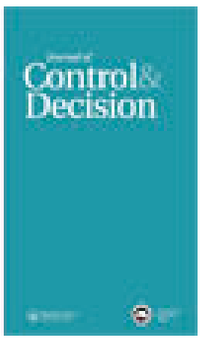


Journal of Control and Decision

ISSN: 2330-7706 (Print) 2330-7714 (Online) Journal homepage: https://www.tandfonline.com/loi/tjcd20

Multi-robot formation control: a comparison between model-based and learning-based methods

Chao Jiang, Zhuo Chen & Yi Guo

To cite this article: Chao Jiang, Zhuo Chen & Yi Guo (2019): Multi-robot formation control: a comparison between model-based and learning-based methods, Journal of Control and Decision, DOI: 10.1080/23307706.2019.1697970

To link to this article: https://doi.org/10.1080/23307706.2019.1697970

Published online: 06 Dec 2019.

JOURNAL OF CONTROL AND DECISION
https://doi.org/10.1080/23307706.2019.1697970

Taylor & Francis
Taylor & Francis Group

Multi-robot formation control: a comparison between model-based and learning-based methods

Chao Jiang*, Zhuo Chen and Yi Guo

Department of Electrical and Computer Engineering, Stevens Institute of Technology, Hoboken, NJ, USA

ABSTRACT

Formation control of multi-robot systems has been extensively studied by model-based methods, where analytic control inputs are constructed based on the kinematics and/or dynamics model and the communication graphs of the multi-robot system. Recently, driven by remarkable advances of robotic learning techniques, emerging studies on learning-based methods for formation control have been developed for adaptive and intelligent control of multi-robot systems. This paper aims to provide a brief overview of our recent development of learning-based formation control, and compare it with a model-based method for a case study of three-robot formation control. Fundamental principles, experimental results and technical challenges are presented, comparing the two different methodologies.

ARTICLE HISTORY
Received 31 August 2019
Accepted 22 November 2019

KEYWORDS
Multi-robot systems; formation control; multi-robot learning

1. Introduction

1.1. Motivation and contribution

Formation control of multi-robot systems is an essential and common problem in a vast range of robotic applications such as cooperative area exploration, self-driving vehicles and security patrols (Guo, 2017). Conventionally, multi-robot formation control is addressed by model-based methods where algorithms are designed to compute analytic robot control inputs using the knowledge of robot kinematic and/or dynamic model and communication graph (Oh, Park, & Ahn, 2015; Qu, 2009). Model-based methods can be implemented efficiently in real time, however, their reliance on model accuracy and communication reliability makes the performance of those methods vulnerable to uncertainties and disturbances of the system and environment. As the complexity of the application domains of formation control constantly increases, the demands for more adaptive and intelligent robot controls are on the rise, thus motivate the development of data-driven control methodologies for multi-robot formation.

Multi-agent learning has long been an active research area where learning algorithms are devised to have multi-agent systems (MAS) attain their optimal actions and/or coordination via experience obtained through interacting with other agents and the environment

CONTACT Yi Guo yguo1@stevens.edu
*Present address: Department of Electrical and Computer Engineering, University of Wyoming, Laramie, WY, USA

(Amato et al., 2016; Liu, Amato, Anesta, Griffith, & How, 2016; Tuyls & Weiss, 2012). Learning from experience provides an alternative methodology for multi-agent control or planning when hand-engineering the policies is difficult or even impossible to obtain in complex environments. The difficulties could stem from the lack of complete knowledge of the agent and environment models, e.g. unknown observation probabilities, unknown behaviours of other agents, and non-deterministic environments. Thus, designing agent behaviours in advance is not appropriate in such scenarios. A growing number of work on multi-agent learning algorithms have emerged in the past a few years, and have been applied in robot soccer, mobile sensor networks and video games (Amato, Chowdhary, Geramifard, Üre, & Kochenderfer, 2013; Barrett & Stone, 2015; Foerster et al., 2017).

Multi-robot formation control as a sub-field of MAS problems could inherently leverage the advancement of multi-agent learning methods to address the emerging challenges faced by conventional model-based. Pioneering studies reported in (Aykin, Knopp, & Diepold, 2018; Hüttenrauch, Adrian, & Neumann, 2019; Jiang, Chen, & Guo, 2019) have explored new methods that use deep learning methods for multi-robot formation control. This paper aims to provide the comparison of the model-based and the learning-based methods. Specifically, two recent work of the different methodologies are briefly overviewed for a three-robot formation control problem, and comparisons are presented in terms of the fundamental principles, experimental results and technical challenges.

1.2. Related work

Formation control of multi-agent systems was initially inspired from the natural phenomena of bird flocking and fish schooling, where a group of agents follow the same direction and speed while maintaining certain geometric shape (Guo, 2017; Oh et al., 2015; Qu, 2009). Existing model-based methods of multi-agent formation control can be classified into *position-, displacement-, and distance-based* controls, where the desired formation is defined by the *absolute position* of each agent with respect to a global reference frame, the desired *displacements* with respect to a global reference frame, and the desired *inter-agent distances*, respectively (Oh et al., 2015). There is a tradeoff between the sensing capability and interaction topology in the above categorisation, and distance-based control requires less sensing capability but more interactions among agents. Recently, researchers proposed vision-based control schemes to alleviate the requirement of inter-robot communication (Gustavi & Hu, 2008; Liang, Wang, Liu, Chen, & Liu, 2017; Vidal, Shakernia, & Sastry, 2004; Wang et al., 2016). For example, in the work by Gustavi and Hu (2008), the dynamics between the leader and follower robots is modelled using the distance, orientation and bearing angle, and a dynamic feedback controller was designed using an observer that estimates the neighbour's speed. These model-based formation control methods either require the availability of full or partial state measurement, or rely on deliberate perception module that explicitly returns measurements (such as bearing and distances) from robot sensors through traditional sensor fusion techniques.

In the last few years, a new research direction has been actively pursued, which utilises the enormous expressive capability of deep neural nets to directly process high-dimensional raw sensing data for self-driving cars (Rausch et al., 2017; Wulfmeier, Rao, Wang, Ondruska, & Posner, 2017). Also, the human-level intelligence for robot control has been proposed with the aim to develop robot decision-making policies that generate

actions directly from raw observation, so as to mimic the functioning and learning process of human brains (Mnih et al., 2015). Recent deep learning methods have achieved great success in learning representation from raw sensing data using deep neural networks (DNN), thus provide an end-to-end framework for learning robot control policies that combine multiple decoupled modules from robot perception to action. To mention a few studies of robot formation control using deep learning methods, Aykin et al. (2018) applied deep Q-learning to the leader–follower formation control problem. A deep neural network was used to represent the robot's policy that maps raw image observations to actions. Jiang et al. (2019) studied the problem of learning decentralised formation control policies for multi-robot formation. A DNN model that processes robot's LIDAR observations to generate motor control was designed and trained in a centralised manner using supervised learning with expert demonstration data generated by a model-based controller. The trained model is then deployed on each robot as a decentralised controller that only relies on local observation to achieve formation without inter-robot communication. Hüttenrauch et al. (2019) proposed a deep state representation based on DNN for end-to-end learning control of robotic swarm systems. The proposed approach addresses the challenges of high and possibly changing dimentionality of robot observations in robot policy learning. The aforementioned work shed light on how to learn formation control from high dimensional observation received by the robots using deep learning methods.

1.3. Paper organisation

The reminder of this paper is as follows. Section 2 presents a method of model-based formation control for a three-robot systems using distance measurements only. Section 3 briefly overviews our recent method on learning decentralised control of robot formation. Section 4 provides the comparison between the model-based and learning-based methods. Section 5 concludes the paper.

2. Model-based method for formation control

The problem of multi-agent formation control can be classified into position-, displacement- and distance-based formation control depending on the system's sensing capabilities and interaction topologies. In this section, we introduce a distance-based formation control method for a three-robot system that was originally developed in Chen, Jiang, and Guo (2019). The main experimental results show stable rigid formation under the distributed control law.

2.1. Formation control for single-integrator modelled robots

2.1.1. Problem statement

Suppose the motion of a group of three robots is governed by

$$\dot{\boldsymbol{p}}_i = \boldsymbol{v}_i, \quad i \in \mathcal{V} \triangleq \{1, 2, 3\}, \tag{1}$$

where $\boldsymbol{p}_i = [x_i, y_i]^{\mathrm{T}} \in \mathbb{R}^2$ and $\boldsymbol{v}_i = [v_{xi}, v_{yi}]^{\mathrm{T}} \in \mathbb{R}^2$ denote, respectively, the state and control of the ith robot. The relative displacement $\tilde{\boldsymbol{p}}_{ij}$ of agent i with respect to agent j is defined

as $\tilde{\boldsymbol{p}}_{ij} = \boldsymbol{p}_i - \boldsymbol{p}_j$. Let $\tilde{\boldsymbol{p}}$ be the stack vector of $\tilde{\boldsymbol{p}}_{ij}$, i.e. $\tilde{\boldsymbol{p}} = [\tilde{\boldsymbol{p}}_{12}^{\mathrm{T}}, \tilde{\boldsymbol{p}}_{23}^{\mathrm{T}}, \tilde{\boldsymbol{p}}_{31}^{\mathrm{T}}]^{\mathrm{T}} \in \mathcal{D}$, where $\mathcal{D}$ is a manifold in $\mathbb{R}^6$ defined by

$$\mathcal{D} = \{\tilde{\boldsymbol{p}} \in \mathbb{R}^6 : \tilde{\boldsymbol{p}}_{12} + \tilde{\boldsymbol{p}}_{23} + \tilde{\boldsymbol{p}}_{31} = \boldsymbol{0}\}. \tag{2}$$

The control objective is for the robots to achieve a desired formation and a desired velocity $\boldsymbol{v}^* = [v_x^*, v_y^*]^{\mathrm{T}} \in \mathbb{R}^2$. Given the desired distances $d_{ij}^* = d_{ji}^* \in (0, +\infty)$ for any $(i,j) \in \mathcal{E}^+$ where $\mathcal{E}^+ = \{(1,2),(2,3),(3,1)\}$, the desired formation is defined as

$$\mathcal{B} = \{\tilde{\boldsymbol{p}} \in \mathcal{D} : \|\tilde{\boldsymbol{p}}_{ij}\| = d_{ij}^*,\ (i,j) \in \mathcal{E}^+\}. \tag{3}$$

The realisability Oh and Ahn (2011) of the desired formation is ensured by the triangle inequality $d_{ij}^* < d_{ik}^* + d_{jk}^*$, for distinct $i, j, k \in \mathcal{V}$.

The interaction topology of the group of three robots can be modelled by a graph denoted $(\mathcal{V}, \mathcal{E})$ where $\mathcal{V}$ is the set of nodes, each corresponding to a robot, and $\mathcal{E} \subseteq \mathcal{V} \times \mathcal{V}$ is the set of edges (i,j). Each agent's accessibility to its neighbour's displacement is signified by the set of edges. Specifically, if an edge (i,j) exists from i to j, robot i has access to robot j's actual displacement $\tilde{\boldsymbol{p}}_{ji}$ relative to robot i and the desired distance d_{ij}^* from robot i at all times. The set of neighbours of the ith robot is defined as $\mathcal{N}_i = \{j \in \mathcal{V} \mid (i,j) \in \mathcal{E}\}$. $\mathcal{N}_i = \{1,2,3\} \setminus \{i\}$. The Laplacian Matrix $\boldsymbol{L} = [l_{ij}] \in \mathbb{R}^{|\mathcal{V}| \times |\mathcal{V}|}$ of the graph is defined as

$$l_{ij} = \begin{cases} -1, & \text{if } i \neq j \wedge (i,j) \in \mathcal{E} \\ 0, & \text{if } i \neq j \wedge \neg(i,j) \in \mathcal{E} \\ \sum_{k \neq i} l_{ik}, & \text{if } i = j \end{cases} \tag{4}$$

In the three-robot system case, the vertex set is $\mathcal{V} = \{1,2,3\}$ and the edge set is $\mathcal{E} = \{(1,2),(1,3),(2,3),(2,1),(3,1),(3,2)\}$. Assume an undirected connected gragh with the Laplacian matrix as:

$$\boldsymbol{L} = \begin{bmatrix} 2 & -1 & -1 \\ -1 & 2 & -1 \\ -1 & -1 & 2 \end{bmatrix} \tag{5}$$

Assumption 2.1 (Initial condition): The initial positions of the robots are not collinear (Oh & Ahn, 2011). Namely, $\tilde{\boldsymbol{p}}(0) \notin \mathcal{C}$, where

$$\mathcal{C} \triangleq \{\tilde{\boldsymbol{p}} \in \mathcal{D} : \det[\tilde{\boldsymbol{p}}_{12}, \tilde{\boldsymbol{p}}_{23}] = 0\} \tag{6}$$

Let the formation separation error e_{ij} between agents i and j be defined as

$$e_{ij} = \|\tilde{\boldsymbol{p}}_{ij}\| - d_{ij}^*, \quad \forall (i,j) \in \mathcal{E}^+. \tag{7}$$

The distance-based formation control problem is defined as follows.

Problem 2.1 (Single integrator): Given a three-agent system with Laplacian matrix (5), single-integrator model (1), and Assumption 2.1, find a control law $\boldsymbol{v}_i$ with $i \in \mathcal{V}$ such that

as $t \to \infty$, each robot reaches the desired velocity

$$\dot{\boldsymbol{p}}_i \to \boldsymbol{v}^*, \quad i \in \mathcal{V} \tag{8}$$

and the system achieves the desired formation $\mathcal{B}$, i.e.

$$e_{ij} \to 0 \quad (i,j) \in \mathcal{E}^+ \tag{9}$$

2.1.2. Control design

With a set of potential functions that have their minimums at the desired distances, a gradient-descent control law can drive the system to its desired formation. A controller for single-integrator modelled agents can be written as

$$\boldsymbol{v}_i = \boldsymbol{v}^* - \nabla_{\boldsymbol{p}_i} \sum_{j \in \mathcal{N}_i} \gamma_{ij}(\|\boldsymbol{p}_j - \boldsymbol{p}_i\|^2), \tag{10}$$

where $\gamma_{ij} : (0, +\infty) \to \mathbb{R}$ is a differentiable potential function with only one minimum at d_{ij}^{*2}. We choose γ_{ij} as

$$\gamma_{ij}(\|\boldsymbol{p}_j - \boldsymbol{p}_i\|^2) = \frac{K}{2} \frac{(\|\boldsymbol{p}_j - \boldsymbol{p}_i\|^2 - d_{ij}^{*2})^2}{\|\boldsymbol{p}_j - \boldsymbol{p}_i\|^2}, \tag{11}$$

where $K > 0$. Note that γ_{ij} quickly approaches infinity as $\|\boldsymbol{p}_j - \boldsymbol{p}_i\|^2 \to 0$. This property enables effective collision avoidance between neighbouring robots.

Let $\beta_{ij}(\tilde{\boldsymbol{p}}) \triangleq \|\tilde{\boldsymbol{p}}_{ij}\|^2 = \|\boldsymbol{p}_j - \boldsymbol{p}_i\|^2$. The partial derivative of γ_{ij} with respect to β_{ij} can be written as

$$\rho_{ij} \triangleq \frac{\partial \gamma_{ij}(\beta_{ij})}{\partial \beta_{ij}} = \frac{K(\beta_{ij}^2 - d_{ij}^{*4})}{\beta_{ij}^2}. \tag{12}$$

The domain of definition of function ρ_{ij} is $\mathcal{D} \setminus \mathcal{Z}$ where $\mathcal{Z} \triangleq \mathcal{Z}_{12} \cup \mathcal{Z}_{23} \cup \mathcal{Z}_{31}$ and

$$\mathcal{Z}_{ij} \triangleq \{\tilde{\boldsymbol{p}} \in \mathcal{D} : \tilde{\boldsymbol{p}}_{ij} = \boldsymbol{0}\}, \quad (i,j) \in \mathcal{E}^+.$$

Then the controller (10) can be rewritten as

$$\boldsymbol{v}_i = \boldsymbol{v}^* - \sum_{j \in \mathcal{N}_i} \frac{\partial \gamma_{ij}(\beta_{ij})}{\partial \beta_{ij}} \frac{\partial \beta_{ij}}{\partial \boldsymbol{p}_i} = \boldsymbol{v}^* - \sum_{j \in \mathcal{N}_i} \rho_{ij} \cdot (\boldsymbol{p}_i - \boldsymbol{p}_j) = \boldsymbol{v}^* - \sum_{j \in \mathcal{N}_i} \rho_{ij} \tilde{\boldsymbol{p}}_{ij}. \tag{13}$$

Proposition 2.1 (Single integrator): *Consider system (1) with Laplacian matrix (5), driven by control law (13). If Assumption* 2.1 *is satisfied, then* $\lim_{t\to\infty} e_{ij}(t) = 0$ *for* $(i,j) \in \mathcal{E}^+$ *and* $\lim_{t\to\infty} \dot{\boldsymbol{p}}_i = \boldsymbol{v}^*$ *for* $i \in \mathcal{V}$. *That is, Problem* 2.1 *is solved by the controller (13).*

The proof of Proposition 2.1 can be found in Chen et al. (2019).

2.2. Formation control for differential drive robots

In this section, we present the generalisation of the controller proposed in Section 2.1 to non-holonomic multi-robot system.

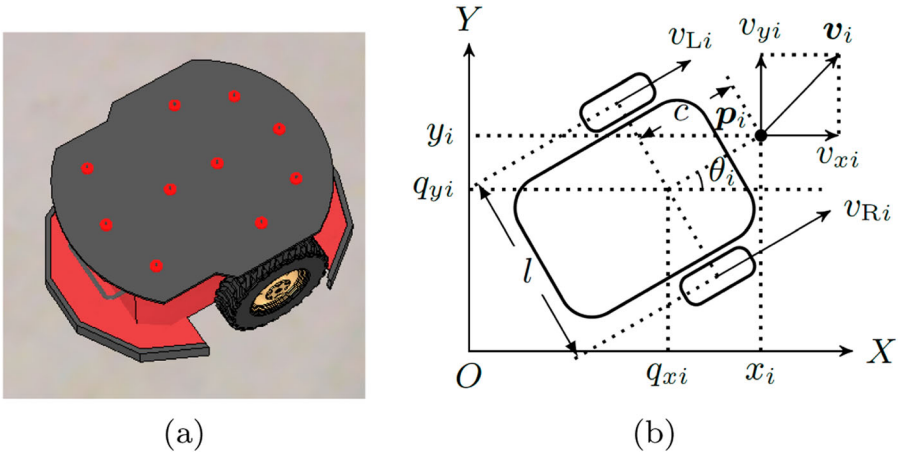


Figure 1. Kinematics of a differential drive robot: (a) Pioneer P3-DX robot; (b) Schematic illustration.

2.2.1. Differential drive robot kinematics

The two-wheel differential drive robot, Pioneer P3-DX as shown in Figure 1(a), is considered for the multi-robot system. Assuming that it moves only on a planar surface, its state equation can be expressed as

$$\dot{\boldsymbol{q}}_i = \boldsymbol{G}(\boldsymbol{q}_i)\,\mathbf{sat}_{v_{\mathrm{m}}}(\boldsymbol{\Gamma}_i), \quad i \in \{1,2,3\}, \tag{14}$$

where the ith robot's pose $\boldsymbol{q}_i = [q_{xi}, q_{yi}, \theta_i]^{\mathrm{T}} \in SE(2)$ consists of the displacement $[q_{xi}, q_{yi}]^{\mathrm{T}} \in \mathbb{R}^2$ of the midpoint of the line segment connecting the robot's two driving wheels and the orientation $\theta_i \in SO(2)$ of the robot. The matrix $\boldsymbol{G}(\boldsymbol{q}_i)$ is written as

$$\boldsymbol{G}(\boldsymbol{q}_i) = \begin{bmatrix} \dfrac{\cos\theta_i}{2} & \dfrac{\cos\theta_i}{2} \\ \dfrac{\sin\theta_i}{2} & \dfrac{\sin\theta_i}{2} \\ -\dfrac{1}{l} & \dfrac{1}{l} \end{bmatrix},$$

where l is the distance between the two driving wheels. The control input $\boldsymbol{\Gamma}_i = [v_{\mathrm{L}i}, v_{\mathrm{R}i}]^{\mathrm{T}} \in \mathbb{R}^2$ represents the wheel velocity input, which is comprised of the left wheel linear speed $v_{\mathrm{L}i}$ and the right wheel linear speed $v_{\mathrm{R}i}$ of the ith robot. The saturation function $\mathbf{sat}_{v_{\mathrm{m}}} : \mathbb{R}^2 \to [-v_{\mathrm{m}}, v_{\mathrm{m}}]^2$ is defined as

$$\mathbf{sat}_{v_{\mathrm{m}}}(\boldsymbol{\Gamma}_i) = \begin{bmatrix} \mathrm{sat}_{v_{\mathrm{m}}}(v_{\mathrm{L}i}) \\ \mathrm{sat}_{v_{\mathrm{m}}}(v_{\mathrm{R}i}) \end{bmatrix},$$

where $\mathrm{sat}_{v_{\mathrm{m}}}(v) \triangleq \mathrm{sgn}(v) \cdot \min\{v_{\mathrm{m}}, |v|\}$ and, and v_m is the maximum speed limit.

2.2.2. Coordinate transformation

As shown in Figure 1(b), we define a reference point located at $\boldsymbol{p}_i = [x_i, y_i]$. If we set $c = l/2$, the nonholonomic kinematic model (14) can be converted to the single-integrator

model (1). That is, in the new coordinate defined by

$$\begin{bmatrix} x_i \\ y_i \end{bmatrix} = \begin{bmatrix} q_{xi} + c\cos\theta_i \\ q_{yi} + c\sin\theta_i \end{bmatrix}. \tag{15}$$

The robot model can be derived as Li, Kong, and Guo (2014):

$$\begin{bmatrix} \dot{x}_i \\ \dot{y}_i \end{bmatrix} = \begin{bmatrix} v_{xi} \\ v_{yi} \end{bmatrix}. \tag{16}$$

The control input $\mathbf{\Gamma}'_i \in \mathbb{R}^2$ is defined as:

$$\mathbf{\Gamma}'_i \triangleq \begin{bmatrix} v'_{\mathrm{L}i} \\ v'_{\mathrm{R}i} \end{bmatrix} = \begin{bmatrix} \sin\theta_i + \dfrac{l}{2c}\cos\theta_i & \sin\theta_i - \dfrac{l}{2c}\cos\theta_i \\ [6pt]\sin\theta_i - \dfrac{l}{2c}\cos\theta_i & \sin\theta_i + \dfrac{l}{2c}\cos\theta_i \end{bmatrix} \begin{bmatrix} v_{xi} \\ v_{yi} \end{bmatrix}. \tag{17}$$

After the coordinate transformation, the control law (13) can be applied to the single-integrator model based on Proposition 2.1.

2.3. Simulation

2.3.1. Experiment environment

The controller is evaluated in experiments using a robot simulator V-REP (Rohmer, Singh, & Freese, 2013). The simulation scenario with three P3-DX robots is shown in Figure 2. Each driving wheel motor of the robot is configured to be able to produce 3 N·m of torque, while the maximum linear speed of the driving wheels are set to 0.7 m/s.

2.3.2. Simulation results

The setup of the simulation is described as follows. The desired formation is set to $d^*_{12} = 1$ m, $d^*_{23} = 2$ m, $d^*_{31} = \sqrt{3}$ m. The maximum wheel linear speed is $v_{\mathrm{m}} = 0.7$ m/s. The common desired velocity is $\boldsymbol{v}^* = [0.25, 0.25\sqrt{3}]$ (m/s). The controller parameter K in (12) is chosen as $K = 0.4$. Note that any positive constant K can stabilise the system, but the performance could vary. K is empirically chosen as $K = 0.4$ for good transient behaviours including small overshoot and short settling time.

The simulation snapshots are shown in Figure 2. One can see the robot team achieves a desired formation and velocity after 6 seconds. Figure 3 shows that simulation results. It can be seen from Figure 3(a) that the formation separation errors reduce from 4 m to 0 within 7 s. Figure 3(b) showss that the robot wheel velocities initially are maxed at 0.7 m/s and then settled to the common desired velocity 0.5 m/s. Figure 3(c) shows the trajectories of the three robots that achieve the triangular formation and then move in consensus at the desired velocity.

3. Learning-based method for formation control

The model-based method uses the robot model and the performance is subject to model uncertainties and external disturbances. When there's a lack of complete knowledge of the robot model, designing robot control is difficult or even impossible. Moreover, the model-based method requires state measurement and hand-engineering the robot perception

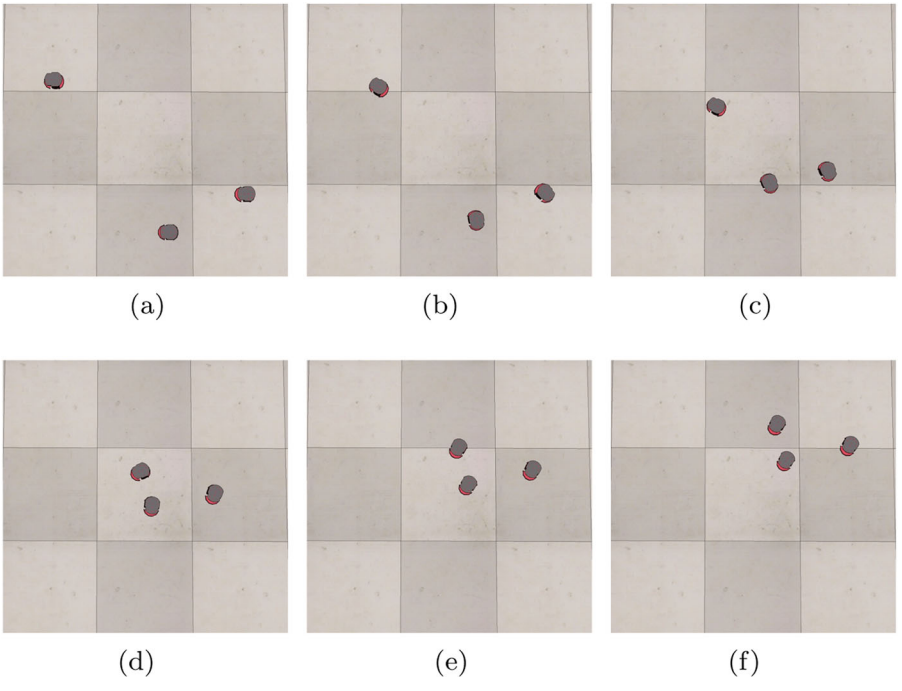


Figure 2. Kinematics of a differential drive robot: (a) Pioneer P3-DX robot; (b) Schematic illustration. (a) $t = 0$ s, (b) $t = 1.5$ s, (c) t = 3 s, (d) $t = 4.5$ s, (e) $t = 6$ s, (f) $t = 7.5$ s.

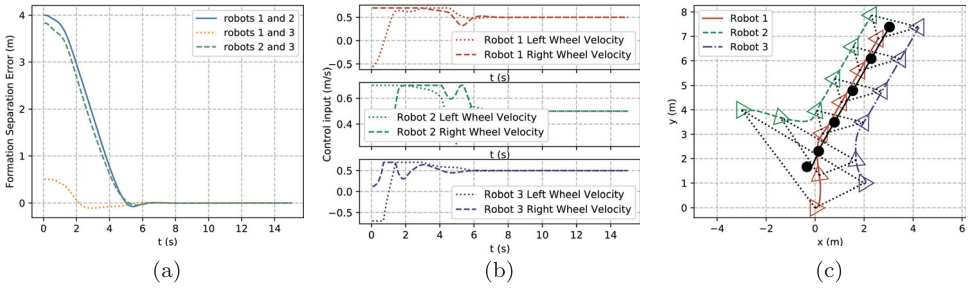


Figure 3. Performance of the proposed control law with $K = 0.4$: (a) Formation separation error $e_{ij}(t)$; (b) Trajectories of the three robots are denoted by solid curves in red, green and blue. Triangles represent the pose of the robots every 3 s; Solid black circles and black curves illustrate the trajectory of their centre of mass. (c) Robot control input $[v_{Li}, v_{Ri}]^T$. (a) $t = 0$ s (b) $t = 1.5$ s (c) $t = 3$ s (colour online).

and control components separately. In this section, we introduce the problem of learning decentralised formation control policies for multi-robot systems, which do not rely on the robot model and can directly operate on individual robots' local perception without inter-robot communication. This method was originally presented in our recent conference paper (Jiang et al., 2019).

3.1. Problem statement

Consider the formation control problem with three mobile robots that are equipped with LIDAR sensors as shown in Figure 4. Each robot has an on-board self-localisation system

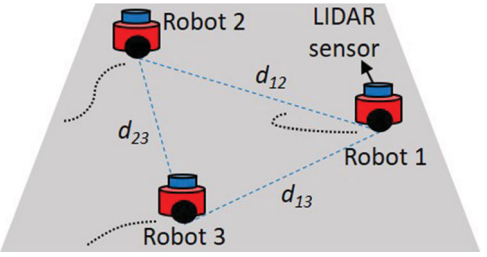


Figure 4. The schematic diagram of the formation control scenario.

that can measure its current orientation, $\phi_i(t)$. The desired formation is defined as the desired orientation, ϕ^*, desired speed (which is a scalar), v^*, and the desired relative distance between any two robots, d_{ij} for $i, j = 1, 2, 3$ and $i \neq j$. For simplicity, it is assumed that $d_{ij} = d^*$ is a positive constant. The three robot team achieves formation if the robots follow the desired orientation ϕ^* and the desired speed v^*, and keep the distance d^* from each other.

The objective is to learn a decentralised control policy and use it for online formation control. The decentralised control policy can be regarded as a generalisation of a decentralised control law, which maps the input of the robot's local sensor observation (i.e. occupancy map from the robot onboard LIDAR sensor) to the output of the robot's motor control. That is

$$\hat{\boldsymbol{u}}_i = \boldsymbol{F}(\boldsymbol{y}_i, \Delta\phi_i, d^*), \tag{18}$$

where the input of the control policy consists of the occupancy map $\boldsymbol{y}_i$ and two auxiliary inputs including the difference between the robot's current orientation and the desired orientation, $\Delta\phi_i = \phi_i - \phi^*$, and the desired formation distance, d^*; and the control policy outputs the robot motor control $\hat{\boldsymbol{u}}_i = [\hat{u}_{il}, \hat{u}_{ir}] \in \mathbb{R}^2$ with $\hat{u}_{il}$ and $\hat{u}_{ir}$ being the left and right motor control, respectively. A DNN model is designed as the parameterised function representation of the control policy $\boldsymbol{F}(\cdot)$ defined in (18).

3.2. Approach

The learning-based approach has two phases, i.e. the training phase and the testing phase as shown in Figure 5.

During the training phase, the expert demonstrations generated by a model-based method are used to train the DNN. Specifically, given the robot states obtained by the measurement system, the model-based controller generates the robot control, $\boldsymbol{u}_i$, as expert demonstration. Meanwhile, the corresponding local observation, $\boldsymbol{y}_i$, from the robot's onboard LIDAR sensor and the auxiliary inputs, $\Delta\phi_i$ and d^*, are recorded. The expert control $\boldsymbol{u}_i$, LIDAR observation $\boldsymbol{y}_i$ and the auxiliary data $\Delta\phi_i$, d^* are then fed to the DNN for training.

The training process is essentially to optimise the DNN parameters $\boldsymbol{\theta}$ such that given the three-tuple $(\boldsymbol{y}_i, \Delta\phi_i, d^*)$, the error between the DNN output, $\hat{\boldsymbol{u}}_i = \boldsymbol{F}(\boldsymbol{y}_i, \Delta\phi_i, d^*; \boldsymbol{\theta})$, and the expert control $\boldsymbol{u}_i$ provided by expert demonstration is minimised. Thus, the loss function

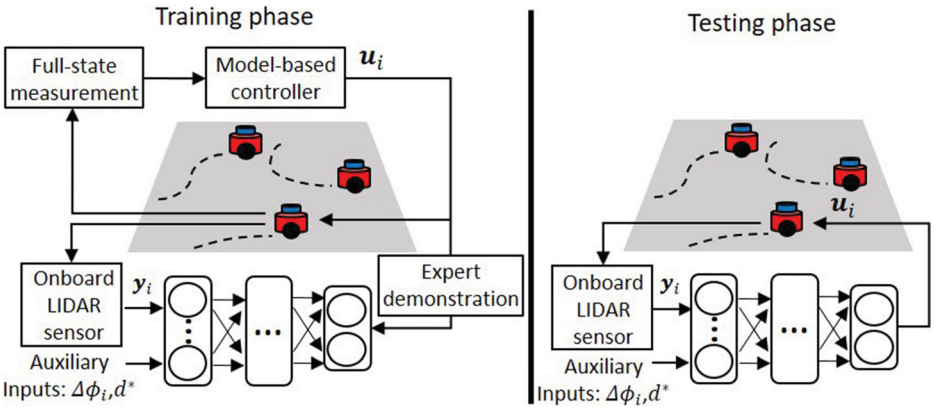


Figure 5. The overview of our proposed policy learning scheme.

for each iteration of the mini-batch learning is defined as the Euclidean loss:

$$\mathcal{L}(\boldsymbol{\theta}) = \frac{1}{N_B}\sum_{j=1}^{N_B} \|\boldsymbol{F}(\boldsymbol{y}_j, \Delta\phi_j, d^*; \boldsymbol{\theta}) - \boldsymbol{u}_j\|^2, \tag{19}$$

where N_B is the mini-batch size. A gradient-descent based training algorithm with Adam optimiser (Kingma & Ba, 2014) is used to learn the optimal parameters of the DNN. After T epochs of training process, the DNN parameters $\boldsymbol{\theta}$ converge to optimum in the sense that the loss function (19) is minimised. The DNN model with the learned parameters is then used by each robot in online testing phase.

In the testing phase, the trained DNN policy is executed on each robot i in a decentralised manner to compute robot control command through a feedforward pass of DNN based on current robot LIDAR observation and auxiliary inputs. The robot motor control law $\boldsymbol{u}_i = K\hat{\boldsymbol{u}}_i$, where K is a constant gain that adjusts the robot's speed to the desired speed v^*. Note that the common speed of the robot team is not an input to the DNN in the training phase, and can be adjusted directly during the testing phase through linear scaling.

3.3. Experiment results

3.3.1. Experiment setup

Robot simulation: The robot simulation is performed in the robot simulator V-REP (Rohmer et al., 2013). Figure 6 shows the snapshots of a simulation experiment in V-REP. The differential drive mobile robot, Pioneer P3-DX, is chosen as the robot platform. The robot is equipped with a Velodyne VPL16 LIDAR sensor to perform laser scan measurement. The data of laser scan measurement in a square region of $5 \times 5\,\text{m}^2$ centred at the sensor position are represented by a 50×50 gray-scale image of 2D occupancy map. Thus, the spatial resolution of the occupancy map is 0.1 m/pixel. Note that, given the same laser scan range, increasing the spatial resolution of the occupancy map results in a larger size of the image and more computational cost. The tradeoff between the image size and the spatial resolution of the occupancy map can be balanced considering practical requirements

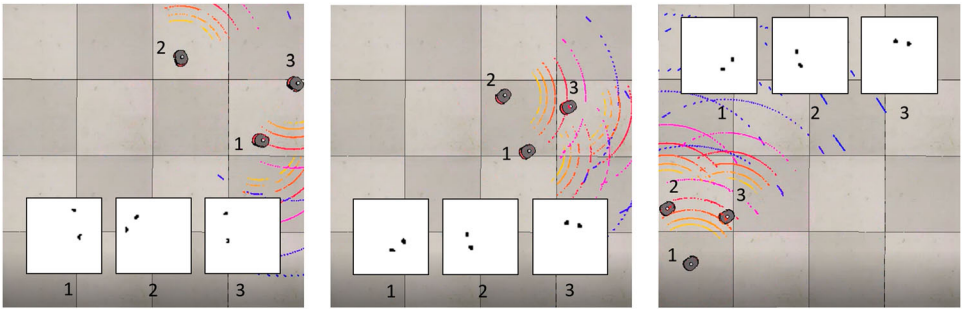


Figure 6. Snapshots of online formation control experiment in V-REP simulator at (1) $t = 0$ s; (b) $t = 5$ s; and (c) $t = 15$ s. The squared images show the occupancy map generated from the observation of the corresponding robot, where the black dots indicate the surrounding robots of each robot. The coloured arcs visualise the LIDAR scanning.

of applications. The robot control algorithms are implemented in a client program written in Python, which communicates with the V-REP simulator via the provided remote API. Thus, the V-REP simulator sends simulation data to the client program for robot control calculation, and computes the robot dynamics given the control commands received from the client program. The simulation frequency is set to 20 Hz.

DNN implementation: The first convolutional layer (Conv1) has 32 filters with filter size 8×8 and stride 4. The output of the Conv1 layer is 32 feature maps of dimension 12×12. The second convolutional layer (Conv2) has 16 filters with size 4×4 and stride 2. The output of the Conv2 layer is 16 feature maps of dimension 5×5. The final convolutional layer (Conv3) has 16 filters with size 3×3 and stride 1. The output of the Conv3 layer is 16 feature maps of dimension 3×3. All three convolutional layers are activated by ReLU. The feature maps of the last convolutional layer are flattened into a 144-dimensional vector which is fed to the FC network. The dimension of each hidden layer of the FC network is 32, and the dimension of the output layer is 2. The first and second layer are followed by ReLU and linear activation, respectively.

3.3.2. Training experiments

Data collection. The training data were collected in the robot simulator V-REP, where a model-based controller was implemented to control each robot to provide expert demonstration data. The distance-based formation controller proposed in Dimarogonas and Johansson (2008) is used to generate the motor control for the robots. A simulation scenario with three mobile robots is created. The robots' initial positions $\boldsymbol{p}_i(0)$ are randomly selected from a circular region with a radius of 3 m, and the initial orientations $\phi_i(0)$ are randomly selected from $[0, 2\pi)$. For each simulation run during data collection, a constant desired orientation ϕ^* is randomly selected from $[0, 2\pi)$, and the desired speed v^* is set to 0.7 m/s. The desired formation distance d^* is randomly selected from the set $\{1, 1.5, 2\}$ m. The duration of each simulation run is 12 s. At each time step t, the control output $\boldsymbol{u}_i(t)$ computed by the model-based controller, the corresponding occupancy map $\boldsymbol{y}_i(t)$ from LIDAR observation, the orientation difference $\Delta\phi_i(t)$ of each robot i, $i = 1, 2,$

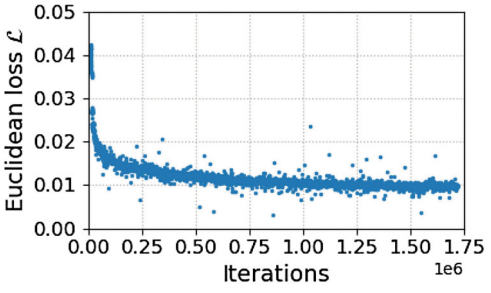


Figure 7. Euclidean loss over training iterations.

3, and the desired formation distance d^* was recorded and stored as one sample of training data. The entire training data set includes 110,160 samples, which were grouped into 3442 mini-batches with the batch size $N_B = 32$. The sampled data collected by all three robots are used to train the control policy in a centralised training framework.

Training results. With the collected training data, the DNN model is trained using supervised learning to find an appropriate control policy. The initial learning rate is set as $\eta_0 = 0.0001$; the exponential decay rates for the moment estimates are initialised as $\beta_1 = 0.9$ and $\beta_2 = 0.999$, respectively. The constant $\hat{\epsilon}$ is set to 10^{-8}. The training epoch is set as $T = 500$. The evolution of loss over training iterations is shown in Figure 7. One can see that the loss of the Adam optimiser converges around 0.009 after 1.2×10^6 iterations of training. The training results demonstrate that our proposed learning scheme is converging and minimises difference between the predicted control output and the expert control. Next, we verify and evaluate the learned policy in online testing experiments.

3.3.3. Testing experiments

After sufficient training, the parameters of the DNN model remain unchanged, and the DNN model is deployed on each robot for online formation control. As shown in Figure 5, the learned policy model calculates the control command from the robot's own observation via the onboard LIDAR sensor and the auxiliary inputs. The learned robot control policy is evaluated in the cases of constant and time-varying desired velocity, respectively. In all the online testing experiments, the magnitude of the desired velocity is set to $v^* = 0.7$ m/s, thus the speed control gain $K = 1$ as the speed after convergence given by the trained control policy is 0.7 m/s.

Case 1 with constant desired velocity. The testing case with a constant desired velocity is shown in Figure 8. The desired orientation is $\phi^* = 3.76$ rad, and the desired formation distance is 2 m. One can see from Figure 8(a) that the formation error between the robots and orientation error of each robot decrease and converge around zero after about 6 s. Figure 8(b) shows the control commands of the robots' left and right wheel speed. It can be seen that the control of the robot wheel speed converges around the desired speed 0.7 m/s of the robot after 6 s.Figure 8(c) shows the trajectories of the robots, where the triangles and the dotted lines represent the position of the robots and the formation at every 3 s, respectively. It can be seen that the group of robots achieves the desired formation and moves at the desired velocity.

Cases 2 and 3 with time-varying desired velocity. In these cases, the desired orientation $\phi^*(t)$ changes as a function of time. Note that, in the training phase, only constant desired

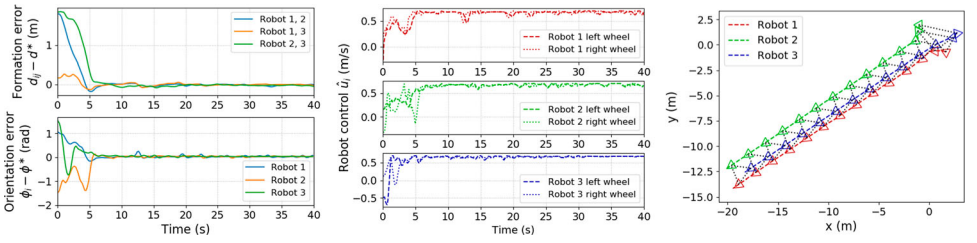


Figure 8. Case 1 with constant desired orientation $\phi^* = 3.76$ rad and desired formation distance $d^* = 2$ m: (a) formation and orientation error; (b) robot control; (c) robot trajectories.

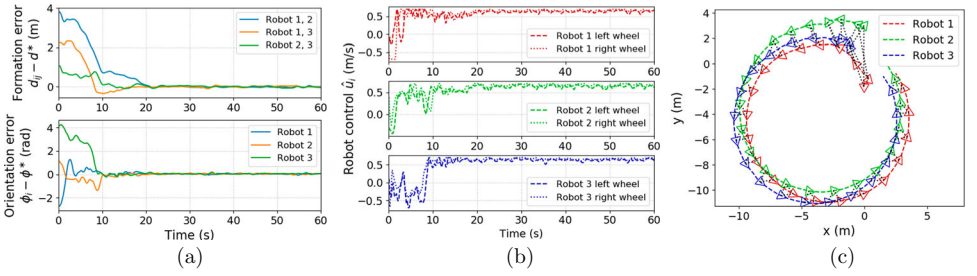


Figure 9. Case 2 with time-varying desired orientation $\phi^*(t) = 0.1t$ rad and desired formation distance $d^* = 1$ m: (a) formation and orientation error; (b) robot control; (c) robot trajectories.

orientations ϕ^* are used to train the DNN, and the time-varying desired velocities are not in the training set. As shown in this subsection, the learned policy can achieve formation with time-varying desired velocities. This is due to the reason that $\Delta\phi$ (i.e. the difference between the robot's current orientation and the desired orientation) is chosen as the input to the DNN that learns the mapping from $\Delta\phi$ after training.

For Case 2, the desired orientation is set as $\phi(t) = \omega \cdot t$ with the angular velocity $\omega = 0.1$ rad/s, and the desired formation distance is 1 m. This setup requires the robot team to keep a desired formation and meanwhile moves in a circular trajectory. The testing results of this case are shown in Figure 9. One can see from Figure 9(a) that the formation and orientation errors decrease and converge around zero after 20 s.Figure 9(b) shows the control output of the robots' left and right wheel speed. It can be seen that the speed control of the robots converges around 0.7 m/s after 20 s. Figure 9(c) shows the trajectories of the robots, where the triangles and the dotted lines represent the position of the robots and the formation at every 3 s, respectively. One can see that the group of robots achieves the desired formation and moves at the desired velocity.

For Case 3, the desired orientation is set as $\phi(t) = \sin \omega t$ with the angular velocity $\omega = 0.2$ rad/s, and the desired formation distance is selected as 1.5 m. This means the robot team is required to keep a desired formation and meanwhile moves in a sinusoidal-like trajectory where the desired orientation is changing sinusoidally. From Figure 10(a) one can see that the formation and orientation errors decrease and converge around zero after 15 s. Figure 10(b) shows the control command of robot wheel speed which converge around 0.7 m/s after 15 s. The trajectories of the robots are shown in Figure 10(c).

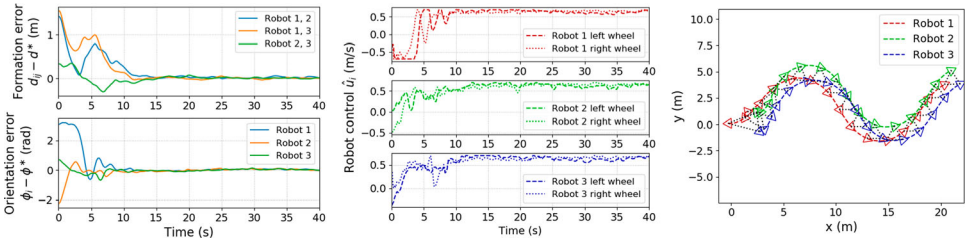


Figure 10. Case 3 with time-varying desired orientation $\phi^*(t) = \sin(0.2t)$ rad and desired formation distance $d^* = 1.5$ m: (a) formation and orientation error; (b) robot control; (c) robot trajectories.

Table 1. Statistical results over 100 runs.

	5% error tolerance	10% error tolerance
Convergence percentage	90%	96%

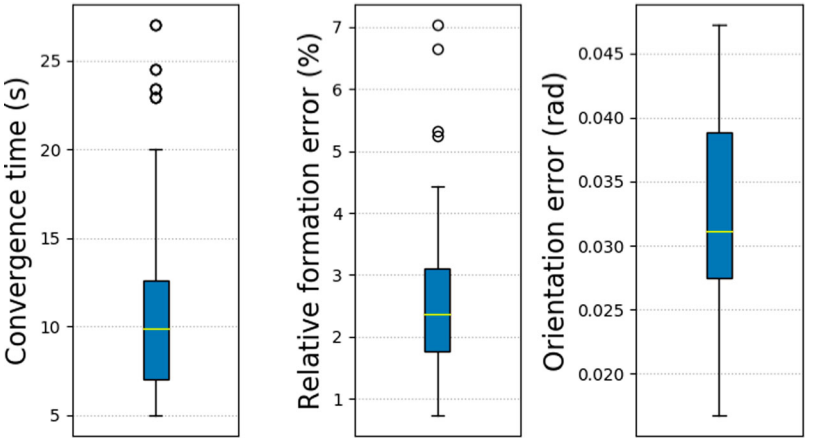


Figure 11. Box plot of convergence time, relative formation error and orientation error after convergence over successful runs.

Statistical results. The statistical results of 100 simulation runs are presented. The simulation time is set to 60 s. A simulation run is considered successful if the formation error between any robot i and j, $i \neq j$, converges in the sense that the temporal average of the relative formation error $|d_{ij} - d^*|/d^*$ over the most recent 20 s is smaller than 5% or 10%. The statistic results of our approach are summarised in Table 1 which shows our approach achieves 90% and 96% success rate under 5% and 10% error metrics, respectively. The statistical results of the convergence time, the relative formation error and the orientation error over all successful runs under 10% error metrics are further shown in Figure 11. One can see that the median convergence time of our approach is about 10 s. The median relative formation error is 2.5% and the median orientation error is 0.031 rad. Compared with existing model-based formation control methods with relative formation errors of 1%–10% in general (Oh et al., 2015), the performance of our learning-based method is satisfactory.

The formation errors in online testing experiments are mainly the result of the quantisation errors introduced when converting the LIDAR sensory data to the occupancy map. In this paper, the spatial resolution of the occupancy maps is 0.1 m/pixel, i.e. any LIDAR scan data within 0.1 m cannot be reflected by the image pixels. Reducing the quantisation errors can be realised by increasing the size of the occupancy map images to increase the spatial resolution of the images. However, larger size of images requires more computational power to process the image input and compute the robot control commands in real time. The tradeoff between the image size and the spatial resolution of the occupancy map can be balanced according to practical requirements of applications.

3.4. Remark on the learning-based control method

The proposed learning-based control approach only considers a three-robot formation scenario with equal desired distance between any two robots. This is due to the reason that we assume no additional information is acquired by each robot to distinguish between its neighbouring robots. The current approach can be scaled to n-robot cases by adopting the semantic labelling methods (Dequaire, Ondrúška, Rao, Wang, & Posner, 2018) to assign distinct IDs to the neighbouring robots. Also, the mean embedding-based state representation (Hüttenrauch et al., 2019) can be used to improve the scalability of the decentralised policy learning method, which is the scope of our future work.

4. Comparison between the two methods

In this section, we compare the model-based method and learning-based method presented in Sections 2 and 3, respectively, from the perspective of fundamental principles, sensing requirements and communication requirements.

4.1. Fundamental principles

The model-based method requires the robot dynamics/kinematics model (e.g. Equations (14) and (1)) and the interaction topology (e.g. the Laplacian matrix (5)) to design the analytic control law. If the robot and environment dynamics are deterministic, the model-based approach can be implemented effectively in real time. However, in practical applications the robot model inevitably includes uncertainties and the environment could be stochastic. The dependence on system model makes the performance of the control law vulnerable to model errors and/or disturbances. Moreover, hand-engineering features could be difficult or even impossible for complex domains when the complete knowledge of the robot and environment models is not available (e.g. unknown observation probabilities, and/or non-deterministic environments).

On the contrary, the learning-based method does not rely on the knowledge of robot dynamics/kinematics model. However, it requires training data that capture the underlying control policy mapping the robot sensor observation to robot control. The control policy in the learning-based methods is represented using deep neural networks, and a supervised learning algorithm is used to train the neural networks to recover the optimal policy for formation control. The learned control policy is used as a generalisation of the decentralised control law. Compared with the model-based method which could perform

poorly under model uncertainties or stochastic environments, the learning-based method uses no information about robot model, thus is not subject to the model uncertainties. Furthermore, learning-based method is able to find robot control policies under environmental uncertainties which are embedded in the training data used to learn the control policies.

4.2. Sensing requirements

The distance-based formation control introduced in Section 2 requires each robot to be able to sense the relative positions of its neighbouring agents with respect to its own local coordinate systems. This requirement basically needs the robots to be able to access the relative position measurements, or have onboard deliberate perception modules that return the measurements explicitly.

In contrast, the learning-based method introduced in Section 3 uses the occupancy map constructed from the raw laser scan measurements as the input to the control policy. Thus, the learning-based method relieves the burden of hand-engineering the robot perception that is required by traditional model-based methods before controllers can be applied. Moreover, learning control policies that operate on raw sensor observations is worth investigating.

4.3. Communication requirements

Model-based methods normally rely on inter-robot communication to different extend, depending on the sensing capability of the robots. For example, the robots in the method discussed in Section 2 needs to communicate with the neighbouring robots for their position information to calculate the required relative positions of the neighbouring robots. Thus, communication reliability and security is critical to the successful operation of control algorithm.

One the other hand, the learning-based method in Section 3 does not need inter-robot communication but only the robot's local observation. In the future, it is of interest to learn the optimal communication to minimise the communication cost for complex multi-robot system (Sukhbaatar, Szlam, & Fergus, 2016).

4.4. Discussion

Both model-based and learning-based control methods come with their own advantages and disadvantages. Conventional model-based control methods are suitable for the problems when reliable system models are available and the system or environmental uncertainties are moderate or easy to model. As learning-based methods relax the reliance on system models for control design, such methods can be considered when the knowledge of the system dynamics or environmental uncertainties are not available. Moreover, incorporated with deep learning techniques, learning-based methods is able to train control policies to operate on non-hand-engineered features. The adaptive learning and feature representation capabilities of the learning-based methods lend themselves to complex intelligent control problems in practical applications. As the emerging method for multi-robot formation, the learning-based method merits further studies on key issues such as

advanced learning algorithms for general formation control, algorithm stability analysis, and generalisability/scalibility.

5. Conclusion

This paper presents and compares two recent works on a three-robot formation control problem using model-based and learning-based methods. In general, the model-based method provides provably stable formation control if the model of robot dynamics/kinematics is known. Analytic control laws derived by the model-based method can be implemented efficiently in real time through online sensing or communication. However, as the complexity of real-world application of robot formation control is increasingly greater, the lack of complete knowledge of the robot and environment dynamics make the problem challenging to solve by model-based methods. The learning-based method provides the solution to adaptive and intelligent control of robot formation, though it relies on data and appropriate design of training strategies to learn the desired control policy. In the future, it is desirable to develop more efficient training schemes using advanced machine learning or deep-learning methods.

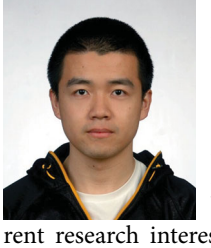
Disclosure statement

No potential conflict of interest was reported by the authors.

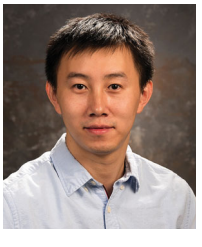
Funding

This work was supported by US National Science Foundation grants CMMI-1825709 (Understanding Pedestrian Dynamics for Seamless Human-Robot Interaction) and IIS-1838799 (SCH: INT: Collaborative Research: Aging In Place Through Enhanced Mobility and Social Connectedness: An Integrated Robot and Wearable Sensor Approach).

Notes on contributors



Chao Jiang received the B.S. degree in measuring and control technology and instrument from Chongqing University, Chongqing, China, in 2009, and the Ph.D. degree in electrical engineering from Stevens Institute of Technology, Hoboken, NJ, USA, in 2019. He worked as a research assistant in the Key Laboratory of Optoelectronic Technology and Systems (Chongqing University), Ministry of Education, China, from 2009 to 2012. He joined the Department of Electrical and Computer Engineering at the University of Wyoming, Laramie, WY, USA, in Fall 2019, where he is currently an Assistant Professor. His current research interests include autonomous robots, human–robot interaction, robotic learning, deep reinforcement learning, and multi-robot systems. He is the recipient of the Innovation & Entrepreneurship Doctoral Fellowship (2012–2016), and the Outstanding Ph.D. Dissertation Award in Electrical Engineering (2019) at Stevens Institute of Technology.



Zhuo Chen received his B.E. in automation from Zhengzhou University, Zhengzhou, China, in 2013, and M.S. in Control Science and Engineering from Harbin Institute of Technology, Harbin, China, in 2015. He is a Ph.D. candidate in electrical engineering at Stevens Institute of Technology, Hoboken, USA. His main research interests include autonomous mobile robotics and human–robot interaction.

Yi Guo received her B.S. and M.Sss. from Xi'an University of Technology, Xi'an, China, in 1992 and 1995, respectively, and Ph.D. from the University of Sydney, Australia, in 1999, all in electrical engineering. She was a Postdoctoral Research Fellow at Oak Ridge National Laboratory from 2000 to 2002, and a Visiting Assistant Professor at the University of Central Florida, Orlando, FL, USA, from 2002 to 2005. She joined the Department of Electrical and Computer Engineering, Stevens Institute of Technology, Hoboken, NJ, USA, in 2005, where she is currently a Professor. Her main research interests include autonomous mobile robotics, distributed sensor networks, and nonlinear control systems. She has published more than 100 peer-reviewed journal and conference papers, authored the book entitled "Distributed Cooperative Control: Emerging Applications" (John Wiley & Sons 2017), and edited a book on micro/nano-robotics for biomedical applications (Springer 2013). She currently serves on the editorial boards of several journals including IEEE Robotics and Automation Magazine and IEEE/ASME Transactions on Mechatronics. She served in Organizing Committees of IEEE International Conference on Robotics and Automation (2015, 2014, 2008, 2006).

References

Amato, C., Chowdhary, G., Geramifard, A., Üre, N. K., & Kochenderfer, M. J. (2013). Decentralized control of partially observable Markov decision processes. *IEEE conference on decision and control* (pp. 2398–2405).

Amato, C., Konidaris, G., Anders, A., Cruz, G., How, J. P., & Kaelbling, L. P. (2016). Policy search for multi-robot coordination under uncertainty. *International Journal of Robotics Research*, *35*(14), 1760–1778.

Aykin, C., Knopp, M., & Diepold, K. (2018). Deep Reinforcement Learning for Formation Control. *IEEE international symposium on robot and human interactive communication* (pp. 1–5).

Barrett, S., & Stone, P. (2015). Cooperating with unknown teammates in complex domains: A robot soccer case study of ad hoc teamwork. *AAAI conference on artificial intelligence* (pp. 2010–2016).

Chen, Z., Jiang, C., & Guo, Y. (2019). Distance-based formation control of a three-robot system. *Chinese control and decision conference* (pp. 5574–5580).

Dequaire, J., Ondrúška, P., Rao, D., Wang, D., & Posner, I. (2018). Deep tracking in the wild: End-to-end tracking using recurrent neural networks. *The International Journal of Robotics Research*, *37*(4-5), 492–512.

Dimarogonas, D. V., & Johansson, K. H. (2008). On the stability of distance-based formation control. *IEEE conference on decision and control* (pp. 1200–1205).

Foerster, J., Nardelli, N., Farquhar, G., Afouras, T., Torr, P. H., Kohli, P., & Whiteson, S. (2017). Stabilising experience replay for deep multi-agent reinforcement learning. *Proceedings of international conference on machine learning* (pp. 1146–1155).

Guo, Y. (2017). *Distributed cooperative control: Emerging applications*. Hoboken, NJ: John Wiley & Sons.

Gustavi, T., & Hu, X. (2008). Observer-based leader-following formation control using onboard sensor information. *IEEE Transactions on Robotics*, *24*(6), 1457–1462.

Hüttenrauch, M., Adrian, S., & Neumann, G. (2019). Deep reinforcement learning for swarm systems. *Journal of Machine Learning Research*, *20*(54), 1–31.

Jiang, C., Chen, Z., & Guo, Y. (2019). Learning decentralised control policies for multi-robot formation. *IEEE/ASME international conference on advanced intelligent mechatronics* (pp. 758–765).

Kingma, D. P., & Ba, J. (2014). *Adam: A method for stochastic optimization*. Preprint arXiv:1412.6980.

Li, S., Kong, R., & Guo, Y. (2014). Cooperative distributed source seeking by multiple robots: Algorithms and experiments. *IEEE/ASME Transactions on Mechatronics*, *19*(6), 1810–1820.

Liang, X., Wang, H., Liu, Y. H., Chen, W., & Liu, T. (2017). Formation control of nonholonomic mobile robots without position and velocity measurements. *IEEE Transactions on Robotics*, *34*(2), 434–446.

Liu, M., Amato, C., Anesta, E. P., Griffith, J. D., & How, J. P. (2016). Learning for decentralised control of multiagent systems in large, partially-observable stochastic environments. *AAAI conference on artificial intelligence* (pp. 2523–2529).

Mnih, V., Kavukcuoglu, K., Silver, D., Rusu, A. A., Veness, J., Bellemare, M. G., ... Hassabis, D. (2015). Human-level control through deep reinforcement learning. *Nature*, *518*(7540), 529–533.

Oh, K. K., & Ahn, H. S. (2011). Formation control of mobile agents based on inter-agent distance dynamics. *Automatica*, *47*(10), 2306–2312.

Oh, K. K., Park, M. C., & Ahn, H. S. (2015). A survey of multi-agent formation control. *Automatica*, *53*, 424–440.

Qu, Z. (2009). *Cooperative control of dynamical systems: Applications to autonomous vehicles*. London: Springer Science & Business Media.

Rausch, V., Hansen, A., Solowjow, E., Liu, C., Kreuzer, E., & Hedrick, J. K. (2017). Learning a deep neural net policy for end-to-end control of autonomous vehicles. *American control conference* (pp. 4914–4919).

Rohmer, E., Singh, S. P., & Freese, M. (2013). V-REP: A versatile and scalable robot simulation framework. *IEEE/RSJ international conference on intelligent robots and systems* (pp. 1321–1326).

Sukhbaatar, S., Szlam, A., & Fergus, R. (2016). Learning multiagent communication with backpropagation. *Advances in neural information processing systems* (pp. 2244–2252).

Tuyls, K., & Weiss, G. (2012). Multiagent learning: Basics, challenges, and prospects. *AI Magazine*, *33*(3), 41–52.

Vidal, R., Shakernia, O., & Sastry, S. (2004). Following the flock [formation control]. *IEEE Robotics & Automation Magazine*, *11*(4), 14–20.

Wang, H., Guo, D., Liang, X., Chen, W., Hu, G., & Leang, K. K. (2016). Adaptive vision-based leader–follower formation control of mobile robots. *IEEE Transactions on Industrial Electronics*, *64*(4), 2893–2902.

Wulfmeier, M., Rao, D., Wang, D. Z., Ondruska, P., & Posner, I. (2017). Large-scale cost function learning for path planning using deep inverse reinforcement learning. *The International Journal of Robotics Research*, *36*(10), 1073–1087.