultaneously generate the product-MDP according to Def. 28 and the associated reward function in Eqn. (5.8). The input of the algorithm is the labeled MDP $\mathcal{M}$ and the LTL$_f$ converted DFA. The output of the algorithm is the product-MDP and the associated reward function. Lines 2 - 5 find the initial state and accepting states of the product-MDP. Lines 6 - 20 start with the initial state and recursively generate the successive states, transitions, and reward functions of the product-MDP. Lines 10 -14 and 17 -19 formulate the reward function $R_p(s_p, a, s'_p)$ according to the reward shaping rules in Eqn. (5.8).

5.5 Bayesian Active Reinforcement learning

In this section, we utilize a Bayesian active RL framework to concurrently learn the unknown parameters $\mathbf{B}_m,\ m = 1, \cdots, M$ and $\mathbf{b}$ of the human trust shaped reward $R(s_p, a, s'_p)$ and explore the optimal policy and trajectory for the human-MRS. Bayesian inference and MCMC sampling first estimate the approximate posterior distribution of the unknown model parameters of the human trust shaped reward in Sec. 5.5.1. Then, Sec. 5.5.2 presents different query strategies to generate the trajectory for human-MRS to travel along and the human to annotate the trust shaped reward under an active RL framework.

Algorithm 7 Reward shaping of product-MDP

Input: labeled MDP $\mathcal{M} = (S,\ A,\ P,\ s_0,\ \mathcal{AP},\ \mathcal{L},\ R,\ \gamma)$, DFA $\mathcal{A} = (Q,\ \Sigma,\ \delta,\ q_0,\ F)$
Output: product-MDP $\mathcal{G}_p = (S_p,\ A_p,\ P_p,\ s_{p,0},\ F_p,\ R_p,\ \gamma)$

```
function PRODUCT-MDP(M, A)
    S_p ← ∅
    Initial state s_{p,0} ← ⟨s_0, q_1⟩ if ∃q_0 --L(s_0)--> q_1
    Accepting states F_p = S × F
    s_p = s_{p,0}, S_p ← add s_{p,0}
    function RECURSION(s_p)
        list1 ← ∅
        successors ← {all s'_p satisfies P_p(s_p, a, s'_p) ≠ 0}
        for s'_p ∈ successors do
            if s_p ≠ s'_p ∧ s_p ∈ F_p then R_p(s_p, a, s'_p) = −c
            else if s'_p ∈ F_p then R_p(s_p, a, s'_p) = c
            else if A(s'_p) = ∅ ∧ s'_p ∉ F_p then R_p(s_p, a, s'_p) = −∞
            else R_p(s_p, a, s'_p) = R_1(s_p, a, s'_p)
            end if
            list1 ← add R_p(s_p, a, s'_p)
            if s'_p ∈ S_p then continue
            end if
            S_p ← add s'_p
            list2 ← RECURSION(s'_p)
            if max(list2) = −∞ ∧ s'_p ∉ F_p then
                list1 ← update R_p(s_p, a, s'_p) = −∞
            end if
        end for
        return list1
    end function
    return S_p, A_p, P_p, s_{p,0}, F_p, R_p, γ
end function
```

5.5.1 Bayesian Inference for Parameters of Reward Function

In this subsection, we estimate the posterior distribution of the unknown parameters $\boldsymbol{\theta}$ of the human trust shaped reward. The estimation requires the environment attributes $\mathbf{Z}_{1:I}^k$ data along a trajectory ρ and the corresponding human trust $\mathbf{x}_{1:I}^k$. We rely on the robot sensors to record the perceived offroad environment attributes $\mathbf{Z}_{1:I}^k$ and present the plots of $\mathbf{Z}_{1:I}^k$ on a human computer interface. The human's trust $\mathbf{x}_{1:I}^k$ is a self-reported time-series value which can exceed the measurement range of the human computer interface. Therefore, we design the human computer interface to measure the human's trust change $\mathbf{y}_{1:I}^k = \left[y_1^k, \ \cdots, \ y_I^k\right]^\top$ with

$$\mathbf{y}_{1:I}^k = \mathbf{x}_{1:I}^k - \mathbf{x}_{1:I}^{k-1} + \boldsymbol{\epsilon}_v^k, \tag{5.10}$$

where the trust $\mathbf{x}_{1:I}^k$ becomes a latent variable; the trust change $\mathbf{y}_{1:I}^k$ is the actual observation on trust $\mathbf{x}_{1:I}^k$; and the vector $\boldsymbol{\epsilon}_v^k$ describes the measurement error of the self-reported trust change.

The human trust change $\mathbf{Y}_{1:I}^{1:K}$ and offroad environment attributes $\mathbb{Z}_{1:I}^{1:K}$ are the collected data at each episode of the active RL. Denote $\mathcal{D}_{s-1}$ as a collection of $s-1$ episodes of data, we use Bayesian inference to infer the latent variable $\mathbf{X}_{1:I}^{1:K}$ and update the posterior distribution of trust model parameters $\boldsymbol{\theta}$ at each episode of the active RL. More specifically, we can first infer a conditional posterior distribution of trust model parameters $\boldsymbol{\theta}$ with equation

$$\begin{aligned}
&\Pr_s(\boldsymbol{\theta} \mid \mathbf{Y}_{1:I}^{1:K}, (\mathbf{X}_{1:I}^{1:K})^{(l-1)}, \mathbb{Z}_{1:I}^{1:K}) \propto \\
&\quad \Pr(\mathbf{Y}_{1:I}^{1:K} \mid (\mathbf{X}_{1:I}^{1:K})^{(l-1)}, \mathbb{Z}_{1:I}^{1:K}, \boldsymbol{\theta}) \Pr((\mathbf{X}_{1:I}^{1:K})^{(l-1)} \mid \\
&\quad \mathbb{Z}_{1:I}^{1:K}, \boldsymbol{\theta}) \Pr_{s-1}(\boldsymbol{\theta} \mid \mathcal{D}_{s-1}).
\end{aligned} \tag{5.11}$$

It is obvious that an estimation of $(\mathbf{X}_{1:I}^{1:K})^{(l-1)}$ is necessary for the posterior distribution $\Pr_s(\boldsymbol{\theta} \mid \mathbf{Y}_{1:I}^{1:K}, (\mathbf{X}_{1:I}^{1:K})^{(l-1)}, \mathbb{Z}_{1:I}^{1:K})$. Hence, we further infer the conditional posterior distribution of $(\mathbf{X}_{1:I}^{1:K})^{(l-1)}$ with equation

$$\begin{aligned}
&\Pr(\mathbf{X}_{1:I}^{1:K} \mid \mathbf{Y}_{1:I}^{1:K}, \mathbb{Z}_{1:I}^{1:K}, \boldsymbol{\theta}^{(l-1)}) = \\
&\quad \Pr(\mathbf{x}_{1:I}^0 \mid \tilde{\mathbf{x}}_{1:I}^1, \boldsymbol{\theta}^{(l-1)}) \prod_{k=2}^{K} \Pr(\tilde{\mathbf{x}}_{1:I}^{k-1} \mid \tilde{\mathbf{x}}_{1:I}^k, \mathbf{Y}_{1:I}^{1:k-1}, \\
&\quad \mathbb{Z}_{1:I}^{1:k-1}, \boldsymbol{\theta}^{(l-1)}) \Pr(\tilde{\mathbf{x}}_{1:I}^K \mid \mathbf{Y}_{1:I}^{1:K}, \mathbb{Z}_{1:I}^{1:K}, \boldsymbol{\theta}^{(l-1)}),
\end{aligned} \tag{5.12}$$

where $\tilde{\mathbf{x}}_{1:I}^{k} = [\mathbf{x}_{1:I}^{k},\ \mathbf{x}_{1:I}^{k-1}]^{\top}$. As a result, the inference of the latent variable $\mathbf{X}_{1:I}^{1:K}$ and model parameters $\boldsymbol{\theta}$ depend on each other. Therefore, we utilize the Markov chain Monte Carlo (MCMC) sampling method to concurrently infer the latent variable $\mathbf{X}_{1:I}^{1:K}$ and the trust model parameters $\boldsymbol{\theta}$. The sampled values can approximate the numerical solution of the posterior distribution of $\mathbf{X}_{1:I}^{1:K}$ and $\boldsymbol{\theta}$.

5.5.2 Trajectory Query Strategies for Bayesian Active RL

Active learning aims to select the most useful samples, i.e., trajectories, from the unlabeled data set and hand them over to the oracle, i.e., human annotator, for labeling [70]. Query strategy of active learning achieves the objective by selecting a trajectory that optimizes a utility function which counts both the labeling costs and the learning performance. Therefore, we present two query strategies for our Bayesian active RL framework to find the most useful trajectory in each episode, i.e, the Thompson sampling based random query (benchmark) and the workload-based query.

5.5.2.1 Thompson Sampling based Random Query

Thompson sampling is a randomized strategy which samples a reward function from the posterior of trust model parameters and selects the trajectory with the highest simulated reward. Therefore, Thompson sampling is easy to implement as long as the the prior hyperparameters of the trust model are available. We apply the Thompson sampling into the query strategy of our Bayesian active RL. In each episode s of the Bayesian active RL, we take the posterior distribution $\Pr_{s-1}(\boldsymbol{\theta} \mid \mathbf{Y}_{1:I}^{1:K}, \mathcal{D}_{s-1})$ as the prior distribution of $\boldsymbol{\theta}$ of the s-th episode. We can sample a parameter $\boldsymbol{\theta}_s$ from $\Pr_{s-1}(\boldsymbol{\theta} \mid \mathbf{Y}_{1:I}^{1:K}, \mathcal{D}_{s-1})$ and formulate a reward functions $R_p(s_p, a, s_p' \mid \boldsymbol{\theta}_s)$ at the s-th episode of the Bayesian active RL. Then, we can rely on the Q-learning to explore the optimal policy π_s^* with $R_p(s_p, a, s_p' \mid \boldsymbol{\theta}_s)$ and obtain the optimal trajectory ρ_s^*.

The Thompson sampling based query strategy can trade off the exploration and exploitation of the Bayesian active RL. Under this query strategy, a trajectory is explored based on how it is likely (under the posterior) to be optimal. On one side, the sampled parameter $\boldsymbol{\theta}_s$ can be any value of the posterior distribution $\Pr_{s-1}(\boldsymbol{\theta} \mid \mathbf{Y}_{1:I}^{1:K}, \mathcal{D}_{s-1})$, which may not be or close to the ground truth value of human trust model. The generated the optimal policy π_s^* and trajectory ρ_s^* under the sampled parameter $\boldsymbol{\theta}_s$ at each episode s doesn't reflect the actual human trusted optimal policy and trajectory. However, it guarantees the trajectory exploration of the product-MDP $\mathbb{G}_p$ and data variety of learning the human trust model. One the other side, the posterior distribution $\Pr_s(\boldsymbol{\theta} \mid \mathbf{Y}_{1:I}^{1:K}, \mathcal{D}_s)$ will become condensed after multiple episodes of online estimation

of human trust model parameter. Then, it will be much more likely to sample the parameter $\boldsymbol{\theta}_s$ at or close to the ground truth value of human trust model. As a result, the optimal policy π_s^* and trajectory ρ_s^* eventually will be stabilized around the human actually trusted ones.

Alg. 8 summarizes the overall process of integrating the Thompson sampling into the Bayesian Active RL. The input of the algorithm is the product-MDP $\mathcal{G}_p$ and the initial prior distribution $\mathrm{Pr}_0(\boldsymbol{\theta} \mid \mathcal{D}_0)$ of the human trust model. The outputs of the algorithm are the optimal policy π_S^* and trajectory ρ_S^*. Lines 4 - 6 describe the process of Thompson sampling based query strategy generating the optimal policy and trajectory at the s-th episode of the Bayesian active RL. Lines 7 - 10 present the Bayesian inference and MCMC based online estimation of human trust model parameter.

Algorithm 8 Bayesian Active RL with Thompson Sampling

Input: prior $\mathrm{Pr}_0(\boldsymbol{\theta} \mid \mathcal{D}_0)$, product-MDP $\mathcal{G}_p$
Output: updated optimal policy π_S^*, posterior $\mathrm{Pr}_S(\boldsymbol{\theta} \mid \mathcal{D}_S)$

1: **function** BARL-TS($\mathrm{Pr}_0(\boldsymbol{\theta} \mid \mathcal{D}_0)$, $\mathcal{G}_p$)
2: $\quad\boldsymbol{\Theta}_0 \leftarrow$ sample $\boldsymbol{\theta} \sim \mathrm{Pr}_0(\boldsymbol{\theta} \mid \mathcal{D}_0)$
3: $\quad$**for** $s = 1, \cdots, S$ **do**
4: $\quad\quad\theta_s \rightarrow \mathrm{Pr}_{s-1}(\boldsymbol{\theta} \mid \mathcal{D}_{s-1})$ ▷ Sample a parameter
5: $\quad\quad R_p(s_p, a, s_p' \mid \boldsymbol{\theta}_s) \leftarrow \boldsymbol{\theta}_s$ ▷ Update reward function
6: $\quad\quad\rho_s^*,\ \pi_s^* \leftarrow$ Q-LEARNING($\mathcal{G}_p, R_p$)
7: $\quad\quad$Obtain data $\mathbb{Z}_{1:I}^{1:K}, \mathbf{Y}_{1:I}^{1:K}$ from path ρ_s^*
8: $\quad\quad\boldsymbol{\Theta}_s \leftarrow \mathrm{MCMC}\big(\mathrm{Pr}_{s-1}(\boldsymbol{\theta} \mid \mathcal{D}_{s-1}), \mathbf{Y}_{1:I}^{1:K}, \mathbb{Z}_{1:I}^{1:K}\big)$
9: $\quad\quad\mathcal{D}_s \leftarrow \mathcal{D}_{s-1} \cup \{(\mathbf{Y}_{1:I}^{1:K}, \mathbb{Z}_{1:I}^{1:K})\}$
10: $\quad\quad\mathrm{Pr}_s(\boldsymbol{\theta} \mid \mathcal{D}_s) \leftarrow \boldsymbol{\Theta}_s$ ▷ Posterior
11: $\quad$**end for**
12: $\quad$**return** $\pi_S^*, \mathrm{Pr}_S(\boldsymbol{\theta})$
13: **end function**

5.5.2.2 Workload-based Query

Many active learning works focus on increasing the diversity of data annotation and decreasing the uncertainty of parameter estimation. However, the selected trajectories under these query strategies can be difficult for human to annotate. Human can have high cognition workload to label correctly and objectively [66]. Limited active learning works discuss the human annotation workload that comes from the data annotation difficulty. According to [70, 66], the labeling difficulty is depends the feature dissimilarity between data objects. A larger feature difference between two sequentially labeled data objects brings in higher human cognition workload. Therefore, we construct a utility function $\mathrm{W}_a(\mathbb{Z}_{1:I}^{1:K} \mid \rho)$ to describe the data annotation workload for the data $\mathbb{Z}_{1:I}^{1:K}$ of a trajectory ρ,

$$\mathrm{W}_a(\mathbb{Z}_{1:I}^{1:K} \mid \rho) = \sum_{i=1}^{I}\sum_{k=1}^{K}\left(1 - e^{-\|\mathbf{z}_i^k - \mathbf{z}_i^{k-1}\|}\right). \tag{5.13}$$

A trajectory ρ will have a high data annotation workload if the neighboring time-steps of data $\mathbf{z}_i^k$ are dissimilar, i.e., a large Euclidean distance.

Furthermore, the human can subject to different levels of operational difficulty and workload when operating the robot r_0 along the trajectories. The learning cost in our human-MRS collaborative task can additionally come from the human operational workload in collaborating with the autonomous robots $r_i,\ i = 1, \cdots, I$ of the MRS. According to [52, 82], human operation workload in collaborating with robots is related to the task complexity and duration of task performing time. The more complex of a task and longer task performing time, the higher of the human operational workload. We relate the task complexity of a trajectory to its environmental attributes $\mathbb{Z}_{1:I}^{1:K}$ and build a utility function $\mathrm{W}_o(\mathbb{Z}_{1:I}^{1:K} \mid \rho)$ to describe the an operational workload as follows,

$$\mathrm{W}_o(\mathbb{Z}_{1:I}^{1:K} \mid \rho) = \sum_{i=1}^{I}\sum_{k=1}^{K}\left(1 - e^{-\|\mathbf{z}_{i,max}^k - \mathbf{z}_i^k\|k}\right), \tag{5.14}$$

where $\mathbf{z}_{i,max}^k$ is the maximal value of the environment attributes. A trajectory ρ will have a high operational workload if the data $\mathbf{z}_i^k$ is less advantageous, i.e., a lower value of the environment attributes.

We develop a workload-based query strategy for the Bayesian active RL to generate the trajectory for human's annotation of the human trust shaped reward. There are three steps for this process. First, we formulate a set of reward functions $R_p(s_p, a, s_p' \mid \boldsymbol{\theta}_{s,l}),\ \boldsymbol{\theta}_{s,l} \in \boldsymbol{\Theta}_{s-1}^b,\ l \in \mathbb{N}^+$ at the s-th episode of the active RL, where $\boldsymbol{\theta}_{s,l} \sim \mathrm{Pr}_{s-1}(\boldsymbol{\theta} \mid \mathcal{D}_{s-1})$. Second, we rely on a general Q-learning algorithm to roll out a pool of optimal policy $\mathcal{H}_\pi$ and trajectories $\mathcal{H}_\rho$ with the above set of reward functions. This process can be summarized with Alg. 9. The input of the algorithm is the product-MDP $\mathcal{G}_p$ and the posterior distribution $\mathrm{Pr}_{s-1}(\boldsymbol{\theta} \mid \mathcal{D}_{s-1})$ of the human trust model at the $s-1$-th episode of the active RL. The posterior distribution $\mathrm{Pr}_{s-1}(\boldsymbol{\theta} \mid \mathcal{D}_{s-1})$ is taken as the prior information of the human trust shaped reward function at the s-th episode of the active RL. The outputs of the algorithm are a set of explored optimal policies $\mathcal{H}_\pi$ and trajectories $\mathcal{H}_\rho$. Each of the optimal policy and trajectory are generated based on a sampled value $\boldsymbol{\theta}_{s,l}$ in $\boldsymbol{\Theta}_{s-1}^b$ and its resultant reward function $R_p(s_p, a, s_p' \mid \boldsymbol{\theta}_{s,l})$, see lines 4 - 6. Third, a workload-based query strategy compares among the rollout trajectories $\mathcal{H}_\rho$ and selects the one that has the minimal data annotation

Algorithm 9 Trajectory Rollout of Product-MDP

Input: product-MDP $\mathcal{G}_p$, posterior $\Pr_{s-1}(\boldsymbol{\theta} \mid \mathcal{D}_{s-1})$
Output: updated policy $\mathcal{H}_\pi$, trajectory pool $\mathcal{H}_\rho$

```
function ROLLOUT(Pr_{s-1}(θ | D_{s-1}), G_p)
    H_ρ ← ∅, H_π ← ∅                                   ▷ Pool of path & policy
    Θ^b_{s-1} → Pr_{s-1}(θ | D_{s-1})
    for θ_{s,l} ∈ Θ^b_{s-1} do
        R_p(s_p, a, s'_p | θ_{s,l}) ← θ_{s,l}          ▷ Reward function
        ρ_l, π_l ← Q-LEARNING(G_p, R_p)
        H_ρ ← H_ρ ∪ {ρ_l}, H_π ← H_π ∪ {π_l}
    end for
    return H_ρ, H_π
end function
```

and operational workload with the following equation

$$\rho^* = \underset{\rho_j \in \mathcal{H}_{\rho,s}}{\operatorname{argmax}} \left[\xi_1 \mathrm{W}_a(\mathbb{Z}_{1:I}^{1:K} \mid \rho_j) + \xi_2 \mathrm{W}_o(\mathbb{Z}_{1:I}^{1:K} \mid \rho_j) \right]. \tag{5.15}$$

We integrate the online update of reward function in Sec. 5.5.1 with the trajectory query strategies in Sec. 5.5.2 and summarize the overall process in Alg. 10. The input and the output of the algorithm are the same to the Alg. 8. Line 4 rolls out a pool of optimal trajectories with Alg. 9. Line 5 refers to Eqn. (5.15) and selects the trajectory that has the minimal workload. Lines 6 - 9 present the Bayesian inference and MCMC based online estimation of human trust model parameter. Line 10 outputs the optimal policy with respect to the selected trajectory in the pool $\mathcal{H}_{\rho,s}$.

Algorithm 10 Bayesian Active RL with Workload-based Query

Input: prior $\Pr_0(\boldsymbol{\theta} \mid \mathcal{D}_0)$, product-MDP $\mathcal{G}_p$
Output: updated policy π_S, posterior $\Pr_S(\boldsymbol{\theta} \mid \mathcal{D}_S)$

```
1  function BARL-W(Pr_0(θ | D_0), G_p)
2      Θ_0 ← sample θ ~ Pr_0(θ | D_0)
3      for s = 1, ··· , S do
4          H_{ρ,s}, H_{π,s} ← ROLLOUT(Θ_{s-1}, G_p)
5          ρ*_s ← Query(H_{ρ,s}, Θ_{s-1})                ▷ Query a path
6          Obtain data Z^{1:K}_{1:I}, Y^{1:K}_{1:I} from path ρ*_s
7          Θ_s ← MCMC(Pr_{s-1}(θ | D_{s-1}), Y^{1:K}_{1:I}, Z^{1:K}_{1:I})
8          D_s ← D_{s-1} ∪ {(Y^{1:K}_{1:I}, Z^{1:K}_{1:I})}
9          Pr_s(θ | D_s) ← Θ_s                           ▷ Posterior
10         π_s ← H_{π,s}                                 ▷ Update policy
11     end for
12     return π_S, Pr_S(θ)
13 end function
```

The probability distribution of the trust model parameters $\boldsymbol{\theta}$ will become more certain as more obser-

vation data on environmental attributes $\mathbb{Z}_{1:I}^{1:K}$ and human trust change $\mathbf{Y}_{1:I}^{1:K}$ become available. Eventually, the corresponding human trust shaped reward function can be very certain, thus generates deterministic trajectory. This deterministic trajectory and the corresponding policy are the desired solution for the human-MRS collaborative task.

5.5.2.3 Decision Field Theory based Query

A decision maker's preference for each option in multialternative choices evolves by comparing among options for their evaluations on respective attributes over time during his/her deliberation process (e.g., uncertainty and risks) [71]. In our work, trajectory exploration based on human's trust can potentially increase the human's willingness to collaborate with the autonomous system and reduce human workload. The human's trust x_i^k can reflect his/her willingness to travel along the path. Therefore, we formulate the human preference value for every path ρ_j based on the predicted human trust value. Then, the active RL can assign the preferable trajectory for human-MRS to perform task and human to label trust.

According to Alg. 2, we can roll out a set of trajectories $\mathcal{H}_{\rho,s}$ based on the lower-confidence and upper-confidence bound values of posterior distribution $\Pr_{s-1}(\boldsymbol{\theta} \mid \mathbf{Y}_{1:I}^{1:K}, \mathbf{X}_{1:I}^{1:K}, \mathbb{Z}_{1:I}^{1:K})$. We quantify human preference value f_s of path ρ_j among multialternative choices $\mathcal{H}_{\rho,s} = \{\rho_1, \cdots, \rho_J\}$ as follows

$$
\begin{aligned}
f_s(\rho_j) &= \lambda \cdot f_{s-1}(\rho_j) + \Delta x_s(\rho_j), \\
\Delta x_s(\rho_j) &= x_s(\rho_j) - \frac{1}{J-1}\left(\sum_{\tilde{j}=1}^{J} x_s(\rho_{\tilde{j}}) - x_s(\rho_j)\right), \\
x_s(\rho_j) &= \frac{1}{KI}\sum_{k=1}^{K}\sum_{i=1}^{I} \boldsymbol{\beta}^\top \tilde{\mathbf{z}}_i^k,
\end{aligned}
\tag{5.16}
$$

where $f_s(\rho_j)$ is the human's dynamic preference value of path ρ_j over all the other paths, the coefficient λ determines the memory of the previous preference $f_{s-1}(\rho_j)$ over the time interval, $\boldsymbol{\beta}^\top \tilde{\mathbf{z}}_i^k$ is the predicted trust value of robot r_i at the k-th step while traveling along path ρ_j, $x_s(\rho_j)$ is the predicted trust value of a path ρ_j at the s-th episode of the active RL, and $\Delta x_s(\rho_j)$ describes the advantages of path ρ_j over all the other paths regarding their predicted trust value at the s-th episode.

If $f_s(\rho_j) > 0$ conditional on $\boldsymbol{\Theta}_{s-1}$, it means path ρ_j is preferred rather than the other paths at the s-th episode. Then, we integrate the above trust-based preference equations into the trajectory exploration

strategy of active RL as

$$\rho^* = \underset{\rho_j \in \mathcal{H}_{\rho,s}}{\operatorname{argmax}} \Pr\left(f_s(\rho_j) > 0 \mid \mathbf{\Theta}_{s-1}\right) \tag{5.17}$$

This function describes the probability of path ρ_j being preferred rather than others under the current belief of human trust model parameters. We take the most likely preferable path $\rho^* \in \mathcal{H}_{\rho,s}$ for the human-MRS at each episode of the active RL.

The trajectory exploration strategy explore the state space of the product-MDP. Initially, the trust model is less certain due to the limited human-MRS collaboration experience. The trajectory exploration strategy can hardly distinguish the human's preference to different paths and can only identify the more obviously advantageous paths. After a period of collaboration, more data is obtained and hence the trust evaluation becomes more determined and consistent. As a result, the trajectory exploration strategy is capable of comparing among the less obviously advantageous paths and finds the preferable one.

5.5.2.4 Situational Awareness-based Query

In each episode of the active RL, it is essential for human to recognize the environment attributes that the MRS are subjecting to and allocate appropriate amount of attention to label the trust value. Situational awareness describes the human's perception of the offroad environment, understanding of the situation, as well as prediction of the future status [28]. It influences the demand and supply of human attention in the collaborative task. Therefore, it is also significant for the query strategy to consider the utility of human situational awareness along a trajectory during the active RL.

Approximate entropy (ApEn) describes the amount of regularity and the unpredictability of fluctuations over time-series data. A higher regularity of the trajectory is more favorable for human-MRS to recognize the environment and build the situational awareness. Hence, we can use the ApEn to identify the trajectory that has repetitive pattern regarding the environment attributes [24]. Denote vector $\mathbf{z}_{i,m}^{1:K}$ as an inclusion of all the K values of the m-th environment attribute that robot r_i is going to subject to along a trajectory ρ. We can extract $u \in \mathbb{N}^+$ vectors $\mathbf{z}_{i,m}^{1:K-u+1}, \cdots, \mathbf{z}_{i,m}^{u:K}$ and arrange them to be a matrix $\mathcal{Z}_{i,m}^{1:K}(u)$ with the shape $(K-u+1) \times u$. Then, we can define

$$dist_{\kappa,k} = \max |\mathcal{Z}_{i,m}^{k}(u) - \mathcal{Z}_{i,m}^{\kappa}(u)|$$

to quantify the distance between any two (e.g., the k-th and κ-th) columns' elements of the matrix $\mathcal{Z}_{i,m}^{1:K}(u)$. We claim that the k-th element $\mathcal{Z}_{i,m}^{k}(u)$ and κ-th element $\mathcal{Z}_{i,m}^{\kappa}(u)$ are similar if $dist_{\kappa,k} < \zeta$, where ζ is a

threshold. Next, we can calculate the ratio of column vectors $\mathcal{Z}_{i,m}^{\kappa}(u), \kappa = 1, \cdots, K$ that is similar to the column vector $\mathcal{Z}_{i,m}^{k}(u)$ with

$$\mathcal{C}(\mathcal{Z}_{i,m}^{k}, u, \zeta) = \frac{1}{K-u+1} \sum_{\kappa=1}^{K-u+1} \mathbb{I}(dist_{k,\kappa} < \zeta),$$

where $\mathbb{I}(\cdot)$ is an indicator function which equals to 1 if the condition is true and 0 otherwise. As a result, we can obtain the regularity of the K values of the vector $\mathbf{z}_{i,m}^{1:K}$ by referring to the similarity ratio $\mathcal{C}(\mathcal{Z}_{i,m}^{k}, u, \zeta)$ of every column vector $\mathcal{Z}_{i,m}^{k}(u)$. The regularity can be formally defined as

$$\phi(\mathbf{z}_{i,m}^{1:K}, u, \zeta) = \frac{1}{K-u+1} \sum_{k=1}^{K-u+1} \log \mathcal{C}(\mathcal{Z}_{i,m}^{k}, u, \zeta). \tag{5.18}$$

Finally, we can quantify the situational awareness level of a trajectory ρ with the ApEn

$$\mathrm{ApEn}\left(\mathbb{Z}_{1:I}^{1:K} \mid \rho\right) = \sum_{i=1}^{I} \sum_{m=1}^{M} \left(\phi(\mathbf{z}_{i,m}^{1:K}, u, \zeta) - \phi(\mathbf{z}_{i,m}^{1:K}, u+1, \zeta)\right). \tag{5.19}$$

The function $\mathrm{ApEn}\left(\mathbb{Z}_{1:I}^{1:K} \mid \rho\right)$ works as a quantification of the rate of regularity in the robot perceived environment attribute values of a trajectory ρ. The environment attribute of a trajectory ρ will have a high regularity if $\mathrm{ApEn}\left(\mathbb{Z}_{1:I}^{1:K} \mid \rho\right)$ has a low value; and a low regularity if the ApEn is large.

Each candidate trajectory in the pool $\mathcal{H}_{\rho}$ has a corresponding ApEn. In an episode of active RL that is guided by the situational awareness based query strategy, we assign the trajectory ρ^* that has the minimum ApEn for human to label the data in the subsequent human-MRS collaboration.

$$\rho^* = \underset{\rho_j \in \mathcal{H}_{\rho,s}}{\operatorname{argmin}} \, \mathrm{ApEn}(\mathbb{Z}_{1:I}^{1:K} \mid \rho_j) \tag{5.20}$$

5.6 Case Study: Bayesian Active Reinforcement Learning for Human Multi-Robot Bounding Overwatch

In this chapter, we extend the case study of Chapter 4.6 on human multi-robot collaborative bounding overwatch tasks in offroad environment. We select different local scenarios of the offroad environment to evaluate the effectiveness of our proposed trust-based active RL framework.

5.6.1 Task Specification of Multi-robot in Offroad Environments

In general, a real-time perception and replanning strategy may not be sufficient to solve the safe planning problem in offroad scenarios. It is necessary to encode a traversability-based requirement into a task specification and prescribe the safe robot behaviors before the real-time perception and replanning. We use the geological height information of an offroad area to create the traversability-related (including obstacle avoidance) robot motion specifications. We can identify the obstacle areas with trees and buildings in the surface map. We label them with "obs" and require that "robots always avoid the areas labeled with obs", where LTL$_f$ formula is $\Box\neg obs$. Then, we consider the situation that robot cannot travel to the high elevation due to the over steep ramps. We label the high elevation areas with "$peak$" and give the requirement that "robots always avoid the motion to a $peak$, which can be encoded with LTL$_f$ formula $\Box\neg peak$. Finally, we synthesize a complex LTL$_f$ formula $\Box\neg peak \wedge \Box\neg obs$ to include all the concerned situations in an uneven terrain.

Besides the traversability, the robots can be subject to the environmental invisibility issues in the area with dense vegetation. We can derive a map about the surface objects' height information from the DSM of an area. Then, we label the low visibility areas with "veg" and require that "robots always avoid these veg regions", which has the LTL$_f$ $\Box\neg veg$. In the bounding overwatch, the robots need to protect themselves from being discovered by the opponents. To enable the robots to have the capability of avoiding the detection of opponent, we can label the interest areas, such as every neighbor of the vegetation, or gully area, with "int" and give the reachability requirement for the robots "robots reach the int area besides a specification φ". Then, we can use an LTL$_f$ $\Box\neg veg \wedge (\Diamond int \wedge \bigcirc\varphi)$ to include all the concerned situations in a vegetation dense terrain.

More specifications can be encoded into the task specification, such as surveillance or sequentially visit a set of labeled regions, avoid moving obstacles, etc. These motion specifications can reduce the risks of being trapped in to the unfavorable regions though does not directly improve the capability of robot hardware in performing the motion task.

5.6.2 Experiment Procedure

We deploy the human-MRS to navigate with the proposed active RL framework under the above experimental setup. The experiment will involve both the simulated human agent and the real human subject. Every episode of the active RL follows the six following steps: (1) A discrete path for the human-MRS is

generated according to a query strategy; (2) the three-robot formed subteam autonomously navigates from the current cell to a temporary destination in the neighboring cell along the selected discrete path. The team then stops and waits for the human-operated robot to catch up; (3) the human operator manipulates the manned ground robot to get close to the autonomous robots along the same discrete path; (4) meanwhile, the autonomous robots overwatch the surrounding environment; (5) the human operator provides trust change in each autonomous robots by referring to the recorded traversability and visibility information of autonomous robots (see Fig. 4.7); (6) the operator repeats steps (2) - (4) until all the ground robots reach the ultimate destination.

5.6.3 Simulated Human Agent

We select a local scenario of the offroad environment as an example, see Figs. 5.5. In this scenario, we require the human-MRS to avoid the "peak" (high elevation) regions as obstacles and utilize the gully regions (int) as a shield while reaching a "$dest$" labeled cell. Therefore, we assign an LTL$_f$ $\varphi = \Box\neg peak \wedge (\Diamond int \wedge \bigcirc\Diamond dest)$ for the human-MRS to satisfy. We assume a simulated human has the known ground truth value of the trust model parameters $\boldsymbol{\beta}_{\text{true}} = [\beta_{-1,\text{true}},\ \beta_{0,\text{true}},\ \beta_{1,\text{true}},\ \beta_{2,\text{true}},\ b_{\text{true}}]^\top$, $\delta^2_{w,\text{true}}$ and $\delta^2_{v,\text{true}}$. We rely on this simulated human agent to provide trust change value in every autonomous robot.

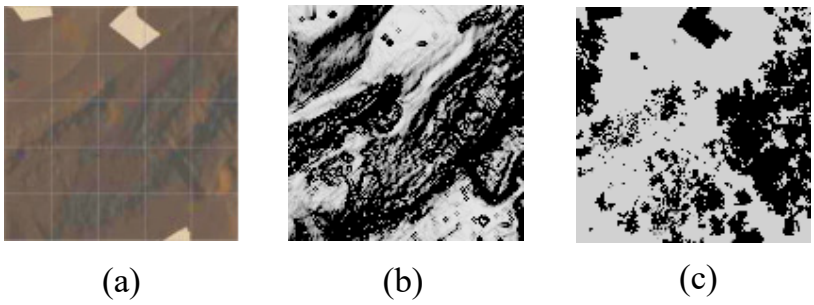


Figure 5.5: Scenario 1: (a) Top view of scenario. (b) Traversability map of the scenario. (c) Visibility map of the scenario.

We provide a non-informative initial prior distribution for the model parameter $\boldsymbol{\beta}$, which has a large variance value $\Sigma^{(0)}$ associated with the randomly assigned mean value $\boldsymbol{\beta}^{(0)}$. We rely on the Bayesian active RL to concurrently plan the optimal trajectory for human agent and learn the human trust-based reward function. Four query strategies (Thompson sampling, workload-based, decision field theory, situational awareness-based) are applied in the Bayesian active RL. Fig. 5.6 show the generated optimal trajectory pool and selected trajectory in the key episodes.

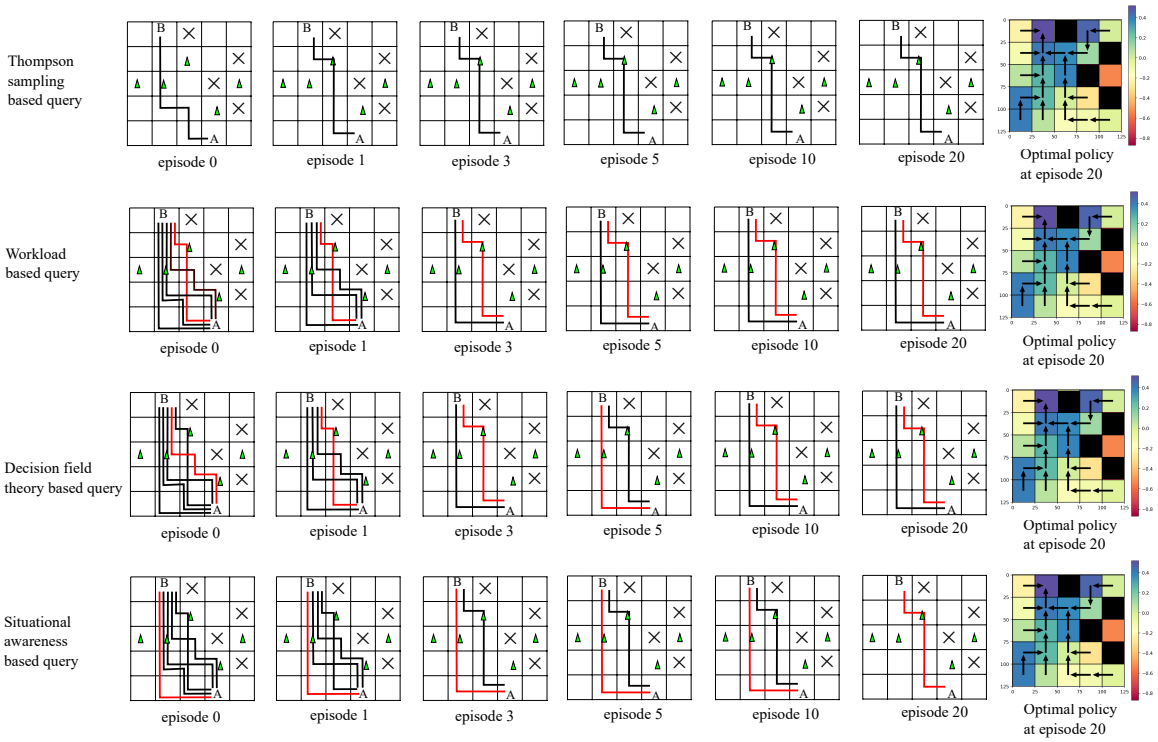
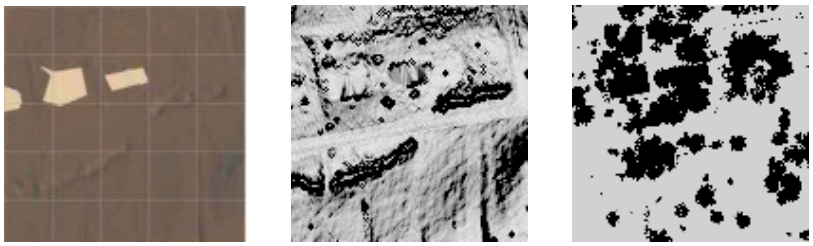


Figure 5.6: The optimal trajectory pool of Bayesian active RL and the selected optimal trajectory (red) of each query strategy. The right most is the ground truth reward function under the simulated human agent. The arrows depict the optimal policy after 20 episodes.

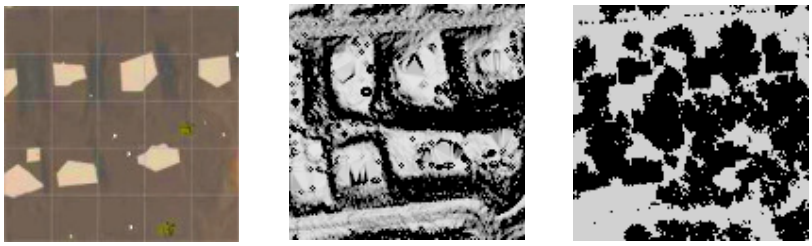
The experiment results with simulated human agents demonstrate that the Bayesian active RL can condense the optimal trajectory pool for human-MRS collaborative offroad motion task. In each episode of Bayesian active RL, different query strategy will select different optimal trajectory based on its own utility function for human to annotate the trust data.

In addition, we also show the scalability of the Bayesian active RL for different scenarios. Scenarios of Fig. 5.7 and Fig. 5.8 are selected and have different geological information from the scenario of Fig. 5.5. The corresponding traversability and visibility map are also associated.



(a) (b) (c)

Figure 5.7: Scenario 2: (a) Top view of scenario. (b) Traversability map of the scenario. (c) Visibility map of the scenario.



(a) (b) (c)

Figure 5.8: Scenario 3: (a) Top view of scenario. (b) Traversability map of the scenario. (c) Visibility map of the scenario.

We use the same simulated human agent to simulate the Bayesian active RL algorithm for the two new scenarios. The optimal trajectory pool results are shown in Figs. 5.9 (bottom) and 5.10 (bottom). We can observe that different amount of optimal trajectories are obtained based on the specific scenario. Scenario 2 with the assigned LTL_f is more sensitive to the reward function change. Hence, the Bayesian active RL can find the ultimate optimal trajectory at episode 6. In comparison, the algorithm cannot find a unique optimal trajectory at the end of episode 20 for the scenario 3 (Fig. 5.10 (bottom)). Therefore, the efficiency of Bayesian active RL is related to the sensitivity of the environment in producing the optimal trajectory.

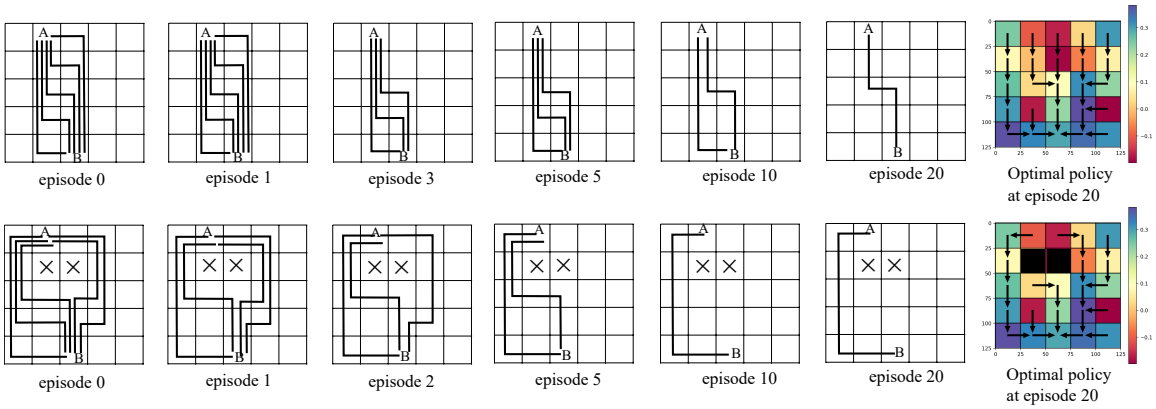


Figure 5.9: Trust-based Bayesian active RL results in scenario 2: (top) update of optimal trajectory pool under LTL$_f$ "$\Diamond dest$". (bottom) update of optimal trajectory pool under LTL$_f$ $\Box\neg peak \wedge (\Diamond gully \wedge \bigcirc\Diamond dest)$.

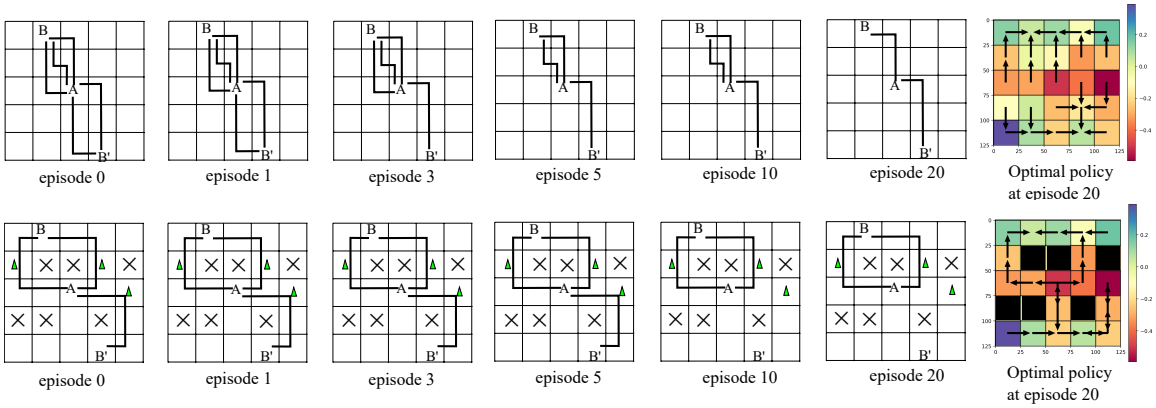


Figure 5.10: Trust-based Bayesian active RL results in scenario 3: (top) update of optimal trajectory pool under LTL$_f$ "$\Diamond dest$". (bottom) update of optimal trajectory pool under LTL$_f$ $\Box\neg peak \wedge (\Diamond int \wedge \bigcirc\Diamond dest)$.

Furthermore, we provide a benchmark task specification for each scenario by simply assigning the human-MRS to reach a "$dest$" labeled cell region with an LTL$_f$ "$\Diamond dest$". The optimal trajectory pool results are shown in Figs. 5.9 (top) and 5.10 (top). We can observe that different optimal trajectories from the "bottom" are obtained based on the benchmark LTL$_f$ specification. The results show that more complex task specifications encoding humans' safety requirements can change the optimal trajectory for human-MRS collaborative offroad motion task.

5.7 Conclusion

In this chapter, we developed a trust-based Bayesian active RL framework for a human multi-robot collaborative system to accomplish an offroad motion task. We first capture the human trust dynamics evolution in the motion task with a computational human-MRS trust model, which can encode the human's trust in the robots as a reward function of the labeled MDP of human-MRS. Then, we utilized LTL$_f$ formulae to encode the human's task requirements for MRS, such as the motion reachability and safety, in the offroad environment. The LTL$_f$ formulae plus the labeled MDP of the robots' motion behaviors synthesize a product-MDP for the human-MRS, which guarantees the provably safe behaviors of human-MRS in the task performing. Next, Thompson sampling based query, workload-based query, decision-field theory, and situational awareness-based query, are developed for human-MRS to simultaneously learn the trust-based reward function and find the optimal trajectory. In the experiment, we present a case study on human-MRS bounding overwatch, a complex multi-robot offroad motion task, to illustrate the effectiveness of our proposed framework.

Chapter 6

Conclusions and Future Work

6.1 Conclusions

The task and motion planning of MRS is extremely complicated and computationally expensive, especially for temporal logic tasks. This dissertation first developed a task decomposition framework to break a global temporal logic task into smaller task pieces, which can significantly reduce the computational space. Then it integrates human trust in a robot with the autonomous motion of MRS, which can improve the human-MRS collaboration performance. Trust-based task allocation and planning are also discussed to demonstrate the usability of trust in a human-MRS collaborated task. The detailed contributions of each chapter are as follows.

Chapter 2 presents a human-supervised task allocation and motion planning framework for MRS to perform multiple parallel subtasks in a human-like decision-making manner. These subtasks are described by automata and conjunction with MRS to synthesize a task allocation automaton. Transitions of task allocation automaton are associated with the estimations of robot performance and human cognitive workload. They are combined with a DBN human-robot trust model, and a maximal trust-encoded task allocation path can be found. This path reflects the maximum trust of the human in the task assignment of MRS. Symbolic motion planning (SMP) is implemented for each robot after the task allocation. The task reallocation is triggered after an action is completed with human permission. The above process is demonstrated effective for MRS task allocation by a simulation with five robots and three parallel subtasks.

Chapter 3 presented a top-down framework for the parallel task and motion planning of MRS to achieve a global task specification with automaton theories. We first extracted a set of sub-automata from the

global task specification and introduced an iterative parallel decomposition algorithm to decompose each of the extracted sub-automata. The decomposition components were a unique set of smallest parallel subtask automata. Each component was assigned a set of heterogeneous robots. A maximum amount of individual and concurrent SPA was then synthesized from these subtask automata and the capability transition systems of the assigned robots. Each SPA provided a minimal-cost task plan for the MRS, and all the task plans were executed in parallel. The task planning process provided a higher level of parallelism in task plans for MRS compared with the centralized approach that directly synthesizes task plans with the global task automaton and robot transition systems. The parallel task planning process was also proved to be more computationally efficient compared to the centralized approach. Furthermore, dynamic concurrent execution was performed for the task plan from each parallel SPA in order to improve the concurrency of the task-performing process.

Chapter 4 developed two LSS models to capture the quantitative relationship between human trust in MRS and offroad environmental characteristics, such as traversability and line of sight. One LSS model, i.e., "CTM1", quantifies the causality of trust by assuming that human's trust in each leading robot does not influence the trust in its succeeding robot in the line formation of MRS team; while the other LSS model, i.e., "CTM2", assumes the existence of such a causality between robots. Bayesian inference and MCMC sampling are used to derive the parameters of each computational trust model. In addition, Bayesian optimization-based experimental design was applied to collect the data, update the trust model parameters and obtain the optimal path for the MRS motion task.

Chapter 5 proposed a trust-based Bayesian active RL algorithm to deal with the human-MRS collaborated offroad motion task under LTL_f specifications. The algorithm shaped the reward functions for human-MRS collaborative offroad motion by referring to the human-to-multi-robot trust model and LTL_f specifications. The Bayesian-based online updating of the reward functions and active query for the optimal trajectory went concurrently to find the stabilized optimal policies and trajectories for human-MRS. The bonding overwatch experiments with the simulated human are conducted in different scenarios to validate the usability of the proposed algorithm.

6.2 Limitations and Future Work

The current work has its limitations. We rely on human operators to report their trust change by sliding the button on the HCI. It is economical but could bring uncertainties in the data collection. Some participants tend to slide the button aggressively while others are more conservative, though both situations

may actually have the same level of trust change. The different measurement scales from participants can directly affect the estimation result of model parameters and the resultant preferable path. Therefore, a future investigation with objective measurements, such as psychophysical signals, of the trust value can reduce the bias [1].

In addition, human decision-making in an adversarial environment can be different from the situation without adversaries. A human-in-the-loop RL algorithm for the multi-robot in the presence of adversaries is worth investigating.

Appendices

Appendix A Equation Derivation

A.1 Derivation of computational trust model without inter-robot trust influence

In Sec. 4.4.1, the computational trust model does not consider the inter-robot trust influence. We have model parameters $\boldsymbol{\beta} = [\beta_0,\ \beta_1,\ \cdots,\ \beta_M,\ b]$, and $\delta_w^2,\ \delta_v^2$ to estimate. According to the Bayes' theorem, we can derive the full posterior distribution of model parameters $\boldsymbol{\theta} = (\boldsymbol{\beta},\ \delta_w^2,\ \delta_v^2)$ as follows,

$$\pi(\boldsymbol{\beta}, \delta_w^2, \delta_v^2 \mid \mathbf{Y}_{1:I}^{1:K}, \mathbf{X}_{1:I}^{1:K}, \mathbb{Z}_{1:I}^{1:K}) \tag{1}$$

$$= \frac{\Pr(\mathbf{Y}_{1:I}^{1:K}, \mathbf{X}_{1:I}^{1:K} \mid \mathbb{Z}_{1:I}^{1:K}, \boldsymbol{\beta}, \delta_w^2, \delta_v^2)\pi_0(\boldsymbol{\beta}, \delta_w^2, \delta_v^2)}{\Pr(\mathbf{Y}_{1:I}^{1:K}, \mathbf{X}_{1:I}^{1:K} \mid \mathbb{Z}_{1:I}^{1:K})} \tag{2}$$

$$\propto \Pr(\mathbf{Y}_{1:I}^{1:K}, \mathbf{X}_{1:I}^{1:K} \mid \mathbb{Z}_{1:I}^{1:K}, \boldsymbol{\beta}, \delta_w^2, \delta_v^2)\pi_0(\boldsymbol{\beta}, \delta_w^2, \delta_v^2), \tag{3}$$

where Eqn. (2) is proportional to Eqn. (3) since $\Pr(\mathbf{Y}_{1:I}^{1:K}, \mathbf{X}_{1:I}^{1:K} \mid \mathbb{Z}_{1:I}^{1:K})$ is a constant. Then, we can use conditional probability to decompose the likelihood function $\Pr(\mathbf{Y}_{1:I}^{1:K}, \mathbf{X}_{1:I}^{1:K} \mid \mathbb{Z}_{1:I}^{1:K}, \boldsymbol{\beta}, \delta_w^2, \delta_v^2)$ in Eqn. (3) to be

$$\Pr(\mathbf{Y}_{1:I}^{1:K}, \mathbf{X}_{1:I}^{1:K} \mid \mathbb{Z}_{1:I}^{1:K}, \boldsymbol{\beta}, \delta_w^2,\ \delta_v^2) \tag{4}$$

$$\begin{aligned} &= \Pr(\mathbf{Y}_{1:I}^{1:K} \mid \mathbf{X}_{1:I}^{1:K}, \mathbb{Z}_{1:I}^{1:K}, \boldsymbol{\beta}, \delta_w^2, \delta_v^2) \\ &\quad \Pr(\mathbf{X}_{1:I}^{1:K} \mid \mathbb{Z}_{1:I}^{1:K}, \boldsymbol{\beta}, \delta_w^2, \delta_v^2) \end{aligned} \tag{5}$$

$$= \Pr(\mathbf{Y}_{1:I}^{1:K} \mid \mathbf{X}_{1:I}^{1:K}, \delta_v^2) \Pr(\mathbf{X}_{1:I}^{1:K} \mid \mathbb{Z}_{1:I}^{1:K}, \boldsymbol{\beta}, \delta_w^2), \tag{6}$$

where Eqn. (5) is simplified to Eqn. (6) because $\mathbf{Y}_{1:I}^{1:K}$ is independent of $\mathbb{Z}_{1:I}^{1:K}, \boldsymbol{\beta}, \delta_w^2$, and $\mathbf{X}_{1:I}^{1:K}$ is independent of δ_v^2. Next, we can derive each component of Eqn. (6) as

$$\begin{aligned} &\Pr(\mathbf{Y}_{1:I}^{1:K} \mid \mathbf{X}_{1:I}^{1:K}, \delta_v^2) \\ &= \prod_{i=1}^{I} \prod_{k=1}^{K} \Pr(y_i^k \mid x_i^k, x_i^{k-1}, \delta_v^2) \\ &= (2\pi\delta_v^2)^{-\frac{IK}{2}} \exp\left(-\frac{\sum_{k=1}^{K} \sum_{i=1}^{I} (y_i^k - (x_i^k - x_i^{k-1}))^2}{2\delta_v^2} \right) \end{aligned} \tag{7}$$

and

$$
\begin{aligned}
&\Pr(\mathbf{X}_{1:I}^{1:K} \mid \mathbb{Z}_{1:I}^{1:K}, \boldsymbol{\beta}, \delta_w^2) \\
&\quad = \prod_{i=1}^{I} \prod_{k=1}^{K} \Pr(x_i^k \mid x_i^{k-1}, \mathbf{z}_i^k, \boldsymbol{\beta}, \delta_w^2) \\
&\quad = (2\pi\delta_w^2)^{-\frac{IK}{2}} \exp\left(-\frac{\sum_{k=1}^{K} \sum_{i=1}^{I} (x_i^k - \boldsymbol{\beta}^\top \tilde{\mathbf{z}}_i^k)^2}{2\delta_w^2} \right).
\end{aligned} \tag{8}
$$

As a result, we can obtain the detailed probabilistic function for the posterior distribution of model parameters as

$$
\begin{aligned}
&\pi(\boldsymbol{\beta}, \delta_w^2, \delta_v^2 \mid \mathbf{Y}_{1:I}^{1:K}, \mathbf{X}_{1:I}^{1:K}, \mathbb{Z}_{1:I}^{1:K}) \\
&= \pi_0(\boldsymbol{\theta})(2\pi\delta_w^2)^{-\frac{IK}{2}} \exp\left(-\frac{\sum_{k=1}^{K} \sum_{i=1}^{I} (x_i^k - \boldsymbol{\beta}^\top \tilde{\mathbf{z}}_i^k)^2}{2\delta_w^2} \right) \\
&(2\pi\delta_v^2)^{-\frac{IK}{2}} \exp\left(-\frac{\sum_{k=1}^{K} \sum_{i=1}^{I} (y_i^k - (x_i^k - x_i^{k-1}))^2}{2\delta_v^2} \right).
\end{aligned} \tag{9}
$$

In this paper, we can assume the initial prior distribution $\pi_0(\boldsymbol{\beta}),\ \pi_0(\delta_w^2)$, and $\pi_0(\delta_v^2)$ are independent. The reason is no information imposes the relevance between the prior distribution of $\boldsymbol{\beta},\ \delta_w^2$, and δ_v^2. Then, given a prior distribution $\pi_0(\boldsymbol{\beta})$, the posterior distribution of $\boldsymbol{\beta}$ can be obtained in the same approach of Eqns.

(1)-(9)

$$
\begin{aligned}
&\pi(\boldsymbol{\beta} \mid \mathbf{Y}_{1:I}^{1:K}, \mathbf{X}_{1:I}^{1:K}, \mathbb{Z}_{1:I}^{1:K}, \delta_w^2, \delta_v^2) \\
&= \frac{\Pr(\mathbf{Y}_{1:I}^{1:K}, \mathbf{X}_{1:I}^{1:K} \mid \mathbb{Z}_{1:I}^{1:K}, \boldsymbol{\beta}, \delta_w^2, \delta_v^2)\pi_0(\boldsymbol{\beta} \mid \delta_w^2, \delta_v^2)}{\Pr(\mathbf{Y}_{1:I}^{1:K}, \mathbf{X}_{1:I}^{1:K} \mid \mathbb{Z}_{1:I}^{1:K}, \delta_w^2, \delta_v^2)} \\
&\propto \Pr(\mathbf{Y}_{1:I}^{1:K}, \mathbf{X}_{1:I}^{1:K} \mid \mathbb{Z}_{1:I}^{1:K}, \boldsymbol{\beta}, \delta_w^2, \delta_v^2)\pi_0(\boldsymbol{\beta} \mid \delta_w^2, \delta_v^2) \\
&= \Pr(\mathbf{Y}_{1:I}^{1:K} \mid \mathbf{X}_{1:I}^{1:K}, \mathbb{Z}_{1:I}^{1:K}, \boldsymbol{\beta}, \delta_w^2, \delta_v^2) \Pr(\mathbf{X}_{1:I}^{1:K} \mid \\
&\qquad \mathbb{Z}_{1:I}^{1:K}, \boldsymbol{\beta}, \delta_w^2, \delta_v^2)\pi_0(\boldsymbol{\beta}) \\
&= \Pr(\mathbf{Y}_{1:I}^{1:K} \mid \mathbf{X}_{1:I}^{1:K}, \delta_v^2) \Pr(\mathbf{X}_{1:I}^{1:K} \mid \mathbb{Z}_{1:I}^{1:K}, \boldsymbol{\beta}, \delta_w^2)\pi_0(\boldsymbol{\beta}) \\
&\propto \Pr(\mathbf{X}_{1:I}^{1:K} \mid \mathbb{Z}_{1:I}^{1:K}, \boldsymbol{\beta}, \delta_w^2)\pi_0(\boldsymbol{\beta}) \\
&= \prod_{i=1}^{I}\prod_{k=1}^{K} \Pr(x_i^k \mid x_i^{k-1}, \mathbf{z}_i^k, \boldsymbol{\beta}, \delta_w^2)\pi_0(\boldsymbol{\beta}) \\
&= (2\pi\delta_w^2)^{-\frac{IK}{2}} \exp\left(-\frac{SSR}{2\delta_w^2}\right)\pi_0(\boldsymbol{\beta}),
\end{aligned}
\tag{10}
$$

where the probability function $\Pr(\mathbf{Y}_{1:I}^{1:K} \mid \mathbf{X}_{1:I}^{1:K}, \delta_v^2)$ is a constant term, and $SSR = ([\mathbf{X}_{1:I}^{1:K}] - [\tilde{\mathbb{Z}}_{1:I}^{1:K}]^\top\boldsymbol{\beta})^\top([\mathbf{X}_{1:I}^{1:K}] - [\tilde{\mathbb{Z}}_{1:I}^{1:K}]^\top\boldsymbol{\beta})$. Here, vector $[\mathbf{X}_{1:I}^{1:K}] = \left[x_1^1, \cdots, x_1^K, \cdots, x_I^1, \cdots, x_I^K\right]_{IK}^\top$ is the flattened form of matrix $\mathbf{X}_{1:I}^{1:K}$ along the row direction; the $IK \times (M+2)$ dimensional matrix $[\tilde{\mathbb{Z}}_{1:I}^{1:K}]$ is the flattened form of $K \times I \times (M+2)$ dimensional matrix $\tilde{\mathbb{Z}}_{1:I}^{1:K}$. Assume $\pi_0(\boldsymbol{\beta})$ is a conjugate prior, then we can select the prior $\boldsymbol{\beta} \sim N(\boldsymbol{\beta}^0, \Sigma^0)$. As a result, we can have the posterior distribution

$$
\boldsymbol{\beta} \mid \mathbf{Y}_{1:I}^{1:K}, \mathbf{X}_{1:I}^{1:K}, \mathbb{Z}_{1:I}^{1:K}, \delta_w^2, \delta_v^2 \sim N(\mathbf{E}, \mathbf{V}) \tag{11}
$$

with the variance $\mathbf{V} = ((\Sigma^0)^{-1} + [\tilde{\mathbb{Z}}_{1:I}^{1:K}]^\top[\tilde{\mathbb{Z}}_{1:I}^{1:K}]/\delta_w^2)^{-1}$, and the mean $\mathbf{E} = \mathbf{V}((\Sigma^0)^{-1}\boldsymbol{\beta}^0 + [\tilde{\mathbb{Z}}_{1:I}^{1:K}]^\top[\mathbf{X}_{1:I}^{1:K}]/\delta_w^2)$.

Let $\eta_w = 1/\delta_w^2$. Given a prior distribution $\pi_0(\eta_w)$, the posterior distribution of η_w is

$$
\begin{aligned}
&\pi(\eta_w \mid \mathbf{Y}_{1:I}^{1:K}, \mathbf{X}_{1:I}^{1:K}, \mathbb{Z}_{1:I}^{1:K}, \boldsymbol{\beta}, \delta_v^2) \\
&= \frac{\Pr(\mathbf{Y}_{1:I}^{1:K}, \mathbf{X}_{1:I}^{1:K} \mid \mathbb{Z}_{1:I}^{1:K}, \boldsymbol{\beta}, \eta_w, \delta_v^2)\pi_0(\eta_w \mid \boldsymbol{\beta}, \delta_v^2)}{\Pr(\mathbf{Y}_{1:I}^{1:K}, \mathbf{X}_{1:I}^{1:K} \mid \mathbb{Z}_{1:I}^{1:K}, \boldsymbol{\beta}, \delta_v^2)} \\
&\propto \Pr(\mathbf{Y}_{1:I}^{1:K}, \mathbf{X}_{1:I}^{1:K} \mid \mathbb{Z}_{1:I}^{1:K}, \boldsymbol{\beta}, \eta_w, \delta_v^2)\pi_0(\eta_w \mid \boldsymbol{\beta}, \delta_v^2) \\
&= \Pr(\mathbf{Y}_{1:I}^{1:K} \mid \mathbf{X}_{1:I}^{1:K}, \mathbb{Z}_{1:I}^{1:K}, \boldsymbol{\beta}, \eta_w, \delta_v^2) \Pr(\mathbf{X}_{1:I}^{1:K} \mid \\
&\qquad \mathbb{Z}_{1:I}^{1:K}, \boldsymbol{\beta}, \eta_w, \delta_v^2)\pi_0(\eta_w \mid \boldsymbol{\beta}, \delta_v^2) \\
&= \Pr(\mathbf{Y}_{1:I}^{1:K} \mid \mathbf{X}_{1:I}^{1:K}, \delta_v^2) \Pr(\mathbf{X}_{1:I}^{1:K} \mid \mathbb{Z}_{1:I}^{1:K}, \boldsymbol{\beta}, \delta_w^2)\pi_0(\eta_w) \\
&\propto \Pr(\mathbf{X}_{1:I}^{1:K} \mid \mathbb{Z}_{1:I}^{1:K}, \boldsymbol{\beta}, \eta_w)\pi_0(\eta_w) \\
&= \prod_{i=1}^{I}\prod_{k=1}^{K} \Pr(x_i^k \mid x_i^{k-1}, \mathbf{z}_i^k, \boldsymbol{\beta}, \delta_w^2)\pi_0(\eta_w) \\
&= (\frac{\eta_w}{2\pi})^{\frac{IK}{2}} \exp\left(-\eta_w \frac{SSR}{2}\right)\pi_0(\eta_w),
\end{aligned}
\tag{12}
$$

where $\Pr(\mathbf{Y}_{1:I}^{1:K} \mid \mathbf{X}_{1:I}^{1:K}, \delta_v^2)$ is also a constant in the above equations. Assume $\pi_0(\eta_w)$ is a conjugate prior, then we can have the η_w follows a gamma distribution, i.e., $\eta_w \sim$ Gamma(a_0, b_0). As a result, the posterior

$$
\pi(\eta_w \mid \mathbf{Y}_{1:I}^{1:K}, \mathbf{X}_{1:I}^{1:K}, \tilde{\mathbb{Z}}_{1:I}^{1:K}, \boldsymbol{\beta}, \delta_v^2) \sim \text{Gamma}(a_K,\ b_K), \tag{13}
$$

where hyperparameters $a_K = a_0 + \frac{IK}{2}$, and $b_K = b_0 + \frac{SSR}{2}$. Finally, we can have the variance δ_w^2 following a inverse gamma distribution, i.e., $\delta_w^2 \sim IG(a_K,\ b_K)$.

Let $\eta_v = 1/\delta_v^2$. Given a prior distribution of $\pi_0(\eta_v)$, the posterior distribution of η_v is

$$
\begin{aligned}
&\pi(\eta_v \mid \mathbf{Y}_{1:I}^{1:K}, \mathbf{X}_{1:I}^{1:K}, \mathbb{Z}_{1:I}^{1:K}, \boldsymbol{\beta}, \delta_w^2) \\
&= \frac{\Pr(\mathbf{Y}_{1:I}^{1:K}, \mathbf{X}_{1:I}^{1:K} \mid \mathbb{Z}_{1:I}^{1:K}, \boldsymbol{\beta}, \delta_w^2, \eta_v)\pi_0(\eta_v \mid \boldsymbol{\beta}, \delta_w^2)}{\Pr(\mathbf{Y}_{1:I}^{1:K}, \mathbf{X}_{1:I}^{1:K} \mid \mathbb{Z}_{1:I}^{1:K}, \boldsymbol{\beta}, \delta_w^2)} \\
&\propto \Pr(\mathbf{Y}_{1:I}^{1:K}, \mathbf{X}_{1:I}^{1:K} \mid \mathbb{Z}_{1:I}^{1:K}, \boldsymbol{\beta}, \delta_w^2, \eta_v)\pi_0(\eta_v \mid \boldsymbol{\beta}, \delta_w^2) \\
&= \Pr(\mathbf{Y}_{1:I}^{1:K} \mid \mathbf{X}_{1:I}^{1:K}, \mathbb{Z}_{1:I}^{1:K}, \boldsymbol{\beta}, \delta_w^2, \eta_v) \Pr(\mathbf{X}_{1:I}^{1:K} \mid \\
&\qquad \mathbb{Z}_{1:I}^{1:K}, \boldsymbol{\beta}, \delta_w^2, \eta_v)\pi_0(\eta_v \mid \boldsymbol{\beta}, \delta_w^2) \\
&= \Pr(\mathbf{Y}_{1:I}^{1:K} \mid \mathbf{X}_{1:I}^{1:K}, \eta_v) \Pr(\mathbf{X}_{1:I}^{1:K} \mid \mathbb{Z}_{1:I}^{1:K}, \boldsymbol{\beta}, \delta_w^2)\pi_0(\eta_v) \\
&\propto \Pr(\mathbf{Y}_{1:I}^{1:K} \mid \mathbf{X}_{1:I}^{1:K}, \eta_v)\pi_0(\eta_v) \\
&= \prod_{k=1}^{K}\prod_{i=1}^{I} \Pr(y_i^k \mid x_i^k, \eta_v)\pi_0(\eta_v) \\
&= (\frac{\eta_v}{2\pi})^{\frac{IK}{2}} \exp\left(-\frac{SSR'}{2}\eta_v\right)\pi_0(\eta_v),
\end{aligned}
\tag{14}
$$

where the probability function $\Pr(\mathbf{X}_{1:I}^{1:K} \mid \mathbb{Z}_{1:I}^{1:K}, \boldsymbol{\beta}, \delta_w^2)$ is the constant term and $SSR' = ([\mathbf{Y}_{1:I}^{1:K}] - ([\mathbf{X}_{1:I}^{1:K}] - [\mathbf{X}_{1:I}^{0:K-1}]))^\top([\mathbf{Y}_{1:I}^{1:K}] - ([\mathbf{X}_{1:I}^{1:K}] - [\mathbf{X}_{1:I}^{0:K-1}]))$. Here, vector $[\mathbf{Y}_{1:I}^{1:K}]$ is the flattened form of matrix $\mathbf{Y}_{1:I}^{1:K} = \left[y_1^1, \ \cdots, \ y_1^K, \ \cdots, y_I^1, \ \cdots, \ y_I^K\right]_{IK}^\top$ along the row direction. Assume $\pi_0(\eta_v)$ is a conjugate prior, then we can have the prior $\eta_v \sim \text{Gamma}(c_0, d_0)$. As a result, the posterior

$$
\pi(\eta_v \mid \mathbf{Y}_{1:I}^{1:K}, \mathbf{X}_{1:I}^{1:K}, \tilde{Z}_{1:I}^{1:K}, \boldsymbol{\beta}, \delta_w^2) \sim \text{Gamma}(c_K, \ d_K), \tag{15}
$$

where hyperparameters $c_K = c_0 + \frac{IK}{2}$, $d_K = d_0 + \frac{SSR'}{2}$. Finally, we can have the variance δ_v^2 following a inverse gamma distribution, i.e., $\delta_v^2 \sim IG(c_K, \ d_K)$

A.2 Kalman filter and smoother

In the Kalman filter of Eqns. (4.8) and (4.9), the one-step ahead prediction mean $\tilde{\mathbf{x}}_{1:I}^{k+1|k}$, covariance $P_{1:I}^{k+1|k}$ and the filtering mean $\tilde{\mathbf{x}}_{1:I}^{k|k}$, covariance $P_{1:I}^{k|k}$ for every step's trust $\tilde{\mathbf{x}}_{1:I}^{k}$ are as follows,

$$
\begin{aligned}
\tilde{\mathbf{x}}_{1:I}^{k+1|k} &= \tilde{\mathbf{B}}_0 \tilde{\mathbf{x}}_{1:I}^{k|k} + \tilde{\mathbf{B}}_1 U^k, \\
P_{1:I}^{k+1|k} &= \tilde{\mathbf{B}}_0 P_{1:I}^{k|k} \tilde{\mathbf{B}}_0^\top + \mathbf{diag}(\Delta_w, \mathbf{0}_{I\times I}), \\
\tilde{\mathbf{x}}_{1:I}^{k+1|k+1} &= \tilde{\mathbf{x}}_{1:I}^{k+1|k} + \boldsymbol{K}^k(\mathbf{y}_{1:I}^k - \boldsymbol{H}\tilde{\mathbf{x}}_{1:I}^{k+1|k}), \\
P_{1:I}^{k+1|k+1} &= (\mathbf{1}_{2I\times 2I} - \boldsymbol{K}^k \boldsymbol{H}) P_{1:I}^{k+1|k}, \\
\boldsymbol{K}^k &= P_{1:I}^{k+1|k} \boldsymbol{H}^\top (\boldsymbol{H} P_{1:I}^{k+1|k} \boldsymbol{H}^\top + \Delta_v)^{-1},
\end{aligned}
$$

where $U^k = [Z_{1:I,1}^k, \cdots, Z_{1:I,M}^k, \mathbf{1}_{I\times 1}]^\top$, $\boldsymbol{H} = [\mathbf{1}_{I\times I}, -\mathbf{1}_{I\times I}]$ and $\mathbf{diag}(\Delta_w, \mathbf{0}_{I\times I})$ is the diagonal matrix composed by Δ_w and $\mathbf{0}_{I\times I}$.

The Kalman smoother of Eqns. (4.8) and (4.9) follows a normal distribution with mean $\boldsymbol{\mu}^k$ and covariance $\boldsymbol{\nu}^k$

$$
\begin{aligned}
\boldsymbol{\mu}^k &= \tilde{\mathbf{x}}_{1:I}^{k|k} + J^k(\tilde{\mathbf{x}}_{1:I}^{k+1} - \tilde{\mathbf{x}}_{1:I}^{k+1|k}), \\
\boldsymbol{\nu}^k &= P_{1:I}^{k|k} - J^k P_{1:I}^{k+1|k} (J^k)^\top,
\end{aligned}
$$

where $J^k = P_{1:I}^{k|k} \tilde{\mathbf{B}}_1^\top (P_{1:I}^{k+1|k})^{-1}$.

A.3 Derivation of computational trust model with inter-robot trust influence

In Subsec. 4.4.3, the computational trust model considers the inter-robot trust influence. We have model parameters $\check{\boldsymbol{\beta}} = [\beta_{-1}, \beta_0, \beta_1, \cdots, \beta_M, b]$, and δ_w^2, δ_v^2 to estimate. The full posterior distribution of model parameters $\check{\boldsymbol{\beta}}, \delta_w^2, \delta_v^2$ is

$$
\begin{aligned}
&\pi(\check{\boldsymbol{\beta}}, \delta_w^2, \delta_v^2 \mid \mathbf{Y}_{1:I}^{1:K}, \mathbf{X}_{1:I}^{1:K}, \mathbb{Z}_{1:I}^{1:K}) \\
&\propto \Pr(\mathbf{Y}_{1:I}^{1:K} \mid \mathbf{X}_{1:I}^{1:K}, \delta_v^2) \Pr(\mathbf{X}_{1:I}^{1:K} \mid \mathbb{Z}_{1:I}^{1:K}, \check{\boldsymbol{\beta}}, \delta_w^2) \pi_0(\check{\boldsymbol{\theta}}) \\
&= \prod_{k=1}^{K} \prod_{i=1}^{I} \Pr(y_i^k \mid x_i^k, x_i^{k-1}, \delta_v^2) \prod_{k=1}^{K} \prod_{i=1}^{I} \Pr(x_i^k \mid x_{i-1}^k, \\
&\quad x_i^{k-1}, \mathbf{z}_i^k, \check{\boldsymbol{\beta}}, \delta_w^2) \pi_0(\check{\boldsymbol{\theta}}) \\
&= (2\pi\delta_v^2)^{-\frac{IK}{2}} \exp\left(-\frac{\sum_{k=1}^{K} \sum_{i=1}^{I} (y_i^k - (x_i^k - x_i^{k-1}))^2}{2\delta_v^2} \right) \\
&\quad (2\pi\delta_w^2)^{-\frac{IK}{2}} \exp\left(-\frac{\sum_{k=1}^{K} \sum_{i=1}^{I} (x_i^k - \check{\boldsymbol{\beta}}^\top \check{\mathbf{z}}_i^k)^2}{2\delta_w^2} \right) \pi_0(\check{\boldsymbol{\theta}}).
\end{aligned} \tag{16}
$$

The estimation of individual model parameters in $\check{\boldsymbol{\theta}}$ is the same with that of $\boldsymbol{\theta}$ in Appendix A.1. Given a prior distribution of $\check{\boldsymbol{\beta}}$, the posterior distribution of $\check{\boldsymbol{\beta}}$ can be obtained with the Bayes rule, i.e.,

$$
\begin{aligned}
&\pi(\check{\boldsymbol{\beta}} \mid \mathbf{Y}_{1:I}^{1:K}, \mathbf{X}_{1:I}^{1:K}, \mathbb{Z}_{1:I}^{1:K}, \delta_w^2, \delta_v^2) \\
&\propto \Pr(\mathbf{X}_{1:I}^{1:K} \mid \mathbb{Z}_{1:I}^{1:K}, \check{\boldsymbol{\beta}}, \delta_w^2) \pi_0(\check{\boldsymbol{\beta}}) \\
&= \prod_{k=1}^{K} \prod_{i=1}^{I} \Pr(x_i^k \mid x_{i-1}^k, x_i^{k-1}, \mathbf{z}_i^k, \check{\boldsymbol{\beta}}, \delta_w^2) \pi_0(\check{\boldsymbol{\beta}}) \\
&= \prod_{k=1}^{K} \prod_{i=1}^{I} (2\pi\delta_w^2)^{-\frac{1}{2}} \exp\left(-\frac{(x_i^k - \check{\boldsymbol{\beta}}^\top \check{\mathbf{z}}_i^k)^2}{2\delta_w^2} \right) \pi_0(\check{\boldsymbol{\beta}}) \\
&= (2\pi\delta_w^2)^{-\frac{IK}{2}} \exp\left(-\frac{S\check{S}R}{2\delta_w^2} \right) \pi_0(\check{\boldsymbol{\beta}}),
\end{aligned} \tag{17}
$$

where the sum square error $S\check{S}R = ([\mathbf{X}_{1:I}^{1:K}] - [\check{\mathbb{Z}}_{1:I}^{1:K}]^\top \check{\boldsymbol{\beta}})^\top ([\mathbf{X}_{1:I}^{1:K}] - [\check{\mathbb{Z}}_{1:I}^{1:K}]^\top \check{\boldsymbol{\beta}})$. Note $[\check{\mathbb{Z}}_{1:I}^{1:K}] = \left[\check{\mathbf{z}}_1^1, \cdots, \check{\mathbf{z}}_1^K, \check{\mathbf{z}}_2^1, \cdots, \check{\mathbf{z}}_I^1, \cdots, \check{\mathbf{z}}_I^K\right]_{IK\times(M+3)}^\top$ is the flattened form of matrix $\check{\mathbb{Z}}_{1:I}^{1:K}$.

Assume $\pi_0(\check{\boldsymbol{\beta}})$ is a conjugate prior, then we can set the prior distribution $\check{\boldsymbol{\beta}} \sim N(\check{\boldsymbol{\beta}}^0, \check{\Sigma}^0)$. As a result, we obtain the posterior distribution

$$
\check{\boldsymbol{\beta}} \mid \mathbf{Y}_{1:I}^{1:K}, \mathbf{X}_{1:I}^{1:K}, \mathbb{Z}_{1:I}^{1:K}, \delta_w^2, \delta_v^2 \sim N(\check{\mathbf{E}}, \check{\mathbf{V}}), \tag{18}
$$

where the variance $\check{\mathbf{V}} = ((\check{\Sigma}^0)^{-1} + [\check{\mathbb{Z}}_{1:I}^{1:K}]^\top[\check{\mathbb{Z}}_{1:I}^{1:K}]/\delta_w^2)^{-1}$, and the mean $\check{\mathbf{E}} = \check{\mathbf{V}}((\check{\Sigma}^0)^{-1}\check{\boldsymbol{\beta}}^0 + [\check{\mathbb{Z}}_{1:I}^{1:K}]^\top[\mathbf{X}_{1:I}^{1:K}]/\delta_w^2)$.

Correspondingly, the posterior distribution of η_w is

$$\begin{aligned}
&\pi(\eta_w \mid y^{1:K}, \mathbf{X}_{1:I}^{1:K}, \mathbb{Z}_{1:I}^{1:K}, \check{\boldsymbol{\beta}}, \delta_v^2) \\
&\propto \Pr(\mathbf{X}_{1:I}^{1:K} \mid \mathbb{Z}_{1:I}^{1:K}, \check{\boldsymbol{\beta}}, \eta_w)\pi_0(\eta_w) \\
&= (\frac{\eta_w}{2\pi})^{\frac{IK}{2}} \exp\left(-\eta_w \frac{S\check{S}R}{2}\right)\pi_0(\eta_w).
\end{aligned} \tag{19}$$

Choose $\pi_0(\eta_w)$ to be a conjugate prior, then we can have the prior $\eta_w \sim$ Gamma(a_0, b_0). As a result, the posterior distribution

$$\pi(\eta_w \mid \mathbf{Y}_{1:I}^{1:K}, \mathbf{X}_{1:I}^{1:K}, \mathbb{Z}_{1:I}^{1:K}, \check{\boldsymbol{\beta}}, \delta_v^2) \sim \text{Gamma}(\check{a}_K,\ \check{b}_K), \tag{20}$$

where hyperparameters $\check{a}_K = a_0 + \frac{IK}{2}$, and $\check{b}_K = b_0 + \frac{S\check{S}R}{2}$. Finally, we can have the variance δ_w^2 following a inverse gamma distribution, i.e., $\delta_w^2 \sim IG(\check{a}_K,\ \check{b}_K)$

Bibliography

[1] Kumar Akash, Wan-Lin Hu, Neera Jain, and Tahira Reid. A classification model for sensing human trust in machines using eeg and gsr. *ACM Transactions on Interactive Intelligent Systems (TiiS)*, 8(4):1–20, 2018.

[2] Kumar Akash, Katelyn Polson, Tahira Reid, and Neera Jain. Improving human-machine collaboration through transparency-based feedback–part i: Human trust and workload model. *IFAC-PapersOnLine*, 51(34):315–321, 2019.

[3] Kumar Akash, Tahira Reid, and Neera Jain. Improving human-machine collaboration through transparency-based feedback–part ii: Control design and synthesis. *IFAC-PapersOnLine*, 51(34):322–328, 2019.

[4] Riad Akrour, Marc Schoenauer, and Michèle Sebag. April: Active preference learning-based reinforcement learning. In *Joint European conference on machine learning and knowledge discovery in databases*, pages 116–131. Springer, 2012.

[5] Derya Aksaray, Austin Jones, Zhaodan Kong, Mac Schwager, and Calin Belta. Q-learning for robust satisfaction of signal temporal logic specifications. In *2016 IEEE 55th Conference on Decision and Control (CDC)*, pages 6565–6570. IEEE, 2016.

[6] Saurabh Arora and Prashant Doshi. A survey of inverse reinforcement learning: Challenges, methods and progress. *Artificial Intelligence*, 297:103500, 2021.

[7] Mohamed GB Atia, Omar Salah, and Haitham Ei-Hussieny. Ogpr: An obstacle-guided path refinement approach for mobile robot path planning. In *2018 IEEE International Conference on Robotics and Biomimetics (ROBIO)*, pages 844–849, 2018.

[8] Hebert Azevedo-Sa, Suresh Kumaar Jayaraman, Connor T Esterwood, X Jessie Yang, Lionel P Robert, and Dawn M Tilbury. Real-time estimation of drivers' trust in automated driving systems. *International Journal of Social Robotics*, pages 1–17, 2020.

[9] M Bagherian and A Alos. 3d uav trajectory planning using evolutionary algorithms: A comparison study. *The Aeronautical Journal*, 119(1220):1271–1285, 2015.

[10] Christel Baier and Joost-Pieter Katoen. *Principles of model checking*. MIT press, 2008.

[11] Christopher Banks, Sean Wilson, Samuel Coogan, and Magnus Egerstedt. Multi-agent task allocation using cross-entropy temporal logic optimization. In *2020 IEEE International Conference on Robotics and Automation (ICRA)*, pages 7712–7718. IEEE, 2020.

[12] Calin Belta, Antonio Bicchi, Magnus Egerstedt, Emilio Frazzoli, Eric Klavins, and George J Pappas. Symbolic planning and control of robot motion [grand challenges of robotics]. *IEEE Robotics & Automation Magazine*, 14(1):61–70, 2007.

[13] Matthew L Bolton, Ellen J Bass, and Radu I Siminiceanu. Using formal verification to evaluate human-automation interaction: A review. *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, 43(3):488–503, 2013.

[14] Alper Kamil Bozkurt, Yu Wang, Michael Zavlanos, and Miroslav Pajic. Model-free reinforcement learning for stochastic games with linear temporal logic objectives. *arXiv preprint arXiv:2010.01050*, 2020.

[15] Alper Kamil Bozkurt, Yu Wang, Michael M Zavlanos, and Miroslav Pajic. Control synthesis from linear temporal logic specifications using model-free reinforcement learning. In *2020 IEEE International Conference on Robotics and Automation (ICRA)*, pages 10349–10355. IEEE, 2020.

[16] Jerome R Busemeyer, Zheng Wang, James T Townsend, and Ami Eidels. *The Oxford handbook of computational and mathematical psychology*. Oxford University Press, New York, 2015.

[17] Olivier Cappé, Eric Moulines, and Tobias Rydén. *Inference in hidden Markov models*. Springer Science & Business Media, New York, 2006.

[18] Min Chen, Stefanos Nikolaidis, Harold Soh, David Hsu, and Siddhartha Srinivasa. Planning with trust for human-robot collaboration. In *Proceedings of the 2018 ACM/IEEE International Conference on Human-Robot Interaction*, pages 307–315, 2018.

[19] Yushan Chen, Xu Chu Ding, and Calin Belta. Synthesis of distributed control and communication schemes from global LTL specifications. In *Decision and Control and European Control Conference (CDC-ECC), 2011 50th IEEE Conference on*, pages 2718–2723. IEEE, 2011.

[20] Yushan Chen, Xu Chu Ding, Alin Stefanescu, and Calin Belta. Formal approach to the deployment of distributed robotic teams. *IEEE Transactions on Robotics*, 28(1):158–171, 2012.

[21] Jin-Hee Cho, Kevin Chan, and Sibel Adali. A survey on trust modeling. *ACM Computing Surveys (CSUR)*, 48(2):1–40, 2015.

[22] Jacob W Crandall, Curtis W Nielsen, and Michael A Goodrich. Towards predicting robot team performance. In *SMC'03 Conference Proceedings. 2003 IEEE International Conference on Systems, Man and Cybernetics. Conference Theme-System Security and Assurance (Cat. No. 03CH37483)*, volume 1, pages 906–911. IEEE, 2003.

[23] Jin Dai, Alessandro Benini, Hai Lin, Panos J Antsaklis, Matthew J Rutherford, and Kimon P Valavanis. Learning-based formal synthesis of cooperative multi-agent systems with an application to robotic coordination. In *Control and Automation (MED), 2016 24th Mediterranean Conference on*, pages 1008–1013. IEEE, 2016.

[24] Alfonso Delgado-Bonal and Alexander Marshak. Approximate entropy and sample entropy: A comprehensive tutorial. *Entropy*, 21(6):541, 2019.

[25] Xu Chu Ding, Marius Kloetzer, Yushan Chen, and Calin Belta. Automatic deployment of robotic teams. *IEEE Robotics & Automation Magazine*, 18(3):75–86, 2011.

[26] Kim Drnec, Amar R Marathe, Jamie R Lukos, and Jason S Metcalfe. From trust in automation to decision neuroscience: applying cognitive neuroscience methods to understand and improve interaction decisions involved in human automation interaction. *Frontiers in human neuroscience*, 10:290, 2016.

[27] Mahmoud Elfar, Haibei Zhu, Mary L Cummings, and Miroslav Pajic. Security-aware synthesis of human-uav protocols. In *2019 International Conference on Robotics and Automation (ICRA)*, pages 8011–8017. IEEE, 2019.

[28] Mica R Endsley, Stephen J Selcon, Thomas D Hardiman, and Darryl G Croft. A comparative analysis of sagat and sart for evaluations of situation awareness. In *Proceedings of the human factors and ergonomics society annual meeting*, pages 82–86, 1998.

[29] Arkady Epshteyn, Adam Vogel, and Gerald DeJong. Active reinforcement learning. In *Proceedings of the 25th international conference on Machine learning*, pages 296–303, 2008.

[30] Fatma Faruq, David Parker, Bruno Laccrda, and Nick Hawes. Simultaneous task allocation and planning under uncertainty. In *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 3559–3564. IEEE, 2018.

[31] Lu Feng, Clemens Wiltsche, Laura Humphrey, and Ufuk Topcu. Synthesis of human-in-the-loop control protocols for autonomous systems. *IEEE Transactions on Automation Science and Engineering*, 13(2):450–462, 2016.

[32] Rico Fischer and Franziska Plessow. Efficient multitasking: parallel versus serial processing of multiple tasks. *Frontiers in psychology*, 6:1366, 2015.

[33] Brian P Gerkey and Maja J Matarić. A formal analysis and taxonomy of task allocation in multi-robot systems. *The International Journal of Robotics Research*, 23(9):939–954, 2004.

[34] Meng Guo and Dimos V Dimarogonas. Multi-agent plan reconfiguration under local LTL specifications. *The International Journal of Robotics Research*, 34(2):218–235, 2015.

[35] Meng Guo and Dimos V Dimarogonas. Task and motion coordination for heterogeneous multi-agent systems with loosely coupled local tasks. *IEEE Transactions on Automation Science and Engineering*, 14(2):797–808, 2017.

[36] Sandra G Hart. Nasa-task load index (nasa-tlx); 20 years later. In *Proceedings of the human factors and ergonomics society annual meeting*, pages 904–908, 2006.

[37] Mohammadhosein Hasanbeig, Alessandro Abate, and Daniel Kroening. Logically-constrained reinforcement learning. *arXiv preprint arXiv:1801.08099*, 2018.

[38] Dominik Henrich and Thomas Honiger. Parallel processing approaches in robotics. In *ISIE'97 Proceeding of the IEEE International Symposium on Industrial Electronics*, volume 2, pages 702–707. IEEE, 1997.

[39] Kevin Anthony Hoff and Masooda Bashir. Trust in automation: Integrating empirical evidence on factors that influence trust. *Human Factors*, 57(3):407–434, 2015.

[40] Jiaming Hu, Yuhui Hu, Chao Lu, Jianwei Gong, and Huiyan Chen. Integrated path planning for unmanned differential steering vehicles in off-road environment with 3d terrains and obstacles. *IEEE Transactions on Intelligent Transportation Systems*, 2021.

[41] Ahmed Hussein, Mohamed Medhat Gaber, Eyad Elyan, and Chrisina Jayne. Imitation learning: A survey of learning methods. *ACM Computing Surveys (CSUR)*, 50(2):1–35, 2017.

[42] Yiannis Kantaros and Michael M Zavlanos. Sampling-based control synthesis for multi-robot systems under global temporal specifications. In *2017 ACM/IEEE 8th International Conference on Cyber-Physical Systems (ICCPS)*, pages 3–14. IEEE, 2017.

[43] Yiannis Kantaros and Michael M Zavlanos. Stylus*: A temporal logic optimal control synthesis algorithm for large-scale multi-robot systems. *The International Journal of Robotics Research*, 39(7):812–836, 2020.

[44] Sertac Karaman and Emilio Frazzoli. Sampling-based algorithms for optimal motion planning with deterministic μ-calculus specifications. In *2012 American Control Conference (ACC)*, pages 735–742. IEEE, 2012.

[45] Mohammad Karimadini and Hai Lin. Guaranteed global performance through local coordinations. *Automatica*, 47(5):890–898, 2011.

[46] W Bradley Knox and Peter Stone. Interactively shaping agents via human reinforcement: The tamer framework. In *Proceedings of the fifth international conference on Knowledge capture*, pages 9–16, 2009.

[47] G Ayorkor Korsah, Anthony Stentz, and M Bernardine Dias. A comprehensive taxonomy for multi-robot task allocation. *The International Journal of Robotics Research*, 32(12):1495–1512, 2013.

[48] David Krueger, Jan Leike, Owain Evans, and John Salvatier. Active reinforcement learning: Observing rewards at a cost. *arXiv preprint arXiv:2011.06709*, 2020.

[49] Geert-Jan M Kruijff, Ivana Kruijff-Korbayová, Shanker Keshavdas, Benoit Larochelle, Miroslav Janíček, Francis Colas, Ming Liu, François Pomerleau, Roland Siegwart, Mark A Neerincx, et al. Designing, developing, and deploying systems to support human–robot teams in disaster response. *Advanced Robotics*, 28(23):1547–1570, 2014.

[50] James R Lewis. Ibm computer usability satisfaction questionnaires: psychometric evaluation and instructions for use. *International Journal of Human-Computer Interaction*, 7(1):57–78, 1995.

[51] Bohao Li and Yunjie Wu. Path planning for uav ground target tracking via deep reinforcement learning. *IEEE Access*, 8:29064–29074, 2020.

[52] Kai Li, Quan Liu, Wenjun Xu, Jiayi Liu, Zude Zhou, and Hao Feng. Sequence planning considering human fatigue for human-robot collaboration in disassembly. *Procedia CIRP*, 83:95–104, 2019.

[53] Xiao Li, Cristian-Ioan Vasile, and Calin Belta. Reinforcement learning with temporal logic rewards. In *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 3834–3839. IEEE, 2017.

[54] James MacGlashan, Mark K Ho, Robert Loftin, Bei Peng, Guan Wang, David L Roberts, Matthew E Taylor, and Michael L Littman. Interactive learning from policy-dependent human feedback. In *International Conference on Machine Learning*, pages 2285–2294. PMLR, 2017.

[55] Maziar Fooladi Mahani, Longsheng Jiang, and Yue Wang. A bayesian trust inference model for human-multi-robot teams. *International Journal of Social Robotics*, pages 1–15, 2020.

[56] Maziar Fooladi Mahani and Yue Wang. Trust-based runtime verification for multi-quad-rotor motion planning with a human-in-the-loop. In *ASME 2018 Dynamic Systems and Control Conference*. American Society of Mechanical Engineers Digital Collection, 2018.

[57] John H McDonald. *Handbook of biological statistics*. Sparky House Publishing, Baltimore, MD, 2014.

[58] Mehdi Mekni. Using gis data to build informed virtual geographic environments (ivge). *Journal of Geographic Information System*, 2013, 2013.

[59] T. K. Moon. The expectation-maximization algorithm. *Signal processing magazine, IEEE*, 13(6):47–60, 1996.

[60] ROGER MUNDRY and JULIA FISCHER. Use of statistical programs for nonparametric tests of small samples often leads to incorrect pvalues: examples from animal behaviour. *Animal behaviour*, 56(1):256–259, 1998.

[61] Anis Najar and Mohamed Chetouani. Reinforcement learning with human advice: a survey. *Frontiers in Robotics and AI*, 8, 2021.

[62] Changjoo Nam, Phillip Walker, Michael Lewis, and Katia Sycara. Predicting trust in human control of swarms via inverse reinforcement learning. In *2017 26th IEEE International Symposium on Robot and Human Interactive Communication (RO-MAN)*, pages 528–533, 2017.

[63] Changjoo Nam, Phillip Walker, Huao Li, Michael Lewis, and Katia Sycara. Models of trust in human control of swarms with varied levels of autonomy. *IEEE Transactions on Human-Machine Systems*, 50(3):194–204, 2019.

[64] Stefanos Nikolaidis and Julie Shah. Human-robot teaming using shared mental models. *ACM/IEEE HRI*, 2012.

[65] Athanasios S Polydoros and Lazaros Nalpantidis. Survey of model-based reinforcement learning: Applications on robotics. *Journal of Intelligent & Robotic Systems*, 86(2):153–173, 2017.

[66] Mattia Racca, Antti Oulasvirta, and Ville Kyrki. Teacher-aware active robot learning. In *2019 14th ACM/IEEE International Conference on Human-Robot Interaction (HRI)*, pages 335–343. IEEE, 2019.

[67] Bhaskar Ramasubramanian, Luyao Niu, Andrew Clark, Linda Bushnell, and Radha Poovendran. Secure control in partially observable environments to satisfy ltl specifications. *IEEE Transactions on Automatic Control*, 2020.

[68] Hossein Rastgoftar, Bingxin Zhang, and Ella M Atkins. A data-driven approach for autonomous motion planning and control in off-road driving scenarios. In *2018 Annual American Control Conference (ACC)*, pages 5876–5883, 2018.

[69] CA: Environmental Systems Research Institute Redlands. Arcgis pro: Version 2.8, 2021.

[70] Pengzhen Ren, Yun Xiao, Xiaojun Chang, Po-Yao Huang, Zhihui Li, Brij B Gupta, Xiaojiang Chen, and Xin Wang. A survey of deep active learning. *ACM computing surveys (CSUR)*, 54(9):1–40, 2021.

[71] Robert M Roe, Jermone R Busemeyer, and James T Townsend. Multialternative decision field theory: A dynamic connectionst model of decision making. *Psychological review*, 108(2):370, 2001.

[72] Dorsa Sadigh, Eric S Kim, Samuel Coogan, S Shankar Sastry, and Sanjit A Seshia. A learning based approach to control synthesis of markov decision processes for linear temporal logic specifications. In *53rd IEEE Conference on Decision and Control*, pages 1091–1096. IEEE, 2014.

[73] B. Sadrfaridpour, H. Saeidi, J. Burke, K. Madathil, and Y. Wang. Modeling and control of trust in human-robot collaborative manufacturing. In *The Intersection of Robust Intelligence (RI) and Trust in Autonomous Systems*. Springer, 2015.

[74] Behzad Sadrfaridpour and Yue Wang. Collaborative assembly in hybrid manufacturing cells: an integrated framework for human–robot interaction. *IEEE Transactions on Automation Science and Engineering*, 15(3):1178–1192, 2017.

[75] Hamed Saeidi and Yue Wang. Incorporating trust and self-confidence analysis in the guidance and control of (semi) autonomous mobile robotic systems. *IEEE Robotics and Automation Letters*, 4(2):239–246, 2018.

[76] Indranil Saha, Rattanachai Ramaithitima, Vijay Kumar, George J Pappas, and Sanjit A Seshia. Automated composition of motion primitives for multi-robot systems from safe LTL specifications. In *Intelligent Robots and Systems (IROS 2014), 2014 IEEE/RSJ International Conference on*, pages 1525–1532. IEEE, 2014.

[77] Philipp Schillinger, Mathias Bürger, and Dimos V Dimarogonas. Decomposition of finite LTL specifications for efficient multi-agent planning. In *Distributed Autonomous Robotic Systems*, pages 253–267. Springer, 2018.

[78] Philipp Schillinger, Mathias Bürger, and Dimos V Dimarogonas. Simultaneous task allocation and planning for temporal logic goals in heterogeneous multi-robot systems. *The international journal of robotics research*, 37(7):818–838, 2018.

[79] Julie Shah and Cynthia Breazeal. An empirical analysis of team coordination behaviors and action planning with application to human–robot teaming. *Human factors*, 52(2):234–245, 2010.

[80] Bobak Shahriari, Kevin Swersky, Ziyu Wang, Ryan P Adams, and Nando De Freitas. Taking the human out of the loop: A review of bayesian optimization. *Proceedings of the IEEE*, 104(1):148–175, 2015.

[81] Laya Shamgah, Ali Karimoddini, and Abdollah Homaifar. A symbolic motion planning approach for the reach-avoid problem. In *2016 IEEE International Conference on Systems, Man, and Cybernetics (SMC)*, pages 003955–003960. IEEE, 2016.

[82] Shuyu Shao, Qianxiang Zhou, and Zhongqi Liu. Mental workload characteristics of manipulator teleoperators with different spatial cognitive abilities. *International Journal of Advanced Robotic Systems*, 16(6):1729881419888042, 2019.

[83] Pedro M Shiroma and Mario FM Campos. Comutar: A framework for multi-robot coordination and task allocation. In *2009 IEEE/RSJ international conference on intelligent robots and systems*, pages 4817–4824. IEEE, 2009.

[84] Robert H Shumway, David S Stoffer, and David S Stoffer. *Time series analysis and its applications*, volume 3. Springer, New York, 2000.

[85] David A Spencer and Yue Wang. Slqr suboptimal human-robot collaborative guidance and navigation for autonomous underwater vehicles. In *American Control Conference (ACC), 2015*, pages 2131–2136. IEEE, 2015.

[86] Jana Tumova and Dimos V Dimarogonas. Decomposition of multi-agent planning under distributed motion and task LTL specifications. In *Decision and Control (CDC), 2015 IEEE 54th Annual Conference on*, pages 7448–7453. IEEE, 2015.

[87] Jana Tumova and Dimos V Dimarogonas. Multi-agent planning under local LTL specifications and event-based synchronization. *Automatica*, 70:239–248, 2016.

[88] Alphan Ulusoy, Stephen L Smith, Xu Chu Ding, Calin Belta, and Daniela Rus. Optimality and robustness in multi-robot path planning with temporal logic constraints. *The International Journal of Robotics Research*, 32(8):889–911, 2013.

[89] Cristian Ioan Vasile and Calin Belta. Sampling-based temporal logic path planning. In *2013 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 4817–4822. IEEE, 2013.

[90] Abraham P Vinod, Yuqing Tang, Meeko MK Oishi, Katia Sycara, Christian Lebiere, and Michael Lewis. Validation of cognitive models for collaborative hybrid systems with discrete human input. In *Intelligent Robots and Systems (IROS), 2016 IEEE/RSJ International Conference on*, pages 3339–3346. IEEE, 2016.

[91] Yue Wang, Laura R Humphrey, Zhanrui Liao, and Huanfei Zheng. Trust-based multi-robot symbolic motion planning with a human-in-the-loop. *ACM Transactions on Interactive Intelligent Systems (TiiS)*, 8(4):1–33, 2018.

[92] Min Wen and Ufuk Topcu. Probably approximately correct learning in adversarial environments with temporal logic specifications. *IEEE Transactions on Automatic Control*, 2021.

[93] Nils Wilde, Erdem Bıyık, Dorsa Sadigh, and Stephen L Smith. Learning reward functions from scale feedback. *arXiv preprint arXiv:2110.00284*, 2021.

[94] Christian Wirth, Riad Akrour, Gerhard Neumann, Johannes Fürnkranz, et al. A survey of preference-based reinforcement learning methods. *Journal of Machine Learning Research*, 18(136):1–46, 2017.

[95] Christian Wirth, Johannes Fürnkranz, and Gerhard Neumann. Model-free preference-based reinforcement learning. In *Thirtieth AAAI Conference on Artificial Intelligence*, 2016.

[96] Eric M Wolff, Ufuk Topcu, and Richard M Murray. Robust control of uncertain markov decision processes with temporal logic specifications. In *2012 IEEE 51st IEEE Conference on Decision and Control (CDC)*, pages 3372–3379. IEEE, 2012.

[97] Keyu Wu, Han Wang, Mahdi Abolfazli Esfahani, and Shenghai Yuan. Achieving real-time path planning in unknown environments through deep neural networks. *IEEE Transactions on Intelligent Transportation Systems*, 2020.

[98] Anqi Xu and Gregory Dudek. Optimo: Online probabilistic trust inference model for asymmetric human-robot collaborations. In *2015 10th ACM/IEEE International Conference on Human-Robot Interaction (HRI)*, pages 221–228. IEEE, 2015.

[99] Osama Zaki and Matthew Dunnigan. A navigation strategy for an autonomous patrol vehicle based on multi-fusion planning algorithms and multi-paradigm representation schemes. *Robotics and Autonomous Systems*, 96:133–142, 2017.

[100] Ashkan Zehfroosh and Herbert Glenn Tanner. A hybrid pac reinforcement learning algorithm for human-robot interaction. *Frontiers in Robotics and AI*, page 22, 2022.

[101] Huanfei Zheng, Zhanrui Liao, and Yue Wang. Human-robot trust integrated task allocation and symbolic motion planning for heterogeneous multi-robot systems. In *Dynamic Systems and Control Conference*, volume 51913, page V003T30A010, 2018.

[102] Huanfei Zheng, Jonathon M Smereka, Dariusz Mikluski, and Yue Wang. Bayesian optimization based trustworthiness model for multi-robot bounding overwatch. *arXiv preprint arXiv:2201.01878*, 2022.

[103] Huanfei Zheng and Yue Wang. A distributed framework for dynamic task allocation of multi-robot symbolic motion planning. In *2019 American Control Conference (ACC)*, pages 3291–3296. IEEE, 2019.

[104] Robert Michael Zlot and Anthony Stentz. *An auction-based approach to complex task allocation for multirobot teams*. PhD thesis, Carnegie Mellon University, The Robotics Institute, 2006.