Doctor's Thesis

Learning Approaches for Flexible and Resilient Multi-robot Cooperative Transport

Kazuki Shibata

March 17, 2023

Program of Information Science and Engineering
Graduate School of Science and Technology
Nara Institute of Science and Technology

A Doctor's Thesis
submitted to Graduate School of Science and Technology,
Nara Institute of Science and Technology
in partial fulfillment of the requirements for the degree of
Doctor of ENGINEERING

Kazuki Shibata

Learning Approaches for Flexible and Resilient Multi-robot Cooperative Transport[*]

Kazuki Shibata

Abstract

Multi-robot transportation has attracted attention in robotics and can be applied in fields such as delivery, logistics, and search and rescue. While previous studies have successfully derived control strategies for various tasks, they still have strong assumptions. Consequently, the robots cannot transport unknown objects or manage unexpected scenarios, where the number of robots differs from the training environment owing to batteries' discharges or actuators' failure. Therefore, this thesis proposes learning approaches for achieving flexible and resilient multi-robot transportation. Firstly, a learning framework is proposed to design communication and control strategies for various numbers of robots. The proposed method exploits a distributed policy to reconstruct global information using local information while determining the timing for communication. Therefore, it can balance communication savings and control performance for various numbers of robots. Secondly, we propose a learning framework of multi-robot task allocation for various numbers of robots and objects with unknown weights. The proposed method exploits a distributed policy that determines the timing for cooperative and independent actions. Therefore, it can reduce the transport time while transporting all the objects for various numbers of robots and objects with unknown weights. Lastly, an approach for object shape estimation is proposed to transport unknown-shaped objects. The proposed method adopts touch-based contact detection using accelerometers to supplement vision sensors. However, this method involves false-positive contact data because it reacts not only to actual contacts but also to the unstable behavior of the robot. Therefore, a robust

[*]Doctor's Thesis, Department of Information Science, Graduate School of Information Science and Technology, Nara Institute of Science and Technology, March 17, 2023.

shape estimation method is proposed to handle such false-positive contact data. I hope this thesis plays an essential role in achieving a flexible and resilient system that can deal with various transportation scenarios in real applications.

Keywords:

Cooperative transport, Task allocation, Shape estimation, Multi-agent reinforcement learning, Consensus algorithm, Gaussian process

柔軟かつレジリエントな協調輸送のための学習法*

柴田一騎

内容梗概

マルチロボットによる協調輸送は配送、物流、捜索救助などの応用が期待されており、ロボティクス分野で注目を集めている。協調輸送の従来研究は、さまざまなタスクで制御戦略を導出できているが、依然としていくつかの強い仮定がある。その結果、ロボットは未知の荷物を輸送できないことや、電池切れやアクチュエータの故障などの予期せぬ状況に対応できないことがある。本論文は、様々な輸送シナリオに対応できる柔軟かつレジリエントな協調輸送のための学習法を提案する。初めに、様々な台数のロボットに対する通信と制御の設計のための学習法を提案する。提案法は、通信のタイミングを決定しつつ、局所情報から大域情報を復元する分散方策モデルを用いる。これによって、提案法は、様々なロボットの台数に対して、制御性能と通信節約を両立できる。次に、荷物の輸送に必要なロボットの台数が未知、かつ様々な数の荷物とロボットに対するマルチロボットの役割分担に取り組む。提案法は、協調と分業のタイミングを決定する分散方策モデルを採用する。これによって、提案法は、荷物の輸送に必要なロボットの台数が未知でも、様々な数の荷物とロボットに対して、与えられた全ての荷物を目的地へ移動しつつ、輸送時間を短縮できる。最後に、未知形状の荷物を輸送するための形状推定に取り組む。提案法は、カメラを補完するため、加速度センサを用いた接触検出法を用いる。しかしながら、この検出法は、実際の衝突だけでなくロボットの不安定な挙動にも反応するため、接触の誤検出データを含む。そのため、誤検出データにも対応可能な形状推定方法を提案する。本論文は、実応用での様々な輸送シナリオに対応できる柔軟かつレジリエントなシステムの実現に重要な役割を果たすことが期待できる。

キーワード

協調輸送、役割分担、形状推定、マルチエージェント強化学習、合意制御、ガウス過程回帰

*奈良先端科学技術大学院大学 先端科学技術研究科 情報科学領域 博士論文, 2023年 3月 17日.

Contents

List of Figures

1. Introduction

1.1. Background

In recent years, multi-robot transportation has attracted attention in robotics and can be applied in fields such as delivery services, warehouse logistics [1, 2], and search and rescue missions [3]. Multi-robot systems have several advantages over single-robot systems regarding load capacity, time efficiency, and robustness to individual robot failures. These systems can handle heavy objects that cannot be transport by a single robot. Moreover, to transport multiple objects over large areas, multi-robot task allocation improves the overall time efficiency.

A key property required for the multi-robot transportation is *resilience*, that is, flexibility for various transportation scenarios. Firstly, the robots should manage unexpected scenarios, where the robots may suffer from the discharging of the batteries, the failure of the robots' actuators, or the disconnection of communication among robots. Secondly, the robots should transport the object whose parameters, including mass, moment of inertia, or center of gravity is unknown because it is difficult to know these parameters by using cameras. Lastly, the robots should transport unknown-shaped objects considering that it is often difficult for the robots to know the accurate shapes by cameras because they may suffer from noises and occlusions.

The aim of this thesis is to construct a resilient multi-robot transportation system that can deal with various unexpected transportation scenarios, as shown in Fig. 1.1.

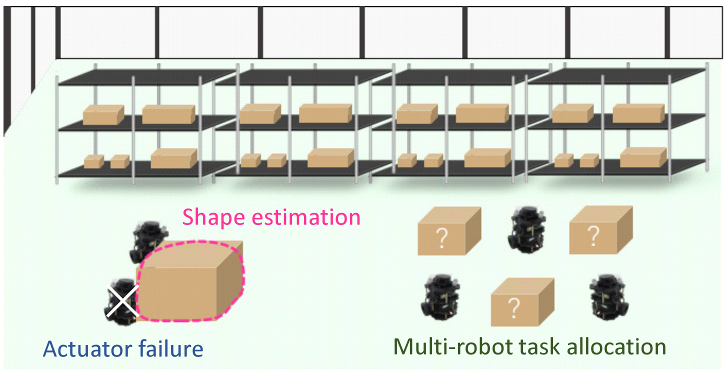


Figure 1.1. Resilient system for multi-robot transportation

1.2. Motivation

Previous studies on multi-robot transportation have successfully derived control strategies for various tasks, including cooperative pushing [4, 5], cooperative manipulation using multiple arm robots [6–8], human and robots [9, 10], and aerial manipulation by multiple quadcopters [11–15]. While these approaches have succeeded in real robot experiments, they still have strong assumptions as follows:

- the number of the robots is the same as that in the training environment to design communication and control strategies.
- the number of robots to execute each task is available.
- the accurate shape of the object is available.

Consequently, the robots cannot transport unknown objects or manage unexpected scenarios, where the number of robots differ from the training environment owing to batteries' discharges or actuators' failure. Therefore, this thesis proposes learning approaches for achieving resilient multi-robot systems. Figure 1.2 shows the basic concept of the proposed approach. The first idea is to adopt a neural network-based policy model that can handle objects with unknown weights. The second idea is to adopt a distributed policy model that can be applied to various numbers of objects and robots. The third idea is to adopt a consensus to reconstruct global information from local information to improve the learning performance. These ideas allows robots to flexibly transport objects even if the

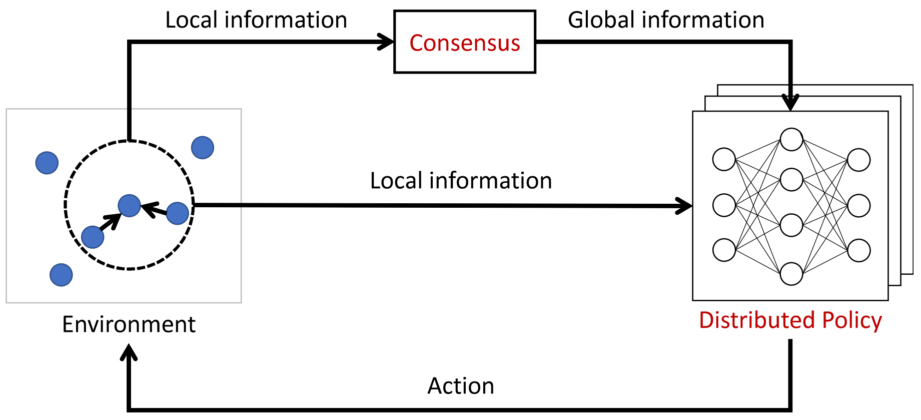


Figure 1.2. Basic concept of the approach

weights of the objects are unknown and the number of objects and robots varies. We confirm the resilience of our approach through through three problems. We first confirm the scalability of the number of robots through multi-robot cooperative transport with various number of robots. Secondly, we confirm the scalability of the number of objects and robots through multi-robot task allocation for multiple objects with unknown and different weights. Finally, we confirm the flexibility for unknown-shaped objects through shape estimation.

We first address a design problem of communication and control strategies for multi-robot cooperative transport with various number of robots. In this problem, we focused on verifying the scalability of the robot with the assumption that the shape of the object is known. Most multi-robot transportation studies have relied on wireless communication to share observations among robots. However, the communication bandwidth can be compressed if multiple robots transmit information at high fixed rates in the same network system. This will increase the probability of message loss and cause long transmission delays [16]. Therefore, it is crucial to minimize communication. Previous studies [17–20] have employed distributed adaptive/robust control to deal with the unknown object dynamics in the multi-robot cooperative transport. However, these studies have strong assumptions and can only work with simple tasks where robots are rigidly attached to the payload. Therefore, it is crucial to consider a data-driven framework without these assumptions. In this study, we explored a multi-agent reinforcement learning (MARL) approach [21, 22] to simultaneously solve the design problems of communication and control strategies for multi-robot cooperative transport. Several studies have proposed communication strategies to reduce the communication frequency [23, 24] and the number of communicating robots and data [25]. These studies assume that the number of robots is equivalent to that in the training environment. However, the number of robots can differ owing to the discharging of the robots' batteries, or additional robots may be introduced to complete tasks quickly. Therefore, the learned strategy must also be applicable to scenarios wherein the number of robots differs from that in the training environment.

Next, we address a task allocation problem for multi-object transport using a multi-robot system. In this thesis, a task corresponds to an object. The existing studies on multi-robot task allocation have adopted deterministic optimization methods [26–29] or auction methods [30–32] under the assumption that the number of robots to execute each task is available. However, these assumptions are sometimes unrealistic. For instance, by using a camera, it may be possible to obtain information on the shape of an object; however, it is challenging to obtain the number of robots required to transport it. In this case, the assumption does not hold. We explore a MARL framework for multi-object transport using a multi-robot system. Each robot selects one object among multiple objects with different and unknown weights. The objective is to transport all the objects to the desired positions as quickly as possible. The existing centralized methods [33, 34] assume the number of robots and tasks to be fixed, which is inapplicable to the scenarios in which the number of robots and tasks differs from the learning environment. Meanwhile, the existing distributed methods limit the minimum number of robots and tasks to a constant value, making them applicable to various numbers of robots and tasks [35]. However, they cannot transport an object whose weight exceeds the load capacity of robots observing the object. Therefore, it is crucial to consider a learning framework that can handle scenarios for various numbers of robots and objects with different and unknown weights.

Finally, we addressed shape estimation to extend the control framework of cooperative transport to an unknown-shaped object. While most previous studies on shape estimation have relied on vision sensors, they are noisy and suffer from occlusion. To supplement the vision-based sensing, tactile sensing was used to improve shape estimation [36–38]. Most touch-to-sense approaches, including grasping [39–41] and manipulation [42–44], have explicitly assumed that a tactile sensor correctly detects the contact occurrence. However, such an assumption may not always be realistic. Given that it is difficult to determine, in advance, which parts of the robot will touch the construction, a number of sensors should be mounted on the robot in an omnidirectional configuration. However, this may be unfeasible regarding the robot's payload, cost, or power consumption. Therefore, We explored an alternative approach to the estimation of the shape of an object by touch *without* the need for tactile sensors. That is, we considered omnidirectional contact detection using an accelerometer incorporated into a robot. This approach would avoid the issues associated with payload, cost, and power consumption. However, the accelerometer may react not only to actual contact with the object but also to unstable behavior induced by gusts of wind, collisions with other robots, etc. As a result, false-positive contact data could be combined with normal contact data. Therefore, when using this approach, it is crucial to incorporate a robust shape estimation method capable of handling such false-positive contact data.

1.3. Contribution

We first propose a novel MARL framework of event-triggered communication and consensus-based control for distributed cooperative transport. The objective of this study was to control the payload to the desired state for scenarios wherein the number of robots differs from that in the training environment, as shown in Fig. 1.3 while minimizing communication among robots. For the settings in this research, the numbers of communication and environmental robots were maintained constant during training but variable during execution. The proposed policy model exploits consensus with local communication robots to reconstruct global information, such as resultant force and torque. Limiting the minimum number of environmental robots to a fixed number of communication robots enables the proposed method to be applicable to a varying number of robots. In particular, our policy model estimates the resultant force and torque applied to an object in a consensus manner [45,46] using the estimates of the resultant force

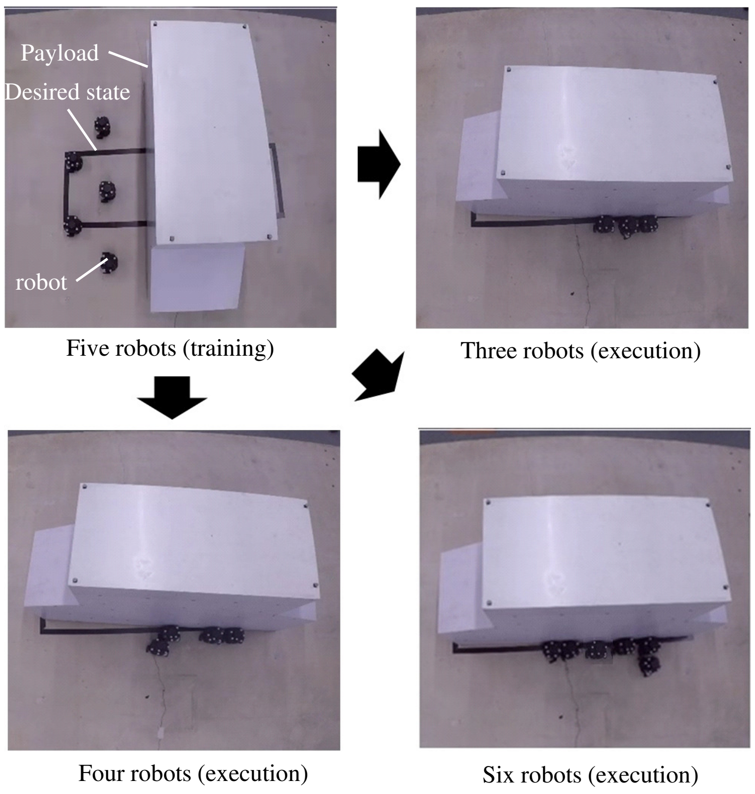
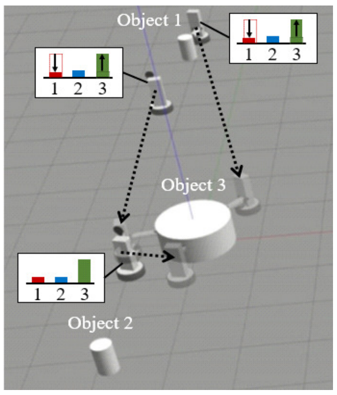


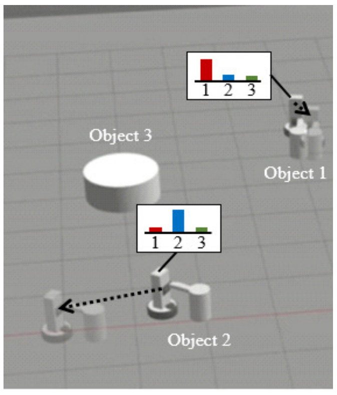
Figure 1.3. Cooperative rotation task using multiple robots. The proposed framework can control the payload to the desired state for scenarios wherein the number of robots differs from that in the training environment.

and torque of the communication robots. Moreover, under local observations and estimates of the resultant force and torque, the proposed framework can compute the control and communication inputs to determine when to communicate with the neighboring robots. Therefore, the control performance and communication savings can be balanced, despite the number of robots being different from that in the training environment. To the authors' knowledge, no similar studies on learning event-triggered control of multi-robot systems have been reported so far.

Next, we propose a learning framework that can handle scenarios for various numbers of robots and objects with different and unknown weights. The proposed framework exploits a structured policy model consisting of 1) the consensus of task priorities with global communication and 2) a neural network-based distributed policy model that determines the timing for consensus. If robots cannot transport the object, they select cooperative actions by reaching a consensus regarding high-



(a) Cooperative action



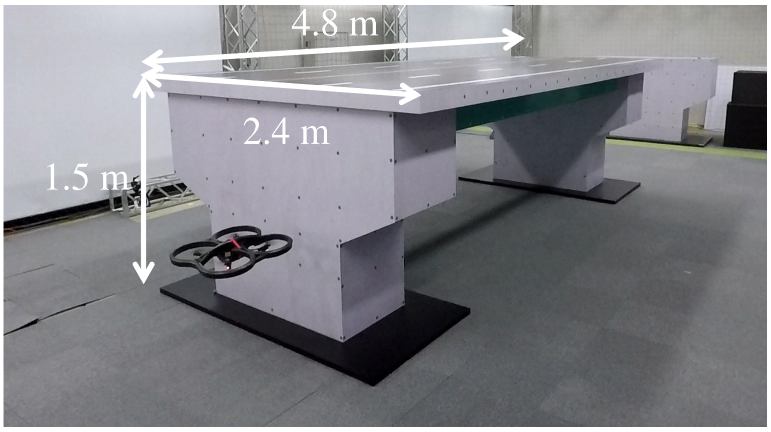
(b) Independent action

Figure 1.4. Multi-object transportation using a multi-robot system. (a) Robots perform cooperative actions by building a consensus on the high-priority object when they cannot move the object. (b) Robots perform independent actions when they can move the selected objects.

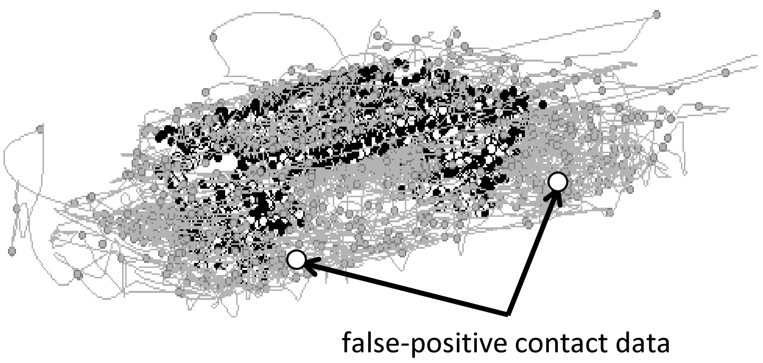
priority object. The distributed policy model determines the timing for consensus, thus balancing the cooperative and independent actions, as illustrated in Fig. 1.4. The policy is optimized by the MARL through trial and error. This structured policy of local learning and global communication makes our framework suitable for scenarios where the numbers of robots and objects vary, and the number of robots required to transport an object is unknown.

Finally, we propose a robust shape estimation method that can handle false-positive contact data, as shown in Fig. 1.5. Our approach involves a robust shape estimation algorithm based on Gaussian process implicit surfaces (GPIS) [47–51], that is, robust GPIS. Because the GPIS assumes that the observations of contact data follow a normal distribution with a constant variance, it estimates incorrect surfaces around the false-positive contact data. In contrast, our GPIS introduces heteroscedasticity into each item of contact data, thus preventing incorrect shape estimates produced by false-positive contact data.

Although we have not achieved cooperative transport of unknown-shaped object, we discuss the future direction to integrate the shape estimation into the learning framework of cooperative transport in Chapter 7.



(a) 3D construction



(b) Observed data and flight trajectory

Figure 1.5. (a) 3D construction. (b) Observed contact data and flight trajectory of quadcopter. The circles, black dots, gray dots, and gray lines represent internal points, contact points, external points, and the flight trajectory of the quadcopter, respectively.

1.4. Structure of thesis

The remainder of this thesis is organized as follows. Chapter 2 introduces related work. Chapter 3 describes the preliminaries. Chapters 4, 5, and 6 propose the learning framework of distributed cooperative transport, multi-robot task allocation, and shape estimation for unknown objects, respectively. Chapter 7 describes the open issues and scope of future work. Finally, Chapter 8 presents the conclusions of this thesis.

2. Related work

2.1. Distributed cooperative transport

In this section, we introduce model-based approaches that derive the control policy using distributed control, based on the dynamics model of cooperative transport. Furthermore, model-free approaches are introduced, which derive the control policy using data-driven approaches without the dynamics model.

2.1.1. Model-based approaches

Previous studies on cooperative transport have derived control strategies based on the dynamics model. A key issue of cooperative transport lies in controlling the payload without prior knowledge of the payload and robots. Franchi et al. [17, 18] proposed a decentralized parameter estimation of an unknown load using observations of the neighboring robots. Based on this algorithm, Petitti et al. [19] proposed a robust control strategy to stabilize the payload in the presence of estimation uncertainties. Marino et al. [52] proposed a distributed control strategy with the estimation of an unknown object without explicit communication and prior knowledge of the number of robots. Culbertson et al. [20] proposed a distributed adaptive control strategy for cooperative transport of an unknown object without parameter estimation. Their control strategy required no communication between robots and made the payload state asymptotically converge to the desired state with a theoretical proof using the Lyapunov function.

Other studies employed variable-rate communication to reduce the communication frequency. Dimarogonas et al. [53] introduced event-triggered control [54, 55] into multi-agent communication to determine the timing of the communication with neighboring agents. Trimpe et al. [56] proposed a distributed control strategy with event-triggered communication to determine both the timing and transmitted data based on the error between the actual measurements and the estimates. Furthermore, they demonstrated the effectiveness by conducting balancing cube experiments using six modules, each having sensors, actuation, and computational units. Dohmann et al. [57] were the first to propose a distributed control strategy with event-triggered communication for cooperative manipulation. Their methods minimized the frequency of receiving positions and velocities of end effectors from neighboring agents while accomplishing several manipulation tasks.

However, these approaches require a dynamics model and cannot be applied to tasks wherein dynamics models are difficult to formulate. In contrast, the proposed method is model-free and can be applied to more cooperative transport tasks compared to these approaches.

2.1.2. Model-free approaches

Several recent studies [34,58,59] have adopted MARL approaches for multi-robot cooperative transport without requiring a dynamics model. However, one of the main problems in MARL is that the variance of the estimated policy becomes large owing to the changing policies of other agents [60]. To address this issue, several authors [21, 22] proposed a learning framework of centralized training and decentralized execution, which learns critics for multi-agents and derives a decentralized policy using observations from each agent; however, these methods cannot determine the timing of communication and can only function with fixed-rate communication.

To learn control strategies while saving communication costs, several authors [23,24] proposed a policy model using event-triggered control to minimize the control signals from a single learning agent to its actuator while achieving the control objective. Demirel et al. [61] proposed DEEPCAS, a reinforcement learning-based control-aware scheduling algorithm in multi-agent setups. In this method, a centralized scheduler called DEEPCAS allocates M communication channels to N agents ($M \ll N$) while designing the agents' controllers beforehand. Our preliminary study [25] extended the policy model [23, 24] in multi-agent setups to reduce the frequency, number of communicating agents, and transmitted data. Although these methods could save communication costs, they adopted a policy model with inputs dependent on the number of agents and were inapplicable to problems wherein the number of agents was different from that in the learning environment.

The proposed framework combines the estimation of the resultant force and torque in a consensus manner and an event-triggered communication to determine the timing of communication into a policy model. Considering that our policy model computes the control and communication inputs under local observations and the estimates of the resultant force and torque of the neighborhood agents, it can be applied to scenarios wherein the number of agents differs from that in the training environment.

While the present study adopts a policy model using event-triggered control, it differs from the methodology proposed in [23–25] as it involves an estimation mechanism and a distributed policy model under local observation. Moreover, the proposed framework can transport the payload to the desired state for varying number of agents.

2.2. Multi-robot task allocation

In this section, we introduce deterministic optimization, metaheuristic and auction methods for multi-robot task allocation. Furthermore, we introduce MARL methods that do not require the number of robots to execute each task.

2.2.1. Deterministic optimization methods

Deterministic optimization formulates the task allocation problem as an optimization problem aimed at minimizing the total travel distance under constraints for the number of robots required for each task. These approaches have adopted various optimization techniques, such as the Hungarian algorithm [26, 27], integer linear programming [28], and mixed integer linear programming [29]. Although these studies can guarantee optimality in terms of the total travel distance, most methods require prior information regarding the number of robots required for each task.

2.2.2. Metaheuristic methods

Metaheuristic methods are inspired by the division of labor exhibited by social insects. A common approach has adopted threshold models [62,63], in which each robot selects a task under local observations using an activation threshold and a stimulus associated with each task. Although these methods can handle varying numbers of robots and tasks, they may allocate unnecessary tasks to robots, thus reducing the time efficiency.

2.2.3. Auction methods

Auction algorithms [64, 65] are common methods for multi-robot task allocation and have been studied via centralized and decentralized approaches. The centralized method [66] adopts the auctioneer, which collects the bids from the bidders, and allocates the highest bidder to the task. In contrast, Choi et al. [67] propose a decentralized auction-based algorithm without the auctioneer. This method adopts a consensus algorithm to estimate the bids of other robots. Then, the robots allocate the task to the highest bidder using the estimated bids. Therefore, each robot can assign a task even if it can locally communicate with other robots. However, their method focuses on the problem where a single robot can execute each task.

Braquet and Bakolas [32] addressed the closest problem to our study, where each task requires multiple robots. Their method adopts the consensus algorithm similar to [67], which estimates the list of selected tasks, the list of winning bids, and the list of completed allocations. Robots assign a task to the robot with the highest bid among the unassigned robots based on the list of completed allocations. Therefore, their method can be applicable to the problem where each task requires multiple robots. However, their methods require a probability of completing each task, which is difficult to compute for objects with unknown weights.

2.2.4. MARL methods

Recent studies [33,34] have addressed task allocation problems using the MARL. These approaches formulate a task allocation problem using the Markov decision process and learn the optimal policies using a multi-agent deep deterministic policy gradient (MADDPG) [21]. However, these methods adopt centralized training assuming that the number of robots and tasks is constant, failing in scenarios with different numbers of robots and tasks. To address this problem, Hsu et al. [35] proposed a distributed policy model, which limits the minimum number of robots and tasks to be constant. The trained policies are applicable to up to 1000 robots and 1000 tasks through multi-target tracking simulations. Although their methods can be applied to various numbers of robots and tasks, they cannot handle a situation where the number of robots required to execute a task exceeds the number of robots observing it.

Although the proposed framework uses distributed policies under local observations, it differs from the method [35] in that our method employs a structured policy model consisting of predesigned dynamic task priorities with global communication and a neural network-based distributed policy model. Therefore, robots can perform all the tasks efficiently even when the number of robots required to complete each task is different and unknown.

2.3. Shape estimation

In this section, we introduce shape estimation approaches using deep learning and GP methods.

2.3.1. Deep learning methods

Several recent studies have addressed shape estimation using deep learning. Wu et al. [68] and Dai et al. [69] proposed the use of 3D-CNN for shape estimation, and Varley et al. [70] proposed a novel grasping framework based on the architecture in [69]. Watkins-Valls et al. [71] proposed an architecture that incorporates depth images and tactile information to improve shape estimation in occluded regions. Through experiments using different types of 3D object shapes, they confirmed improvements in shape estimation errors and in the success rate of grasping compared with the GPIS. While these studies obtained impressive performance in shape estimation, all except for Lundell et al. [72] are based on the deterministic model. Therefore, it is not straightforward to apply this architecture to a model where there is estimation uncertainty and different kinds of noise.

2.3.2. GP methods

Other studies have focused on the use of GPIS by representing object shapes in a GP model. Caccamo et al. [48] and Mahler et al. [49] combined visual data with haptic measurements using GPIS. In addition, using the basic framework of GPIS, Dragiev et al. [50] employed a combination of laser and tactile sensing while Gerardo-Castro [51] used a combination of lasers and radar.

Basically, most of the GPIS methods are based on a homoscedastic GP, in which the distribution of observations is assumed to follow a normal distribution with a constant variance. However, this assumption may be unsuitable for application to cases involving some outliers because both normal and outlier data are handled equally.

To overcome this issue, several authors proposed heteroscedastic GPs (HGPs), for which the variation in the input or output noise is considered to be non-uniform. Kuss et al. [73] and Jylanki et al. [74] proposed a robust GP, which is a HGP that assumes that the noise variance is data-dependent. McHutchon et al. [75] proposed a GP with input noise (NIGP). This is a HGP that assumes that the noise variance differs for each input dimension. Lazaro-Gredilla et al. [76] proposed a variational heteroscedastic GP (VHGP), which is a HGP that assumes that the noise variance is input-dependent. Among the available HGPs, we found that the robust GP is particularly suitable for our purpose given that it provides us with the uncertainty of data.

Among practical research using a robust GP as a mathematical model, Martinez-Cantin et al. [77] proposed robust Bayesian optimization for active exploration of optimal policy parameters in a humanoid robot walking under the condition that there exist outliers of rewards caused by perturbations. While our methodology used a common robust GP as a mathematical model, it differs from the methodology in [77] in terms of shape reconstruction and false-positive contact detection when considering the uncertainty for each item of contact data.

3. Preliminaries

3.1. Distributed cooperative transport

3.1.1. Notation

Let the position, yaw angle, velocity, angular velocity, desired position, and the desired yaw angle of the payload in world coordinates be denoted by $\boldsymbol{x} \in \mathbb{R}^2$, $\theta \in \mathbb{R}$, $\boldsymbol{v} \in \mathbb{R}^2$, $\omega \in \mathbb{R}$, $\boldsymbol{x}^* \in \mathbb{R}^2$, and $\theta^* \in \mathbb{R}$, respectively.

The position, yaw angle, and control input of the robot i $(i = 1, \cdots, N)$ are represented by $\boldsymbol{x}_i \in \mathbb{R}^2$, $\theta_i \in \mathbb{R}$, and $\boldsymbol{u}_i \in \mathbb{R}^2$, respectively. Robot i applies a force $\boldsymbol{f}_i \in \mathbb{R}^2$ and torque $\tau_i \in \mathbb{R}$ on the payload.

3.1.2. Problem formulation

We consider a team of N robots pushing a rigid payload with an unknown mass and moment of inertia. In our setting, N is constant during the training phase but variable during the execution phase. The objective of this problem is to control the payload to its desired state while reducing communication with other robots for varying numbers of robots during the execution phase.

According to [20], we assumed the following:

- all robots know $\boldsymbol{x}^*$ and θ^*;
- robot i can observe $\boldsymbol{x}$, θ, $\boldsymbol{v}$, ω, $\boldsymbol{x}_i$, θ_i, $\boldsymbol{f}_i$, and τ_i.

Moreover, we assumed the following:

- all robots know the object shape.
- all robots know N.

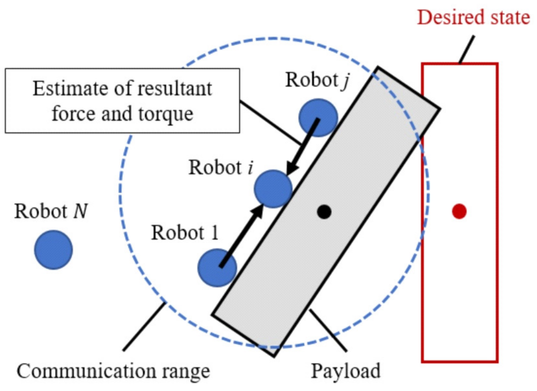


Figure 3.1. Distributed cooperative transport.

- the robot can communicate the estimates of the resultant force and torque with the K nearest robots, as shown in Fig. 3.1, where K is constant.

3.1.3. MARL settings

We introduce the MARL setting for distributed cooperative transport according to a Markov decision process.

Let us denote the state, observation, and action of robot i as $\boldsymbol{s}_i$, $\boldsymbol{o}_i$ and $\boldsymbol{a}_i$, respectively. Robot i selects action $\boldsymbol{a}_i$ under its local observation $\boldsymbol{o}_i$ depending on a policy π_i. After N robots select the current actions $[\boldsymbol{a}_1, \cdots, \boldsymbol{a}_N]$, the current states $[\boldsymbol{s}_1, \cdots, \boldsymbol{s}_N]$ transition to the next states $[\boldsymbol{s}'_1, \cdots, \boldsymbol{s}'_N]$. At current step t, robot i receives a reward $r_t \in \mathbb{R}$, which is defined by the error between the current and desired state of the payload and the communication costs. Robot i updates its policy by maximizing the expected reward $\mathbb{E}[R_t] = \mathbb{E}\left[\sum_{k=0}^{T-1} \gamma^k r_{t+k}\right]$, where $\gamma \in [0, 1]$ is a discount factor, and T is the total number of control steps per episode.

3.1.4. Communication topology

Let a binary variable be defined by γ_{ij}; using this, robot i receives data from robot j. Specifically, $\gamma_{ij} = 1$ if robot i receives data from robot j; otherwise, $\gamma_{ij} = 0$.

The entries of the adjacency matrix $\boldsymbol{A} \in \mathbb{R}^{N \times N}$, i.e., A_{ij} $(i, j \in \{1, \cdots, N\})$, are given by

$$A_{ij} \leftarrow \begin{cases} 1, \text{ if } \gamma_{ij} = 1 \\ 0, \text{ otherwise} \end{cases} .$$

The degree matrix $\boldsymbol{D} \in \mathbb{R}^{N \times N}$ is a diagonal matrix whose entries D_{ij} $(i, j \in \{1, \cdots, N\})$ are given by

$$D_{ij} \leftarrow \begin{cases} d_i, \text{ if } i = j \\ 0, \text{ otherwise} \end{cases} ,$$

where d_i represents the total number of robots that communicate with robot i.

Herein, we define several terms related to the communication topology based on graph theory. The communication topology is undirected if the communication between robots is bi-directional; otherwise, the communication topology is directed. Moreover, the communication topology is connected if communication is possible for any robot when starting from any robot at adjacent robots; otherwise, the communication is disconnected.

According to [78], the communication topology is connected if the following condition is satisfied.

$$\mathrm{rank}(\boldsymbol{L}) = N - 1, \tag{3.1}$$

where $\boldsymbol{L} := \boldsymbol{D} - \boldsymbol{A}$ is the graph Laplacian.

3.1.5. Consensus algorithm

We consider N robots with a vector $\boldsymbol{c} := [\boldsymbol{c}_1, \cdots, \boldsymbol{c}_N]$. The objective of this problem is to converge N vectors to the same value. One common method is Laplacian averaging [79], which is used to average the estimates of N robots. This algorithm achieves a consensus given by

$$\boldsymbol{c}[s+1] = \boldsymbol{c}[s] - k\boldsymbol{L}\boldsymbol{c}[s], \tag{3.2}$$

where $\boldsymbol{c}[s]$ represents the vector at step s, and k is a positive constant. Using Eq. (3.2), we can make $\boldsymbol{c}$ converge to the average value after m iterations, as follows:

$$\lim_{m\to\infty} \boldsymbol{c}[s+m] = \frac{\mathbf{1}\mathbf{1}^\top}{N}\boldsymbol{c}[s] \tag{3.3}$$

According to [80], the consensus in Eq. (3.3) can be guaranteed if the following conditions are satisfied:

- The communication topology is undirected and connected.
- $0 < k < \frac{2}{N}$

3.2. Multi-robot task allocation

3.2.1. Notation

We denote the position of robot $i\,(i = 1, \cdots, N)$ by $\boldsymbol{x}_i \in \mathbb{R}^2$. The position, velocity, and desired position of the object $l\,(l = 1, \cdots, M)$ are represented by $\boldsymbol{z}_l \in \mathbb{R}^2$, $\boldsymbol{v}_l \in \mathbb{R}^2$ and $\boldsymbol{z}_l^* \in \mathbb{R}^2$, respectively. Robot i can observe robots $j \in \mathcal{N}_i^{\text{Robot}} := \{j_{i1}, \cdots, j_{iK}\}$ and objects $l \in \mathcal{N}_i^{\text{Load}} := \{l_{i1}, \cdots, l_{iK}\}$, whose positions are K nearest from $\boldsymbol{x}_i$.

3.2.2. Problem formulation

We consider a team of N robots. Each of these robots selects one object simultaneously among the M objects with different and unknown weights. Multi-object transportation requires a huge amount of learning time compared to the single-object transportation. Therefore, we learned the multi-robot policies that derive multi-robot task allocation and communication strategies while the control strategy is not considered. In this study, we simplify the problem such that the robots can move the object if the total load capacity of the robot within a certain distance from the object exceeds the mass of the object.

The objective is to transport all the objects to the desired positions as quickly as possible.

We made the following assumptions:

- Robots know M and N
- Robots know the current and desired positions of M objects
- Robots can communicate with other robots if necessary

3.2.3. MARL settings

To address the multi-robot task allocation problem for multi-object transport, we describe the MARL settings using a Markov decision process.

Let us denote the state, action, and observation of robot i $(i = 1, \cdots, N)$ as $\boldsymbol{s}_i$, $\boldsymbol{a}_i$, and $\boldsymbol{o}_i$, respectively. Robot i selects action $\boldsymbol{a}_i$ under local observation $\boldsymbol{o}_i$ including robots $j \in \mathcal{N}_i^{\text{Robot}}$ and objects $l \in \mathcal{N}_i^{\text{Load}}$. Action $\boldsymbol{a}_i$ includes a variable to compute the priority of objects $l \in \mathcal{N}_i^{\text{Load}}$ and variables to determine communicating task priorities with other robots. Robot i updates the task priorities by computing the current actions $\boldsymbol{a}_i$, then selects the object with the highest priority among the M objects. After robot i moves to the selected object for a certain control period, $\boldsymbol{s}_i$ transitions to the next state $\boldsymbol{s}_i'$. Simultaneously, robot i receives reward r_t at every step t when moving the object or carrying it to the desired position. Robot i updates its policy by maximizing the expected reward $\mathbb{E}[R_t] = \mathbb{E}\left[\sum_{k=0}^{T-1} \gamma^k r_{t+k}\right]$, where $\gamma \in [0, 1]$ is a discount factor and T is the total number of steps per episode.

3.3. Touch-based object shape estimation using accelerometers

3.3.1. Problem formulation

This section introduces the shape estimation problem considered in the present study. The objective of this problem is to estimate the shape of an unknown object placed in a given exploration region $Q \in \mathbb{R}^3$.

In [47], an implicit surface is used to describe the shape of an object by means of a real-value shape potential function. Based on this shape potential function, we can determine whether each point is located on the surface of, inside, or some distance from the object. As described in [50], we defined the observation of a shape potential value y_i $(i = 1, \cdots, n)$ at the observed point $\boldsymbol{x}_i \in \mathbb{R}^3$ as

$$y_i = \begin{cases} 0, \boldsymbol{x}_i \text{ on the surface} \\ 1, \boldsymbol{x}_i \text{ outside the surface} \\ -1, \boldsymbol{x}_i \text{ inside the surface.} \end{cases}$$

In this study, the shape of the object is computed offline. Moreover, we assumed that the position of the quadcopter is accurate. This assumption holds true when using high-precision positioning systems such as motion capture systems or RTK-GPS.

3.3.2. Observation model

Observations were obtained by detecting events. An illustration of this event detection when using an acceleration is shown in Fig. 3.2. In this example, an event takes the form of the quadcopter colliding with the object or becoming unstable. If an event occurs and the acceleration becomes larger than the threshold, y_i is labeled as being zero. When physical contact occurs, false-negative contact data can occur when the quadcopter collides with the object at low speed. Moreover, false-positive contact data are detected when the quadcopter becomes unstable.

As shown in Fig. 3.2, it is difficult to determine the threshold so that the number of false-negative and false-positive contact data will be zero. Although the number of items of false-positive contact data could be reduced by constructing more sophisticated classifiers, it is unrealistic to classify false-positive contact data with a 100 % accuracy rate for objects made of any material. Therefore, in the present study, we explored another approach by assuming that the generation of false-positive contact data is unavoidable.

3.3.3. Distribution of observations

With GPs, we are required to assume the distribution of observations. With homoscedastic GPs, the distribution is assumed to follow a normal distribution,

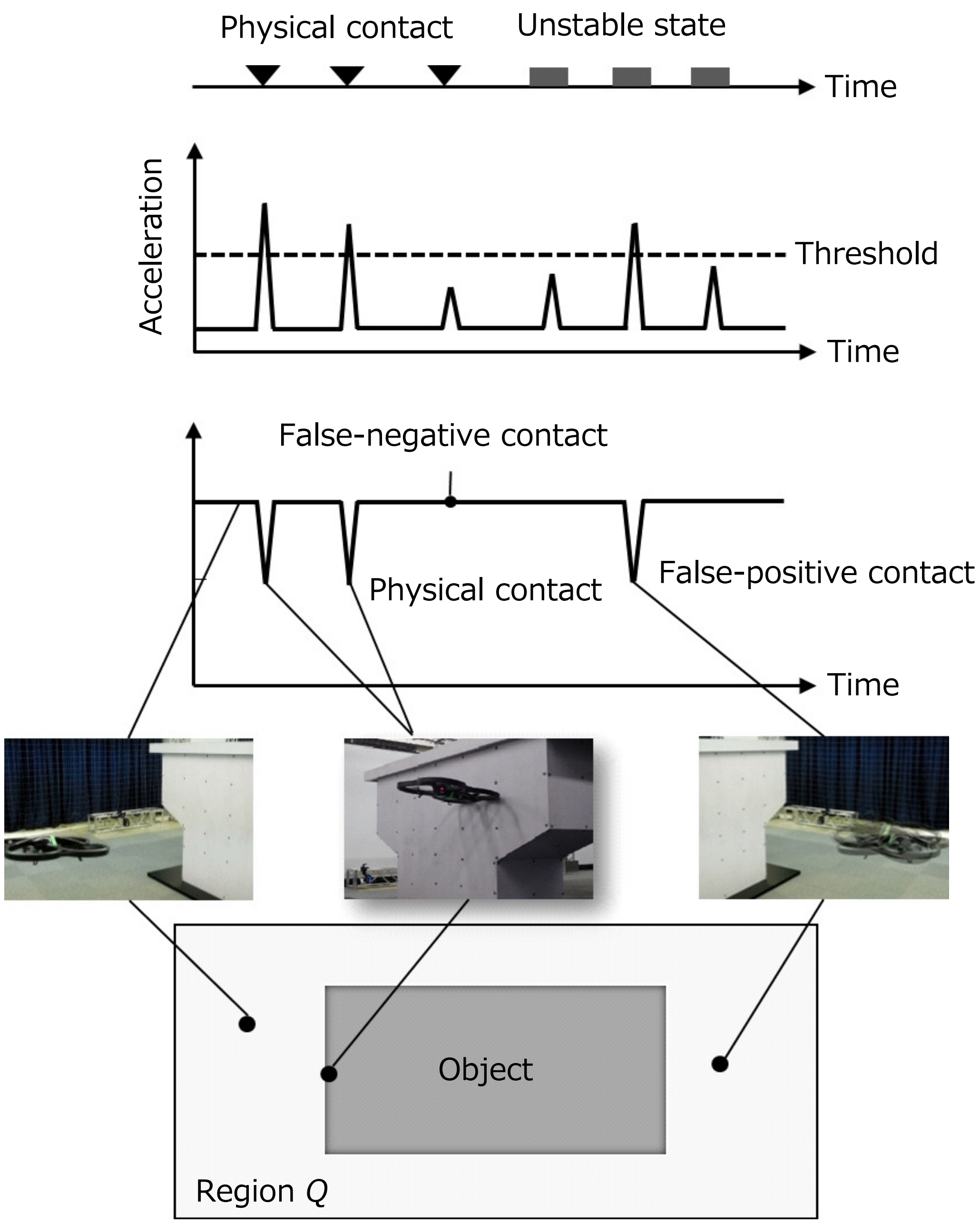


Figure 3.2. Event detection and contact detection based on acceleration. When physical contact occurs, the acceleration becomes larger than a threshold (dashed line), such that the measurement y will be zero. In this case, false-negative contact data can be detected when the quadcopter collides with an object at low speed. Moreover, false-positive contact data are detected when the quadcopter becomes unstable and the acceleration exceeds the threshold.

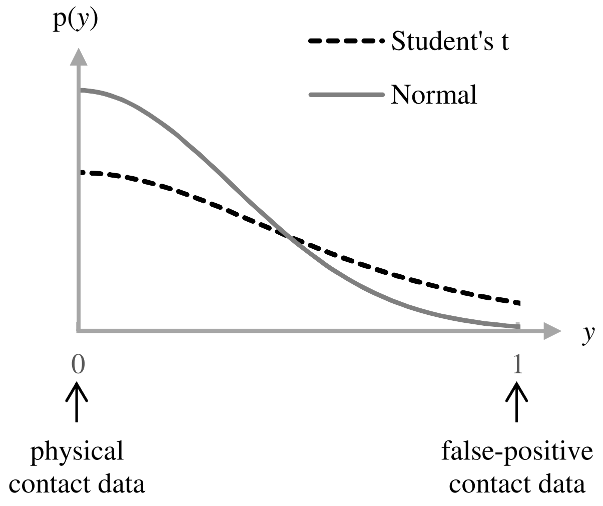


Figure 3.3. Comparison of Student's t- and normal distribution.

as defined by

$$\mathcal{N}(y_i \mid f_i, \sigma^2) = \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left[-\frac{(y_i - f_i)^2}{2\sigma^2}\right],$$

where f_i and σ^2 represent the shape potential function and the noise variance with a constant value, respectively. This distribution may be unsuitable for application to our problem; contact data with false-positive samples cannot be explained by such a light-tailed distribution, owing to the nature of the false-positive contact data and its representation with three natural numbers (-1/0/1). Thus, it may be unsuitable for use with false-positive contact data.

Therefore, we introduce the Student's t-distribution for the observations, given by

$$\mathcal{T}(y_i \mid f_i, \lambda, \nu) = \frac{\Gamma(\frac{\nu+1}{2})}{\Gamma(\frac{\nu}{2})} \left(\frac{\lambda}{\pi\nu}\right)^{\frac{1}{2}} \left[1 + \frac{\lambda(y_i - f_i)^2}{\nu}\right]^{-\frac{\nu+1}{2}},$$

where ν, λ and Γ represent the degree of freedom, a scale parameter and gamma function, respectively. As shown in Fig. 3.3, false-positive contact data can be explained with a greater degree of probability, owing to its heavy-tailed distribution.

4. Deep reinforcement learning of event-triggered communication and consensus-based control for distributed cooperative transport

In this chapter, we propose a multi-agent reinforcement learning framework of event-triggered communication and consensus-based control for distributed cooperative transport. The proposed policy model estimates the resultant force and torque in a consensus manner using the estimates of the resultant force and torque with the neighborhood robots. Moreover, it computes the control and communication inputs to determine when to communicate with the neighboring robots under local observations and estimates of the resultant force and torque. Therefore, the proposed framework can balance the control performance and communication savings in scenarios wherein the number of robots differs from that in the training environment.

We demonstrate the effectiveness of the proposed algorithm through cooperative transport and cooperative rotation tasks. We confirm the versatility of our framework through cooperative transport task using two robots for randomly arranged initial and desired positions of the payload in the simulations. Moreover, we confirm the scalability of our framework by using a maximum of eight and six robots in the simulations and experiments, respectively.

The remainder of this chapter is organized as follows. Section 4.1 introduces the proposed framework of event-triggered communication and consensus-based control. Section 4.2 demonstrates the effectiveness of our algorithm through numerical simulations. Section 4.3 demonstrates the effectiveness of our method through real robot experiments. Finally, Section 4.4 presents the conclusions of this chapter.

4.1. Method

This section introduces a MARL framework that can be applied to cooperative transport with varying numbers of robots.

4.1.1. Distributed policy model

Overview

Figure 4.1 presents the proposed policy model of event-triggered communication and consensus-based control for distributed cooperative transport. Our method exploits a distributed policy model that computes the communication and control inputs using local observations and the resultant force and torque with consensus estimation.

Robot i clusters N robots into K nearest robots in the group $\mathcal{N}_i$. To ensure that the policy is scalable to the number of robots, each robot estimates the resultant force and torque using those of the neighborhood robots. Note that the robot can control the payload using the velocity of the payload if the payload moves. However, the robot cannot use the velocity because it will always be 0 if the load cannot be moved. Therefore, we considered the resultant force and resultant torque, which are the internal information of the payload. Based on the consensus algorithm, robot i estimates the resultant force $\boldsymbol{F}_i \in \mathbb{R}^2$ and torque $T_i \in \mathbb{R}$ using $\boldsymbol{F}_j$ and T_j obtained from robot $j \in \mathcal{N}_i$ via communication. Event-triggered communication (ETC) determines when to communicate with robot $j \in \mathcal{N}_i$ at every control step. Policy π_i calculates the control input $\boldsymbol{u}_i \in \mathbb{R}^2$ and communication input $\alpha_i \in \mathbb{R}$ using local observation $\boldsymbol{o}_i = [\boldsymbol{e}^\top, \boldsymbol{v}^\top, \omega, \boldsymbol{x}_i^\top \theta_i]^\top$, $\boldsymbol{F}_i$ and T_i, where $\boldsymbol{e} = [\boldsymbol{x}^\top - \boldsymbol{x}^{*\top}, \theta - \theta^*]^\top$ is the error vector.

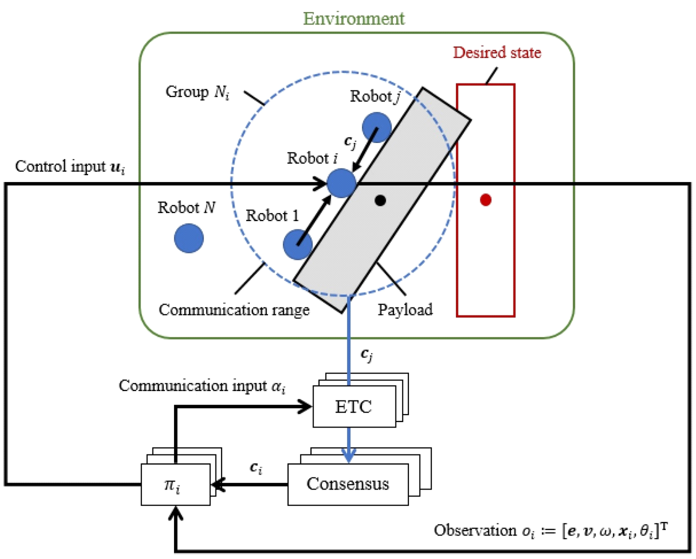


Figure 4.1. The proposed policy model of event-triggered communication and consensus-based control for distributed cooperative transport

Considering that our policy model computes the control and communication inputs under local observations and the estimates of the resultant force and torque of the neighborhood robots, it can be applied to scenarios wherein the number of robots differs from that in the training environment.

Consensus estimation

This subsection details the estimation of the resultant force and torque in a consensus manner. Let denote the estimates of robot i by $\boldsymbol{c}_i = \frac{1}{N}[\boldsymbol{F}_i^\top, T_i]^\top$. By communicating with nearest robots, $\boldsymbol{c}_i$ can be estimated by

$$\boldsymbol{c}_i[t + \Delta t] = \boldsymbol{c}_i[t] + k \sum_{j \in \mathcal{N}_i} (\boldsymbol{c}_j[t] - \boldsymbol{c}_i[t]), \tag{4.1}$$

where Δt is the consensus period. At every control period ΔT ($> \Delta t$), robot i updates $\boldsymbol{c}_i$ using $\boldsymbol{c}_i \leftarrow \left[\boldsymbol{f}_i^\top, \tau_i\right]^\top$. We adopted the average value to compute the resultant force and torque by multiplying the average value by N. By estimating the resultant force and torque, the robots can transport the object even if the number of robots varies.

In the present study, we cannot theoretically guarantee the convergence of the consensus estimation. Instead, we verify the manner in which a communication topology using our method may satisfy the first condition of the average consensus through numerical simulations.

Considering Δt in Eq. (4.1) is smaller than ΔT, the communication costs may increase owing to the high-rate communication required for the estimation. To address this issue, we introduce an event-triggered architecture that determines the timing of communication with neighborhood robots while controlling the payload to the desired state.

Event-triggered communication and consensus-based control

In this subsection, we introduce the ETC and consensus-based control that balances the transport performance and communication savings for varying number of robots.

In ETC, robot i receives $\boldsymbol{c}_j$ from robot $j \in \mathcal{N}_i$ by

$$\boldsymbol{c}_j = \begin{cases} \boldsymbol{c}_j, \text{ if } \gamma_{ij} = 1 \\ 0, \text{ if } \gamma_{ij} = 0 \end{cases} . \tag{4.2}$$

According to [23,24], the timing of communication is decided based on a trigger law given by

$$\gamma_{ij} = 1 \Longleftrightarrow \alpha_i \geq 0 \tag{4.3}$$

where $\alpha_i \in [-1, 1]$ is an output of the policy of robot i. Using Eq. (4.3), robot i makes the communication decision with robot $j \in \mathcal{N}_i$ by

$$\gamma_{ij} = \begin{cases} 1, \text{ if } \alpha_i \geq 0 \text{ and } j \in \mathcal{N}_i \\ 0, \text{ otherwise} \end{cases} , \tag{4.4}$$

Moreover, to ensure that the communication topology is undirected, robot i

updates γ_{ij}, given as follows:

$$\gamma_{ij} \leftarrow \begin{cases} 1, \text{ if } \gamma_{ji} = 1 \\ \gamma_{ij}, \text{ otherwise} \end{cases}. \tag{4.5}$$

Our ETC and consensus-based control involves a policy that calculates the communication and control inputs using local observations, as well as the resultant force and torque with consensus estimation, given as follows:

$$\boldsymbol{a}_i = \left[\boldsymbol{u}_i^\top, \alpha_i\right]^\top = \pi_i\left(\boldsymbol{o}_i, \boldsymbol{c}_i\right), \tag{4.6}$$

where π_i is computed by a deep neural network. The communication input α_i is used to make the communication decision with other robots in the next control step using Eq. (4.4).

The calculation steps used in the ETC and consensus-based control is shown in Algorithm 1.

Algorithm 1: ETC and consensus-based control

```
Initialize $\boldsymbol{x}$, $\theta$, $\boldsymbol{x}^*$, $\theta^*$, $\boldsymbol{x}_i$, $\theta_i$ $(i = 1, \cdots, N)$
for $t = 1, ..., T$ do
    for $i = 1, ..., N$ do
        /* procedure for grouping $\mathcal{N}_i$
        Calculate $K$ nearest robots among $N$ robots
        /* procedure for ETC
        Determine the robots to communicate using Eqs. (4.2) and (4.5)
        /* procedure for consensus-based control
        Update $\boldsymbol{c}_i \leftarrow [\boldsymbol{f}_i^\top, \tau_i]^\top$
        for $s = t, ..., t + \Delta T$ do
            Estimate $\boldsymbol{c}_i$ using Eq. (4.1)
        Compute $\boldsymbol{u}_i$ and $\alpha_i$ using Eq. (4.6)
```

4.1.2. Reward design

To balance the control performance and communication savings, we designed the reward of robot i, given as follows:

$$r_i = -\|\boldsymbol{e}\|_2 - \lambda_1 \sum_{j \in \mathcal{N}_i} \gamma_{ij} - \lambda_2 p_i \tag{4.7}$$

$$p_i = \begin{cases} 1, & \text{if} \ \ \|\boldsymbol{x}_i - \boldsymbol{x}\|_2 > \delta \\ 0, & \text{otherwise} \end{cases} \tag{4.8}$$

where δ and λ_i $(i = 1, 2)$ are the positive constant and hyperparameters, respectively. The second term in Eq. (4.7) minimizes the communication with neighboring robots at every control step. Moreover, we added the third term in Eq. (4.7) to improve the learning efficiency by making robots move within a certain distance δ from the payload's position.

4.1.3. Policy optimization

The weight parameters in the policy networks in the learning process are optimized to maximize the expected reward. In this study, we optimized the multi-agent policies using the multi-agent deep deterministic policy gradient (MADDPG) [21], which is a multi-agent variant of the deep actor-critic algorithms.

One of the primary problems in MARL is that the variance of the policy gradient can be large when the number of agents increases in partially observable environments. To address this issue, the MADDPG algorithm adopts a learning framework called "centralized training and decentralized execution." The critic networks approximate the optimal Q-value functions using observations and actions of all agents. In contrast, the policy networks are optimized using a policy gradient method, wherein each actor network can access its own observations and actions. Once the policies are trained, each policy network can compute the action under local observations. Details of the policy optimization steps can be found in [21].

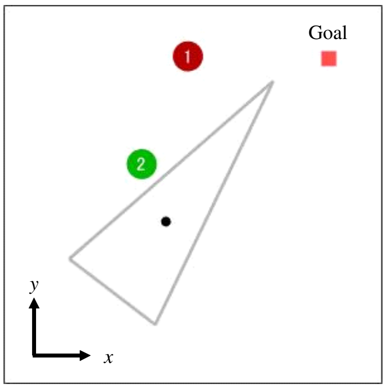


Figure 4.2. Cooperative transport task. The colored circles, gray triangle, and black dot represent the robots, payload, and the position of the payload, respectively.

4.2. Simulation

We demonstrate the effectiveness of the proposed algorithm through cooperative transport and cooperative rotation tasks in a simulation. We confirmed the versality of the framework through a cooperative transport task using two robots for randomly arranged initial and the desired positions of the payload. Moreover, we confirmed the scalability of our framework through a cooperative rotation task using varying number of robots.

4.2.1. Cooperative transport task

Setup

We began with a 2D cooperative transport task to confirm that the proposed algorithm could balance the control performance and communication saving for randomly generated initial positions and the desired positions of the payload, as shown in Fig. 4.2.

We used a triangular object with side lengths of 1.0, 1.0, and 0.4 m. The mass and moment of inertia were set to 1.0×10^1 kg and 6.0×10^{-2} kg·m^2, respectively. The shape of the robot was circular. The radius, mass, and moment of inertia were set to 0.10 m, 1.1 kg and 5.3 kg·m^2, respectively. The control input of robot i was $\boldsymbol{u}_i = [u_i^v, u_i^\omega]^\top$, where $u_i^v \in \mathbb{R}$ and $u_i^\omega \in \mathbb{R}$ are the linear and angular velocity inputs, respectively. We also set $\mid u_i^v \mid \leq 0.4$ and $\mid u_i^\omega \mid \leq 2.0$.

We set the number of robots and neighborhood robots to $N = 2$ and $K = 1$, respectively. The initial positions of the payload and robots were randomly generated within region $Q := \{(x, y) \mid 2.0 \leq x \leq 3.0, 2.0 \leq y \leq 3.0\}$, whereas the desired position of the payload was randomly generated within region Q.

Further, numerical simulations were carried out using the code in [81] and the dynamics presented in [82]. Table 4.1 lists the simulation conditions. The parameters used in the MARL method were set by trial and error.

To confirm the effectiveness of the proposed framework, we compared our algorithm with two communication topologies, as follows:

- **Full**: each robot receives the resultant force and torque at every control step.
- **Nocom**: each robot never receives force and torque from other robots.

We carried out three trainings for each communication topology under the same conditions.

Moreover, to evaluate each communication topology quantitatively, we defined the control error and communication cost as follows:

$$E = \frac{1}{M} \sum_{m=1}^{M} \|\boldsymbol{e}_m(T)\|_2 \tag{4.9}$$

$$C = \frac{1}{M} \sum_{m=1}^{M} \sum_{i=1}^{N} \sum_{j=1}^{N} \sum_{k=1}^{T} \gamma_{ij}^{m}(k) \frac{\Delta T}{\Delta t} \tag{4.10}$$

where M is the number of trials. We set $M = 1.0 \times 10^3$ in the evaluations.

Training performance

Figure 4.3 shows the comparison of the cumulative reward of the first term in (4.7) at each episode when applying each method to the cooperative transport task. The results showed that the proposed method achieved a cumulative reward as high as that of the **Full** method. Moreover, it was greater than that of the **Nocom** method. These results indicate that the estimates of the resultant force and torque could improve the cumulative reward.

Table 4.1. Simulation conditions of the cooperative transport task

Variable	Value
Control period [s]	0.1
Consensus period [s]	0.05
k in Eq. (4.1)	0.5
Number of steps per episode	1.5×10^2
Number of episode	8.0×10^5
Number of hidden layers (critic)	4
Number of hidden layers (actor)	4
Number of units per layer	64
Activation function of hidden layers	ReLU
Activation function of output layers (critic)	linear
Activation function of output layers (actor)	tanh
Discount factor	0.99
Batch size	4096
Replay buffer	1.0×10^6

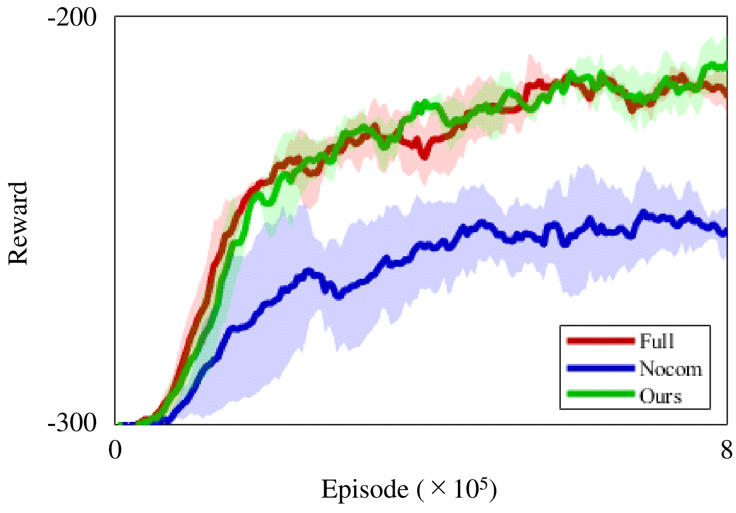
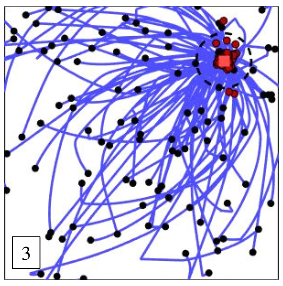
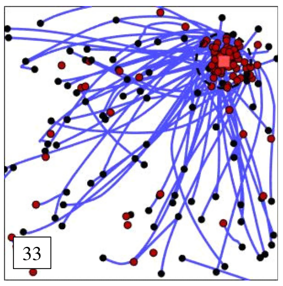


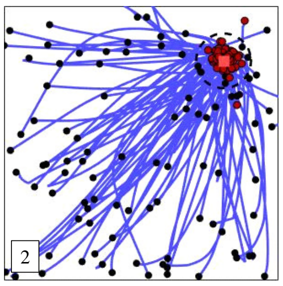
Figure 4.3. Comparison of the cumulative reward of the first term in Eq. (4.7) when applying each method to the cooperative transport task.



(a) **Full**



(b) **Nocom**



(c) **Ours**

Figure 4.4. Results of position control for randomly arranged initial positions of the payload when applying each method to the cooperative transport task. The blue lines, black dots, red dots, and dashed circles denote the trajectories of the payload, positions of the payload at the initial and last steps, and the areas within 0.1 m from the desired position, respectively. The number denotes the total number of trials where the payload fails to be controlled within 0.1 m from the desired position.

Table 4.2. Comparisons of the control error and communication cost when applying each method to the cooperative transport task.

	Full	**Nocom**	**Ours**
E	0.05 m	0.22 m	**0.05** m
C	4.0×10^2	0.0	$\mathbf{1.5}\times10^2$

Transport performance and communication saving

Figure 4.4 shows the results of cooperative transport for randomly arranged initial positions of the payload when each method is applied. We performed 100 trials for each method. The results showed that the **Full** and **Ours** successfully controlled the position of the payload within 0.1 m from the desired position for most trials, whereas the **Nocom** method failed for 33 trials.

Table 4.2 shows the comparisons of the mean absolute error and communication cost for each communication topology. The results showed that our method achieved errors as small as that of the **Full** method. Moreover, compared to the **Full** method, our method saved communication costs by 63%.

Overall, our method achieved transport performance as good as that of full communication topology while reducing communication costs for randomly generated initial positions and desired positions of the payload.

Communication and estimation

Herein, we verify that our method can determine communication timing while estimating the resultant force and torque. Figure 4.5a shows the trajectories and communication occurrence at each time. The results showed that communication occurred while robots determined the edge that they should push. During this period, robot 1 changed the pushing position on the same edge as robot 2 to move the payload. Afterwards, the robots kept pushing the payload without communication until the payload reached the desired position.

Figures 4.5b and 4.5c show the communication occurrences and the estimates of the resultant force and torque by two robots. The results showed that the communication rate was high a few seconds after the start of the control, whereas it was low at the end of the control. Moreover, our method estimated the resultant force and torque closer to the true values, compared to the method without consensus when communication occurred in the red-colored regions.

Overall, our method could determine the timing of communication of neighborhood robots while estimating the resultant force and torque using communication.

4.2.2. Cooperative rotation task

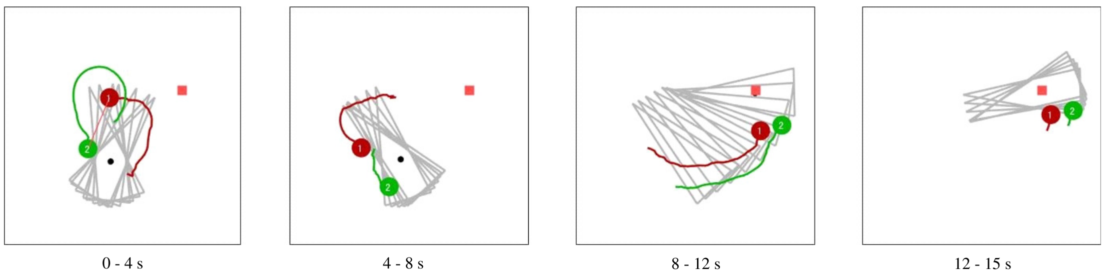
Setup

The second simulation is a cooperative rotation task to confirm that our algorithm can balance control performance and communication savings for varying number of robots, as shown in Fig. 4.6.

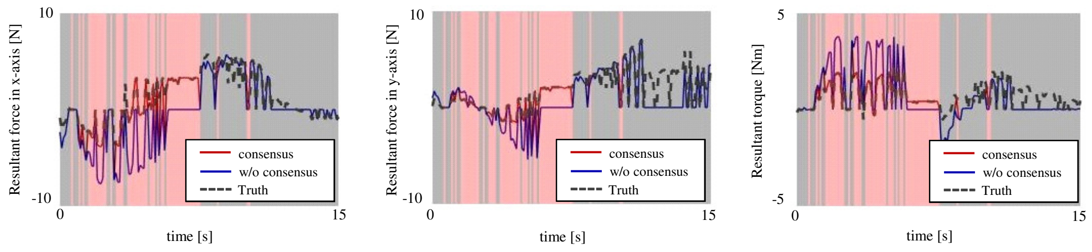
The mass and moment of inertia were set to 2.0×10^1 kg and 7.3 kg·m^2, respectively. The shape, radius, mass, and moment of inertia were set to the same value as in the previous simulation. Furthermore, we set $\mid u_i^v \mid \leq 0.2$ and $\mid u_i^\omega \mid \leq 2.0$.

The yaw angle in the world coordinate is defined as shown in Fig. 4.6. The center position of the payload is fixed to [2.0 m, 2.0 m]$^\top$. The initial yaw angle of the payload was randomly generated within $[0.4\pi, 0.6\pi]^\top$, whereas the desired yaw angle was set to 0 or π. The initial positions of the robots were randomly set to [1.35 m, 1.0 m]$^\top$ or [1.35 m, 3.0 m]$^\top$ with a 50% probability.

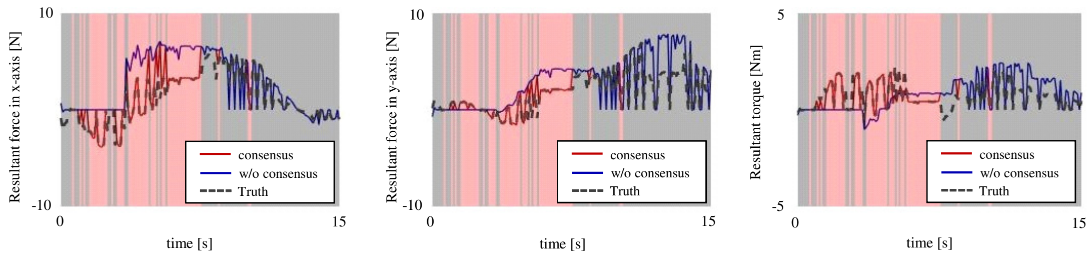
The simulation conditions are listed in Table. 4.3. We set the number of robots and neighborhood robots to $N = 5$ and $K = 2$, respectively, in the training



(a) Trajectories and communication occurrence



(b) Robot 1



(c) Robot 2

Figure 4.5. Trajectories, communication occurrence, and the estimation results when applying the proposed method to the cooperative transport task. The red lines in Fig. 4.5a represent the communication occurrence. In Figs. 4.5b and 4.5c, each robot receives the resultant force and torque estimates from other robots in red-colored regions. In contrast, the robots do not receive them in gray-colored regions.

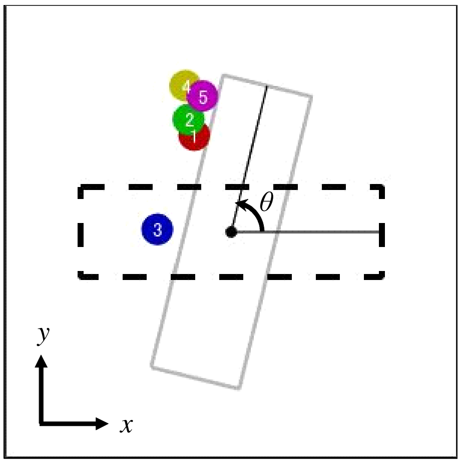


Figure 4.6. Cooperative rotation task. The colored circles, gray rectangular, and dashed rectangular denote the robots, payload, and the desired state of the payload, respectively.

phase. To confirm the scalability of our method, we executed the trained policies for $N \in \{3, 4, 5, 6, 7, 8\}$. Moreover, we set the parameters in Eqs. (4.7) and (4.8) to $\lambda_1 = 0.02$, $\lambda_2 = 0.1$ and $\delta = 1.2$ by trial and error, respectively.

To confirm the effectiveness of our algorithm, we compared our algorithm with three communication topologies as follows:

- **Full**: each robot receives the resultant force and torque at every control step.
- **ETC**: each robot determines the timing for robots to receive the force and torque using the event-triggered communication [25].
- **Nocom**: no robot receives force and torque from any other robots.

We carried out three trainings for each communication topology under the same conditions.

Moreover, we evaluated the control error E, communication cost C, and transport time P, which were defined by the average time to control the error of the yaw angle within ± 10 deg for 1.0×10^3 trials.

Training performance

Figure 4.7 compares the cumulative reward of the first term in Eq. (4.7) when applying each method to the cooperative rotation task. The results showed that

Table 4.3. Simulation conditions of the cooperative rotation task

Variable	Value
Control period [s]	0.1
Consensus period [s]	0.02
k used in the consensus	0.2
Number of steps per episode	2.0×10^2
Number of episode	6.0×10^5
Number of hidden layers (critic)	4
Number of hidden layers (actor)	4
Number of units per layer	64
Activation function of hidden layers	ReLU
Activation function of output layers (critic)	linear
Activation function of output layers (actor)	tanh
Discount factor	0.99
Batch size	4096
Replay buffer	1.0×10^6

our method achieved cumulative rewards as high as that of the **Full** method. Moreover, the rewards were greater than those of the **ETC** and **Nocom** methods. These results show that the estimates of the resultant force and torque improved the cumulative reward.

Scalability analysis

Figure 4.8a compares the control errors when applying each method for varying number of robots. The results showed that the control errors of the **Nocom** and **ETC** methods became considerably large when the number of robots was small. This can be attributed to the fact that the yaw angle could not reach the desired value in the given control steps, considering the resultant torque decreased as the number of robots decreased. In contrast, the **Full** and our methods reduced these errors compared to the **Nocom** and **ETC** methods. As the number of robots increased, the differences among the **Full**, **ETC**, and proposed methods decreased. Meanwhile, the **Full** and proposed methods achieved shorter transport time compared to the **ETC** method, as shown in Fig. 4.8b.

Figure 4.8c compares the communication costs. The results showed that the

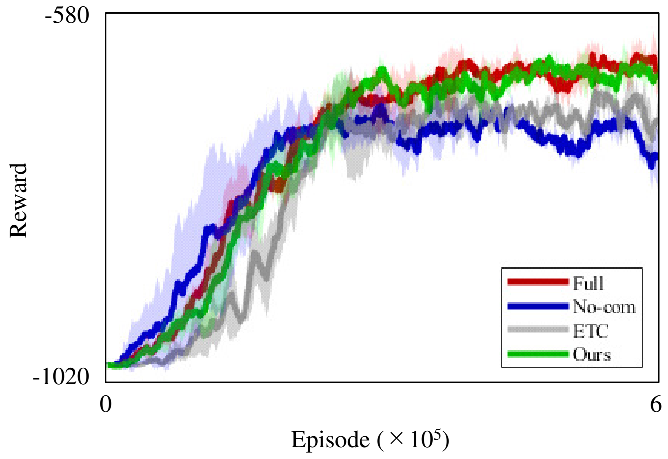


Figure 4.7. Comparisons of the cumulative reward of the first term in Eq. (4.7) when applying each method to the cooperative rotation task.

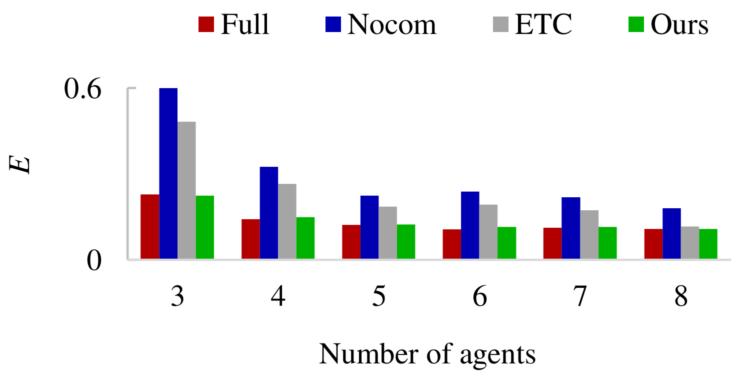
communication cost of the proposed method was higher than that of the **ETC** method, considering that our method required high-rate communication to estimate the resultant force and torque. Meanwhile, our method decreased the communication cost significantly compared to the **Full** communication, whose control performance was almost as good as that of our method.

Overall, our method achieved transport performance as good as that of the **Full** communication topology while saving the communication costs for varying number of robots.

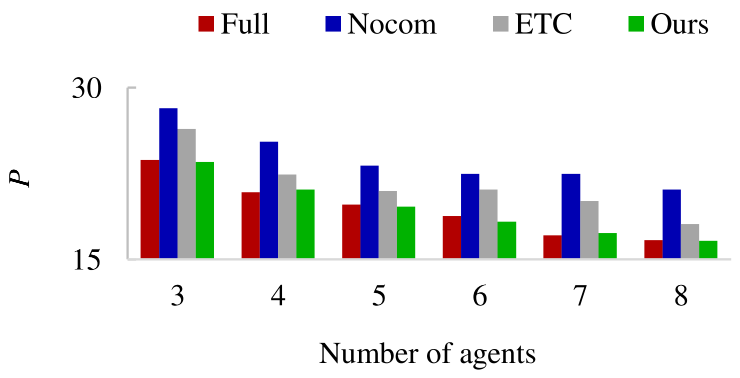
Communication and estimation

Herein, we verify that our method can determine the communication timing while estimating the resultant torque. Figure 4.9a shows the trajectories and communication occurrence at each time. The results showed that communication occurred while robots determine the direction in which they rotate the payload. During this period, robot 4 changed the pushing position to rotate the payload in the same direction as the other robots. Following this, robots kept pushing the payload until the yaw angle of the payload reached the desired value.

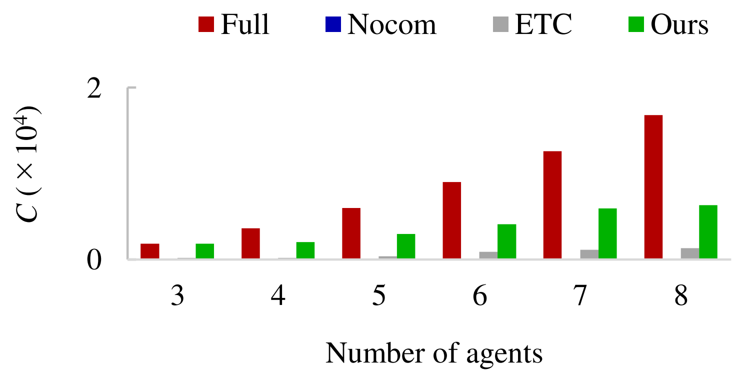
Figure 4.9b depicts the communication occurrence and the estimates of the resultant torque by all robots. The robots adopted high-rate communication to



(a) Control error



(b) Transport time



(c) Communication cost

Figure 4.8. Comparison of the control error, transport time, and communication cost when applying each method to the cooperative rotation using various number of robots.

determine the direction in which they rotate the payload a few seconds after the start of the control. Once the payload rotated clockwise, they adopted low-rate communication, considering that they pushed the payload until the yaw angle of the payload reached the desired value. Moreover, our method estimated the resultant force and torque closer to the true values compared to those of the method without consensus when communication occurred in red-colored regions.

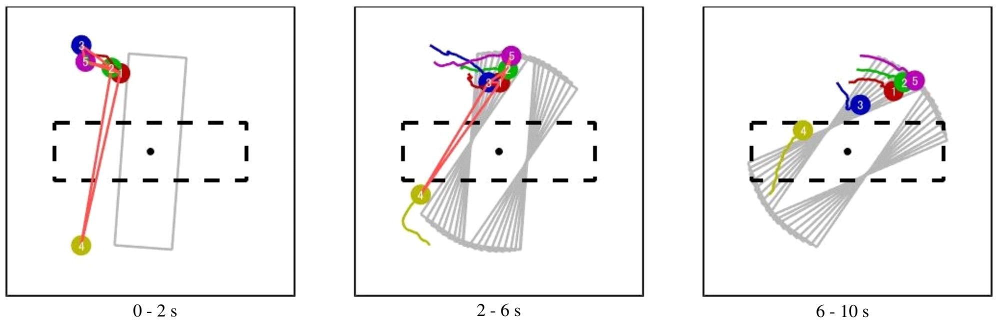
Although the estimated values do not perfectly converge to the true values, the robots could transport the object to the desired value. Because we considered both object errors and communication costs in the reward design, the polices were trained so that they could balance communication savings and transport performance at the same time.

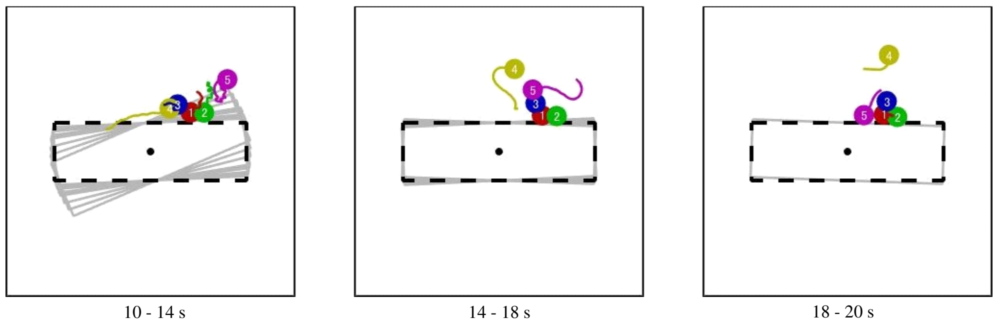
Overall, the proposed method could determine communication timing with the neighboring robots while estimating the resultant torque using local observations.

Connectivity of communication topology

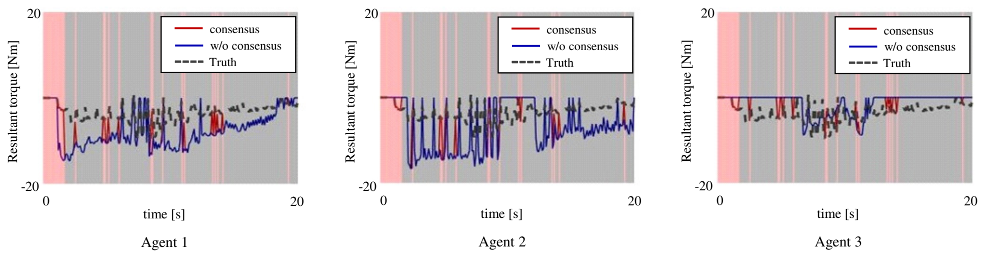
Next, we verify the manner in which a communication topology using our method may satisfy the first condition of the average consensus by introducing a connectivity metric R_c, as described Appendix B.

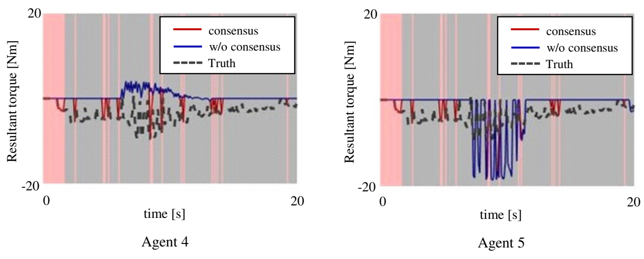
Table 4.4 shows a comparison of R_c for each method. We compared our method with a **Random** method, where we randomly skipped communication among robots with a probability of 50%. The proposed method improves the estimation accuracy if the communication topology satisfies the connectivity condition. Moreover, the estimation accuracy improves the transport performance. Consequently, our method achieved a value of $R_c = 0.7$ although we did not promote the connectivity in the reward. In contrast, the **Nocom**, **ETC**, and **Random** methods resulted in R_c values that were lower than the value of our method as they did not require the connectivity condition. These results show that our method could obtain the necessary condition to estimate the resultant force and torque using the estimates of the resultant force and the torque of the neighborhood robots.





(a) Trajectories and communication occurrence





(b) Resultant torque estimated by each robot

Figure 4.9. Trajectories, communication occurrence, and the estimation results when the proposed method is applied to the cooperative rotation task. The red lines in Fig. 4.9a show that mutual communications of the estimated resultant force and torque occur between robots. In Fig. 4.9b, each robot receives the resultant force and torque estimates from other robots in red-colored regions. In contrast, the robots do not receive them in gray-colored regions.

Table 4.4. The ratio of the connectivity

	Nocom	**ETC**	**Random**	**Ours**
R_c	0.0	0.0	0.23	**0.7**

4.3. Real robot experiment

This section shows the effectiveness of our learning method through real robot experiments using multiple ground robots to confirm that the settings in the simulation are realistic and the scalability of our method holds true in real robot environments.

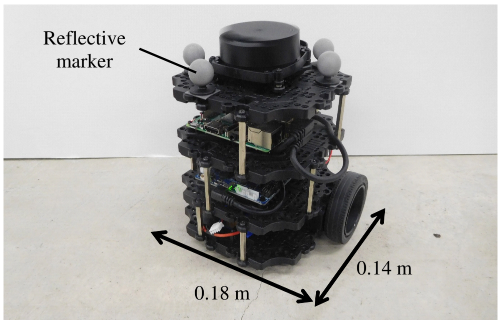
4.3.1. Setup

The real robot demonstration was performed using Turtlebot3 Burger robots, and the experimental configuration is shown in Figs. 4.10a and 4.10b.

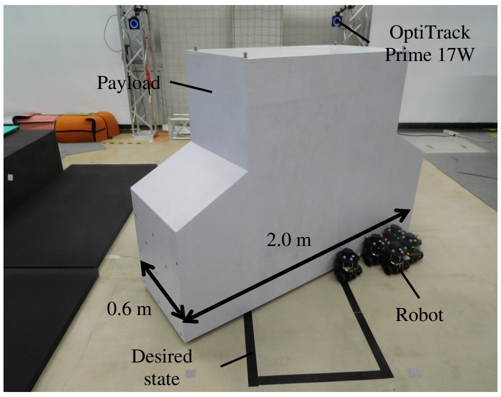
Our experimental system utilized an OptiTrack Prime 17 W motion capture system (Natural Point, Inc., Corvallis, OR) to observe the positions and yaw angles of the payload and robots at 10 Hz. The linear and angular velocities of the payload and robots were calculated using the measured positions and yaw angles. We trained the policies by setting the number of robots and neighborhood robots to $N = 5$ and $K = 2$, respectively, in the simulation and executed the trained policies for $N \in \{3, 4, 5, 6\}$ in the real environment. The trained policies calculated the control inputs in a PC with an 8-core Intel ® Core$^{\text{TM}}$ i7 (2.80 GHz) with 32 GB RAM. The control inputs were transmitted from the control PC to each robot using Wi-Fi communication at 10 Hz.

4.3.2. Result

Figure 4.11 shows the mean absolute error of the yaw angle when applying our method to various numbers of robots. The results show that our method can control the yaw angle of the payload to the desired value for various numbers of robots. Moreover, the time for controlling the yaw angle becomes shorter as the number of robots increases.



(a) Robot



(b) Experimental configuration

Figure 4.10. Robot and experimental configuration used in the experiment. The positions and yaw angles of the payload and robots were observed using a motion capture system.

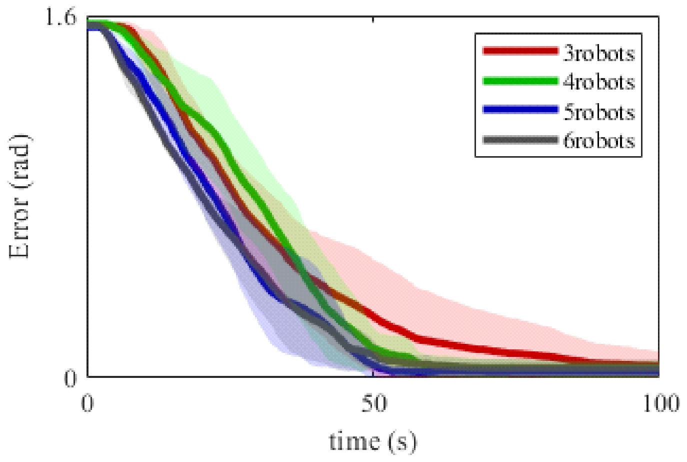


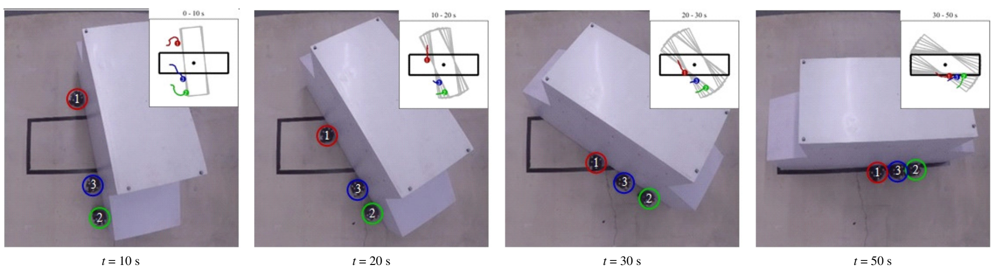
Figure 4.11. Mean absolute error of the yaw angle when applying our method to various number of robots for 10 trials.

Figure 4.12 depicts the trajectories of the payload and robots for various number of robots. After several robots came in contact with the payload, the payload began to rotate clockwise. Once the payload rotated, several robots changed the pushing positions to rotate the payload in the same direction as the other robots. The robots kept rotating the payload collaboratively until the yaw angle of the payload reached the desired value for various number of robots.

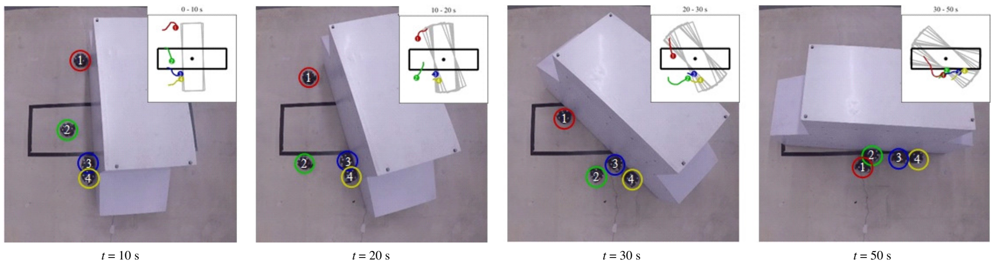
Overall, we verified that the settings in the simulation were realistic and the scalability of our method holds true in real robot environments.

4.4. Summary of Chapter 4

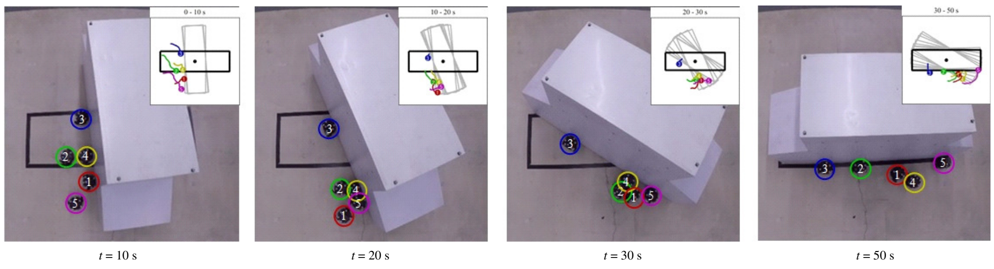
In this chapter, we proposed a learning framework of ETC and consensus-based control for distributed cooperative transport. The proposed method achieved transport performance as good as that of full communication while saving the communication costs through cooperative transport tasks using two robots for randomly arranged initial and desired positions of the payload. Moreover, our method achieved transport performance as good as that of full communication while saving the communication costs through cooperative rotation tasks in scenarios wherein the number of robots differed from that in the training environment in the simulations and experiments.



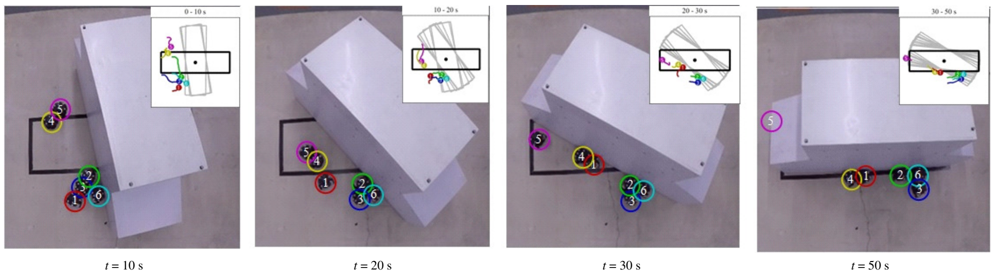
(a) Three robots



(b) Four robots



(c) Five robots



(d) Six robots

Figure 4.12. Trajectories of the payload and robots when applying our method to various number of robots. We also show the top view of the trajectories of the payload and robots. The black line denotes the desired state of the payload.

5. Deep reinforcement learning of multi-robot task allocation for cooperative transport

In this chapter, we propose a learning framework that can handle scenarios for various numbers of robots and objects with different and unknown weights. First, we introduce a structured policy model consisting of 1) predesigned dynamic task priorities with global communication and 2) a neural network-based distributed policy model that determines the timing for coordination. The distributed policy builds consensus on the high-priority object under local observations and selects cooperative or independent actions. Then, the policy is optimized by multi-agent reinforcement learning through trial and error. This structured policy of local learning and global communication makes our framework applicable to various numbers of robots and objects with different and unknown weights.

We demonstrated the effectiveness of the proposed algorithm through multi-object transport simulations using multiple robots. We confirm the scalability of our framework using various numbers of robots and objects with different and unknown weights. Moreover, we confirm the versatility of our framework by varying the proportion of heavy and light objects.

The remainder of this chapter is organized as follows. Section 5.1 details MARL and the proposed learning framework. Section 5.2 shows the effectiveness of our framework through multi-robot transport simulations. Section 5.3 presents the conclusions of this chapter.

5.1. Method

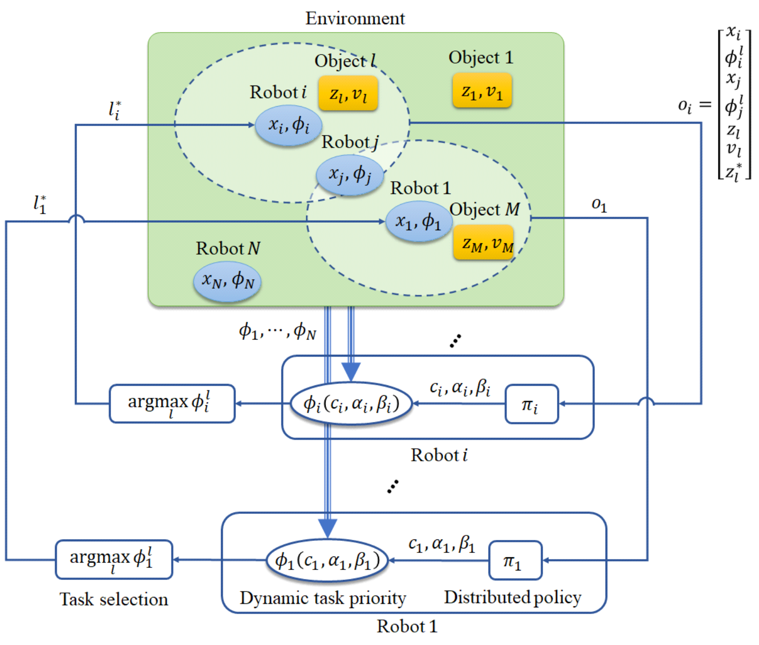
In this section, we introduce the proposed MARL framework that can handle a varying number of robots and objects with different and unknown weights.

5.1.1. Overview

Fig. 5.1 shows the overview of the learning framework. Each robot has a distributed policy, which differs from those of other robots. Robot i has task priority $\boldsymbol{\phi}_i := \left[\phi_i^1, \cdots, \phi_i^M\right]^\top \in \mathbb{R}^M$, where $\phi_i^l \in [0,1]$ is the priority of the lth object possessed by robot i. Robot i updates the priority of the neighboring object $l \in \mathcal{N}_i^{\mathrm{Load}}$ under local observation $\boldsymbol{o}_i = [\boldsymbol{x}_i, \phi_i^{l_{i1}}, \cdots, \phi_i^{l_{iK}}, \boldsymbol{x}_{j_{i1}}, \phi_{j_{i1}}^{l_{i1}}, \cdots, \phi_{j_{i1}}^{l_{iK}}, \cdots, \boldsymbol{x}_{j_{iK}}, \phi_{j_{iK}}^{l_{i1}}, \cdots, \phi_{j_{iK}}^{l_{iK}}, \boldsymbol{z}_{l_{i1}}, \boldsymbol{v}_{l_{i1}}, \boldsymbol{z}_{l_{i1}}^*, \cdots, \boldsymbol{z}_{l_{iK}}, \boldsymbol{v}_{l_{iK}}, \boldsymbol{z}_{l_{iK}}^*]$ using $\boldsymbol{c}_i = \left[c_i^1, \cdots, c_i^K\right]^\top \in \mathbb{R}^K$, where $c_i^l \in [0,1]$ is the reference value of ϕ_i^l. The reference value $\boldsymbol{c}_i$ is computed by policy π_i. Limiting the minimum number of robots and objects to a constant value makes the policy applicable to varying numbers of robots and objects. However, this policy cannot transport an object whose weight exceeds the load capacity of robots observing the object because it cannot update the priorities of the object $l \notin \mathcal{N}_i^{\mathrm{Load}}$.

The proposed framework introduces dynamic task priorities with global communication and a neural network-based distributed policy model. The distributed policy computes communication inputs $\alpha_i \in [0,1]$ and $\beta_i \in [0,1]$ under local observations, where α_i is the parameter by which the robot i receives task priority from other robots, and β_i is the parameter by which the robot i sends $\boldsymbol{\phi}_i$ to other robots. Because the observations involve the velocity of the object, the robot sends the priorities to the other robots if they cannot move the selected object. Moreover, if the other robots receives the task priorities, the dynamic task priority makes the agents establish a consensus on the high-priority object and select cooperative actions. Otherwise, robots select independent actions. Therefore, robots can transport all objects efficiently without knowing the number of robots required to transport objects. Robot i selects the object l_i^*, which has the highest priority among M objects. Then, the policy is optimized by MARL through trial and error.

Note that we focus on the transport performance and the communication sav-



$$o_i = \begin{bmatrix} x_i \\ \phi_i^l \\ x_j \\ \phi_j^l \\ z_l \\ v_l \\ z_l^* \end{bmatrix}$$

Figure 5.1. Overview of learning framework. Robot i updates task priorities of the neighboring objects using $\boldsymbol{c}_i$ while building consensus on the high-priority object using α_i and β_i according to the distributed policy π_i under local observations $\boldsymbol{o}_i$. Robot i selects the object l_i^* which has the highest priority among M objects.

ings are not considered in this study.

5.1.2. Dynamic task priority with global communication

This subsection introduces the dynamic task priority with global communication to select an object among various candidates.

We design the dynamic task priority such that the robot i can update ϕ_i^l ($l \in \mathcal{N}_i^{\text{Load}}$) according to its policy while updating ϕ_i^l ($l \notin \mathcal{N}_i^{\text{Load}}$) using the priorities of the N robots. In this case, the robots should balance cooperative and independent actions to transport all the objects efficiently.

To this end, we design the dynamic task priority of object l for robot i given by

$$\dot{\phi}_i^l = \begin{cases} k_\phi(c_i^l - \phi_i^l) + \sigma_i \sum_{j=1}^N d_j k_\phi(\phi_j^l - \phi_i^l), & \text{if } l \in \mathcal{N}_i^{\text{Load}} \\ \sigma_i \sum_{j=1}^N d_j k_\phi(\phi_j^l - \phi_i^l), \text{otherwise} \end{cases} \tag{5.1}$$

where $k_\phi > 0$, d_i and σ_i are equal to 0 or 1. We introduced the first-order linear time-delay system to avoid the occurrences of chattering, where the robots travel back and forth between different objects. $k_\phi(c_i^l - \phi_i^l)$ induces an independent action while $k_\phi(\phi_j^l - \phi_i^l)$ induces a cooperative action. If $\sigma_i = 1$ and $d_j = 1$, $k_\phi(\phi_j^l - \phi_i^l)$ makes ϕ_i^l asymptotically converge to ϕ_j^l, establishing consensus on the task priority. Otherwise, $k_\phi(c_i^l - \phi_i^l)$ makes ϕ_i^l asymptotically converge to c_i^l according to its own policy. The distributed policy calculates σ_i and d_i to reach a consensus on the high-priority object as well as $\boldsymbol{c}_i$ under local observations.

5.1.3. Distributed policy model

We introduce a distributed policy model under local observations $\boldsymbol{o}_i$ given by

$$\boldsymbol{a}_i = \left[\boldsymbol{c}_i^\top, \alpha_i, \beta_i\right]^\top = \pi_i(\boldsymbol{o}_i), \tag{5.2}$$

where π_i is computed by a deep neural network. Agent i determines the reference values of ϕ_i^l using c_i^l for K local objects while maintaining the priority of the M objects.

Using α_i and β_i in Eq. (5.2), request signal d_i and response signal σ_i are calculated by the event-triggered law in [23, 25] given by

$$d_i(\alpha_i) = \begin{cases} 1, \text{ if } \alpha_i > 0.5 \ \& \ \|\boldsymbol{v}_{l_i^*}\|_2 = 0 \\ 0, \text{ otherwise} \end{cases}, \tag{5.3}$$

$$\sigma_i(\beta_i) = \begin{cases} 1, \text{ if } \beta_i > 0.5 \ \& \ \|\boldsymbol{v}_{l_i^*}\|_2 = 0 \\ 0, \text{ otherwise} \end{cases}, \tag{5.4}$$

where robot i can transmit and receive the priority when it cannot move the selected object l_i^*. Fig. 5.2 illustrates the communication of the task priority using our distributed policy under local observation. Using the triggering law in Eqs. (5.3) and (5.4), robot i can receive $\boldsymbol{\phi}_j$ transmitted by robot j and then reach consensus on the high-priority object using Eq. (5.1).

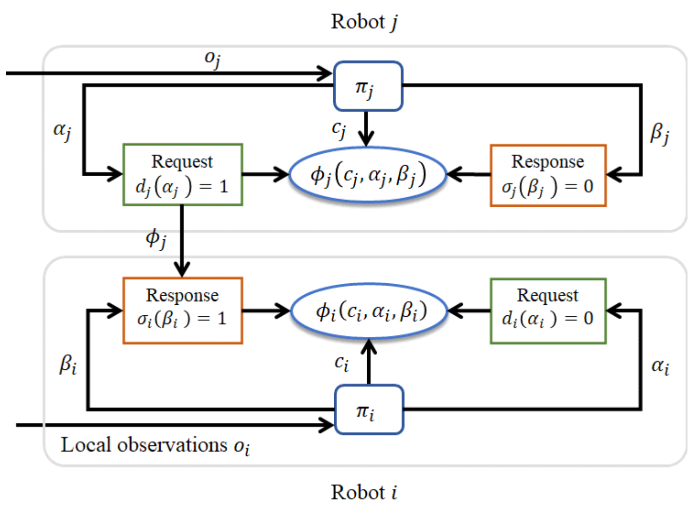

Robot j
o_j
π_j
α_j
c_j
β_j
Request $d_j(\alpha_j) = 1$
$\phi_j(c_j, \alpha_j, \beta_j)$
Response $\sigma_j(\beta_j) = 0$
ϕ_j
Response $\sigma_i(\beta_i) = 1$
$\phi_i(c_i, \alpha_i, \beta_i)$
Request $d_i(\alpha_i) = 0$
β_i
c_i
α_i
π_i
Local observations o_i
Robot i

Figure 5.2. Example of the communication of the task priority using the proposed distributed policy under local observation. When $d_j(\alpha_j) = 1$ and $\sigma_i(\beta_i) = 1$, robot i receives $\boldsymbol{\phi}_j$ transmitted by robot j.

5.1.4. Task selection

This subsection introduces the procedure for the selection of an object based on its priority. Robot i selects the object with the highest priority among M objects using $l_i^* = \arg\max_l \phi_i^l$. Moreover, we set the priority of the object that has reached close to the desired position using $\phi_i^l \leftarrow 0$, if $\|\boldsymbol{z}_l - \boldsymbol{z}_l^*\|_2 < \delta$, where $\delta > 0$ represents a threshold to determine whether the object reaches the desired position.

5.1.5. Reward Design

To transport all the objects to the desired positions as quickly as possible, we designed a reward function given by

$$r = \sum_{l=1}^{M} P_l + \lambda \sum_{l=1}^{M} \|\boldsymbol{v}_l\|_2, \tag{5.5}$$

where λ is a positive constant. $P_l = 1$ if $\|\boldsymbol{z}_l - \boldsymbol{z}_l^*\|_2 < \delta$; otherwise $P_l = 0$. The first term in (5.5) aims to transport all the objects to the desired position, while the second term aims to move as many objects as possible.

5.1.6. Policy Optimization

In this study, we optimized the multi-agent policies using multi-agent deep deterministic policy gradient (MADDPG) [21], which is one of the deep actor-critic algorithms for multi-agent systems.

A common issue of MARL is that the learning becomes unstable as the number of unobservable agents increases. The MADDPG algorithm addressed this problem using a learning framework called "centralized training and decentralized execution." During training, the weight parameters of the critic networks are optimized through the Q-learning algorithm [83] using the observations and actions of all the agents. At the same time, the weight parameters of the actor networks are optimized through a policy gradient method using its observations and actions. During execution, the actor networks compute actions under local observations. See [21] for the details of the policy optimization steps.

5.2. Simulation

We conducted multi-object transport simulations using multiple robots to confirm the scalability and versatility of the proposed framework for various numbers of robots and objects and various proportions of heavy and light objects.

5.2.1. Simulation setup

We show the simulation scenario in Fig. 5.3. We randomly generated the initial positions of the robots and objects in the region $Q := \{(x, y) \mid 2 \leq x \leq 8, 2 \leq y \leq 8\}$. The desired positions of the objects were evenly arranged on a circumference with a center and radius of $[5.0, 5.0]^\top$ and 4.0 m, respectively. We set the load capacity of the robot to 1 kg.

During training, we set $K = 2$, $N = 3$, and $M = 6$, while setting the object's mass to 1 or 3 kg with 50 % probability. To confirm the scalability of the algorithm, we evaluated $N \in \{3, 6\}$ and $M \in \{4, 6, 8, 10\}$.

The multi-agent policies were optimized using MADDPG, a deep actor-critic algorithm. We used the MADDPG code in [81] and set the simulation parameters as listed in Table 5.1.

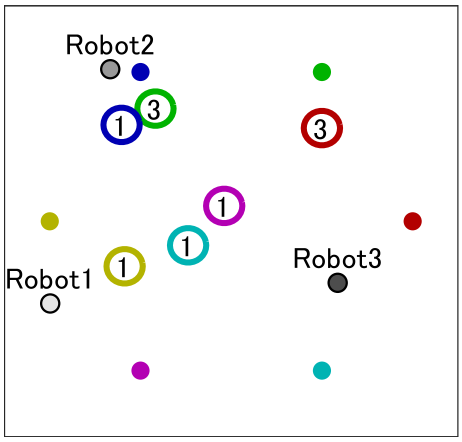


Figure 5.3. Simulation scenario. The colored circles, dots, and numbers indicate the objects, their desired positions, and the number of robots required to transport the object, respectively. Robots should transport each object to the desired position with the same color.

We set k_ϕ in Eq. (5.1) to 0.2 such that the priority changed according to a first-order delay with the time constant of 5 s, which was longer than the selection period. The threshold δ was set to 0.05 for the positions of the objects to be controlled within 0.05 m from the desired positions. We set the weight parameter λ in Eq. (5.5) to 3.0×10^2 such that transporting a different object obtained almost the same reward as locating an object to the desired position.

To confirm the effectiveness of our framework, we conducted comparisons through the following methods:

- **Nearest**: Each robot selects the nearest object
- **One**: Each robot is randomly assigned an object from the M objects
- **Local**: Our method without dynamic task priority with global communication by setting $\sigma_i = 0$ in Eq. (5.1).
- **Nearest-one**: Each robot selects the object closest to its current position. When the robot does not move the object for a specific time t_s, the robot picks the same object as the robot, unable to carry the load for the longest time. We set $t_s = 1.0$ s for all the robots.
- **No-com**: **Local** method under local observations without the task priorities.

Table 5.1. Simulation parameters

Parameter	Value
Selection period [s]	1.0
Number of steps per episode	150
Number of episodes	2.0e5
Number of hidden layers (critic)	4
Number of hidden layers (actor)	4
Number of units per layer	64
Activation function of hidden layers	ReLU
Activation function of output layers (critic)	linear
Activation function of output layers (actor)	tanh
Discount factor	0.99
Batch size	1024

- **No-dynamics**: **No-com** method without the dynamics of the task priority by setting $\phi_i^l = c_i^l$ ($l \in \mathcal{N}_i^{\mathrm{Load}}$).

To evaluate our approach quantitatively, we used the following measures:

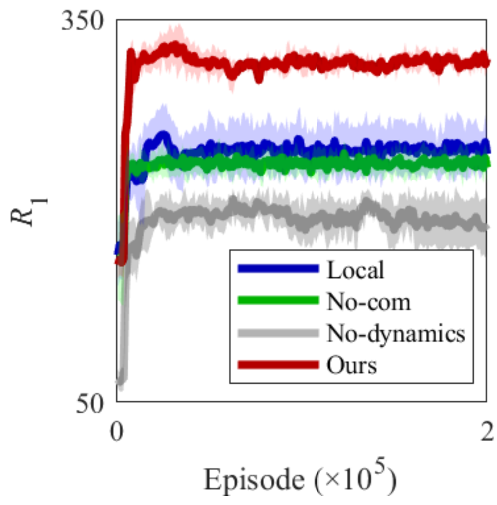
- Success rate (SR): The ratio of trials to 100 trials, in which robots can transport all the objects to the desired positions within 10 min. We considered 10 min for method **One** to achieve a 100 % success rate for various numbers of robots and objects.

- Transport time (TT) [s]: Average time required to move all the objects to the desired positions within 10 min.

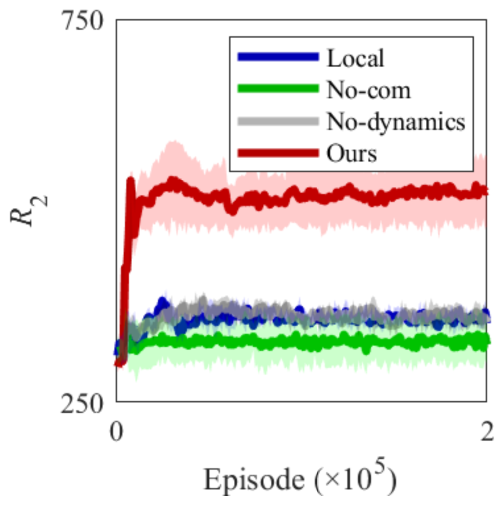
5.2.2. Comparisons of training performance

We evaluated the effects of dynamic task priority and communication on the training performance by comparing our framework with methods **Local**, **No-com**, and **No-dynamics**. For each method, we repeated the training three times.

Fig. 5.5 shows the cumulative rewards of the first and second terms in Eq. (5.5), which are denoted as R_1 and R_2, respectively. When applying method **No-dynamics**, we confirmed the occurrences of chattering where the robots travel



(a) First term



(b) Second term

Figure 5.4. Cumulative rewards of various evaluated methods.

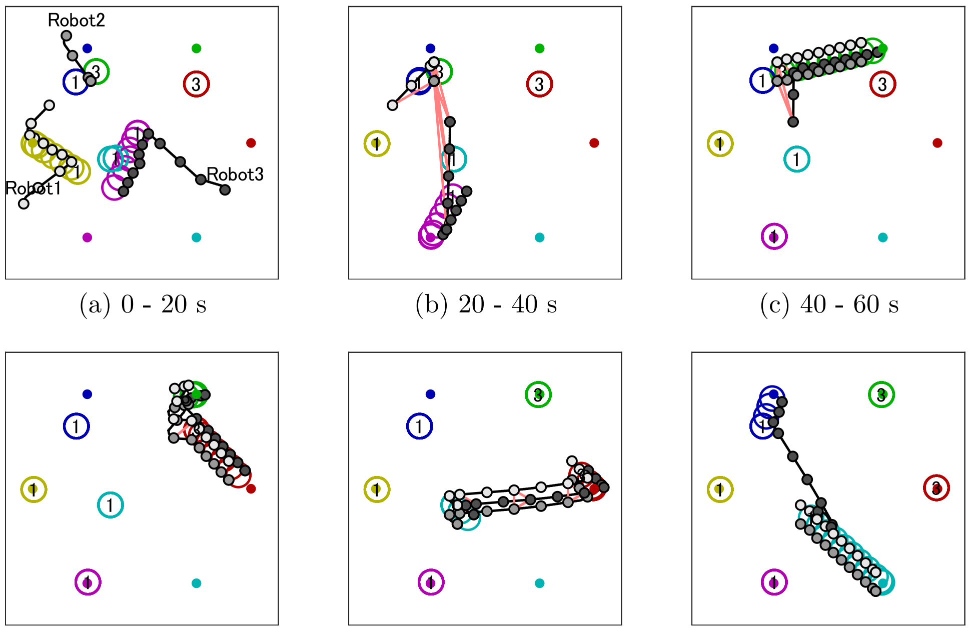
back and forth between different objects. As a result, this method made R_1 achieve smaller values compared to those in other methods.

Method **Local** achieves slightly higher R_1 and R_2 values than method **No-com**. Therefore, training the policy with the priority of neighboring robots improves the training performance. Moreover, the proposed framework achieves higher values than the other methods. These results indicate that the dynamic task priority with global communication in Eq. (5.1) has a greater impact on the training performance of our framework than the local communication of task priorities.

5.2.3. Emergence of cooperative and independent actions

We confirmed the emergence of cooperative and independent actions when applying the proposed framework. We show trajectories and communication occurrences when applying the framework in Fig. 5.5.

At the initial stage, robots 1 and 3 transport different objects, while robot 2 cannot move the object, which requires three robots to transport, as shown in Fig. 5.5a. To prevent this situation, robot 2 transmits its priority to other robots, as shown in Fig. 5.5b. While robots 1 and 3 receive the priority of robot 2, their priorities gradually approach that of robot 2, as shown in Figs. (5.5a–5.5c). Once



(a) 0 - 20 s (b) 20 - 40 s (c) 40 - 60 s

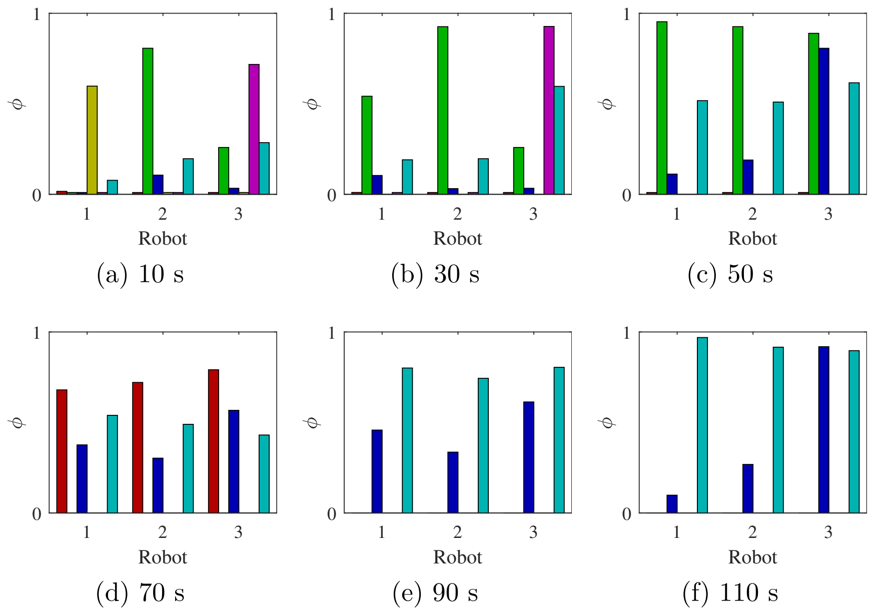
(d) 60 - 80 s (e) 80 - 100 s (f) 100 - 120 s

Figure 5.5. Trajectories and communication occurrences. We offset the overlapping robot trajectories for clarity. The black and red lines show the trajectories of the robots and occurrences of priority communication.

the priority of the green object is the highest for the three robots, the object is transported to the desired position, as shown in Fig. 5.5c. Fig. 5.6d shows that the priority of the red object is the highest in the corresponding period for all the robots, which transport the object to the desired position, as shown in Fig. 5.5d.

Finally, two objects remain to be transported by three robots, as shown in Fig. 5.5e. While three robots transport the light-blue object according to the priority in Fig. 5.6e, the priority of the blue object is the highest for robot 3, as shown in Fig. 5.6f. Hence, robot 3 moves the blue object, and the two objects can be transported to the desired positions, as shown in Fig. 5.5f.

Overall, our framework can balance cooperative and independent actions by determining the timing of priority communication.



(a) 10 s (b) 30 s (c) 50 s (d) 70 s (e) 90 s (f) 110 s

Figure 5.6. Task priorities per robot. The colors correspond to those of objects shown in Fig. 5.5.

5.2.4. Scalability analysis

We evaluated the success rate and transport time when using our framework and other methods for various numbers of robots and objects.

Table 5.2 shows the quantitative results for various numbers of robots and objects when applying each method. Methods **Nearest** and **Local** cannot achieve a 100 % success rate for various numbers of robots and objects. In contrast, **One**, **Nearest-one**, and **Ours** can achieve a 100 % success rate for various numbers of robots and objects. When applying Method **One**, the transport time is the longest because all the robots select the same object. To confirm the effectiveness of our method, we evaluated the average time t_a for carrying two or more objects simultaneously when applying each method to $(N, M) = (6, 10)$ for 100 trials. The large t_a value indicates that the robots select more independent actions. Our method achieves $t_a = 6.0\times10^1$ while Method **Nearest-one** achieves $t_a = 4.9\times10^1$. The results indicate that our method can promote more independent actions compared to Method **Nearest-one**.

Overall, compared to other methods, our framework can reduce the transport

Table 5.2. Quantitative results for various numbers of robots and objects

(N, M)	Metrics	Nearest	One	Local	Nearest-one	**Ours**
(3,4)	SR	0.59	**1.0**	0.93	**1.0**	**1.0**
	TT ($\times 10^2$)	**1.1**	1.4	1.3	1.2	1.2
(3,6)	SR	0.34	**1.0**	0.85	**1.0**	**1.0**
	TT ($\times 10^2$)	**1.7**	2.0	2.2	1.9	1.8
(3,8)	SR	0.17	**1.0**	0.67	**1.0**	**1.0**
	TT ($\times 10^2$)	**2.2**	2.7	2.8	2.5	2.4
(3,10)	SR	0.07	**1.0**	0.54	**1.0**	**1.0**
	TT ($\times 10^2$)	**2.5**	3.4	3.5	3.1	3.0
(6,4)	SR	0.97	**1.0**	**1.0**	**1.0**	**1.0**
	TT ($\times 10^2$)	0.99	1.3	0.91	0.88	**0.86**
(6,6)	SR	0.88	**1.0**	0.98	**1.0**	**1.0**
	TT ($\times 10^2$)	1.5	2.0	**1.4**	**1.4**	**1.4**
(6,8)	SR	0.82	**1.0**	0.88	**1.0**	**1.0**
	TT ($\times 10^2$)	1.9	2.7	1.9	1.9	**1.8**
(6,10)	SR	0.74	**1.0**	0.72	**1.0**	**1.0**
	TT ($\times 10^2$)	2.6	3.6	2.5	2.4	**2.2**

time while transporting all the objects to the desired positions for various numbers of robots and objects.

5.2.5. Versatility analysis by varying proportion of heavy and light objects

Additionally, we verified the versatility of our framework by varying the proportion of heavy and light objects. We set $N = 6$ and $M = 10$ while setting the mass of the objects to 1 or 3 kg. We evaluated each method by generating 3 kg objects with probabilities of 0 %, 25 %, 50 %, 75 %, and 100 %.

Table 5.3 shows the quantitative results for various proportions of heavy and light objects. When applying methods **Nearest** and **Local**, the success rate becomes lower with the increasing proportion of heavy objects. In contrast, **One**, **Nearest-one**, and **Ours** can achieve a 100 % success rate for various proportions of heavy objects. Method **One** increases the transport time compared with **Nearest-one** and our methods because all the robots select a common object regardless of its weight. Moreover, our method achieves a lower transportation time than method **Nearest-one** for various proportions of heavy objects because our

Table 5.3. Quantitative results for various proportions of heavy and light objects. P represents the proportion of heavy objects.

P	Metrics	Nearest	One	Local	Nearest-one	**Ours**
0.0	SR	**1.0**	**1.0**	**1.0**	**1.0**	**1.0**
	TT ($\times 10^2$)	1.2	3.3	1.1	1.2	**1.0**
0.25	SR	0.94	**1.0**	0.98	**1.0**	**1.0**
	TT ($\times 10^2$)	1.9	3.5	1.7	1.8	**1.6**
0.5	SR	0.7	**1.0**	0.76	**1.0**	**1.0**
	TT ($\times 10^2$)	2.5	3.5	2.4	2.4	**2.2**
0.75	SR	0.6	**1.0**	0.36	**1.0**	**1.0**
	TT ($\times 10^2$)	3.1	3.7	3.2	2.9	**2.8**
1.0	SR	0.29	**1.0**	0.14	**1.0**	**1.0**
	TT ($\times 10^2$)	3.5	3.8	3.7	3.4	**3.3**

framework can promote more independent actions than Method **Nearest-one**.

Overall, compared to other methods, our framework can reduce the transport time while transporting all the objects to their desired positions when handling objects with various weights.

5.3. Summary of Chapter 5

In this chapter, we propose a learning framework that can handle scenarios for various numbers of robots and objects with different and unknown weights. The distributed policy model builds consensus on the high-priority object under local observations, thus balancing the cooperative and independent actions. Therefore, compared to other methods, our framework can reduce the transport time while transporting all the objects to their desired positions for various numbers of robots and objects with different and unknown weights.

6. Robust shape estimation with false-positive contact detection

In this chapter, we propose a means of omni-directional contact detection using accelerometers instead of tactile sensors for object shape estimation using touch. Unlike tactile sensors, our contact-based detection method tends to induce a degree of uncertainty with false-positive contact data because the sensors may react not only to actual contact but also to the unstable behavior of the robot. Therefore, it is crucial to consider a robust shape estimation method capable of handling such false-positive contact data. To realize this, we introduce the concept of heteroscedasticity into the contact data and propose a robust shape estimation algorithm based on GPIS. While the GPIS assumes that the observations of contact data follow a normal distribution with a constant variance, our GPIS introduces heteroscedacity into each item of contact data. Thus, we can prevent incorrect shape estimates produced by false-positive contact data.

To demonstrate the effectiveness of our approach, we constructed an experimental system using a quadcopter, using its built-in accelerometer for contact detection. We confirmed that our approach is better than GPIS for a 3D construction when using a quadcopter.

The remainder of this chapter is organized as follows: Section 6.1 introduces the mathematical formula for the GPIS and robust GPIS method, then Section 6.2 shows the effectiveness of our algorithm as demonstrated by numerous simulations. Section 6.3 introduces the constructed experimental system and shows a few of the results of the experiments performed with the quadcopter. Finally, Section 6.4 summarizes this chapter.

6.1. Method

In this section, we describe our robust shape estimation with false-positive contact data. We start with a brief introduction of GPIS, and then discuss the issues associated with the method. Moreover, we propose a robust GPIS method by modifying the distributions of the observations made with the GPIS method.

6.1.1. GPIS

In a GPIS framework, the shape potential function f is modeled by GP regression for a given data set $\mathcal{D} = \{\boldsymbol{x}_i, y_i\}$ $(i = 1, \cdots, n)$. In the GPIS, a prior of the shape potential functions $\boldsymbol{f} := [f(\boldsymbol{x}_1), \cdots, f(\boldsymbol{x}_n)]^\top$ is assumed to follow a normal distribution: $p(\boldsymbol{f} \mid \boldsymbol{x}) = \mathcal{N}(\boldsymbol{f} \mid \boldsymbol{0}, \boldsymbol{K})$, where $\boldsymbol{x} := [\boldsymbol{x}_1, \cdots, \boldsymbol{x}_n]^\top$, and $\boldsymbol{K} \in R^{n\times n}$ is the training data covariance matrix. Moreover, the probability of observations $\boldsymbol{y} := [y_1, \cdots, y_n]^\top$ is given by the normal distribution with constant variance: that is, $p(\boldsymbol{y} \mid \boldsymbol{f}) = \mathcal{N}(\boldsymbol{y} \mid \boldsymbol{f}, \sigma^2\boldsymbol{I})$. Upon combining these probabilities, the posterior of the shape potential function is given by

$$p(\boldsymbol{f} \mid \mathcal{D}) = \frac{\mathcal{N}(\boldsymbol{y} \mid \boldsymbol{f}, \sigma^2\boldsymbol{I})\mathcal{N}(\boldsymbol{f} \mid \boldsymbol{0}, \boldsymbol{K})}{p(\boldsymbol{y} \mid \boldsymbol{x})}, \tag{6.1}$$

for which the marginal likelihood $\int \mathcal{N}(\boldsymbol{y} \mid \boldsymbol{f}, \sigma^2\boldsymbol{I})\mathcal{N}(\boldsymbol{f} \mid \boldsymbol{0}, \boldsymbol{K})d\boldsymbol{f}$ in Eq. (6.1) can be computed analytically as described in [84, 85]. Moreover, the predictive mean and variance can be computed as

$$\mu(\boldsymbol{x}^*) = \boldsymbol{k}^\top(\boldsymbol{x}^*)(\boldsymbol{K} + \sigma^2\boldsymbol{I})^{-1}\boldsymbol{y}, \tag{6.2}$$

$$\sigma^2(\boldsymbol{x}^*) = k(\boldsymbol{x}^*) - \boldsymbol{k}^\top(\boldsymbol{x}^*)(\boldsymbol{K} + \sigma^2\boldsymbol{I})^{-1}\boldsymbol{k}(\boldsymbol{x}^*), \tag{6.3}$$

where $k(\boldsymbol{x}_i, \boldsymbol{x}_j)$ is the covariance between the observed points $\boldsymbol{x}_i$ and $\boldsymbol{x}_j$, $i, j \in \{1, \cdots, n\}$, $\boldsymbol{k}(\boldsymbol{x}^*)$ is the covariance vector between all the observed points $\boldsymbol{x}_i$ $(i = 1, \cdots, n)$ and the test point $\boldsymbol{x}^*$, $\boldsymbol{K}$ has entries $K_{ij} = k(\boldsymbol{x}_i, \boldsymbol{x}_j)$, $i, j \in \{1, \cdots, n\}$. An estimate of the object's shape is obtained by finding those points $\boldsymbol{x}^*$ where $\mu(\boldsymbol{x}^*) \approx 0$ in Eq. (6.2).

Because all the diagonal terms of $\sigma^2\boldsymbol{I}$ in Eqs. (6.2) and (6.3) have the same value, false-positive contact data have the same influence on the predictive distribution as normal data. An issue, however, is whether the distribution of the observations is assumed to conform to a normal distribution with constant variance.

6.1.2. Proposed method

In this subsection, we propose a robust GPIS with false-positive contact data based on the GPIS method.

To realize this, we used the Student's t-distribution as the distribution of the observations, given by

$$p(\boldsymbol{y} \mid \boldsymbol{f}) = \mathcal{T}(\boldsymbol{y} \mid \boldsymbol{f}, \lambda, \nu). \tag{6.4}$$

Given the prior of the shape potential functions with a normal distribution, $p(\boldsymbol{f} \mid \boldsymbol{x}) = \mathcal{N}(\boldsymbol{f} \mid \boldsymbol{0}, \boldsymbol{K})$, the posterior of the shape potential function is given by

$$p(\boldsymbol{f} \mid \mathcal{D}, \boldsymbol{\psi}, \lambda, \nu) = \frac{\mathcal{T}(\boldsymbol{y} \mid \boldsymbol{f}, \lambda, \nu)\mathcal{N}(\boldsymbol{f} \mid \boldsymbol{0}, \boldsymbol{K})}{p(\boldsymbol{y} \mid \boldsymbol{x}, \boldsymbol{\psi}, \lambda, \nu)}, \tag{6.5}$$

where $\boldsymbol{\psi}$ represents the hyperparameters in the covariance matrix $\boldsymbol{K}$.

In the same way as in the GPIS method, an estimate of the object's shape can be obtained from the posterior mean. However, the marginal likelihood $\int \mathcal{T}(\boldsymbol{y} \mid \boldsymbol{f}, \lambda, \nu)\mathcal{N}(\boldsymbol{f} \mid \boldsymbol{0}, \boldsymbol{K})d\boldsymbol{f}$ in Eq. (6.5) cannot be computed analytically. Thus, we conform to the variational approximation for GP regression with the Student's t-distribution, as described in [73].

The Student's t-distribution in Eq. (6.4) is a scale-mixture of an infinite number of normal distributions, given by

$$\mathcal{T}(y \mid f, \lambda, \nu) = \int_0^\infty \mathcal{N}(y \mid f, \sigma^2)\mathrm{Inv}\Gamma(\sigma^2 \mid \alpha, \beta)d\sigma^2, \tag{6.6}$$

where $\alpha = \nu/2$ and $\beta = \nu/2\lambda$ represent the shape and inverse scale parameter, respectively. The Student's t-distribution becomes heavy-tailed when α becomes small or β becomes large, as described in Appendix A. Since outliers are more likely to occur from a heavy-tailed distribution, it is suitable to explain false-positive contact data. As described later, these parameters are optimized such that the estimated shape best fits the observed data. Moreover, as α becomes small or β becomes large, the noise variance, σ^2, becomes large with high probability, as described in Appendix A.

Using Eq. (6.6), we approximate the posterior in the factorized form given by $p(\boldsymbol{f}, \boldsymbol{\sigma}^2 \mid \mathcal{D}, \boldsymbol{\psi}, \boldsymbol{\theta}) \approx q(\boldsymbol{f})q(\boldsymbol{\sigma}^2)$, where $\boldsymbol{\theta} := [\alpha, \beta]^\top$ and $\boldsymbol{\sigma}^2 = [\sigma_1^2, \cdots, \sigma_n^2]^\top$. The approximate posterior of $\boldsymbol{f}$ and $\boldsymbol{\sigma}^2$ is given by

$$q(\boldsymbol{f}) = \mathcal{N}(\boldsymbol{f} \mid \boldsymbol{m}, \boldsymbol{A}), \tag{6.7}$$

$$q(\boldsymbol{\sigma}^2) = \prod_{i=1}^{n} \mathrm{Inv}\Gamma(\sigma_i^2 \mid \tilde{\alpha}_i, \tilde{\beta}_i), \tag{6.8}$$

where $\boldsymbol{m}$ and $\boldsymbol{A}$ represent the mean and covariance of $q(\boldsymbol{f})$, and $\tilde{\alpha}_i$ and $\tilde{\beta}_i$ represent the shape and inverse scale for each item of data i ($i = 1, \cdots, n$), respectively. If we could approximate the posterior in an analytically tractable form, we could estimate an object's shape from its posterior mean.

To realize this, we introduce the KL divergence between the approximation and the posterior distribution, given by

$$\mathrm{KL}(q \| p) = \int q(\boldsymbol{f})q(\boldsymbol{\sigma}^2) \ln \frac{p(\boldsymbol{f}, \boldsymbol{\sigma}^2 \mid \mathcal{D}, \boldsymbol{\psi}, \boldsymbol{\theta})}{q(\boldsymbol{f})q(\boldsymbol{\sigma}^2)} d\boldsymbol{f} d\boldsymbol{\sigma}^2.$$

The KL divergence is minimized using an expectation maximization (EM) algorithm [86]. In the EM algorithm, the approximate posterior and parameter values are updated by repeating the two steps. In the expectation step (E-step), $\boldsymbol{m}$, $\boldsymbol{A}$ in Eq. (6.7), $\tilde{\alpha}_i$ and $\tilde{\beta}_i$ in Eq. (6.8) are iteratively updated to minimize the KL divergence for given parameters $\boldsymbol{\theta}$ and $\boldsymbol{\psi}$. In the maximization step (M-step), performed after each E-step, $\boldsymbol{\theta}$ and $\boldsymbol{\psi}$ are iteratively updated to minimize the KL divergence for fixed $q(\boldsymbol{f})$ and $q(\boldsymbol{\sigma}^2)$. The EM algorithm iterates the E- and M-steps until the KL divergence converges. Details of the method used to derive the update laws can be found in [73].

After the variational approximation, the mean and variance of the approximate posterior are computed as

$$\mu(\boldsymbol{x}^*) = \boldsymbol{k}^\top(\boldsymbol{x}^*)(\boldsymbol{K} + \boldsymbol{\Sigma})^{-1}\boldsymbol{y}, \tag{6.9}$$

$$\sigma^2(\boldsymbol{x}^*) = k(\boldsymbol{x}^*) - \boldsymbol{k}^\top(\boldsymbol{x}^*)(\boldsymbol{K} + \boldsymbol{\Sigma})^{-1}\boldsymbol{k}(\boldsymbol{x}^*), \tag{6.10}$$

where $\boldsymbol{\Sigma}$ represents a diagonal matrix with entries $\Sigma_{ii} = \tilde{\beta}_i/\tilde{\alpha}_i$ $(i = 1, \cdots, n)$. Since the diagonal terms of $\boldsymbol{\Sigma}$ in Eqs. (6.9) and (6.10) differ from each other, we can introduce heteroscedacity into each item of contact data.

The object shape is reconstructed by finding those points $\boldsymbol{x}^*$ that satisfy $\mu(\boldsymbol{x}^*) \approx 0$ in Eq. (6.9). As $\tilde{\beta}_i/\tilde{\alpha}_i$ becomes large, the contact data i has little influence on $\mu(\boldsymbol{x}^*)$. Therefore, such contact data has little influence on the shape estimates.

Finally, we present the false-positive contact detection. After the variational approximation, we can obtain the uncertainty of data i, given by

$$u_i = \tilde{\beta}_i/\tilde{\alpha}_i. \tag{6.11}$$

The noise variance, σ^2, in Eq. (6.6) becomes large with high probability as u_i becomes large. Therefore, we can judge the uncertainty for each item of contact data using Eq. (6.11).

The main steps used in the shape estimation and false-positive contact detection are shown in Algorithm 2.

6.2. Simulation

We conducted numerous simulations with 2D objects to confirm the reduction in the shape estimation errors and the false-positive contact detection when applying our algorithm.

6.2.1. Experimental setup

We prepared three different object shapes, as shown in Fig. 6.1. The contact points were set on the edges at intervals of 0.01 [m], as shown in Figs. 6.2a and 6.2c, while they were set on the circumference at an interval of 3[deg] in Fig. 6.2b. The internal and external points were randomly set in the given region

Algorithm 2: Robust GPIS

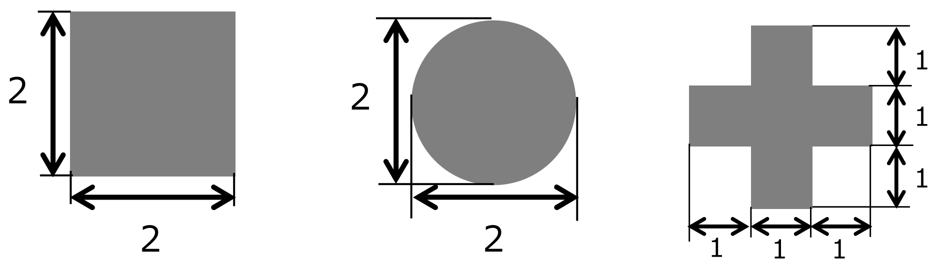
```
Inputs: $x_i$, $y_i$ $(i = 1, \cdots, n)$, $\boldsymbol{\theta}$, $\boldsymbol{\psi}$, $\boldsymbol{x}^*$
Outputs: $\boldsymbol{x}^*$ s.t. $\mu(\boldsymbol{x}^*) \approx 0$, $u_i$ $(i = 1, \cdots, n)$
Initialisation: set $\tilde{\alpha}_i \leftarrow 1$, $\tilde{\beta}_i \leftarrow 1$, $\boldsymbol{\theta}$ and $\boldsymbol{\psi}$ to random positive values.
/* procedure for robust GP
Compute $\boldsymbol{K}$ from $\boldsymbol{x}$ and $\boldsymbol{\psi}$
E-step:
repeat
    $\boldsymbol{m}$, $\boldsymbol{A}$, $\tilde{\alpha}_i$ and $\tilde{\beta}_i$ are iteratively updated for fixed parameters $\boldsymbol{\theta}$ and $\boldsymbol{\psi}$
    using Eqs. (5.34)-(5.36) in [73].
until KL$(q\|p)$ converges
M-step:
repeat
    $\boldsymbol{\theta}$ and $\boldsymbol{\psi}$ are iteratively updated for fixed parameters $\boldsymbol{m}$, $\boldsymbol{A}$, $\tilde{\alpha}_i$ and $\tilde{\beta}_i$
    using Eqs. (5.39)-(5.43) in [73].
until KL$(q\|p)$ converges
Compute $\mu(\boldsymbol{x}^*)$ and $\sigma^2(\boldsymbol{x}^*)$ using Eqs. (6.9) and (6.10).
/* procedure for 3D shape reconstuction
Estimate object shape by finding those points $\boldsymbol{x}^*$ that satisfy $\mu(\boldsymbol{x}^*) \approx 0$.
/* procedure for false-positive contact detection
Estimate the uncertainty of data $i$ from Eq. (6.11)
```

$Q := \{(x_1, x_2) \mid (-3 \leq x_1 \leq 3, -3 \leq x_2 \leq 3)\}$. To consider the false-positive contact data, we replaced the external points with a pair of outliers of contact and internal points with a 2% chance. The outlier of the internal point was randomly set within a certain distance 0.1 [m] from the outlier of the contact point.

We performed 50 simulations for each object shape. The parameters used in the simulations are listed in Table 6.1. The parameters used in the GPIS methods were set by trial and error. The test data were set on a grid with an interval of 0.02 [m].

6.2.2. Shape estimation error

One of the 50 simulations is shown in Fig. 6.3. As shown in the figure, the object shapes could be roughly captured using the GPIS. At the same time, however, surfaces were incorrectly estimated around a few of the outliers, located sufficiently far from the object. The predicted shape potential values (colored contours) show that the mean values became negative around the outliers, which



(a) Square (b) Circle (c) Cross

Figure 6.1. 2D object models used in simulation experiments. All figures are in units of m.

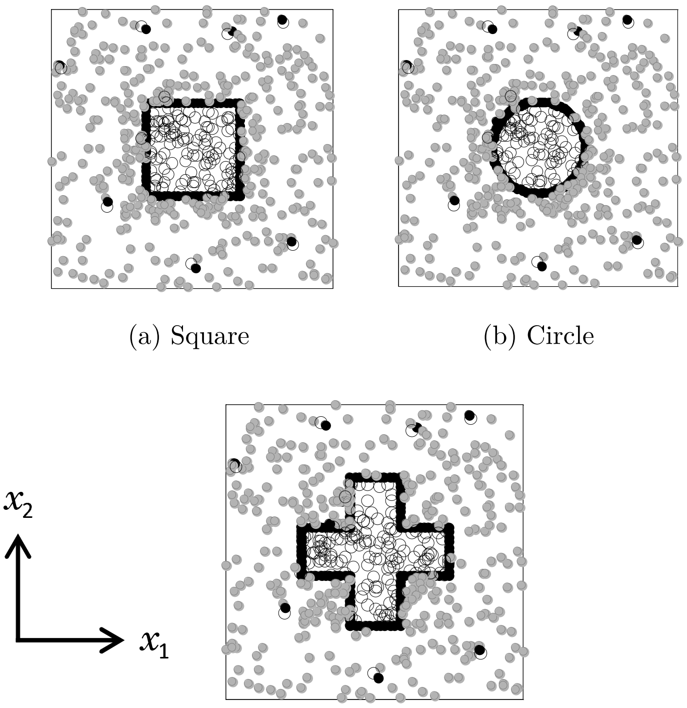
Table 6.1. Simulation conditions

Variable	Symbol	Value
Number of observed points	n	5.9×10^2
Number of test points	n_t	9.1×10^4
Initial value of variance of Gaussian kernel	-	0.25^2
Initial value of shape parameter	α	2.0
Initial value of inverse scale	β	4.0

indicates that the predictive distributions were greatly influenced by the outliers.

In contrast, a surface was not incorrectly estimated around the outliers when using the robust GPIS, as shown in Fig. 6.3. Considering the uncertainty values, those values around the outliers became remarkably large compared with those around the normal data. As a result, the predicted shape potential values around the outliers were sufficiently greater than zero, indicating that the outliers had little influence on the predicted mean values.

To evaluate the accuracy of shape estimation using the GPIS methods, we introduce metrics for determining the shape estimation error (i.e., the difference between the estimated shape and the true shape). Using Eqs. (6.2) and (6.9), we selected the test point $i = \{1, \cdots, n_s\}$ whose mean value $\mu(\boldsymbol{x}_i)$ is almost equal to zero ($-0.01 < \mu(\boldsymbol{x}_i) < 0.01$), where $\boldsymbol{x}_i$ is the position of the test point i. After computing the minimum distance d_i between the test point i and the surface of



(a) Square (b) Circle

(c) Cross

Figure 6.2. Data acquired in one of the simulations. The circles, black dots, and gray dots represent internal points, contact points, and external points, respectively.

the true shape, we computed the mean error of the distance d_i, defined by

$$e = \frac{1}{n_s} \sum_{i=1}^{n_s} d_i. \tag{6.12}$$

The results obtained with the t-test for the 50 simulations used to compare the robust GPIS with the GPIS are listed in Table 6.2. These results reveal significant differences between the robust GPIS and the GPIS in terms of the shape estimation errors, since the p-values are all less than 0.01. Therefore, for all the prepared objects, our algorithm produced smaller shape errors than the GPIS.

Table 6.2. Results of t-tests comparing the GPIS and the robust GPIS with respect to shape estimation error. The degree of freedom is 98 in all cases. (**: p<0.01)

Object shape	p-value
Square	7.7E-11**
Circle	7.7E-9**
Cross	1.2E-9**

6.2.3. False-positive contact detection

This subsection shows the clear detection of false-positive contact data when applying our algorithm. With the robust GPIS, the uncertainty of the data around the outliers became remarkably large compared with those around the normal data, as shown in Fig. 6.3.

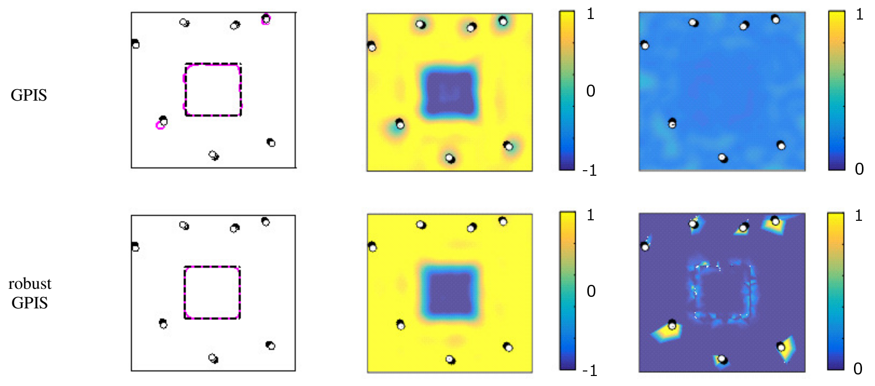
Here we introduce metrics for evaluating the false-positive contact detection. While the robust GPIS can estimate the uncertainty of data, the GPIS cannot directly do so. Therefore, we regard the predicted variance at each observed data as the uncertainty of the data. To define the clarity of the false-positive contact detection for each outlier o, we define the metrics given by

$$Q_o = \frac{1}{n_o} \sum_{j \in \mathcal{R}_o} \frac{u_j}{u_o}, \tag{6.13}$$

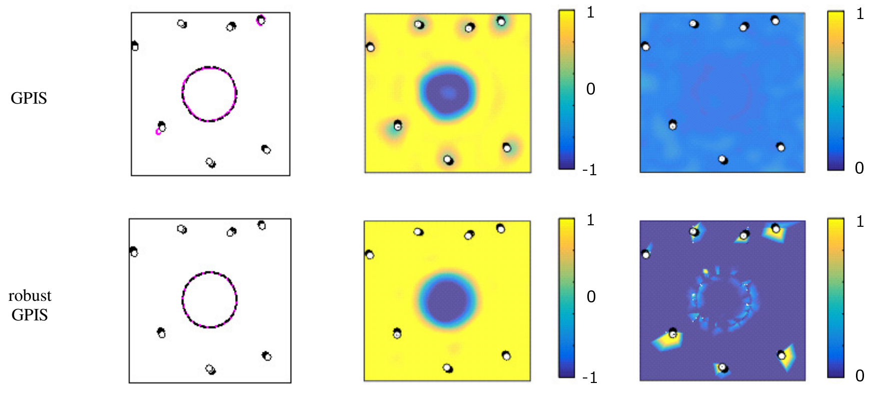
where u_o, $\mathcal{R}_o$ and n_o represent the uncertainty of the data o, the neighborhood region within certain distance d_o from the outlier o, and the number of the normal data within the region $\mathcal{R}_o$, respectively. In the simulations, we set $d_o = 0.50$ [m], which is equal to the length and width of the quadcopter used in the experiment.

Similar to the evaluation of the shape estimation error, we performed a t-test for the 50 simulations by comparing the robust GPIS with the GPIS. The results listed in Table 6.3 reveal significant differences between the robust GPIS and the GPIS in terms of false-positive contact detection, since the p-values are all less than 0.01.

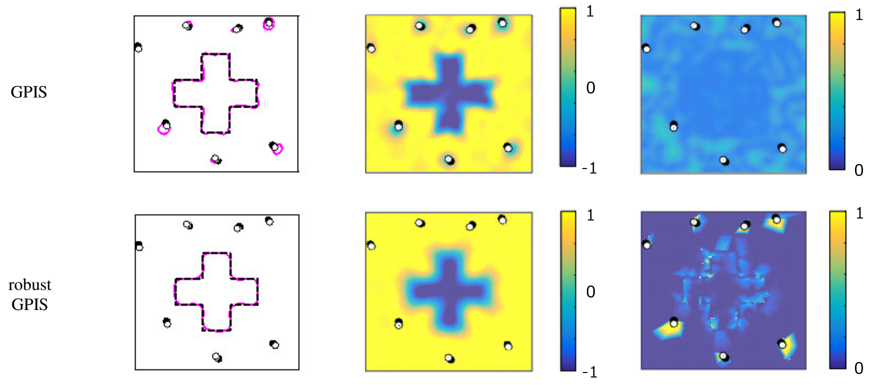
Therefore, for all the prepared objects, our algorithm could detect false-positive contacts more clearly than the GPIS.



(a) Square



(b) Circle



(c) Cross

Figure 6.3. Estimated shapes of 2D objects (left), predicted mean (center), and uncertainty of data (right) when applying the GPIS methods. The dashed lines, black dots, and circles represent the real object shapes, outliers of contact points, and outliers of internal points, respectively.

Table 6.3. Results of t-tests comparing the GPIS and the robust GPIS with respect to false-positive contact detection. ($**$: $p<0.01$)

Object shape	p-value
Square	1.4E-104**
Circle	2.5E-111**
Cross	8.3E-121**

6.3. Experiment

In this section, we will demonstrate the use of the algorithm through experiments using a quadcopter, specifically, a Parrot AR.Drone 2.0. We introduce the experimental configuration and conditions, as well as the omni-directional contact detection using the accelerometers implemented in the experiment. We then present the experimental results.

6.3.1. Experimental configuration and conditions

The position of the quadcopter was observed using an OptiTrack Prime 17W motion-capture system (Natural Point, Inc., Corvallis, OR) operating at 200 Hz. In the experiment, the ground-truth shape of the 3D object is obtained by placing markers at the vertexes of the polygons that compose the object surfaces. Moreover, the object shape is considered in the ground-fixed coordinate system. Therefore, translation and rotation were not considered when comparing the estimated shape with the true object shape.

The acceleration and attitude were measured using mounted sensors operating at 200 Hz. The quadcopter was controlled using a personal computer (PC) with an 8-core Intel Core i7 (2.80 GHz) processor, 32 GB of RAM, and a joystick (Extreme 3D Pro, Logitech, Inc., Lausanne, Switzerland). The measurement data and control signals were exchanged between the PC and the quadcopter via WiFi communication.

The exploration region Q is given by

$$\boldsymbol{x} \in Q := \{\boldsymbol{x} \mid -4 \leq x_1 \leq 4, -2 \leq x_2 \leq 2, 0 \leq x_3 \leq 2\},$$

Table 6.4. Experimental conditions

Variable	Symbol	Value
Threshold of contact detection [G]	-	0.6
Thickness parameter [m]	d_{in}	0.1
Number of observed points	n	2.7×10^3
Number of test points	n_t	5.4×10^5
Initial value of variance of Gaussian kernel	-	0.25^2
Initial value of shape parameter	α	2.0
Initial value of inverse scale	β	4.0

where x_1, x_2, and x_3 represent the longitudinal, transverse, and vertical axes, respectively.

We controlled the quadcopter using the joystick, which allowed us to collect as much contact data as possible in Q at 25 Hz while collecting non-contact data as uniformly as possible at 0.5 Hz. The limitations imposed by the battery capacity forced us, for each experiment, to perform five flights of 8 min each.

To validate the effectiveness of our algorithm, we generated some ground truth outliers by applying a gust of wind in some regions. We conducted two flights with artificial pertubations and prepared two observed data sets combined with the other four flights without artificial pertubations. The initial values for the hyperparameters used with the GPIS methods are listed in Table 6.4.

6.3.2. Omni-directional contact detection using accelerometers

This subsection introduces omni-directional contact detection based on the acceleration of the quadcopter. Based on the measured acceleration, the shape potential value is obtained using

$$y_i = \begin{cases} 0, \text{if } \ \|{}^{\mathrm{B}}\hat{\boldsymbol{a}}_i\| > a_0 \\ \\ 1, \text{otherwise} \end{cases},$$

where ${}^{\mathrm{B}}\hat{\boldsymbol{a}}_i$ and a_0 represent the acceleration of the robot in the body-fixed coordinate system B and the positive threshold to detect the transition state from non-contact to contact, respectively.

The flowchart of the contact detection is shown in Fig. 6.4. If y_i is equal to zero and its true value y_i^* is equal to one, the data i is the false-positive contact data. If y_i is equal to one and y_i^* is equal to zero, the data i is the false-negative contact data. The occurrence of these data depend on the the threshold a_0. If a_0 is set too low, false-positive contact detection can occur easily since sudden acceleration and deceleration can be detected as false-positive contacts. In the opposite case, false-negative contact detection can occur. In this study, we focus on the false-positive contacts since the false-negative contacts have much less influence on the false-positive contacts. We searched for the minimum value of a_0 by trial and error, considering that the number of false-positive contacts is as small as possible.

Furthermore, the outward normal vector on the surface is estimated by

$$\hat{\boldsymbol{n}}_i = \begin{cases} \frac{{}^{\mathrm{E}}\hat{\boldsymbol{a}}_i}{\|{}^{\mathrm{E}}\hat{\boldsymbol{a}}_i\|}, \text{if} \quad y_i = 0 \\ \\ \boldsymbol{0}, \text{otherwise} \end{cases}, \tag{6.14}$$

where ${}^{\mathrm{E}}\hat{\boldsymbol{a}}_i$ represents the acceleration of the robot in the ground fixed coordinate system E. Acceleration with mounted sensors is observed in the body-fixed coordinate system B, such that ${}^{\mathrm{E}}\hat{\boldsymbol{a}}$ in Eq. (6.14) is converted from ${}^{\mathrm{E}}\hat{\boldsymbol{a}}_i = \boldsymbol{R}(\hat{\phi})\boldsymbol{R}(\hat{\theta})\boldsymbol{R}(\hat{\psi})^{\mathrm{B}}\hat{\boldsymbol{a}}_i$, where $\boldsymbol{R}$, $\hat{\phi}$, $\hat{\theta}$, and $\hat{\psi}$ represent the rotation matrix, and the measured values of the roll, pitch, and yaw angle, respectively.

Using the center of gravity of the robot and the estimated normal vector, we approximated a contact position $\boldsymbol{x}_c$, as shown in Fig. 6.5. Consider a circumscribed cylinder to the quadcopter with the radius r and the height h. If one of the planar components of the estimated vector $\hat{\boldsymbol{n}}_i$ is largest, the contact position is calculated by $\boldsymbol{x}_{c_i} = \boldsymbol{x}_i - r\hat{\boldsymbol{n}}_i$. Otherwise, the contact position is calculated by $\boldsymbol{x}_{c_i} = \boldsymbol{x}_i - h\hat{\boldsymbol{n}}_i$.

Because the internal points of the object cannot be obtained directly from

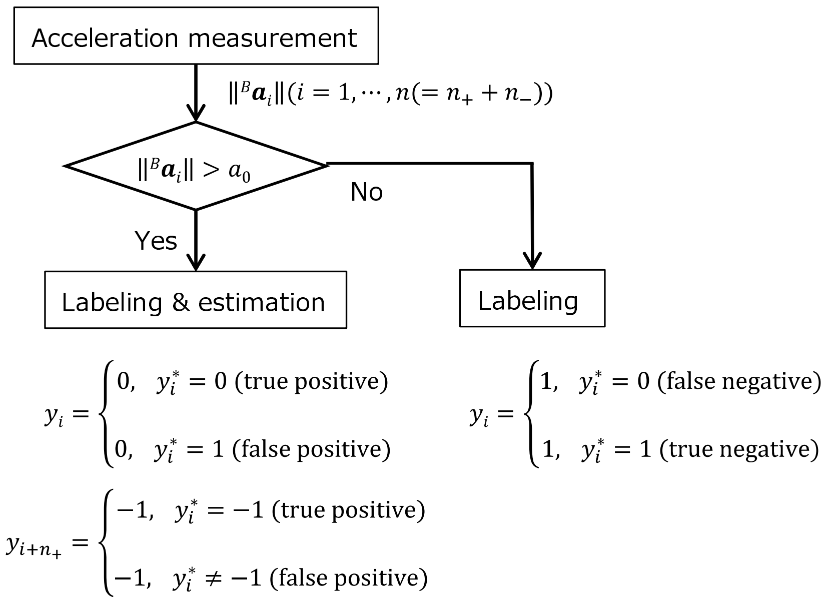

Acceleration measurement
$\|{}^{B}\boldsymbol{a}_i\| (i = 1, \cdots, n(= n_+ + n_-))$
$\|{}^{B}\boldsymbol{a}_i\| > a_0$
No
Yes
Labeling & estimation
Labeling
$y_i = \begin{cases} 0, & y_i^* = 0 \text{ (true positive)} \\ 0, & y_i^* = 1 \text{ (false positive)} \end{cases}$
$y_i = \begin{cases} 1, & y_i^* = 0 \text{ (false negative)} \\ 1, & y_i^* = 1 \text{ (true negative)} \end{cases}$
$y_{i+n_+} = \begin{cases} -1, & y_i^* = -1 \text{ (true positive)} \\ -1, & y_i^* \neq -1 \text{ (false positive)} \end{cases}$

Figure 6.4. Flowchart of omni-directional contact detection. n_+ and n_- represent the number of contact and external points and the number of internal points, respectively.

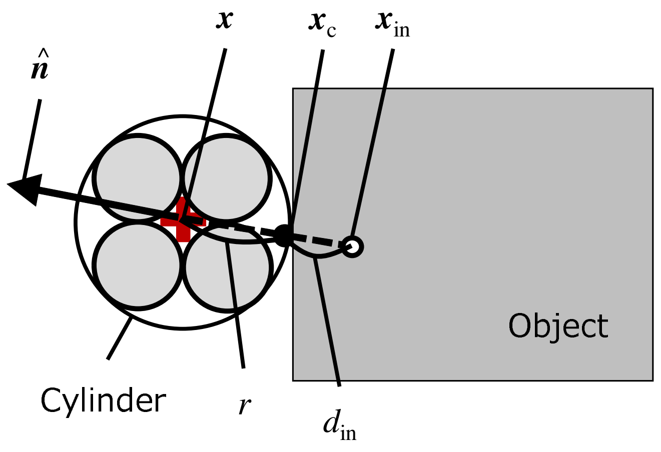


Figure 6.5. Estimation of contact and internal points. When the robot collides with the object, a contact point (colored dot) is estimated using the center of gravity (cross), the estimated vector (arrow) and the circumscribed cylinder to the robot. The internal point (circle) is estimated on the extension line through the center of gravity of the robot and the contact point.

the contact, we estimate the position of the internal point $\boldsymbol{x}_{\text{in}}$, as shown in Fig. 6.5. Only if $y(\boldsymbol{x}) = 0$, the position of the internal point and its potential value are calculated as $\boldsymbol{x}_{\text{in}} = \boldsymbol{x}_{\text{c}} - d_{\text{in}}\hat{\boldsymbol{n}}(\boldsymbol{x})$ and $y(\boldsymbol{x}_{\text{in}}) = -1$, where d_{in} represents the thickness parameter determined by the minimum thickness of the object. The training data was produced by merging the contact, external, and internal points while the test data was set on a grid with an interval of 0.05 [m].

6.3.3. Shape estimation error

The results of shape estimation when applying the GPIS methods are shown in Fig. 6.6.

As shown in Fig. 6.6a, the shape of the 3D construction could be roughly captured when applying GPIS. However, surfaces were generated at points sufficiently far from the construction. The predicted mean values of the shape potential function were all zero around the outliers. In contrast, the robust GPIS was able to avoid the incorrect generation of surfaces around the outliers, as shown in Fig. 6.6b. This result indicates that the predicted mean values of the shape potential function were sufficiently greater than zero around the outliers.

Furthermore, the shape estimation errors computed by Eq. (6.12) when applying the GPIS methods are listed in Table 6.5. The results indicate that the use of the robust GPIS incurred smaller errors than the GPIS.

6.3.4. False-positive contact detection

This subsection focuses on the results of false-positive contact detection when applying the GPIS methods.

The distribution of the uncertainty of data, shown in Fig. 6.6a, indicates that the uncertainty around the outliers was as small as that around the normal data with the application of GPIS. In contrast, we can confirm that the uncertainty around the outliers in Fig. 6.6b could be distinguished very clearly when our algorithm was applied.

To evaluate the false-positive contact detection of the GPIS methods, we evaluated false-positive contact detection for each outlier using Eq. (6.13). In the experiment, we regarded any detected contact points which were 0.5 meters or

Table 6.5. Mean error of the estimated shapes

Algorithm	GPIS	Robust GPIS
Experiment#1	0.10 m	**0.07** m
Experiment#2	0.12 m	**0.07** m

Table 6.6. False-positive contact detection

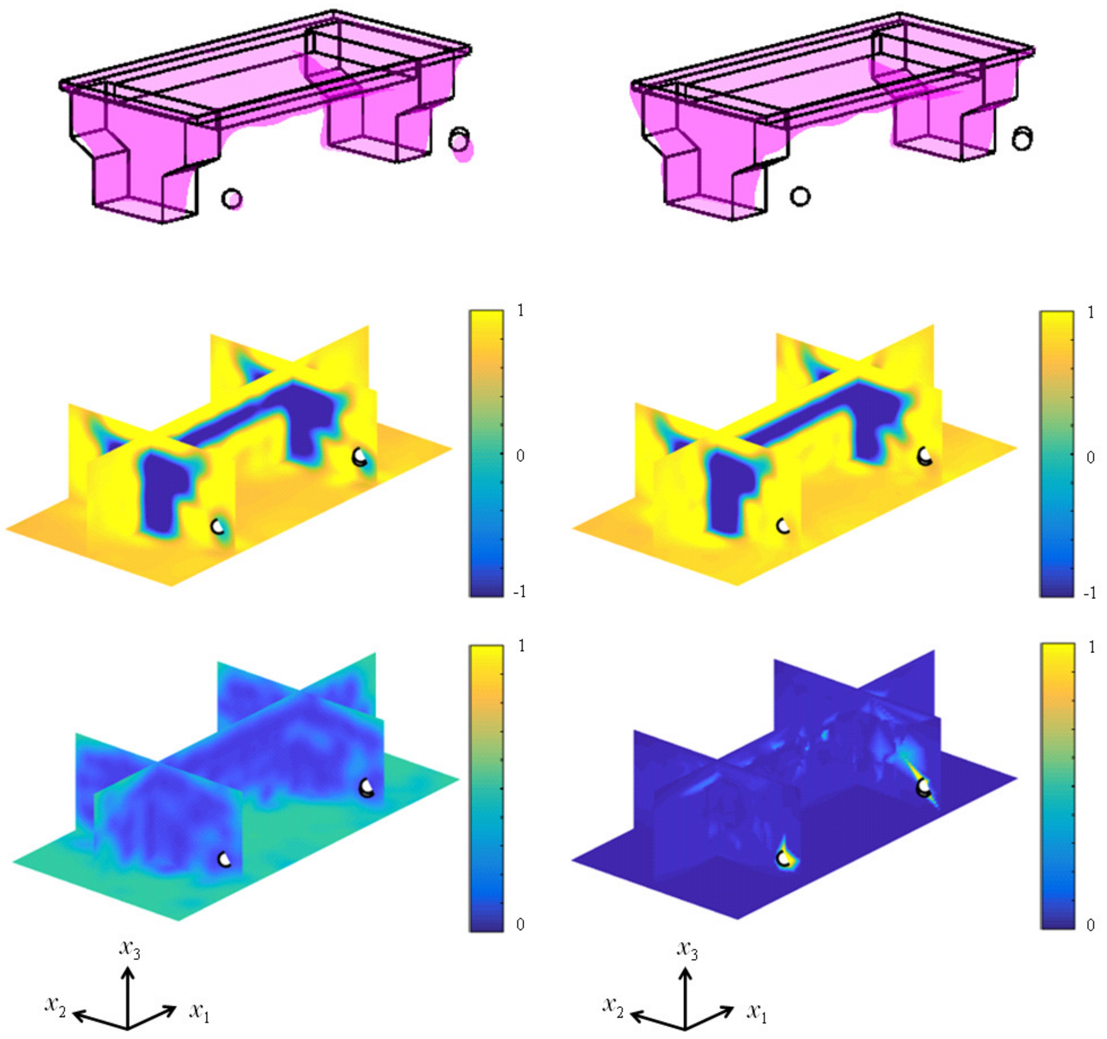
Algorithm	GPIS	Robust GPIS
Mean values	0.96	**0.045**

more from the object as being outliers. This value was equivalent to the length and width of the quadcopter.

The mean values for eight outliers in the two experiments are listed in Table 6.6. The results show that the mean value obtained when applying the GPIS were almost equal to one, which indicates that the GPIS could not distinguish between the outliers and the normal data. In contrast, the mean values were much smaller than one, which indicates that our algorithm could very clearly distinguish the outliers from the normal data.

6.4. Summary of Chapter 6

In this chapter, we have proposed an approach for object shape estimation based on touch with omni-directional contact detection using accelerometers. Because our contact detection method tends to induce a degree of uncertainty due to the presence of false-positive contact data, we proposed a robust shape estimation method capable of handling such false-positive contact data. We confirmed that our algorithm could reduce shape estimation errors caused by false-positive contact data through simulations and actual experiments using a quadcopter. Moreover, our algorithm could distinguish false-positive contact data more clearly than the GPIS, suggesting that false-positive contact detection which is possible with our proposed algorithm can be useful for other applications, such as active object exploration.



(a) GPIS (b) robustGPIS

Figure 6.6. Estimated shapes (top), predicted mean (middle), and uncertainty of data (bottom) when applying the GPIS methods to a 3D construction. The circles and bold lines represent the false-positive contact data and the true object shape, respectively. The cross-sections are placed in order to visualize the distribution of each value.

7. Discussions

7.1. Open issues and future scope for distributed cooperative transport

In this section, we describe the open issues and scope of future work for distributed cooperative transport.

Through simulations, it has been confirmed that the proposed method can tolerate a certain degree of fluctuations in the number of agents. However, there is room for improvement when increasing the number of agents. Because our method cannot theoretically guarantee the connectivity of communication topology, the connectivity may decrease when the number of agents increases. To address this issue, we will combine the proposed method with a learning method to maximize the connectivity achieved in a previous study [87]. Furthermore, the convergence speed decreases as the number of agents increases [88], which can degrade control performance. Therefore, it is crucial to consider a framework that guarantees a certain level of convergence speed.

In the present study, the trained multi-agent policies performed worse in real experiments because the physical parameters of the simulations are different from those of the experiments. Moreover, we should consider the uncertainties of the observations, including the position and yaw angle of the payload and robots, which were not considered in this study. To address these problems, we should make multi-agent policies more robust to these uncertainties using domain randomization [89]. Further, the trained multi-agent policies performed worse in real experiments as the number of robots increased, considering that our model neglected the collision avoidance between different robots. To address this issue, we plan on combining our algorithm with decentralized multi-robot collision avoidance presented in [90].

Reducing the sample complexity in the current system would be an important line of research. While we could optimize the multi-agent policies for a small number of robots, the computational costs could significantly increase as the number of robots increases. To address this issue, we will combine our algorithm with a more sample efficient algorithm such as a multi-agent model-based RL algorithm [91]

In the present study, the proposed framework requires to determine what to establish consensus for each task, it cannot be applied to various cooperative transport tasks. Therefore, it is crucial to propose a learning framework that can determine what to establish consensus in general setups.

It would be interesting to apply the proposed method to a three-dimensional cooperative transport task using multiple quadcopters. To achieve such a system, we need to consider two issues. The first issue is to estimate the torque applied to the object. To calculate the three-dimensional resultant torque, it is necessary to know the object's moment of inertia. To address this issue, we will combine the proposed method with the moment of inertia estimation technique proposed in [18]. The second issue is to consider the convergence speed required for estimation. Because the convergence speed is determined by the eigenvalues of the graph Laplacian defined by the robot's communication topology [92], it does not depend on the dimension. However, the convergence speed may decrease as the number of robots increases. Therefore, it is crucial to consider a learning framework that guarantees convergence speed in order to extend to 3D cooperative transport tasks using large-scale multi-robot systems.

7.2. Open issues and future scope for multi-robot task allocation

In this section, we describe the open issues and scope of future work for multi-robot task allocation.

Because our learning framework requires global communication between robots, the communication bandwidth can be compressed as the number of robots increases. To address this issue, we should decentralize the communication structure using techniques such as an attentional communication channel [90].

In the present study, we assume that each robot knows the positions of all the objects. If there exist some objects which are not observed by robots, the robots cannot update the dynamic task priorities of those objects, which can degrade the transport performance. To address this issue, we may combine the consensus of objects' positions with our framework and confirm its effectiveness under partial observations with several unknown object positions.

Herein, we discuss the effectiveness of our policy model compared to other policy models. We can apply the centralized policy model which selects an object for all the robots. However, because the centralized policy model depends on the number of objects and robots, it requires to learn the policy from scratch when varying the number of objects and robots. We can employ the same distributed policy model for all robots, using a broadcast signal including all robots' observations. However, this policy model also depends on the number of objects and robots. In contrast, our policy model adopts the distributed policy model that limits the minimum number of environmental objects and robots to a fixed number of objects and robots. Therefore, it can be applied to a varying number of objects and robots.

It would be interesting to apply our algorithm to task allocation of heterogeneous robots. While our framework can be applied to the task allocation for given robots, it cannot determine the configuration and number of heterogeneous robots. Therefore, it is crucial to consider a framework that can derive teaming and individual policies at the same time.

7.3. Open issues and future scope for object shape estimation

In this section, we describe the open issues and scope of future work for object shape estimation.

The current experimental system has room for improvement when estimating more complex object shapes. In a more complex scenario such as some concave formations, multiple contacts might occur at different locations on the robot. In such a situation, contacts are detected around the inside part of the corners because the current system estimates only one contact position at any one time using the resultant force of multiple contacts. As a result, the estimated shape around the corner tends to be rounded. To overcome this issue, more precise contact localization should be implemented using the methods described in [93, 94]. In addition, we should extend our contact detection to handle multiple contacts, as proposed by Sommer et al. [95].

In the present study, we assumed that the position of the robot is accurate. However, in real robot environments, it is necessary to consider the uncertainty of localization in the shape estimation. To address this issue, we aim to eliminate the assumption by combining the GP model with the input noise proposed by Girard et al. [96].

Finally, we discuss the usefulness of the false-positive contact detection. Let us consider active object exploration [97–100] with false-positive contact data. In this problem, the outliers might degrade any prediction, resulting in inefficient exploration since robots might excessively explore some areas around the incorrect surfaces. Moreover, false-negative contacts could degrade the efficiency of the object exploration although they were not considered in the present study. If contacts were not detected at the surface, the GPIS methods could predict that no surface exists around the false-negative contacts. As a result, those locations were difficult to sample when applying Bayesian optimization [101]. To deal with this issue, we can remove the outliers based on our false-positive contact detection. To apply our algorithm to active object exploration, we should reduce the amount of observed data using a sparse Gaussian process regression methods as described in [102, 103].

7.4. Open issues and future scope for cooperative transport of unknown-shaped object

Our current system requires few modifications to transport unknown shaped objects. Firstly, we should incorporate shape estimation into the control loop of multi-robot cooperative transport. In the current system, the robot computes the force and torque using a normal vector to the object surface according to the

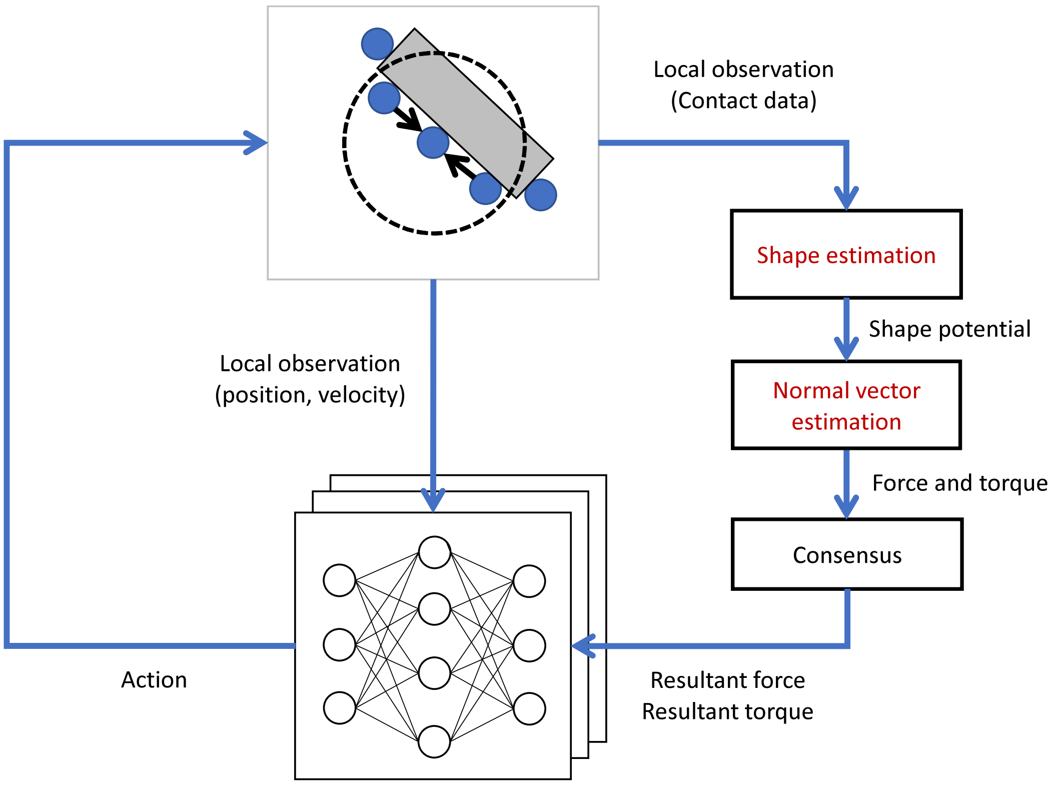


Figure 7.1. Control loop for cooperative transport of unknown-shaped object

object shape model. Therefore, we are required to replace the shape model with the shape estimation, as shown in Fig. 7.1. Moreover, we should combine our method with a method of estimating the normal vector from the predicted shape potential function [50].

8. Conclusion

In this thesis, we proposed learning approaches for flexible and resilient multi-robot cooperative transport.

First, we proposed a MARL framework for multi-robot cooperative transport with various numbers of robots. The proposed method exploits a distributed policy to reconstruct global information using consensus with the local communication robots while determining the timing for communication. Thus, the proposed method could balance communication savings and control performance in scenarios wherein the number of robots differed from that in the training environment through simulations and experiments.

Secondly, we proposed a MARL framework that can handle scenarios for various numbers of robots and objects with different and unknown weights. The proposed method exploits a distributed policy that determines the timing for cooperative and independent actions. Therefore, our framework can reduce the transport time while transporting all the objects to their desired positions for various numbers of robots and objects with different and unknown weights through simulations.

Finally, we proposed an approach for shape estimation to transport unknown-shaped objects. The proposed method adopts touch-based contact detection using accelerometers to supplement vision sensors. However, this method involves false-positive contact data because it reacts not only to actual contacts but also to the unstable behavior of the robot. Therefore, we proposed a robust shape estimation method capable of handling such false-positive contact data. We confirmed that our algorithm could reduce shape estimation errors caused by false-positive contact data through simulations and actual experiments using a quadcopter.

In summary, this thesis is expected to play an essential role in achieving a flexible and resilient system that can deal with various transportation scenarios in real applications.

A. Appendix: Student's t- and inverse gamma distribution

This appendix introduces how the parameters α and β affect the Student's t- and inverse gamma distributions in Eq. (6.6). The Student's t-distribution for different combinations of α and β is shown in Fig. A.1. Based on the results, the Student's t-distribution gets heavy-tailed when α becomes small or β becomes large.

The inverse gamma distribution is given by

$$\mathrm{Inv}\Gamma(\sigma^2 \mid \alpha, \beta) = \frac{\beta^\alpha}{\Gamma(\alpha)}(\sigma^2)^{-(1+\alpha)} \exp\left(-\frac{\beta}{\sigma^2}\right).$$

The inverse gamma distribution for different combinations of α and β values is shown in Fig. A.2. From the result, the noise variance, σ^2, becomes large with high probability as α becomes small or as β becomes large.

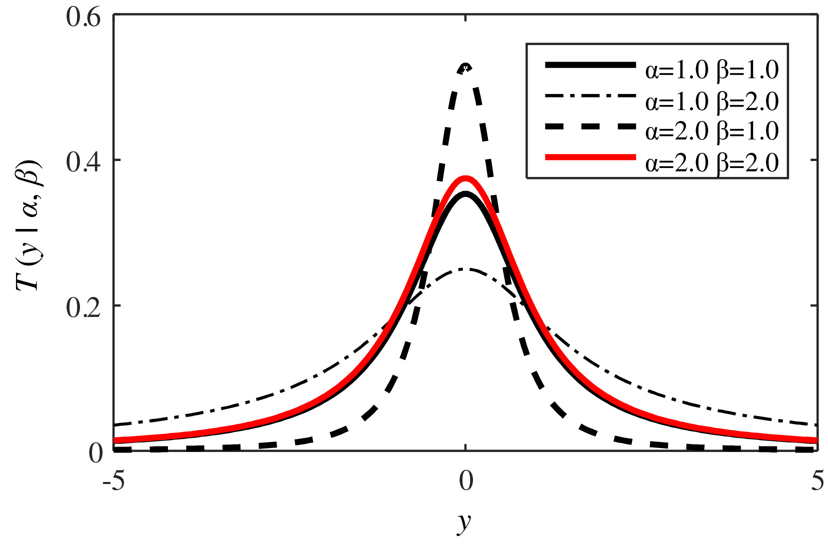


Figure A.1. Student's t-distribution

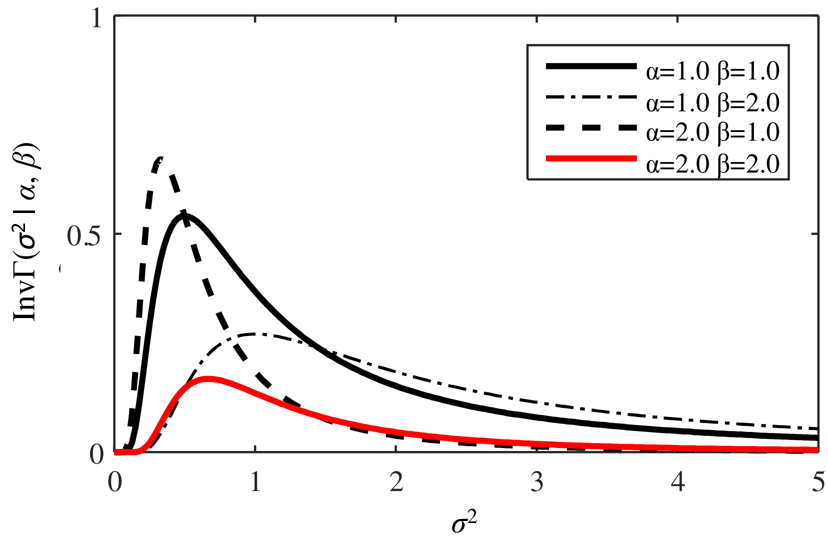


Figure A.2. Inverse gamma distribution

B. Appendix: Connectivity metrics

This appendix introduces the connectivity metrics of communication topology.

At every control step k, we check the satisfaction of the connectivity given by

$$\beta(k) = \begin{cases} 1, & \text{if Eq. (3.1) is satisfied} \\ 0, & \text{otherwise} \end{cases}.$$

Moreover, we evaluate the ratio of the connectivity given by

$$R_c = \frac{1}{N} \sum_{i=1}^{N} \frac{\sum_{k=1}^{T} \beta(k)}{\sum_{k=1}^{T} b_i(k)},$$

where $b_i(k) = 1$ if agent i communicates with other agents at control step k; otherwise $b_i(k) = 0$.

Acknowledgements

In my doctoral course at the Nara Institute of Science and Technology, I had a lot of invaluable experiences. I want to thank all the people who have supported me.

First, I would like to thank Professor Takamitsu Matsubara. He taught me a lot of learning techniques, academic writing, and presentation skills. I also appreciate him for discussing the direction when I was in trouble with my research. I appreciate Visiting Professor Kenji Sugimoto. He supported me as a supervisor in my 1st and 2nd year during my doctoral course. I also appreciate the invaluable comments and suggestions. I appreciate Professor Takahiro Wada reviewing this thesis. His comments on real robot applications gave me an invaluable opportunity to reconfirm the usefulness of my research. I appreciate Assistant Professor Kenta Hanada for reviewing this thesis. His comments on the control theory of multi-agent systems gave me an invaluable opportunity to improve my research. I appreciate Assistant Professor Yoshihisa Tsurumine for reviewing this thesis. His comments on reinforcement learning approaches gave me an invaluable opportunity to improve my research. I also thank him for arranging the presentation in lab meetings in consideration of my work.

I appreciate all members of the Robot Learning Lab. At the weekly lab meetings, I had a good opportunity to receive many invaluable questions and comments. Finally, I want to thank my family, friends, and company members.

References

[1] R. D'Andrea, "A revolution in the warehouse: A retrospective on Kiva systems and the grand challenges ahead," *IEEE Transactions on Automation Science and Engineering*, vol. 9, no. 4, pp. 638–639, 2012.

[2] E. Cardarelli, V. Digani, L. Sabattini, C. Secchi, and C. Fantuzzi, "Cooperative cloud robotics architecture for the coordination of multi-AGV systems in industrial warehouses," *Mechatronics*, vol. 45, pp. 1–13, 2017.

[3] J. P. Queralta, J. Taipalmaa, B. C. Pullinen, V. K. Sarker, T. N. Gia, H. Tenhunen, M. Gabbouj, J. Raitoharju, and T. Westerlund, "Collaborative multi-robot search and rescue: Planning, coordination, perception, and active vision," *IEEE Access*, vol. 8, pp. 191617–191643, 2020.

[4] J. Fink, N. Michael, and V. Kumar, "Composition of vector fields for multi-robot manipulation via caging," in *Robotics: Science and Systems*, vol. 3, 2007.

[5] J. Fink, M. A. Hsieh, and V. Kumar, "Multi-robot manipulation via caging in environments with obstacles," in *2008 IEEE International Conference on Robotics and Automation*, pp. 1471–1476, 2008.

[6] Z. Wang and M. Schwager, "Multi-robot manipulation with no communication using only local measurements," in *2015 54th IEEE Conference on Decision and Control (CDC)*, pp. 380–385, 2015.

[7] Z. Wang and M. Schwager, "Kinematic multi-robot manipulation with no communication using force feedback," in *2016 IEEE international conference on robotics and automation (ICRA)*, pp. 427–432, 2016.

[8] Z. Wang and M. Schwager, "Force-amplifying n-robot transport system (force-ants) for cooperative planar manipulation without communication," *The International Journal of Robotics Research*, vol. 35, no. 13, pp. 1564–1586, 2016.

[9] D. Sieber, S. Musić, and S. Hirche, "Multi-robot manipulation controlled by a human with haptic feedback," in *2015 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 2440–2446, 2015.

[10] M. Gienger, D. Ruiken, T. Bates, M. Regaieg, M. MeiBner, J. Kober, P. Seiwald, and A.-C. Hildebrandt, "Human-robot cooperative object manipulation with contact changes," in *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 1354–1360, 2018.

[11] N. Michael, J. Fink, and V. Kumar, "Cooperative manipulation and transportation with aerial robots," *Autonomous Robots*, vol. 30, no. 1, pp. 73–86, 2011.

[12] Q. Jiang and V. Kumar, "The inverse kinematics of cooperative transport with multiple aerial robots," *IEEE Transactions on Robotics*, vol. 29, no. 1, pp. 136–145, 2013.

[13] K. Sreenath and V. Kumar, "Dynamics, control and planning for cooperative manipulation of payloads suspended by cables from multiple quadrotor robots," in *Robotics: Science and Systems*, 2013.

[14] D. Mellinger, M. Shomin, N. Michael, and V. Kumar, "Cooperative grasping and transport using multiple quadrotors," in *Distributed autonomous robotic systems*, Springer, 2013.

[15] G. Loianno and V. Kumar, "Cooperative transportation using small quadrotors using monocular vision and inertial sensing," *IEEE Robotics and Automation Letters*, vol. 3, no. 2, pp. 680–687, 2018.

[16] X.-M. Zhang, Q.-L. Han, and X. Yu, "Survey on recent advances in networked control systems," *IEEE Transactions on industrial informatics*, vol. 12, no. 5, pp. 1740–1752, 2015.

[17] A. Franchi, A. Petitti, and A. Rizzo, "Distributed estimation of the inertial parameters of an unknown load via multi-robot manipulation," in *53rd IEEE Conference on Decision and Control*, pp. 6111–6116, 2014.

[18] A. Franchi, A. Petitti, and A. Rizzo, "Decentralized parameter estimation and observation for cooperative mobile manipulation of an unknown load using noisy measurements," in *2015 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 5517–5522, 2015.

[19] A. Petitti, A. Franchi, D. Di Paola, and A. Rizzo, "Decentralized motion control for cooperative manipulation with a team of networked mobile manipulators," in *2016 IEEE international conference on robotics and automation (ICRA)*, pp. 441–446, 2016.

[20] P. Culbertson and M. Schwager, "Decentralized adaptive control for collaborative manipulation," in *2018 IEEE international conference on robotics and automation (ICRA)*, pp. 278–285, 2018.

[21] R. Lowe, Y. I. Wu, A. Tamar, J. Harb, O. Pieter Abbeel, and I. Mordatch, "Multi-agent actor-critic for mixed cooperative-competitive environments," in *Advances in neural information processing systems*, vol. 30, 2017.

[22] J. Foerster, G. Farquhar, T. Afouras, N. Nardelli, and S. Whiteson, "Counterfactual multi-agent policy gradients," in *Proceedings of the AAAI conference on artificial intelligence*, vol. 32, 2018.

[23] D. Baumann, J.-J. Zhu, G. Martius, and S. Trimpe, "Deep reinforcement learning for event-triggered control," in *2018 IEEE Conference on Decision and Control (CDC)*, pp. 943–950, 2018.

[24] N. Funk, D. Baumann, V. Berenz, and S. Trimpe, "Learning event-triggered control from data through joint optimization," *IFAC Journal of Systems and Control*, vol. 16, 100144, 2021.

[25] K. Shibata, T. Jimbo, and T. Matsubara, "Deep reinforcement learning of event-triggered communication and control for multi-agent cooperative transport," in *2021 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 8671–8677, 2021.

[26] H. W. Kuhn, “The Hungarian method for the assignment problem,” *Naval research logistics quarterly*, vol. 2, no. 1-2, pp. 83–97, 1955.

[27] L. Liu and D. A. Shell, “Assessing optimal assignment under uncertainty: An interval-based algorithm,” *The International Journal of Robotics Research*, vol. 30, no. 7, pp. 936–953, 2011.

[28] L. Sabattini, V. Digani, C. Secchi, and C. Fantuzzi, “Optimized simultaneous conflict-free task assignment and path planning for multi-AGV systems,” in *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 1083–1088, IEEE, 2017.

[29] E. F. Flushing, L. M. Gambardella, and G. A. Di Caro, “Simultaneous task allocation, data routing, and transmission scheduling in mobile multi-robot teams,” in *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 1861–1868, IEEE, 2017.

[30] A. Ahmed, A. Patel, T. Brown, M. Ham, M.-W. Jang, and G. Agha, “Task assignment for a physical agent team via a dynamic forward/reverse auction mechanism,” in *International Conference on Integration of Knowledge Intensive Multi-Agent Systems, 2005.*, pp. 311–317, IEEE, 2005.

[31] M. Ham and G. Agha, “Market-based coordination strategies for large-scale multi-agent systems,” *System and Information Sciences Notes*, vol. 2, no. 1, pp. 126–131, 2007.

[32] M. Braquet and E. Bakolas, “Greedy decentralized auction-based task allocation for multi-agent systems,” *IFAC-PapersOnLine*, vol. 54, no. 20, pp. 675–680, 2021.

[33] H. Qie, D. Shi, T. Shen, X. Xu, Y. Li, and L. Wang, “Joint optimization of multi-UAV target assignment and path planning based on multi-agent reinforcement learning,” *IEEE Access*, vol. 7, pp. 146264–146272, 2019.

[34] T. Niwa, K. Shibata, and T. Jimbo, “Multi-agent reinforcement learning and individuality analysis for cooperative transportation with obstacle removal,” in *International Symposium Distributed Autonomous Robotic Systems*, pp. 202–213, 2021.

[35] C. D. Hsu, H. Jeong, G. J. Pappas, and P. Chaudhari, "Scalable reinforcement learning policies for multi-agent control," in *2021 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 4785–4791, 2021.

[36] J. Ilonen, J. Bohg, and V. Kyrki, "Fusing visual and tactile sensing for 3-D object reconstruction while grasping," in *2013 IEEE International Conference on Robotics and Automation*, pp. 3547–3554, 2013.

[37] S. Luo, W. Mou, K. Althoefer, and H. Liu, "Localizing the object contact through matching tactile features with visual map," in *2015 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 3903–3908, 2015.

[38] M. Björkman, Y. Bekiroglu, V. Högman, and D. Kragic, "Enhancing visual perception of shape through tactile glances," in *2013 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 3180–3186, 2013.

[39] H. P. Saal, J.-A. Ting, and S. Vijayakumar, "Active estimation of object dynamics parameters with tactile sensors," in *2010 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 916–921, 2010.

[40] M. Li, Y. Bekiroglu, D. Kragic, and A. Billard, "Learning of grasp adaptation through experience and tactile sensing," in *2014 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 3339–3346, 2014.

[41] M. Li, K. Hang, D. Kragic, and A. Billard, "Dexterous grasping under shape uncertainty," *Robotics and Autonomous Systems*, vol. 75, pp. 352–364, 2016.

[42] J. Sanchez, C. M. Mateo, J. A. Corrales, B.-C. Bouzgarrou, and Y. Mezouar, "Online shape estimation based on tactile sensing and deformation modeling for robot manipulation," in *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 504–511, 2018.

[43] H. van Hoof, T. Hermans, G. Neumann, and J. Peters, "Learning robot in-hand manipulation with tactile features," in *2015 IEEE-RAS 15th International Conference on Humanoid Robots (Humanoids)*, pp. 121–127, 2015.

[44] S. Chitta, J. Sturm, M. Piccoli, and W. Burgard, “Tactile sensing for mobile manipulation,” *IEEE Transactions on Robotics*, vol. 27, no. 3, pp. 558–568, 2011.

[45] R. Olfati-Saber and R. M. Murray, “Consensus problems in networks of agents with switching topology and time-delays,” *IEEE Transactions on Automatic Control*, vol. 49, no. 9, pp. 1520–1533, 2004.

[46] W. Ren and R. Beard, “Consensus seeking in multiagent systems under dynamically changing interaction topologies,” *IEEE Transactions on Automatic Control*, vol. 50, no. 5, pp. 655–661, 2005.

[47] O. Williams and A. Fitzgibbon, “Gaussian process implicit surfaces,” in *Gaussian Processes in Practice*, pp. 121–127, 2006.

[48] S. Caccamo, Y. Bekiroglu, C. H. Ek, and D. Kragic, “Active exploration using Gaussian random fields and Gaussian process implicit surfaces,” in *2016 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 582–589, 2016.

[49] J. Mahler, S. Patil, B. Kehoe, J. van den Berg, M. Ciocarlie, P. Abbeel, and K. Goldberg, “GP-GPIS-OPT: Grasp planning with shape uncertainty using Gaussian process implicit surfaces and sequential convex programming,” in *2015 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 4919–4926, 2015.

[50] S. Dragiev, M. Toussaint, and M. Gienger, “Gaussian process implicit surfaces for shape estimation and grasping,” in *2011 IEEE International Conference on Robotics and Automation*, pp. 2845–2850, 2011.

[51] M. P. Gerardo-Castro, T. Peynot, F. Ramos, and R. Fitch, “Robust multiple-sensing-modality data fusion using Gaussian process implicit surfaces,” in *17th International Conference on Information Fusion (FUSION)*, pp. 1–8, 2014.

[52] A. Marino and F. Pierri, “A two stage approach for distributed cooperative manipulation of an unknown object without explicit communication and

unknown number of robots," *Robotics and Autonomous Systems*, vol. 103, pp. 122–133, 2018.

[53] D. V. Dimarogonas, E. Frazzoli, and K. H. Johansson, "Distributed event-triggered control for multi-agent systems," *IEEE Transactions on Automatic Control*, vol. 57, no. 5, pp. 1291–1297, 2011.

[54] W. P. Heemels, K. H. Johansson, and P. Tabuada, "An introduction to event-triggered and self-triggered control," in *2012 IEEE 51st IEEE Conference on Decision and Control (CDC)*, pp. 3270–3285, IEEE, 2012.

[55] M. Miskowicz, *Event-based control and signal processing.* CRC press, 2018.

[56] S. Trimpe and R. D'Andrea, "An experimental demonstration of a distributed and event-based state estimation algorithm," *IFAC Proceedings Volumes*, vol. 44, no. 1, pp. 8811–8818, 2011.

[57] P. B. g. Dohmann and S. Hirche, "Distributed control for cooperative manipulation with event-triggered communication," *IEEE Transactions on Robotics*, vol. 36, no. 4, pp. 1038–1052, 2020.

[58] M. Rahimi, S. Gibb, Y. Shen, and H. M. La, "A comparison of various approaches to reinforcement learning algorithms for multi-robot box pushing," in *International Conference on Engineering Research and Applications*, pp. 16–30, 2018.

[59] K. Miyazaki, N. Matsunaga, and K. Murata, "Formation path learning for cooperative transportation of multiple robots using MADDPG," in *2021 21st International Conference on Control, Automation and Systems (ICCAS)*, pp. 1619–1623, 2021.

[60] P. Hernandez-Leal, B. Kartal, and M. E. Taylor, "A survey and critique of multi-agent deep reinforcement learning," *Autonomous Agents and Multi-Agent Systems*, vol. 33, no. 6, pp. 750–797, 2019.

[61] B. Demirel, A. Ramaswamy, D. E. Quevedo, and H. Karl, "Deepcas: A deep reinforcement learning algorithm for control-aware scheduling," *IEEE Control Systems Letters*, vol. 2, no. 4, pp. 737–742, 2018.

[62] G. Theraulaz, E. Bonabeau, and J. Denuebourg, "Response threshold reinforcements and division of labour in insect societies," *Proceedings of the Royal Society of London. Series B: Biological Sciences*, vol. 265, no. 1393, pp. 327–332, 1998.

[63] M. J. Krieger and J.-B. Billeter, "The call of duty: Self-organised task allocation in a population of up to twelve mobile robots," *Robotics and Autonomous Systems*, vol. 30, no. 1-2, pp. 65–84, 2000.

[64] B. P. Gerkey and M. J. Matarić, "A formal analysis and taxonomy of task allocation in multi-robot systems," *The International Journal of Robotics Research*, vol. 23, no. 9, pp. 939–954, 2004.

[65] M. B. Dias, R. Zlot, N. Kalra, and A. Stentz, "Market-based multirobot coordination: A survey and analysis," *Proceedings of the IEEE*, vol. 94, no. 7, pp. 1257–1270, 2006.

[66] A. M. Kwasnica, J. O. Ledyard, D. Porter, and C. DeMartini, "A new and improved design for multiobject iterative auctions," *Management science*, vol. 51, no. 3, pp. 419–434, 2005.

[67] H.-L. Choi, L. Brunet, and J. P. How, "Consensus-based decentralized auctions for robust task allocation," *IEEE transactions on robotics*, vol. 25, no. 4, pp. 912–926, 2009.

[68] Z. Wu, S. Song, A. Khosla, F. Yu, L. Zhang, X. Tang, and J. Xiao, "3D shapenets: A deep representation for volumetric shapes," in *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 1912–1920, 2015.

[69] A. Dai, C. R. Qi, and M. Nießner, "Shape completion using 3D-encoder-predictor CNNs and shape synthesis," in *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 6545–6554, 2017.

[70] J. Varley, C. DeChant, A. Richardson, J. Ruales, and P. Allen, "Shape completion enabled robotic grasping," in *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 2442–2447, 2017.

[71] D. Watkins-Valls, J. Varley, and P. Allen, “Multi-modal geometric learning for grasping and manipulation,” in *2019 International Conference on Robotics and Automation (ICRA)*, pp. 7339–7345, 2019.

[72] J. Lundell, F. Verdoja, and V. Kyrki, “Robust grasp planning over uncertain shape completions,” in *2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 1526–1532, 2019.

[73] M. Kuß, *Gaussian process models for robust regression, classification, and reinforcement learning.* PhD thesis, Technische Universität, 2006.

[74] P. Jylänki, J. P. Vanhatalo, and A. Vehtari, “Robust Gaussian process regression with a Student-t likelihood,” *J. Mach. Learn. Res.*, vol. 12, pp. 3227–3257, 2011.

[75] A. Mchutchon and C. Rasmussen, “Gaussian process training with input noise,” in *Advances in Neural Information Processing Systems*, vol. 24, 2011.

[76] M. Lázaro-Gredilla and M. K. Titsias, “Variational heteroscedastic Gaussian process regression,” in *Proceedings of the 28th International Conference on International Conference on Machine Learning*, pp. 841–848, 2011.

[77] R. Martinez-Cantin, M. J. McCourt, and K. Tee, “Robust Bayesian optimization with Student-t likelihood,” *arXiv preprint arXiv:1707.05729*, 2017.

[78] R. Olfati-Saber, J. A. Fax, and R. M. Murray, “Consensus and cooperation in networked multi-agent systems,” *Proceedings of the IEEE*, vol. 95, no. 1, pp. 215–233, 2007.

[79] N. Elhage and J. Beal, “Laplacian-based consensus on spatial computers,” in *Proceedings of the 9th International Conference on Autonomous Agents and Multiagent Systems: volume 1-Volume 1*, pp. 907–914, 2010.

[80] M. D. Kennedy III, L. Guerrero, and V. Kumar, “Decentralized algorithm for force distribution with applications to cooperative transport,” in *In-*

ternational Design Engineering Technical Conferences and Computers and Information in Engineering Conference, vol. 57144, V05CT08A013, 2015.

[81] R. Lowe, Y. I. Wu, A. Tamar, J. Harb, O. Pieter Abbeel, and I. Mordatch, "MADDPG." `https://github.com/openai/maddpg`, 2017. [Accessed: 3-Nov-2021].

[82] Z. Wang, G. Yang, X. Su, and M. Schwager, "Ouijabots: Omnidirectional robots for cooperative object transport with rotation control using no communication," in *Distributed Autonomous Robotic Systems*, pp. 117–131, Springer, 2018.

[83] C. J. Watkins and P. Dayan, "Q-learning," *Machine learning*, vol. 8, no. 3, pp. 279–292, 1992.

[84] C. E. Rasmussen and C. K. I. Williams, *Gaussian Processes for Machine Learning.* MIT Press, 2006.

[85] C. M. Bishop, *Pattern Recognition and Machine Learning.* Springer, 2007.

[86] A. P. Dempster, N. M. Laird, and D. B. Rubin, "Maximum likelihood from incomplete data via the EM algorithm," *JOURNAL OF THE ROYAL STATISTICAL SOCIETY, SERIES B*, vol. 39, no. 1, pp. 1–38, 1977.

[87] D. Mox, V. Kumar, and A. Ribeiro, "Learning connectivity-maximizing network configurations," *IEEE Robotics and Automation Letters*, vol. 7, no. 2, pp. 5552–5559, 2022.

[88] A. Olshevsky and J. N. Tsitsiklis, "Convergence speed in distributed consensus and averaging," *SIAM journal on control and optimization*, vol. 48, no. 1, pp. 33–55, 2009.

[89] J. Tobin, R. Fong, A. Ray, J. Schneider, W. Zaremba, and P. Abbeel, "Domain randomization for transferring deep neural networks from simulation to the real world," in *2017 IEEE/RSJ international conference on intelligent robots and systems (IROS)*, pp. 23–30, 2017.

[90] Y. Zhai, B. Ding, X. Liu, H. Jia, Y. Zhao, and J. Luo, "Decentralized multi-robot collision avoidance in complex scenarios with selective communication," *IEEE Robotics and Automation Letters*, vol. 6, no. 4, pp. 8379–8386, 2021.

[91] D. Willemsen, M. Coppola, and G. C. de Croon, "MAMBPO: Sample-efficient multi-robot reinforcement learning using learned world models," in *2021 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 5635–5640, IEEE, 2021.

[92] S.-i. Azuma, M. Nagahara, H. Ishii, N. Hayashi, K. Sakurama, and T. Hatanaka, "Control of multi-agent systems," *Corona Publishing CO., LTD*, 2015.

[93] W. McMahan, J. M. Romano, and K. J. Kuchenbecker, "Using accelerometers to localize tactile contact events on a robot arm," in *Proceedings of Workshop on Advances in Tactile Sensing and Touch-Based Human-Robot Interaction, ACM/IEEE International Conference on Human-Robot Interaction*, 2012.

[94] A. Molchanov, O. Kroemer, Z. Su, and G. S. Sukhatme, "Contact localization on grasped objects using tactile sensing," in *2016 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 216–222, 2016.

[95] N. Sommer and A. Billard, "Multi-contact haptic exploration and grasping with tactile sensors," *Robotics and Autonomous Systems*, vol. 85, pp. 48–61, 2016.

[96] A. Girard, C. Rasmussen, J. Q. Candela, and R. Murray-Smith, "Gaussian process priors with uncertain inputs application to multiple-step ahead time series forecasting," in *Advances in neural information processing systems*, vol. 15, 2002.

[97] Z. Yi, R. Calandra, F. Veiga, H. van Hoof, T. Hermans, Y. Zhang, and J. Peters, "Active tactile object exploration with Gaussian processes," in

2016 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), pp. 4925–4930, 2016.

[98] S. Dragiev, M. Toussaint, and M. Gienger, "Uncertainty aware grasping and tactile exploration," in *2013 IEEE International Conference on Robotics and Automation*, pp. 113–119, 2013.

[99] U. Martinez-Hernandez, T. J. Dodd, M. H. Evans, T. J. Prescott, and N. F. Lepora, "Active sensorimotor control for tactile exploration," *Robotics and Autonomous Systems*, vol. 87, pp. 15–27, 2017.

[100] T. Matsubara and K. Shibata, "Active tactile exploration with uncertainty and travel cost for fast shape estimation of unknown objects," *Robotics and Autonomous Systems*, vol. 91, pp. 314–326, 2017.

[101] J. Mockus, *Bayesian Approach to Global Optimization: Theory and Applications.* Mathematics and its Applications, Springer Netherlands, 2011.

[102] E. Snelson and Z. Ghahramani, "Sparse Gaussian processes using pseudo-inputs," in *Advances in Neural Information Processing Systems*, vol. 18, 2005.

[103] M. Titsias, "Variational learning of inducing variables in sparse Gaussian processes," in *International Conference on Artificial Intelligence and Statistics*, vol. 5, pp. 567–574, 2009.

Publication List

Journal

1. Kazuki Shibata, Tatsuya Miyano, Tomohiko Jimbo, and Takamitsu Matsubara, “Robust shape estimation with false-positive contact detection,” Robotics and Autonomous Systems, vol. 129, 103527, 2020 (Chapter 6).

2. Kazuki Shibata, Tomohiko Jimbo, and Takamitsu Matsubara, “Deep reinforcement learning of event-triggered communication and consensus-based control for distributed cooperative transport,” Robotics and Autonomous Systems, vol. 159, 104307, 2023 (Chapter 4).

International Conference

1. Kazuki Shibata, Tomohiko Jimbo, and Takamitsu Matsubara, "Deep reinforcement learning of event-triggered communication and control for multi-agent cooperative transport," IEEE International Conference on Robotics and Automation (ICRA), pp. 8671-8677, 2021 (Chapter 4).

2. Kazuki Shibata, Tomohiko Jimbo, Tadashi Odashima, Keisuke Takeshita, and Takamitsu Matsubara, "Learning locally, communicating globally: reinforcement learning of multi-robot task allocation for cooperative transport," IFAC World Congress, 2023, Accepted (Chapter 5).