automation

MDPI

Article

Accelerated Transfer Learning for Cooperative Transportation Formation Change via SDPA-MAPPO (Scaled Dot Product Attention-Multi-Agent Proximal Policy Optimization)

Almira Budiyanto [1], Keisuke Azetsu [1] and Nobutomo Matsunaga [2,*]

[1] Graduate School of Science and Technology, Kumamoto University, Kumamoto 860-8555, Japan
[2] Faculty of Advanced Science and Technology, Kumamoto University, Kumamoto 860-8555, Japan
* Correspondence: matunaga@cs.kumamoto-u.ac.jp

Abstract: A method for cooperative transportation, which required formation change in a traveling environment, is gaining interest. Deep reinforcement learning is used in formation changes for multi-robot cases. The MADDPG (Multi-Agent Deep Deterministic Policy Gradient) method is popularly used for recognized environments. On the other hand, re-learning may be required in unrecognized circumstances by using the MADDPG method. Although the development of MADDPG using model-based learning and imitation learning has been applied to reduce learning time, it is unclear how the learning results are transferred when the number of robots changes. For example, in the GASIL-MADDPG (Generative adversarial self-imitation learning and Multi-agent Deep Deterministic Policy Gradient) method, how the results of three robot training can be transferred to the four robots' neural networks is uncertain. Nowadays, Scaled Dot Product Attention (SDPA) has attracted attention and is highly impactful for its speed and accuracy in natural language processing. When transfer learning is combined with fast computation, the efficiency of edge-level re-learning is improved. This paper proposes a formation change algorithm that allows easy and fast multi-robot knowledge transfer using SDPA combined with MAPPO (Multi-Agent Proximal Policy Optimization), compared to other methods. This algorithm applies SDPA to multi-robot formation learning and performs fast learning by transferring the acquired knowledge of formation changes to a certain number of robots. The proposed algorithm is verified by simulating the robot formation change and was able to achieve dramatic high-speed learning capabilities. The proposed SDPA-MAPPO (Scaled Dot Product Attention-Multi-Agent Proximal Policy Optimization) learned 20.83 times faster than the Deep Dyna-Q method. Furthermore, using transfer learning from a three-robot to five-robot case, the method shows that the learning time can be reduced by about 56.57 percent. A scenario of three-robot to five-robot is chosen based on the number of robots often used in cooperative robots.

Keywords: multi-robots; formation change; Scaled Dot Product Attention; transfer learning

check for updates

Citation: Budiyanto, A.; Azetsu, K.; Matsunaga, N. Accelerated Transfer Learning for Cooperative Transportation Formation Change via SDPA-MAPPO (Scaled Dot Product Attention-Multi-Agent Proximal Policy Optimization). *Automation* **2024**, *5*, 597–612. https://doi.org/10.3390/automation5040034

Academic Editor: Duc Truong Pham

Received: 13 September 2024
Revised: 22 November 2024
Accepted: 22 November 2024
Published: 27 November 2024

1. Introduction

Multi-agent systems have strengths in many areas such as research, robotics, autonomous driving, warehouse systems, and game playing. Especially in robotics, multi-agent systems require cooperative behavior to coordinate from one agent to another in order to reach a specific task. On the path to completing a specific task, formation change and control are essential in multi-agent systems. Various environmental conditions require different formations of the multiple agents, including obstacle avoidance, size and shape tracks, and various sizes and shapes of object transportation [1–7].

One example of multi-agent application is cooperative transport, in which the formation change of multiple robots is adjusted based on the environment. Nowadays, this concept is receiving increasing attention [8]. Formation changes can be made at any arbitrary point on the trajectory, and deep reinforcement learning is used. In multi-agent systems, reinforcement learning is making rapid advancements in autonomous learning. Reinforcement learning is a framework that effectively solves sequential decision-making tasks by allowing a learning agent to interact with the environment and learn from mistakes by using trial and error to improve performance. Challenges of reinforcement learning in multi-agent systems include the long learning process; the gained knowledge is based on the circumstance and limitations in the ability of robots to store knowledge [2,3].

Since the formation change requires a long learning time, many speed-up methods based on Multi-Agent Deep Deterministic Policy Gradient (MADDPG) [9] have been considered. We have also applied a model-based learning method, namely Deep Dyna-Q [10], which estimates the state based on the states collected by the agent, and imitation learning [11], which imitates the trajectory to approximate the state of an expert. Both methods have led to improved learning times compared to MADDPG. Deep Dyna-Q effectively explored formation control through both simulations and actual experiments. In the simulations, the rate of successful formation changes was influenced by factors such as the discount rate, learning rate, the number of collisions among agents, and collisions between agents and transport objects. In actual experiments, the implementation of a Model Error Compensator (MEC) reduced movement errors in the robots. As a result, the robots were able to achieve successful formation changes.

Transfer learning (TL) is developed in reinforcement learning to overcome the long learning time and accelerate the learning process. Transfer learning [12] is the process of sharing information from one agent to another agent. By utilizing pertinent prior knowledge, transferable knowledge from a source task speeds up learning and improves the quality of solutions in target tasks. Various learning strategies and communication protocols have been implemented to overcome this issue and to enhance stability, training, and observability [13–15].

In transfer learning, single-agent transfer learning has shown positive transfer, but multi-agent transfer learning is still developing. Conversely, irresponsible knowledge reuse could result in a detrimental transfer. Therefore, it is essential to develop flexible, safe, and robust methods for reusing knowledge in multi-agent applications. Transfer learning should take into account autonomous, cooperative learners and stochastic environments where joint rewards and local observations are used as feedback. This approach is applicable and realistic for a wide range of multi-agent domains. The popular transfer learning strategies applied in reinforcement learning are imitation learning and policy transfer.

Well-known examples of imitation learning include behavioral cloning, which imitates behavioral trajectories precisely, and GAIL [16], which estimates rewards from expert behavioral trajectories and uses them for reinforcement learning. There is also self-imitation learning, which uses past good experiences as the behavioral trajectories of experts [4,5,17]. However, when the number of robots used or the transport environment changes, the input vector changes, making it difficult to transfer the learning results.

In recent years, Scaled Dot Product Attention (SDPA) [18], which has high accuracy and speed, has gained attention in natural language processing. The SDPA algorithm is fast because it can calculate input in parallel and provide accurate translations by understanding the context while grasping the anaphoric relationships between words. Therefore, we are considering using this SDPA scheme to learn robot formation. SDPA can learn quickly when the attention mechanism, which focuses on the weights of features, can be applied to reinforcement learning, Additionally, we can expect more robust knowledge transfer against environmental changes, such as in the number of robots.

In this paper, a formation change algorithm is proposed, which allows easy and fast multi-robot knowledge transfer by using SDPA combined with MAPPO (Multi-Agent Proximal Policy Optimization). The proposed algorithm is verified by simulating the robot formation change. The contribution of this paper is that it not only achieved fast learning by using the SDPA algorithm but also made it easy to transfer the learning results from a neural network, which was difficult with reinforcement learning.

2. Formation Change Using SDPA with Transfer Learning

Conventional search algorithms for formation changes have difficulty transferring knowledge when the number of robots changes during transport. Therefore, we will focus on the SDPA algorithm, which considers the weights of features. With this scheme, we can expect a learning algorithm that can respond to changes in the input vector.

2.1. SDPA-MAPPO Algorithm

MAPPO Algorithm

Once the action is determined, all uncertainties are determined by the policy π based on the Markov decision process so that the expected value can be obtained. The policy gradient method learns a policy that maximizes this expected value. Then, the policy gradient method learns to maximize the objective function. The policy gradient method has the problem of instability since it is difficult to determine an appropriate update size, and large updates may cause the policy to deteriorate. Therefore, TRPO (Trust Region Policy Optimization) [19] has been proposed, which trains by restricting the update width so that the output of the policy network does not change too much before and after the update. TRPO is an excellent algorithm, but it has problems such as complex calculations and difficulty sharing parameters. In contrast, PPO [20] suppresses the updated width by clipping instead of the constraint conditions adopted in TRPO. In this paper, we use PPO to reduce the update width. The clipping conditions are as follows in Equation (1) [20]:

$$L^{CLIP}(\theta) = \hat{E}[min(r(\theta)\hat{A},\ clip(r(\theta), 1-e, 1+e)\hat{A})] \tag{1}$$

where $\hat{E}$ is the estimate, r(θ) is the previously mentioned policy, θ is the policy parameter, e is the hyperparameter, and $\hat{A}$ is the advantage function. PPO is a method suitable for learning a single agent. However, multi-agent learning cannot achieve high performance [21]. In this paper, we use Multi-Agent Proximal Policy Optimization (MAPPO), which is an improved version of PPO for MAS (Multi-agent system), as a deep reinforcement learning algorithm [22]. Equations (2) and (3) [22] are value functions for agent i and agent j.

$$V^i_{joint} = \epsilon V^i(O^i_t) + (1-\epsilon)V^j(O^j_t) \tag{2}$$

$$V^j_{joint} = \epsilon V^j(O^j_t) + (1-\epsilon)V^i(O^i_t) \tag{3}$$

The O^i, V^i, and V^i_{joint} are the observed value, the value function, and the value function combined by weighted average, respectively. V^i_{joint} contains its observation information and the observation information of other agents.

2.2. SDPA Algorithm

The input vector in each robot in the simulation used in this experiment is defined as shown in Figure 1. $P(x, y)$ is the robot's coordinates, $V(x, y)$ is the robot's velocity, $G(x, y)$ is the relative distance between the goals, and $R(x, y)$ is the relative distance between the robots. These elements make up the input dimensions of the neural network (NN). For the case of three robots, the input dimension would consist of 7-dimension inputs, which include $P(x, y)$, $V(x, y)$, $G_1(x, y)$, $G_2(x, y)$, $G_3(x, y)$, $R_1(x, y)$, and $R_2(x, y)$ for each robot.

Figure 2 illustrates an overview of the SDPA-MAPPO approach for the three-robot case. The overview shows the input process, embedding, SDPA process, and deep reinforcement learning.

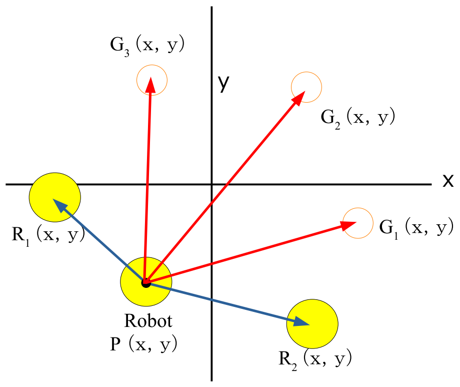

Figure 1. Definition of input vector [9].

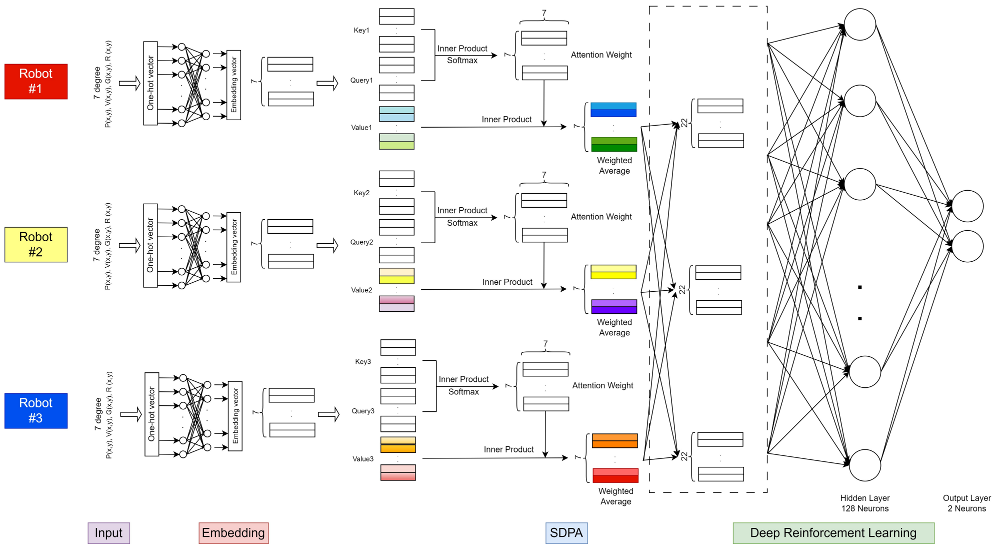

Figure 2. Overview of SDPA-MAPPO approach.

First, the input process is discussed. In a neural network (NN), the input is typically represented by a vector. One simple way to represent the input is by using a one-hot vector. A one-hot vector is a vector in which one component is 1, and the rest are all 0. Although it can be expressed concisely, the dimension of the vector depends on the dimension of the input. Then, in the embedding block in Figure 2, the fixed-size vector is created. A fixed-size (embedding dimension) vector is formed using a distributed representation. The distributed representation is shown in the following Figure 3.

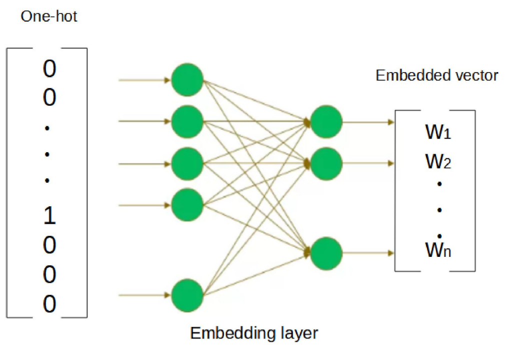

Figure 3. Distributed representation.

Distributed representation is a linear mapping of a one-hot vector onto a lower-order vector space. The transformation matrix $W^T_{embedding}$ is expressed as (size of the embedding dimension) × (size of the input dimension). By using the transformation matrix for the one-hot vector, we can calculate the vector using the embedding dimension size.

In Figure 2, the SDPA block performs several functions: multiplying embeddings, calculating similarity and importance, and generating a vector to focus attention on. Utilizing this input in a deep reinforcement learning NN enables the creation of more complex combinations of features. In this way, SDPA plays an auxiliary role in deep reinforcement learning NNs, so learning is expected to be faster.

Attention can be described as follows: Q, K, and V represent Query, Key, and Value, where $\sqrt{d_K}$ is the number of dimensions of Key, as shown in Equation (4) [18].

$$Attention(Q, K, V) = softmax(\frac{Q \times K^T}{\sqrt{d_K}})V \tag{4}$$

First, Key, Query, and Value are created based on the vectors generated by distributed representation. Next, the Q matrix and the transposed matrix of the Key matrix are multiplied. The matrix obtained by the inner product has a size of (input dimension) × (input dimension). The matrix is then weighted with the activation function. The weighted matrix is used as attention weight to calculate the similarity of the Values. Figure 4 depicts a schematic diagram of the derivation of the attention weight.

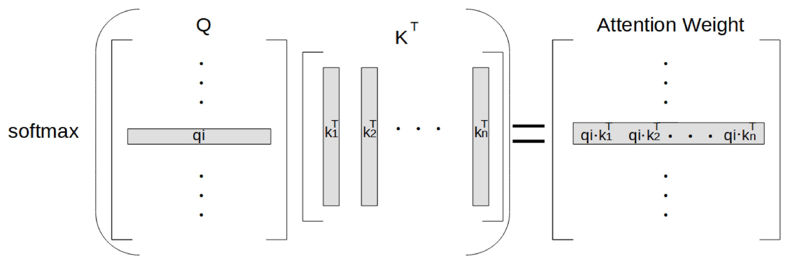

Figure 4. Derivation of attention weight.

As an example, Figure 5 displays the attention weight when learning with three robots. Akshat et al. [23] used curriculum learning, but in this paper, a more complex formation change problem can be realized using deep reinforcement learning with SDPA.

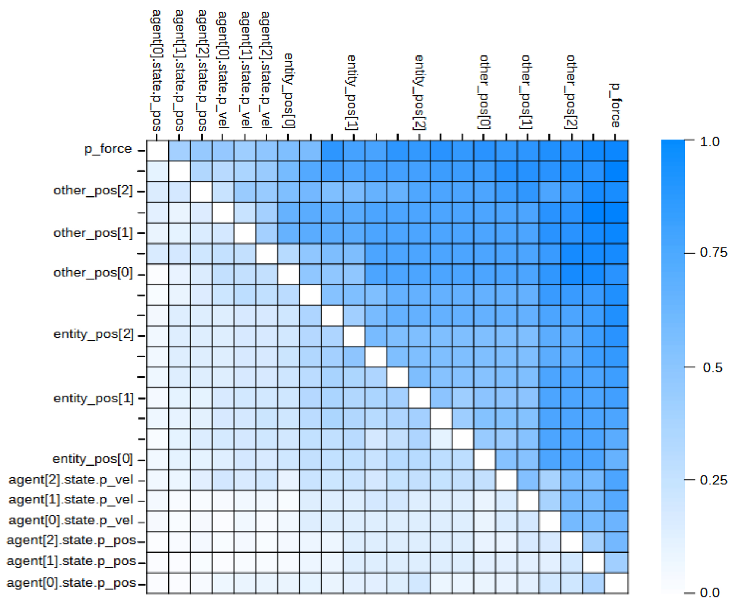

Figure 5. Visualization of attention weight.

The weighted average of the value is taken using the obtained attention weight. Figure 6 shows the process for obtaining the weighted average.

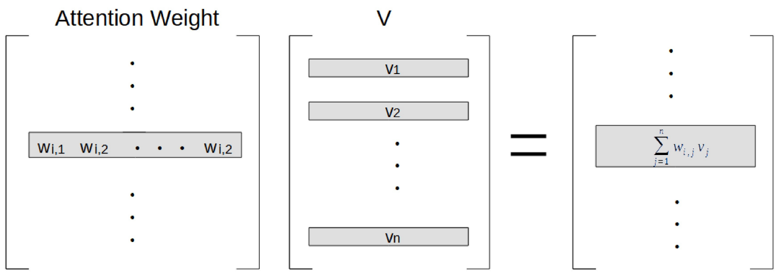

Figure 6. The weighted average of the value.

The weighted average, resulting from the SDPA process, is used as input for deep reinforcement learning. The feature of this algorithm is that the reinforcement learning input for each agent is modified by the SPDA weights. When the number of robots is increased, a new input layer is added for new learning. So, in the new reinforcement learning, the input layer weights learned by three robots can be transferred to new reinforcement learning.

The dashed line in Figure 2 represents the input position from deep reinforcement learning. In Figure 2, the neural network (NN) configuration is explained. The middle layer consists of one layer, and the number of neurons is 128. The output layer represents the target acceleration vector $a(x, y)$, with the number of neurons in the output layer being 2. The activation function used in the NN is the ReLU function, which produces output values between 0 and 1.

3. Learning Speed of SDPA

3.1. Transition in Reward Value During Learning

The problem dealt with in this paper is a formation change problem where the robot moves from the right transport position to the left transport position (black), as shown in Figure 7 [24]. The robot reaches the new transport position (black) within the given time without colliding with the object or other robots. The robot's starting positions (R, G, B) are known, but the robots can then move to any of the three positions for transport. The area of the experimental environment is (1 × 1), the robot's diameters are 0.08, and the target point diameters are 0.05.

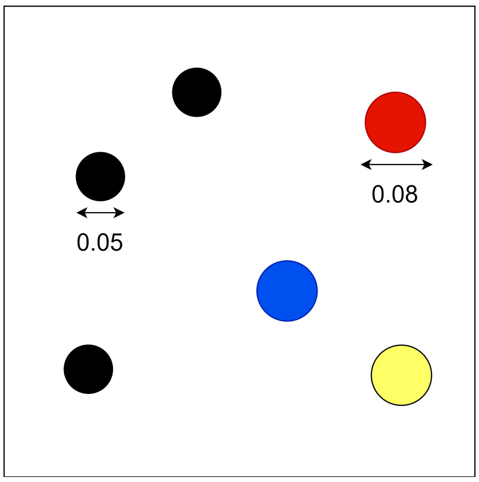

Figure 7. Environment without the obstacles in simulation: The starting positions of the robots are indicated by the yellow, red, and blue circles, while the target positions are represented by the black circles.

In the following analysis, we present the reward values of formation learning from the initial state using SDPA-MAPPO. Figures 8–10 show the changes in reward values when there are three, four, and five robots, respectively. The vertical axis of the graph represents the average reward value, while the horizontal axis indicates the number of episodes.

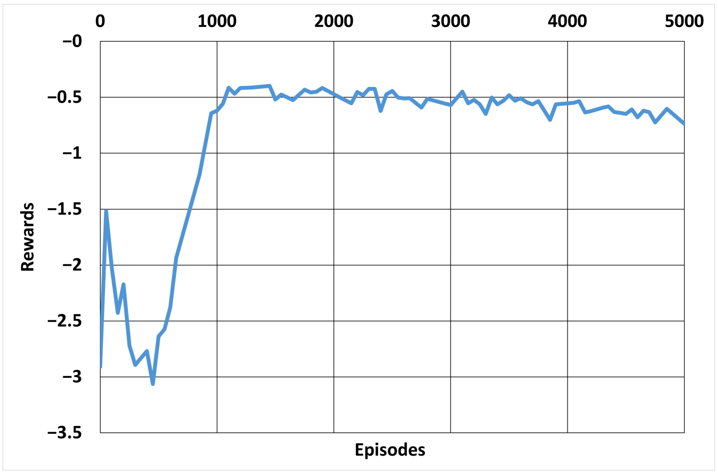

Figure 8. Transition of reward on SDPA-MAPPO(3).

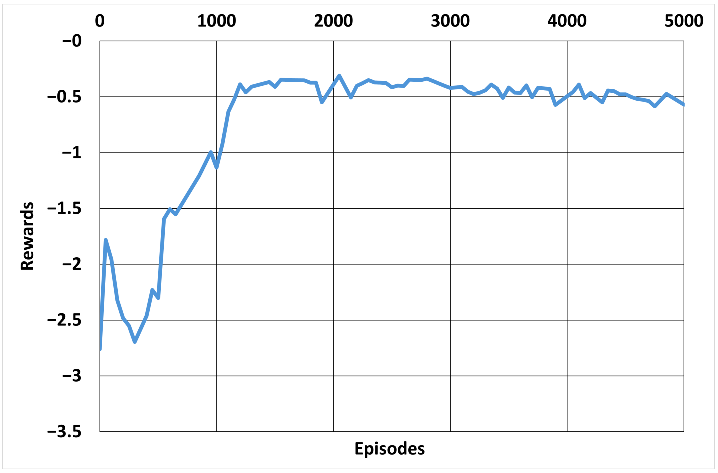

Figure 9. Transition of reward on SDPA-MAPPO(4).

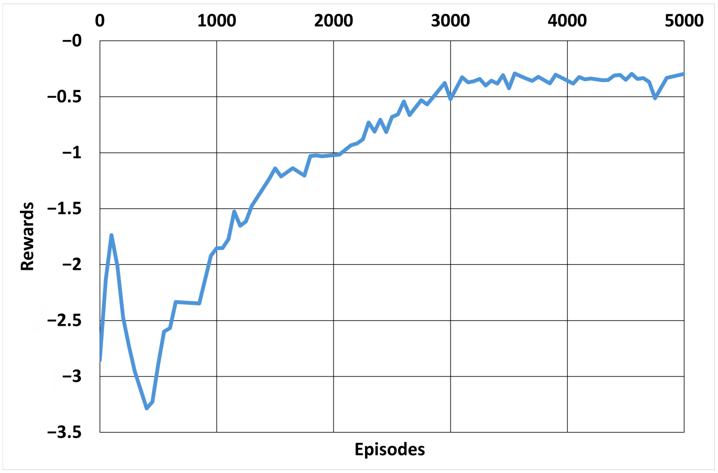

Figure 10. Transition of reward on SDPA-MAPPO(5).

Figure 8, depicts the rapid increase in reward value from approximately –3.4 to about –0.54 over 1000 episodes, followed by stabilization and convergence around –0.36, indicating completion of learning at 1200 episodes. Meanwhile, in the case of the transition reward of SDPA-MAPPO in four robots, the reward value increases rapidly from about –2.7 to about –0.4 in 1500 episodes. It can be seen in Figure 9 that the reward is stabilized and converged around –0.36 at 2000 episodes. For five robot simulations, as shown in Figure 10, the reward converged around –0.34, so the learning process is completed at 3200 episodes.

3.2. Evaluation of Learning

In each episode, we calculated the difference in distance between the robot's final position and the target point. The final distance shows the distance average of all robots' positions to the target point. A smaller final distance indicates that the robot is closer to the target, showing better system performance.

Since the target point size is 0.05, the goal condition requires that the errors in the distance between the robot and the target point are within 0.05. Figures 11–13 are the final distances of SDPA-MPPO for three, four, and five robots, respectively. All robots have successfully reached the goal, as shown in these figures. In Figure 11, the initial distance between the robot and the target point is 9 mm, which then reached 0.4 mm final distance in about 1000 episodes for three robots. For four robots and five robots, both final distances are about 0.05 and reached in 1500 and 3200 episodes, respectively.

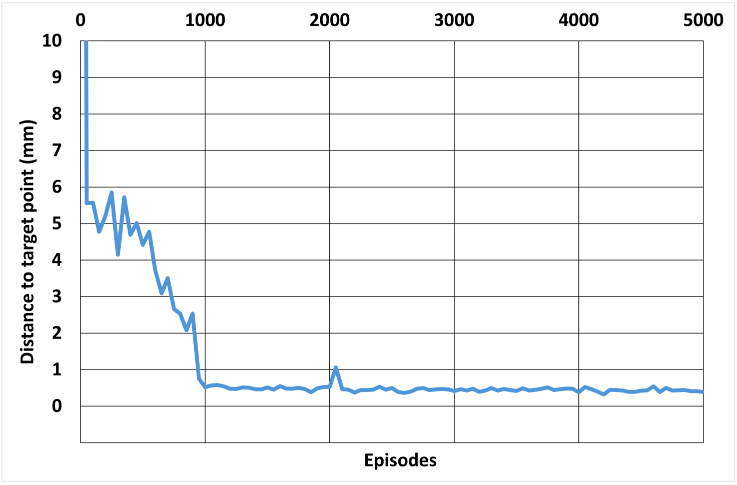

Figure 11. Final distance in three robots.

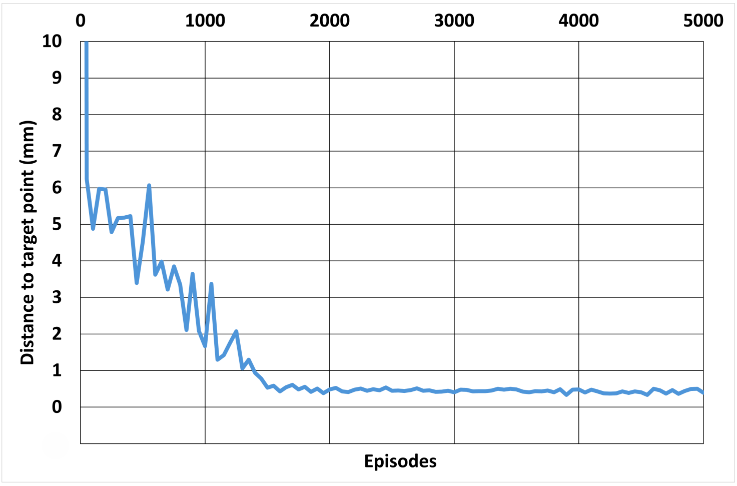

Figure 12. Final distance in four robots.

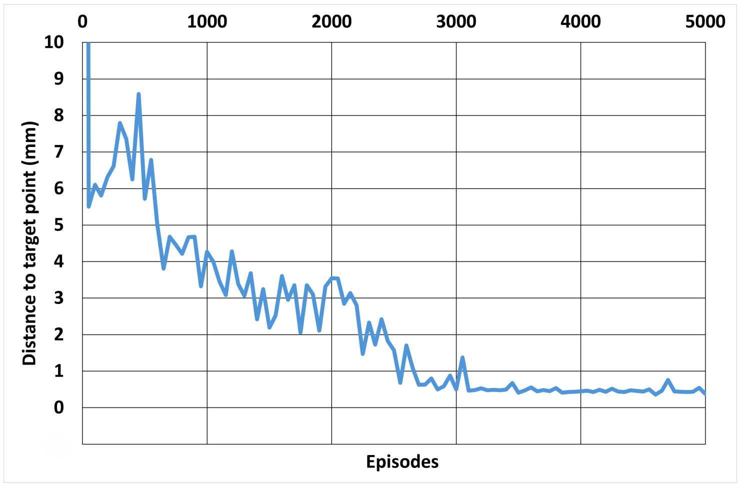

Figure 13. Final distance in five robots.

Table 1 compares the learning performance of the proposed method using SDPA-MAPPO and previous studies conducted under similar conditions. Learning convergence, measured in the number of episodes, indicates convergence speed, with smaller numbers representing faster convergence. First, we compare the learning end times from the three random states with SDPA-MAPPO(3) (address as SDPA(3) on the Table 1), GASIL-MADDPG(3), and Deep Dyna-Q(3). For formation learning involving three robots, SDPA(3) is about 20.83 times faster than Deep Dyna-Q. It is noted the learning time increases as the number of robots increases in SDPA-MAPPO. For example, the learning time with five units is 2.6 times longer than that with three units.

Table 1. Number of episodes for learning convergence.

	SDPA(3)	SDPA(4)	SDPA(5)	GASIL-MADDPG(3)	Deep Dyna-Q(3)
Learning convergence (128 steps)	1200	1500	3200	25,000	40,000

3.3. Result of Formation Change Using SDPA

Figures 14 and 15 show an example of three robots changing their transport formation. Figure 14 illustrates the formation in simulation with three robots and Figure 15 illustrates the formation change for four robots [25]. The solid red, blue, yellow, green, and purple lines represent the trajectories of the robots.

There are four split pictures in Figures 14 and 15. In Figure 14-1, the green, red, and blue squares show the initial position of robots. All the robots started to move to the target positions in Figure 14-2 and Figure 14-3. We can see that robot 1 is maneuvering to avoid the transportation target. Finally, in Figure 14-4, all the robots reached the target positions. In Figure 15-1, all the robots are in the starting position and start to move from Figure 15-2. The green and blue robots rapidly moved to the nearest position and reached the target position in Figure 15-3. For longer trajectories, the yellow and red robots reached the target position in Figure 15-4. Each robot is finally adjacent to the target, and the formation change is achieved.

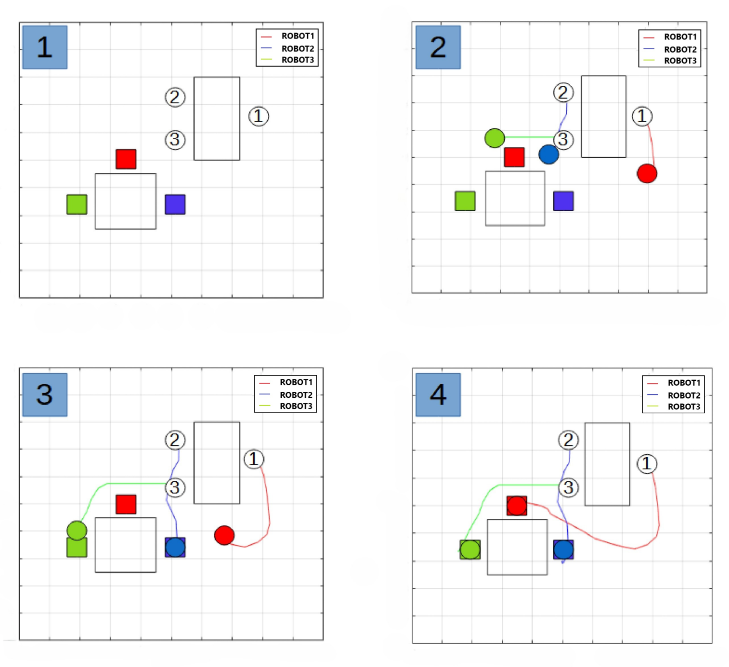

Figure 14. Formation change for three robots using SDPA: The initial positions of the robots are represented by numbered circles 1 to 3, while their target positions are indicated by the red, blue, and green squares.

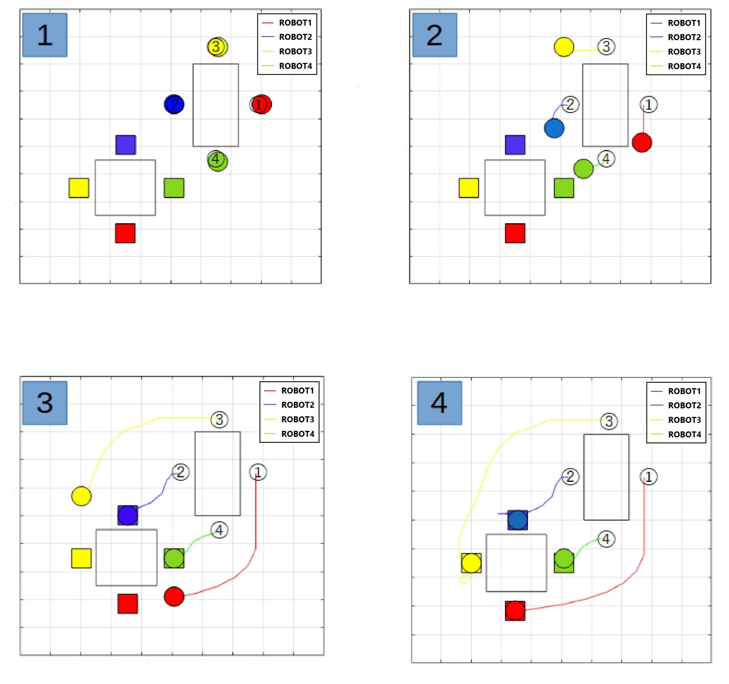

Figure 15. Formation change for four robots using SDPA: The initial positions of the robots are represented by numbered circles 1 to 4, while their target positions are indicated by the red, blue, yellow, and green squares.

4. Transfer Learning of SDPA-MPPO in Case of Number Change in Robots

4.1. Existing Transfer Learning

In this section, the existing transfer learning method is summarized. Much of the existing transfer learning has been developed, including action advising, human-focused transfer, reward shaping and heuristics, and learning from demonstrations. Additionally, there are popular inter-agent transfer methods, namely imitation, curriculum learning, inverse reinforcement learning, transfer in deep reinforcement learning, and scaling learning to complex problems [15]. These methods were developed in order to accelerate the learning time.

One of the recognized transfer learning methods is imitation learning. Like a human who imitates others by observing their behavior, transfer learning in multi-agent also relies on imitation. This method is as simple as learning from a demonstration or from an expert and applying it to the agent. In this section, the Deep Dyna-Q and GASIL-MADDPG method is discussed in comparison to the proposed method in the paper and the non-transfer learning method.

Before SDPA-MAPPO, one of the popular learning methods for cooperative transportation is Deep Dyna-Q. Deep Dyna-Q is reinforcement learning that uses model-based learning. The result of the investigation of Deep Dyna-Q is the system has high-speed learning and excellent performance results. This method was also investigated in actual experiments. On the other hand, for trajectory learning in a large workspace, the Deep Dyna-Q method is not fast enough.

The updated research after Deep Dyna-Q is GASIL-MADDPG. Besides using based learning, the GASIL-MADDPG combines based learning with imitation learning. The GASIL-MADDPG extracts past successful experiences, and as a result, the learning time is speeded up. Even though the GASIL-MADDPG has sped up the learning time, the method still has inflexibility when the number of robots changes. The SDPA-MAPPO tries to solve the problem of inflexibility when the number of robots changes, which is the transfer learning method. So, these two methods, as rationality, are representative comparisons for each kind of method to compare to the SDPA-MAPPO method, which is the transfer learning method. Regarding fairness, the environmental system is built as closely as possible for every method. As shown in Figure 16, the results of Deep Dyna-Q and GASIL-MADDPG were tuned under the same conditions. So that the results are representative of the effectiveness of each comparison method.

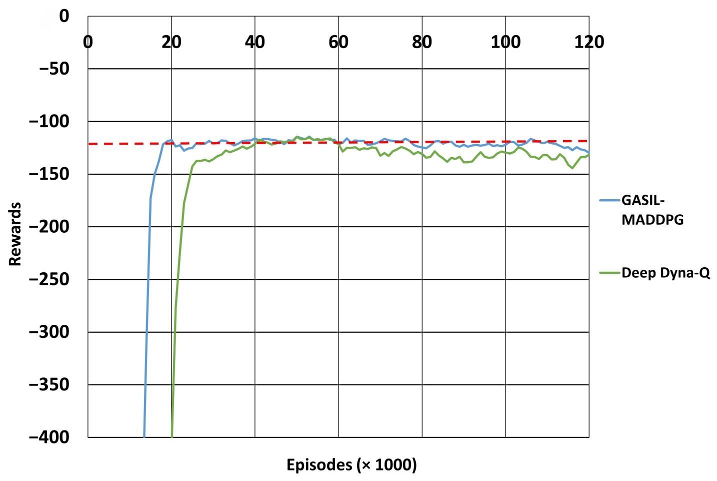

Figure 16. Transition Reward on GASIL-MADDPG vs. non-transfer learning method [26].

GASIL-MADDPG [26] is a method that combines imitation learning and model-based learning at the same time and then applies them to MADDPG. GASIL stands for Generative adversarial self-imitation learning, a combination of self-imitation learning and GAIL (Generative adversarial imitation learning). In self-imitation learning, the method uses past successful experiences as an expert's behavioral reference. In GAIL, rewards are calculated based on the experts' trajectory.

The transition rewards of GASIL-MADDPG are compared to the non-transfer learning method, Deep Dyna-Q, in Figure 16. These transition rewards are based on the environment of three robots in Figure 7. The rewards were calculated based on the reward function. Compared to the non-transfer learning methods, GASIL-MADDPG had the fastest learning speed. The GASIL-MADDPG method had a speed learning of about 25,000 episodes. In comparison, other methods show a speed learning of about 40,000 episodes for Deep Dyna-Q methods.

For converged rewards, GASIL-MADDPG has achieved −125 rewards, while Deep Dyna-Q is below −125. A more positive reward indicates better performance for the agent. A more positive reward indicates better performance of the agent, a closer distance between the goal and the agents, and fewer collisions.

Under this circumstance, the GASIL-MADDPG as existing learning showed good performance in the learning speed process. On the other hand, the number change in robots in applications is unavoidable. When the number of robots changes, the training results from three robots cannot be transferred as previous successful experiences. Therefore, alternative methods are required to address this situation.

4.2. Reward Value of Transfer Learning

We show that the proposed SPDA-MAPPO is an algorithm that can flexibly transfer learning results even if the number of NN inputs changes. First, we will train the NNs with SPDA-MAPPO using three robots. As mentioned above, transfer learning becomes difficult with conventional methods because the NN input changes when the number of robots differs.

By using the input of NNs generated from SPDA-MAPPO, it is possible to easily transfer the weights of three units. Next, we use SPDA-MAPPO to learn from the weights of the NNs learned by the three robots and the initial weights of the newly added NNs. The reward values of four robots are shown in Figure 17, and the reward values of five robots are shown in Figure 18. From this result, it can be seen that the learning results of the three machines are transferred.

The comparison of transition rewards between non-transfer learning and transfer learning for four robots can be seen in Figures 9 and 17. Without transfer learning, the system converged in about 2000 episodes, compared with transfer learning, which converged from fewer than 1000 episodes, specifically in 650 episodes. This comparison showed that with transfer learning, the learning process time is significantly reduced. This is because the transfer learning is extracted from the last successful experience as a reference, allowing transition reward with transfer learning to converge in a speedy process. In the same case with four robots transfer learning, the five robots transfer learning converged from 850 episodes. The comparison for five robot transition rewards without and with transfer learning can be seen in Figures 10 and 18.

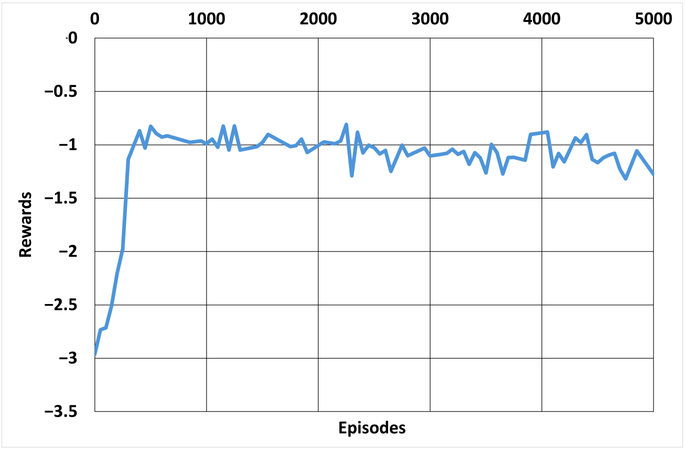

Figure 17. Transition of reward on transfer learning (4).

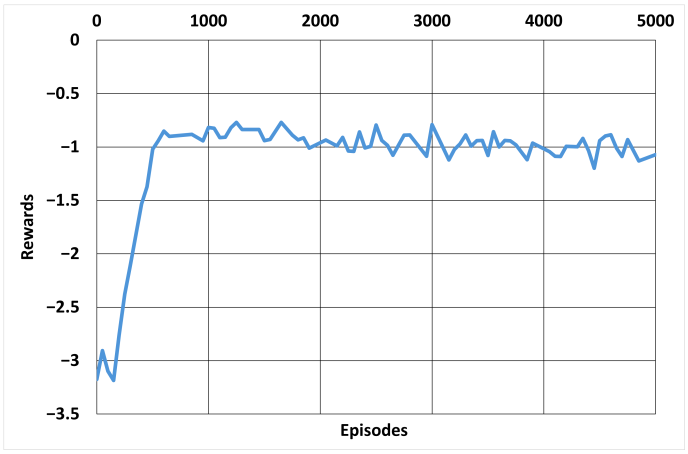

Figure 18. Transition of reward on transfer learning (5).

4.3. *Evaluation of Transfer Learning*

Figures 19 and 20 are the results of transfer learning of the learning results of three robots to four and five robots, respectively. Based on Figures 19 and 20, transfer learning reaches the goal faster than without transfer learning. Compared to Figure 12 for four robots, Figure 19 reached the converged final distance from about 650 episodes. The same result about speedy convergence for five robots compared to Figures 13 and 20 with transfer learning, the distance converged from about 850 episodes.

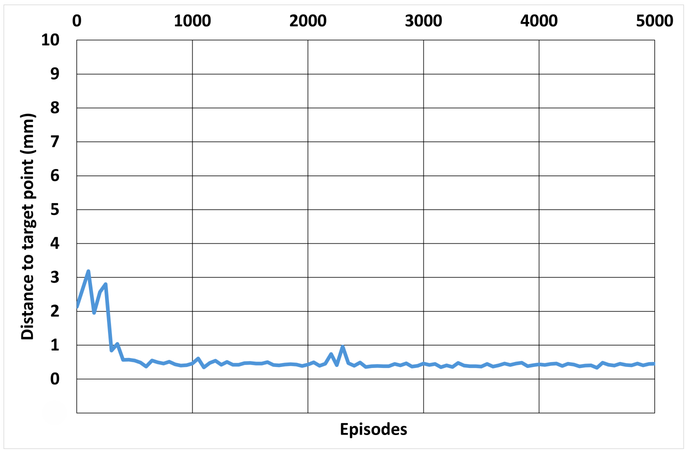

Figure 19. Final distance in four robots based on transfer learning.

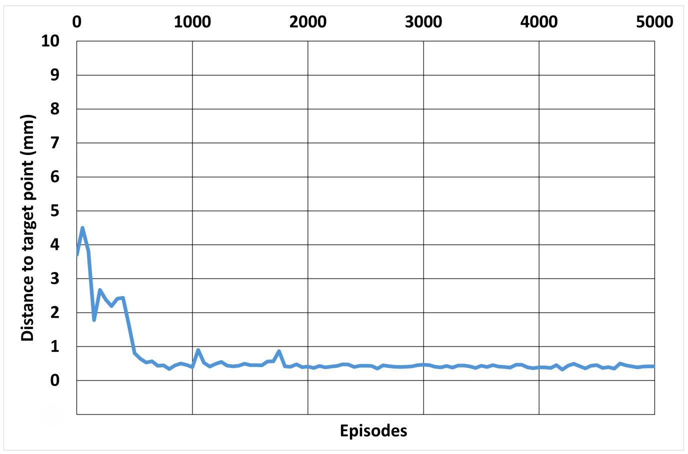

Figure 20. Final distance in five robots based on transfer learning.

Table 2 compares the case without and with transfer learning. The learning speed of conventional reinforcement learning has not been improved by using transfer learning until now. The reason is that the input vector changes when the number of units changes, and the weights of the conventional NN cannot be used. By using the proposed SDPA-MAPPO, the learning time of SDPA can be further reduced by 56.67% compared to the previously mentioned learning time.

Table 2. Comparison results of transfer learning.

	Conventional Learning	**Transfer Learning**
Learning convergence (4 robots)	1500	650
Learning convergence (5 robots)	3200	850

4.4. Result of Formation Change with Transfer Learning

Here is an example of transfer learning using SDPA-MAPPO. Figure 21 shows the results of training four transport robots using transfer learning from the results of three transport robots. In Figure 21, the added robot 4 moves to the goal while mainly avoiding the object to be transported.

Figure 22 shows the results of learning five transport robots using transfer learning from the results of three robots. From Figure 22, robot 1 and robot 4 move to the goal while largely avoiding the objects to be transported. Despite the complexity of the problem of transporting five vehicles, SDPA-MAPPO was shown to learn approximately 20 times faster than the results in Figure 10.

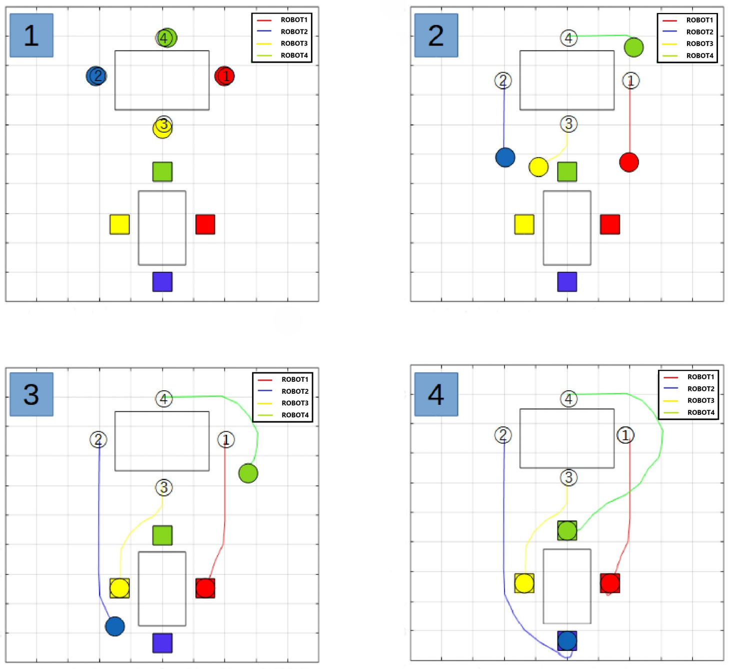

Figure 21. Formation change of four robots using transfer learning: The initial positions of the robots are represented by numbered circles 1 to 4, while their target positions are indicated by the red, blue, yellow, and green squares.

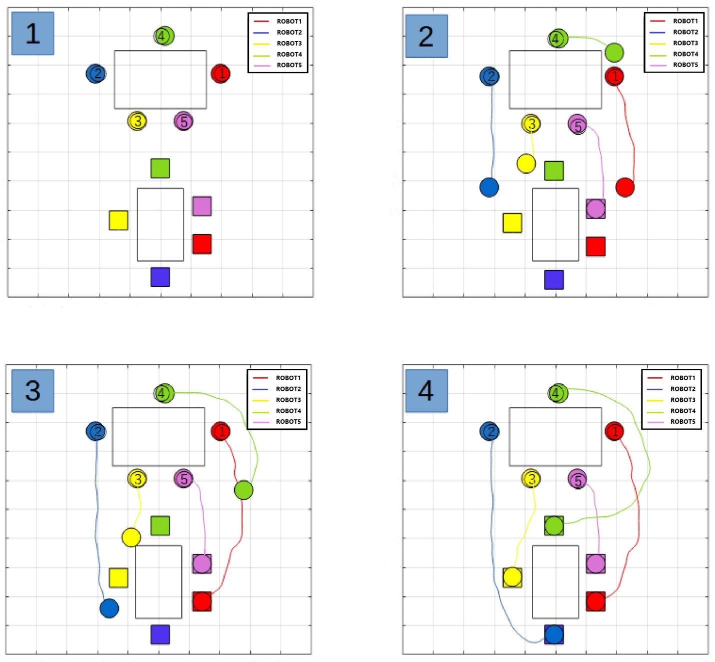

Figure 22. Formation change of five robots using transfer learning: The initial positions of the robots are represented by numbered circles 1 to 5, while their target positions are indicated by the red, blue, yellow, green, and purple squares.

5. Conclusions

In this paper, a formation change algorithm is proposed. It allows easy and fast multi-robot knowledge transfer by using SDPA combined with MAPPO (Multi-Agent Proximal Policy Optimization), compared to other methods. The proposed algorithm is verified by simulating the robot formation change. The contribution of this paper is that it not only achieved fast learning by using the SDPA algorithm but also made it easy to transfer the learning results from a neural network, which was difficult with reinforcement learning. The proposed SDPA-MAPPO learned 20.83 times faster than Deep Dyna-Q. Furthermore, using transfer learning from a three-robot to five-robot case, the method shows that the learning time can be reduced by about 56.57 percent.

Author Contributions: Conceptualization, N.M.; methodology, N.M., A.B., and K.A.; software, A.B. and K.A.; validation, N.M., A.B., and K.A.; investigation, N.M., A.B., and K.A.; writing—original draft preparation, N.M., A.B., and K.A.; writing—review and editing, N.M. and A.B.; visualization, N.M., A.B., and K.A. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Data Availability Statement: Restrictions apply to the datasets.

Conflicts of Interest: The authors declare no conflicts of interest.

References

1. Zhu, Z.; Lin, K.; Jain, A.K.; Zhou, J. Transfer Learning in Deep Reinforcement Learning: A Survey. *IEEE Trans. Pattern Anal. Mach. Intell.* **2023**, *45*, 133443362. [CrossRef] [PubMed]
2. Sutton, R.S.; Barto, A.G. *Reinforcement Learning: An Introduction*; MIT Press: Cambridge, MA, USA, 2018.
3. Kono, H.; Murata, Y.; Kamimura, A.; Tomita, K.; Suzuki, J.T. Transfer Learning Method Using Ontology for Heterogeneous Multi-agent Reinforcement Learning. *Int. J. Adv. Comput. Sci. Appl. (IJACSA)* **2014**, *5*, 156–164. [CrossRef]
4. Brackett, P.; Liu, S.; Liu, Y. C-MAIRL: Semi-Centralized Multi-Agent Imitation Reinforcement Learning. *IEEE Access* **2023**, *11*, 579657976. [CrossRef]
5. Didi, S.; Nitschke, G. Multi-Agent Behavior-Based Policy Transfer. In *EvoApplications 2016: Applications of Evolutionary Computation*; Springer: Berlin/Heidelberg, Germany, 2016; p. 18197.

6. Omidshafiei, S.; Pazis, J.; Amato, C.; How, J.P.; Vian, J. Deep Decentralized Multi-task Multi-Agent Reinforcement Learning under Partial Observability. In Proceedings of the ICML7: Proceedings of the 34th International Conference on Machine Learning, Sydney, Australia, 6–11 August 2017.
7. Bai, C.; Yan, P.; Pan, W.; Guo, J. Learning-Based Multi-Robot Formation Control With Obstacle Avoidance. *IEEE Trans. Intell. Transp. Syst.* **2022**, *23*, 11811–11822. [CrossRef]
8. NEC. NEC Develops Autonomous Mobile Robot Control Technology That Doubles Efficiency While Maintaining Safety. Available online: https://www.nec.com/en/press/202201/global_20220127_01.html (accessed on 9 December 2023).
9. Miyazaki, K.; Matsunaga, N.; Murata, K. Formation path learning for cooperative transportation of multiple robots using MADDPG. In Proceedings of the 21st International Conference on Control, Automation and Systems, Jeju, Republic of Korea, 12–15 October 2021.
10. Budiyanto, A.; Matsunaga, N. Deep Dyna-Q for Rapid Learning and Improved Formation Achievement in Cooperative Transportation. *Automation* **2023**, *4*, 210–231. [CrossRef]
11. Hao, X.; Wang, W.; Hao, J.; Yang, Y. Independent Generative Adversarial Self-Imitation Learning in Cooperative Multiagent Systems. In Proceeding of the 18th International Conference on Autonomous Agents and Multiagent Systems, Montreal, QC, Canada, 13–17 May 2019.
12. Yang, T.; Wang, W.; Tang, H.; Hao, J.; Meng, Z.; Mao, H.; Li, D.; Liu, W.; Chen, Y.; Hu, Y.; et al. An Efficient Transfer Learning Framework for Multiagent Reinforcement Learning. In Proceedings of the 35th Conference on Neural Information Processing Systems, New York, NY, USA, 6–14 December 2021.
13. Liu, Y.; Hu, Y.; Gao, Y.; Chen, Y.; Fan, C. Value Function Transfer for Deep Multi-Agent Reinforcement Learning Based on N-Step Returns. In Proceedings of the Twenty-Eighth International Joint Conference on Artificial Intelligence (IJCAI-19), Macao, China, 10–16 August 2019; p. 45763.
14. Chu, T.; Chinchali, S.; Katti, S. Multi-Agent Reinforcement Learning for Networked System Control. In Proceedings of the ICLR 2020, Addis Ababa, Ethiopia, 26–30 April 2020. .
15. Leno, F.; Silva, D.; Costa, A.H.R. A Survey on Transfer Learning for Multiagent Reinforcement Learning Systems. *J. Artif. Intell. Res.* **2019**, *64*, 645–703.
16. Ho, J.; Ermon, S. Generative Adversarial Imitation Learning. In Proceedings of the 30th Conference on Neural Information Processing Systems (NIPS 2016), Barcelona, Spain, 5–10 December 2016; p. 19.
17. Torabi, F.; Warnell, G.; Stone, P. Behavioral Cloning from Observation. In Proceedings of the Twenty-Seventh International Joint Conference on Artificial Intelligence (IJCAI-18), Stockholm, Sweden, 13–19 July 2018.
18. Vaswani, A.; Shazeer, N.; Parmar, N.; Uszkoreit, J.; Jones, L.; Gomez, A.N.; Kaiser, L.; Polosukhin, I. Attention Is All You Need. In Proceedings of the 31st Conference on Neural Information Processing Systems, Long Beach, CA, USA, 4–9 December 2017.
19. Schulman, J.; Levine, S.; Moritz, P.; Jordan, M.; Abbeel, P. Trust Region Policy Optimization. In Proceedings of the 32nd International Conference on Machine Learning, Lille, France, 6–11 July 2015.
20. Schulman, J.; Wolski, F.; Dhariwal, P.; Radford, A.; Klimov, O. Proximal policy optimization algorithm. *arXiv* **2017**, arXiv:1707.06347.
21. Goodfellow, I.; Pouget-Abadie, J.; Mirza, M.; Xu, B.; Warde-Farley, D.; Ozair, S.; Courville, A.; Bengio, Y. Generative adversarial nets. In Proceedings of the 27th International Conference on Neural Information Processing Systems, Montreal, QC, Canada, 8–13 December 2014; Volume 2.
22. Yu, C.; Velu, A.; Vinitsky, E.; Gao, J.; Wang, Y.; Bayen, A.; Wu, Y. The Surprising Effectiveness of PPO in Cooperative Multi-Agent Games. In Proceedings of the 36th Conference on Neural Information Processing Systems Track on Datasets and Benchmarks, Virtual, 28 November 2022.
23. Agarwal, A.; Kumar, S.; Sycara, K.; Lewis, M. Learning Transferable Cooperative Behavior in Multi-Agent Teams. In Proceedings of the 19th International Conference on Autonomous Agents and MultiAgent Systems, Auckland, New Zealand, 9–13 May 2020.
24. Lowe, R.; Wu, Y.; Tamar, A.; Harb, J.; Abbeel, P.; Mordatch, I. Multi-agent actor-critic for mixed cooperative-competitive environments. In Proceedings of the 31st International Conference on Neural Information Processing Systems (NIPS'17), Long Beach, CA, USA, 4–9 December 2017.
25. Azetsu, K.; Budiyanto, A.; Matsunaga, N. Fast Transfer Learning using Scaled Dot Product Attention for Formation Change of Transport Robots. In Proceedings of the 2024 63rd Annual Conference of the Society of Instrument and Control Engineers of Japan (SICE), Kochi, Japan, 27–30 August 2024.
26. Azetsu, K.; Budiyanto, A.; Matsunaga, N. Fast Learning for Multi-Agent with Combination of Imitation Learning and Model-Based Learning for Formation Change of Transport Robots. In Proceedings of the International Joint Conference on Neural Networks, Yokohama, Japan, 30 June–5 July 2024.