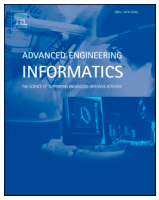
Advanced Engineering Informatics 54 (2022) 101800

Contents lists available at ScienceDirect

Advanced Engineering Informatics

journal homepage: www.elsevier.com/locate/aei



Full length article

Reward shaping in multiagent reinforcement learning for self-organizing systems in assembly tasks

Bingling Huang, Yan Jin *

Dept. of Aerospace and Mechanical Engineering, University of Southern California, McClintock Avenue, OHE-400, Los Angeles, CA 90089-1453, USA

ARTICLE INFO

Keywords:
Multiagent systems
Assembly task
Reinforcement learning
Design
Reward shaping
Collision avoidance

ABSTRACT

Self-organizing systems feature flexibility and robustness for tasks that may endure changes over time. Various methods, e.g., applying task-field and social-field, have been proposed to capture the complexity of task environments so that agents can remain simple. To expand to complex task domains, the multiagent reinforcement learning (MARL) approach has been taken to train agent teams to be more capable and intelligent, permitting reduced complexity in task descriptions. MARL depends on the design of reward functions, which has been a challenging endeavour thus far. This paper investigates the impact of reward shaping in the context of an "L-shape" assembly task that involves collision avoidance. After introducing a general form of reward shaping function, various types of reward shaping fields are studied empirically with agent teams of different sizes. The experiment results have shown that reward shaping can be highly effective, and the singularities, the proper forms of the fields, and the suitable shaping field gradients are essential for successful agent team training. Furthermore, the effect of reward shaping functions depends highly on the size of agent teams.

1. Introduction

During the late 20th century, industry 4.0 was proposed to show an era in which industrial production follows the latest developments [1]. In the factory, smart digital devices are networked [1] for higher flexibility and productivity [2]. Meanwhile, collaborative robots are developed with a higher degree of artificial intelligence and are expected to be smarter to cooperate with each other, even humans. They have advantages like performance boost, reliability enhancement, and simplifying the design [3]. Therefore, they are assigned to more repetitive tasks, allocated to higher workloads, and placed in hazardous environments [2]. Collaborative robots mark a departure from traditional robots that fulfill independent functions [4].

The assembly tasks involve physically coupling multiple separated parts together to form a new sub-component or finished product [5]. With the increasing demands for solving complex assembly tasks in the changing environment with unpredictable, undesirable, and must-avoid conditions [6], collaborative robotic teams are gradually taking the place of human beings. However, there exist challenges like bidirectional recognition [6] and letting robots adopt human beings [7]. In order to achieve the desired autonomy, self-organizing approaches have attracted more researchers' attention thanks to the high-level adaptability of the self-organizing systems [8].

Self-organization refers to an ability of a class of self-organizing systems (SOS) to change their internal structure and/or function in response to changing external circumstances [9]. One significant advantage of the self-organizing systems approach is that each agent, or robot, in a self-organizing system can be kept relatively simple, i.e., possessing minimal knowledge, and the emergent behaviour of the overall system can be expected to be sophisticated enough to deal with various demanding tasks. Previous work has demonstrated that a field-based behaviour regulation (FBR) mechanism can be devised to allow simple agents to self-organize by following the guidance of artificial fields [10]. Depending on the complexity level of a given task, the fields can be composed of a task-field, formed based on the task and environment information [10], or a social-field, formed based on the agents involved and their relationships [11], or both [10 11].

Although the field-based behaviour regulation approach is effective, developing definitions of both task-field and social-field for complex task domains can be challenging due to a lack of prior knowledge of the effect of possible actions. In order to overcome this issue, a machine learning approach has been proposed to let agents acquire their behaviour knowledge through reinforcement learning (RL) [8]. In this approach, the multiagent reinforcement learning (MARL) method is

* Corresponding author.
E-mail addresses: binglinh@usc.edu (B. Huang), yjin@usc.edu (Y. Jin).

https://doi.org/10.1016/j.aei.2022.101800
Received 2 June 2022; Received in revised form 17 October 2022; Accepted 24 October 2022
Available online 9 November 2022

employed, where multiple agents learn how to accomplish the shared task collaboratively by maximizing a shared common reward function. This approach assumes that composing reward functions is more attainable than defining the task-field and social-field functions.

In MARL, reward functions play a crucial role in encouraging exploration and provide a gradient in the learning process, which greatly influences training efficiency, learning speed, and the task performance of the trained systems. For agents to learn about their action behaviors for relatively complex tasks from the reward functions, one needs to understand the properties of various types of reward functions for given task domains. While reward function design highly relies on designers' instincts, experiences, observations, and prior knowledge [12], the lack of a systematic understanding of how reward functions and problem properties interact may lead to systems inefficiency and failure risks.

This paper explores the problem of reward function design in the context of engineering assembly tasks. It especially focuses on investigating how various reward signals and different forms of reward functions may impact the agent teams' training and task performance. Thus, this paper addresses the following questions: *How do different reward shaping fields impact the task performance of agent teams? How may such an impact interact with agent team sizes?*

In the rest of the paper, the related work in RL reward function design, especially reward shaping, is reviewed in Section 2. Section 3 presents the methodology of this study on MARL and reward shaping and introduces a 2D "L-shape" assembly task and its reward function designs. In Section 4, the design of simulation-based reward shaping experiments is described with the definitions of independent, dependent, and control variables. The details of case studies are presented in Section 5 with illustrative results, evaluation metrics, and in-depth discussions. The conclusions are drawn in Section 6, together with the directions for future work.

2. Related work

Robots are gradually replacing humans to conduct assembly tasks. Much research has been done to solve various challenges in assembly tasks. Prasad et al. propose a computative strategic planning projections algorithm to solve the assembly sequence problem [13]. By extracting directions that are required to test geometric feasibility and detecting collisions between parts, it can help test the geometric feasibility. Kumar et al. also focus on assembly sequence problems in oblique orientations [14]. The authors propose an Oblique-directional interference matrix (ODIM) [14] to generate the optimal solution of assembly. In [15], authors propose a stability concept to mitigate the computational complexity problem in parallel assembly sequence planning. It defines various kinds of stabilities between mating parts in a matrix to identify valid subassemblies.

However, the above methods are developed targeting computer-aided design CAD tools, which limits the application range. It requires that engineers prepare all parts as CAD format files first. Besides, those methods rely on" if-else-then" logic rules to process geometric relationships among parts. That puts high requirements on the designer's knowledge to manually classify, analyze, summarize, and process different geometrical relationships. In order to overcome those drawbacks, a more automated and universal method is expected to be developed.

The RL method enlightens a new direction in solving assembly tasks, and many techniques are booming to support its realization. Oikawa et al. proposed a method to select pre-defined stiffness matrices to control the movement of pegs or gears in insertion tasks [16]. It applies deep RL to output optimal local trajectory modifications based on the sensor data, such as tip position and external force. However, the designed reward function is sparse and only provides reward/punishment in accordance with the success/failure of the task. The reward function provides gradients for iterations, and it is crucial for learning equality. As the tasks become more complex, how to enrich reward functions under the condition of limited resources is an unavoidable question to be answered.

When applying RL in self-organizing systems to solve assembly tasks, the goal is to find an optimal policy for each agent to map state and action. The individual agent follows the policy to cooperate and behave in the team to achieve the team's goal. The set of action policies can collectively maximize the return defined by the reward function. One major challenge for both RL and MARL is how to design a proper reward function as the task complexity rises. It has drawn great attention from researchers since reward functions play crucial roles in conveying a designer's preference for the system by training ignorant agents into intelligent ones.

Reward shaping (RS) is deemed an important method to incorporate additional information into the system when solving complex problems [17] by accelerating the learning process and improving the training quality. There are two main methods of RS, i.e., potential-based reward shaping (PBRS) [17] and difference reward shaping (DiRS) [18].

PBRS is an effective way to apply the designer's heuristics to inform agents of the system's preference. Wiewiora et al. proposed two forms of PBRS, look-ahead advice and look-back advice [19]. The methods shape rewards targeting different state-action pairs and recommend using them under different conditions [19]. Plan-based reward shaping has been proposed and applied in both single-agent [20] and MARL [21]. This method applies a reasoning technique to search for a path from the initial state to the goal state. And the trajectory of states can be used to define a potential field. Badnava et al. presented a novel potential-based reward shaping method to accelerate the learning speed by making a reward function changing each step that can push agents to make progress frequently in the task [22]. Brys et al. proposed a method that uses expert demonstrations to speed up learning by biasing exploration through shaping reward by calculating the similarity between state-action pairs [23]. Mannion et al. [24] discussed the theoretical implications of applying these shaping approaches to cooperative multi-objective MARL problems and evaluated their efficacy using two benchmark domains.

DiRS uses a shaping reward signal to help an agent learn the consequences of its actions on the system objective by removing a large amount of the noise created by the actions of other agents active in the system [25 26]. Its applications are restricted to certain problem domains and demonstrated to work only in cooperative tasks. Devlin et al. have combined the two methods in a novel reward function to leverage their benefits [27].

All methods described above require domain-specific heuristic knowledge [22 28], and previous applications of potential-based reward shaping to MARL have been implemented in relatively simple problem domains. Thus, this paper focuses on developing a better understanding of how the reward shaping signals and different forms of reward shaping fields may impact the training process of agent teams and lead to different results in the task performance of the trained agent teams. By applying our findings, the enhanced MARL with reward shaping fields is expected to be more effective in utilizing heuristic knowledge in the learning process and lead to a more intelligent self-organizing system. Besides, it excels at high flexibility in designers' knowledge level on tasks and the number of performers.

3. Methodology

3.1. Reinforcement learning

Reinforcement learning (RL) is a paradigm that makes an agent learn from trials in the process of maximizing the reward. In a finite MDP, a tuple of $< S, A, P, R, \gamma >$ describes the interaction between an agent and an environment in a sequence of discrete-time steps [29]. At each time step t, the agent selects an action $A_t \in A$ based on some representation of the environment's state $S_t \in S$ it receives [29]. The dynamics of the MDP

are defined by a transition matrix P that maps the state and action to the next step's state and reward with a certain probability. R defines a reward function that tells numerical rewards that agents can receive based on their states and actions, and γ is a discount factor that is used for calculating return values, $G_t = \sum_{t+1}^{T} \gamma^{T-t-1} R_t$, *where t stands for current timestep and T represents the timestep when an episode ends*. An agent receives an immediate reward R_t at each time step, its goal is to maximize the total rewards it receives, which means the cumulative reward in the long run [29]. The problem of solving an MDP is to find a policy π that can maximize the accumulated reward [30].

Q-learning [31] is a popular RL algorithm that updates the cumulative rewards of actions based on the temporal difference of value estimations:

$$Q(S_t, A_t) \leftarrow Q(S_t, A_t) + \alpha[R_{t+1} \gamma max_a Q(S_{t+1}, \alpha) - Q(S_t - A_t)] \quad (1)$$

Deep Q-learning [32 33] has been developed in recent years to replace the Q-table with a Q-network with weights θ_i to process complex situations in an end-to-end way. The Q-network updates its weights θ_i at each iteration i by minimizing the loss function shown as:

$$L_i(\theta_i) = E\left[(R_{t+1} + \gamma max_a Q(S_{t+1}, \alpha) - Q(S_t, A_t))^2\right] \quad (2)$$

3.2. Multiagent reinforcement learning

While the single-agent RL is concerned with one agent, multiagent reinforcement learning (MARL) addresses the RL of multiple agents in a shared environment. In MARL, joint action learners and multiple individual learners are two common approaches to solving the problem [34]. Joint action learners apply specific algorithms for multiagent systems (MAS) to learn joint actions. However, it suffers an exponential increase in computational resources since its value functions consider all the possible combinations of actions by all agents [35]. In the case of multiple individual learners, each agent of the system performs its single-agent RL algorithm [31] and the system actions are a joint set $A = A_1 \times \cdots \times A_n$ [36], where n is the number of agents. In cooperative tasks, all agents share a team reward function for a common goal but train separate neural networks to behave and learn their own roles in the teamwork [37].

This paper takes a multi-individual learner's approach and considers cooperative task domains. For such MARL tasks, the goal is to maximize the accumulative reward resulting from all agents' actions over time. Their separate neural networks contribute to capturing behavior codes on how to benefit the team and impart agents with the intelligence of interacting with a dynamic environment influenced by others. The trained neural networks can be reused for different teams of the same size, or even different sizes, in similar tasks if agents are homogeneous [6].

3.3. Reward shaping

Reward functions provide a numerical score to offer a gradient for an RL algorithm's iteration, guide agents' learning, and encourage exploration. It usually includes local and global rewards to direct agents' behaviors [27]. While local rewards regulate a partial system that an agent can directly observe around itself, global rewards judge the outcome of agents' performances. Local and global rewards alone are often insufficient to guide agents' learning for complex tasks, such as assembly task that requires specific configurations. Agents need more information for effective learning.

Reward shaping is a technique to provide additional reward signals that can improve agents' learning efficiency and final performance [24 38]. It provides a way to incorporate knowledge from different sources to guide searches in RL.

In MARL, difference-rewards can be introduced as a reward signal that helps an agent learn the consequences of its own actions on the system after removing the other agents' influences or noises [26], as shown below.

$$D_i(z) = G(z) - G(z_{-i}) \quad (3)$$

where z represents a general term of states or state-action pairs. $G(z)$ is a return of all agents and $G(z_{-i})$ is a return without the work of agent i.

Difference reward shaping (DiRS) [18] focuses on quantifying each agent's contribution to the system, but it works only for cooperative tasks. For a team with homogeneous agents, DiRS has limited power to stimulate individual agents' learning and is ineffective in learning speed improvement since each agent's outcome may be minor compared to the team, making $G(z)$ very close to $G(z_{-i})$.

Potential-based reward shaping (PBRS) [17] helps express a preference for agents to reach a particular state by constructing a potential field Φ. The field function Φ can be defined over different terms, including state s, action a, or state-action pair (s, a) based on different algorithms. Generally, it can be presented as:

$$PBRS = \gamma \Phi(s', a') - \Phi(s, a) \quad (4)$$

where γ is a discount factor, and Φ is a field function.

The potential field Φ is usually built manually by system designers based on their understanding of the tasks. A field is efficient for incorporating heuristic knowledge and provides the gradient to deliver the information [41]. However, this method requires designers to possess high-level prior knowledge of the tasks, and developing the field function can be mathematically challenging, making applying this method a challenge.

Targeting to solve the difficulties of reward function design in assembly tasks, we remodel the shaping reward in the form of multivariate fields, which will be introduced in detail in the next section. It is designed for several improvements. First, it provides a universal way for assembly scenarios with geometrical precision requirements. Secondly, unlike the sparse problems that previous methods may face, it has a high tolerance to be effective when designers hold incomplete task knowledge. Thirdly, it is flexible to be scaled up to fit cases like higher task complexity or heterogeneous agents.

The design situation is usually full of challenges of limited resources. Therefore, it is meaningful to seek ways of a better information carrier and fill the understanding gap between human beings and artificial agents. It is a necessary step toward solving more complex engineering tasks.

3.4. Task description and shaping reward design

In order to empirically investigate how reward shaping influences the learning process and the task performance of MARL self-organizing systems, a specific MARL task is introduced, and the shaping reward design issues for the task are explored.

The Task: The target problem is an "L-shape" assembly task, in which an agent team learns to push each part from separation to the goal configuration, i.e., the "L" shape. At the same time, the task requires agents to avoid collision with surrounding walls in the process.

Fig. 1 shows the task environment of a 1000 × 1000 pixels field. The upper-left rectangle is a dynamic box that agents can push, and its mass equals 1 and size is 180 × 60. The middle-right rectangle is a static box that is a target to form an" L" shape. The agent team (the green squares in the figure) with limited sensing capabilities needs to spontaneously organize themselves to push the dynamic box towards the target box, end with an "L" shape, and avoid the box colliding with the surrounding walls. Detailed settings of the task environment are introduced in Table 1.

The obstacles can be added in the field shown in Fig. 1. Those obstacles increase the complexity of tasks. Considering that the effect of task complexity will not be discussed in this paper, we only show the results in the scenarios with surrounding walls.

In assembly tasks, the precision of the final configuration is

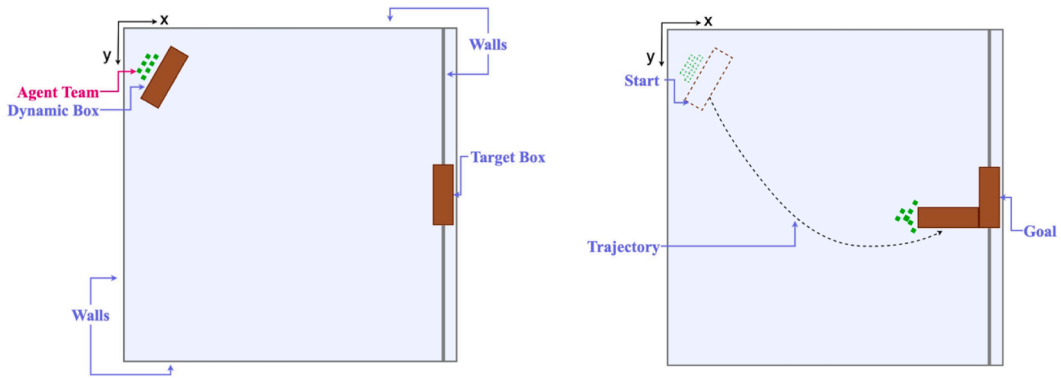


Fig. 1. Task of 'L-shape' assembly: (a) task environment, (b) task start and goal status illustration.

Table 1
Environment settings in Task: "L-shape" assembly.

Field size (pixel)	1000 * 1000
Box size (pixel)	180 * 60
Target box size (pixel)	180 * 60
Box start center coordinates.	(150, 180)
Target box center coordinates	(950, 500)
Box mass (kg)	1
Push impulse (N · s)	1

important. To achieve the task's goal, robot teams are supposed to master strategies not only about the dynamic box's trajectory but also its rotation and displacement. They need to cooperatively control the box's orientation along with the final "L" configuration. When all agents push the box together at each step, the box's movement depends on joint impulses. That increases tasks' difficulty and puts a higher requirement on the group's intelligence. Considering above, two boxes are initiated with different orientations and put away at a certain distance in the field, which provides agent teams with space to try and adjust pushing strategies.

The agents are homogenous with the same action space and sensing capability. To complete the task, they self-organize themselves to work cooperatively as a team and are controlled by their own neural networks. During the training, they consider other agents' influence as a part of the environment. The self-organizing system is trained by deep Q-learning, and an ε–greedy strategy is applied to explore optimal policy [37]. The hyperparameters of the algorithm are shown in Table 2.

State Space and Action Space: The state space is set as a 63-digit tuple, $S = < vicinity\,situation,\ v_x,\ v_y,\ v_{angular} >$. The vicinity situation is sensed by sensors on the box in the range of 200-pixel. The action space is as $A = < a_1, a_2, a_3, a_4, a_5, a_6 >$, which separately represents pushing the box with 1 $N{\cdot}s$ impulse in six different positions: two on each of the two long sides of the box and one on each of the two short sides.

Local Reward: The local reward includes *rotation reward* $R_{rotation}$, which works to control the rotational dynamic, as shown in equation (5).

$$R_{rotation} = C_r{\cdot}(cos(\Delta\rho_i) - cos(\Delta\rho_{control})) \tag{5}$$

Table 2
Hyperparameter settings.

Training episodes	16,000
Discount factor	0.99
Memory buffer size	1000
Mini-batch size	32
Target network update frequency	200
Learning rate	0.001
Neural network size	(63, 64, 128, 6)
epsilon	1 → 0.01

where ρ is the angle of the dynamic box's rotation. $\Delta\rho_i$ is the change of rotational angle of the box at step *i*. $\Delta\rho_{control}$ is a hyperparameter that is set manually before the task begins. It represents an angle change threshold for receiving punishment when exceeded; it is set as 11° in all cases. C_r is a hyperparameter that works for tuning the collective effect with other rewards. It is set as 10 in our cases. Although $\Delta\rho_i$ is globally available information, each agent observes its own values and updates its policy. This allows us to introduce local noises and assess the impact of partially observable states.

Global Reward: The global rewards in this study are calculated based on the global parameter values and fed to each agent simultaneously as a reward signal. They include *goal rewards* R_{goal}, *distance rewards* $R_{distance}$ and *collision rewards* $R_{collision}$.

The *goal reward* R_{goal} gives a big positive reward when the agents achieve the goal (form an "L" shape with two boxes). It is set as:

$$R_{goal} = C_{goal}{\cdot}\mathrm{I}_{GOAL} \tag{6}$$

where I_{GOAL} is a unit indicator of the event GOAL (the "L"-assembly goal is achieved). C_{goal} is a hyperparameter and is set as 1000 in our cases.

The *collision reward* $R_{collision}$ works to avoid any collisions that happen during the task, and it is set as:

$$R_{collision} = C_{collide-wall}{\cdot}\mathrm{I}_{COLLIDE-wall} + C_{collide-obs}{\cdot}\mathrm{I}_{COLLIDE-obs} \tag{7}$$

where $\mathrm{I}_{COLLIDE-wall}$ is a unit indicator of the event COLLISION_WALL (the dynamic box collides onto the walls) and $\mathrm{I}_{COLLIDE-obs}$ an indicator of the event COLLISION_OBS (the dynamic box collides with the obstacles). The $C_{collide-wall}$ and $C_{collide-obs}$ are hyperparameters and are set as −100 and −200, respectively, in all cases.

The *distance reward* $R_{distance}$ is designed to encourage agents to push the box toward the target box and punish them if they go in the opposite direction. It is set as:

$$R_{distance} = C_{distance}{\cdot}\Delta d_i \tag{8}$$

where Δd_i is the distance difference from the current box's position to the target box at step *i*. It is a positive number when the box moves nearer to the target at step *i*. $C_{distance}$ is a hyperparameter and is set as 0.1 in our cases.

Shaping Reward: The reward shaping field is designed to allow the exploration of multiple different shaping reward categories. Considering the task's goal ("L-shape" configuration) and sensing capability, two parameters are selected to form a signal vector, $\vec{s} = < \alpha, \beta >$. The first one, named α, is the box's rotation angle from the vertical line, as shown in Fig. 2-(a). Since, in the beginning, the moving box's longitudinal axis is aligned with the vertical (or north) line, the α angle changes in the range from 0° to 180°. The second parameter, named β, is the angle between the horizontal line and the line connecting the centers of the two boxes, as shown in Fig. 2-(a).

These two angles are chosen as reward signals because 1) they

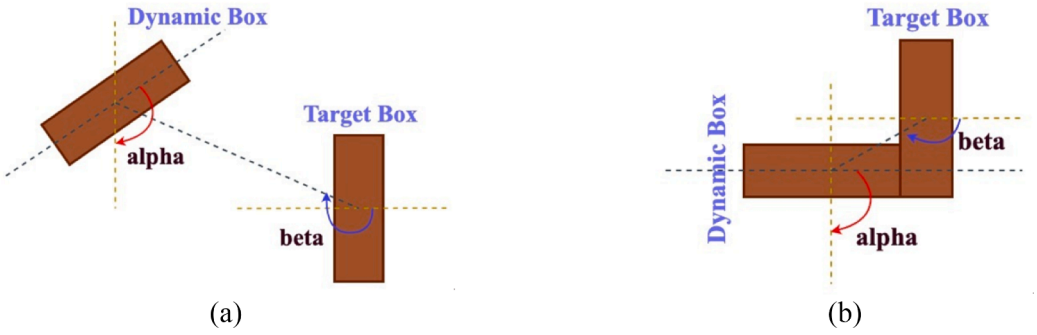


Fig. 2. Box self-angle α and relative angle β: (a) angles in a pushing process, (b) angles at the goal position.

together fully define both the transient process (i.e., changes of the parameter values) and the final angular relation of the two boxes, while other reward terms are employed to reward either the local (e.g., the rotational angle change) or the final positions of the boxes (e.g., collision or goal states); 2) they both can be obtained directly from the sensors on the box, no need to add new sensors for the shaping purpose.

From the geometrical relationship of the two boxes, we know that when the angle α equals 90° ($\pi/2$rad) and angle β equals 135° ($3\pi/4$rad), the final configuration is shaped as "L" perfectly, as shown in Fig. 2-(b). Thus, two hyperparameters can be defined in equations (9) and (10).

$\alpha_{goal} = 90^{\circ}$ *or* $\pi/2rad$ (9).

$\beta_{goal} = 135^{\circ}$ *or* $3\pi/4rad$ (10).

Based on the above, one can devise mathematical functions to describe the task's preference. And the shaping reward can be written as:

$$R_{shaping} = C_{shaping} \cdot f(t) \cdot \Phi(\vec{s}) \tag{11}$$

$$\Phi(\vec{s}) = h(\alpha) \cdot g(\beta) \tag{12}$$

where $f(t)$ represents the task progress in controlling the iteration steps. $h(\alpha)$ and $g(\beta)$ are functions that represent the task's preference for parameters α and β, respectively.

The simplest form that can be used for angle α is:

$$h(\alpha) = |sin\alpha| \tag{13}$$

That is consistent with the system's preference, which means it provides the max reward when the angle α equals 90°, and the reward gradually decreases when the angle α gets deviates from 90°. Some other forms that can be used for angle α are:

$$h(\alpha) = exp\left(-C_{\alpha} \cdot |\alpha - \alpha_{goal}|\right) \tag{14}$$

where α_{goal} is a constant that means our training goal for angle α and $\alpha_{goal} = 90^{\circ}$. C_{α} is a constant coefficient to tune the gradient.

Compared to the reward functions without reward shaping terms, the reward shaping items provide additional information that is coded based on the system designer's heuristic knowledge on how the system should move through the transient process (e.g., equations (11) and (12)) into the goal state (i.e., equations (9) and (10)). Therefore, reward shaping provides space and opportunities to explore the reward landscape to deepen our understanding of task dynamics from a learning perspective, making it possible for us to develop a reward field in which the agents can learn about the task and social fields [5 6].

Similar steps can be carried out with the reward shaping with regard to the parameter β. Specifically, $g(\beta)$ can be set as:

$$g(\beta) = exp\left(-C_{\beta} \cdot |\beta - \beta_{goal}|\right) \tag{15}$$

where β_{goal} is a constant to indicate our training goal and its value is shown in equation (10). C_{β} is a constant coefficient to tune the gradient.

In this study, the resulting "L" configuration is good or not only can be known at the end of the task progress; thus, we set $f(t) = 1$, meaning to check the configuration when the task ends. To investigate how different reward shaping impacts the learning process and task performance, we examine different reward shaping fields of different forms. Following is a list of the reward shaping fields being studied.

$$\boldsymbol{P}0: \quad No\ Reward\ Shaping\ Field \tag{16}$$

$$\boldsymbol{P}1: \quad R_{shaping} = C_{shaping} \cdot |sin\alpha| \tag{17}$$

$$\boldsymbol{P}2: \quad R_{shaping} = C_{shaping} \cdot |sin\alpha| \cdot exp\left(-C_{\beta} \cdot |\beta - \beta_{goal}|\right) \tag{18}$$

$$\boldsymbol{P}3: \quad R_{shaping} = C_{shaping} \cdot exp\left(-C_{\alpha} \cdot |\alpha - \alpha_{goal}|\right) \cdot exp\left(-C_{\beta} \cdot |\beta - \beta_{goal}|\right) \tag{19}$$

$$\boldsymbol{P}4: \quad R_{shaping} = C_{shaping} \cdot exp\left(-C_{\alpha} \cdot |\alpha - \alpha_{goal}|\right) \tag{20}$$

$$\boldsymbol{P}5: \quad R_{shaping} = C_{shaping} \cdot exp\left(-C_{\beta} \cdot |\beta - \beta_{goal}|\right) \tag{21}$$

Fig. 3 and Fig. 4 are visualizations of these fields, in which the x-axes are angles of α and/or β, and y-axes are field values.

The difference between the $|\sin(x-G)|$ field and $\exp(-(x-G))$ field indicates two different training tendencies of the agents toward the goal. The former implies the fast approaching early on and slow adjustments at the end around the goal. The latter is the opposite, slowly approaching early on and significant adjustments at the end. Furthermore, the cases of P1, P4, and P5 simulate situations where only a partial signal, α or β, is available for the purpose of reward shaping.

Based on the state space, action space, and the reward function introduced previously in this section, the deep Q-learning algorithm (hyperparameters shown in Table 2) with the designed reward shaping fields (in equation (16) - (21)) inserted in the reward function (R_{t+1} in equation (2)) can be applied to train. Agent teams learn to conduct the" L"-assembly task throughout thousands of episodes, and their optimal policies are saved and retrieved for outputting the task-performing procedure. The agent teams' performances are supposed to be different as the shaping rewards vary. In order to find effective ways of transiting the task information to artificial agents, the influences of various forms of reward shaping fields are analyzed from several perspectives. The experiments are designed for that goal and are introduced in the following section.

It is worth mentioning that the proposed reward shaping methodology is applicable based on several prerequisites. Firstly, the selected signals in the shaping reward (e.g., α, β in our cases) are assumed to be known by robots in some ways, either from their own sensors or global controllers. Secondly, the preferable values of the selected signals (like α_{goal} and β_{goal} in our cases) are supposed to be constants and can be inputted as hyperparameters in the algorithm. Lastly, the preferences of all selected signals do not exist in conflict, but the signals are NOT required to be independent of each other.

In our cases, we are shaping one equation of an angle ($\beta_{goal} = 135^{\circ}$) and one perpendicular relationship ($\alpha_{goal} = 90^{\circ}$). Besides, the method can be applied to various geometrics based on task requirements if a

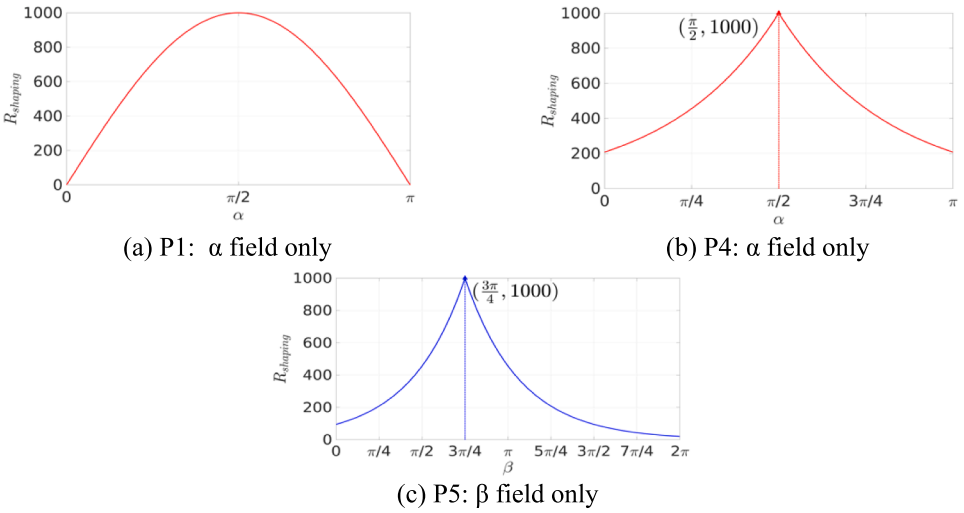


Fig. 3. Illustrations of reward shaping fields P1, P4 and P5.

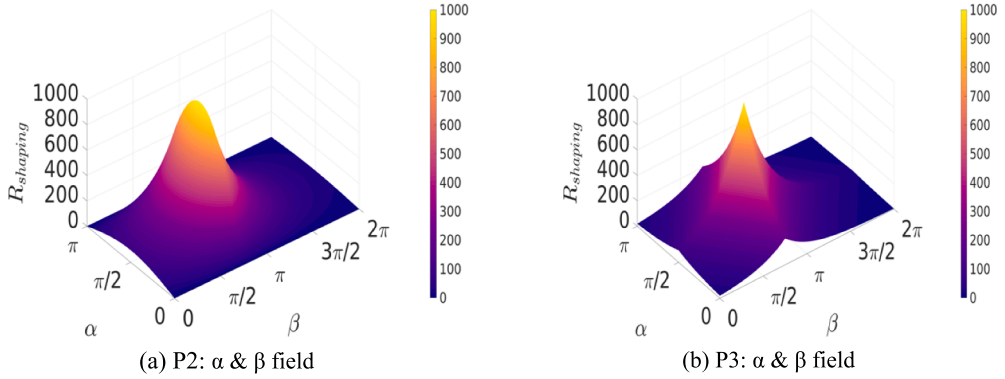


Fig. 4. Illustrations of reward shaping fields P2 and P3.

geometry can be written in an equation in terms of detectable signals. For example, "parallel" can be transformed as a function of two isotropic angles.

4. Experiment design

The physical dynamics of the box movement are simulated in *pygame* [39] and *pymunk* [40] models. The early simulation studies have demonstrated that the *local rewards* and *global rewards* alone frequently failed to guide agents to form needed configurations with desired precision in assembly tasks. Therefore, the reward shaping potential fields introduced above are investigated to further our understanding of the impact of various shaping potential fields in the context of different team sizes. Specifically, the experiment design intends to address the above-mentioned research questions related to *the impact of different reward shaping fields,* and its *interaction with the size of agent teams*.

Fig. 5 illustrates the experiment design of this study. All experiments are based on solving the 2D "L-shape" assembly task introduced in Section 3.4. In order to address the research questions, three of independent variables are set as inputs of the studies, which are *RS fields* shown in Fig. 3 and Fig. 4, *RS field gradients* ranging from 0.1 to 10, and *Agent team size* with possible values of 3, 5, 7 and 9. Correspondingly, *Task performance* is applied as metrics for evaluation. Details of these variables are described in the following subsections.

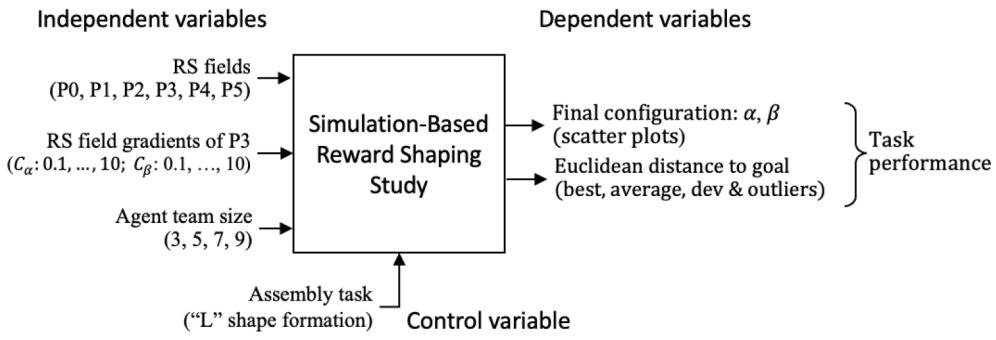


Fig. 5. Experiment design.

4.1. Independent variables

RS fields: The first research question asks how different kinds of reward shaping signals influence the learning process and, consequently, task performance. Although there can be unlimited forms of possible reward shaping functions, our study focuses on differentiating between "convex vs concave" functions, "continuous vs with a singularity (cusp)" functions, as well as "partial vs complete" reward shaping signals. "Concave vs convex" signifies different ways of approaching task goals: fast-first-then-gradual or gradual-first-then-fast; "continuous vs w/singularity" categorizes two groups of functions that treat the goal point significantly differently; and "partial vs complete" checks if limited reward shaping is still meaningful. It is further expected that the impact of these RS fields will interact with the size of the agent teams. The independent variable "*RS fields*" shown in Fig. 5 has six possible variable formations with different properties (see Fig. 3 and Fig. 4 and equations through (16) to (21)):

- P0 (baseline),
- P1 ("concave", "continuous", "partial"),
- P2 ("concave", "continuous", "complete"),
- P3 ("convex", "w/singularity", "complete"),
- P4 ("convex", "w/singularity", "partial"),
- P5 ("convex", "w/singularity", "partial"),

RS field gradient: Our initial simulation studies by varying RS fields from P0 to P5 revealed significant results with P3. In order to further explore the details of RS fields' impact on the system learning and task performance, we introduced another independent variable, "RS field gradient." This variable is composed of two hyperparameters C_α and C_β shown in equation (19). The possible values of these two parameters range from 0.1 to 10 to regulate the field gradients. The results are expected to give hints on designing an RS field with appropriate gradients for a certain task and control its output in a predicted way. Again, it is expected that the team size will interact with the gradients.

Agent team size: Our research question addresses the interaction with team size in MARL, which is a critical aspect to be considered in multiagent system design. For further understanding of how the impact of RS fields interacts with team size, "*agent team size*" is introduced as the third independent variable, as shown in Fig. 5. The possible values of "*agent team size*" are set to be 3, 5, 7, and 9 in the context of various RS fields. Given the task complexity, computational resources, and the box's dynamic movements, the minimum team size is set as 3 due to the dynamic box shown in Fig. 2-(a) requires both displacement and rotation, and the maximum team size is set as 9 for current studies due to the referential experiences from previous studies [37]. As the task complexity grows in future studies, bigger sizes of agent teams will be considered.

4.2. Dependent variables

Task performance, shown in Fig. 5, is an important aspect of evaluation. *Task performance* focuses on how well an assigned task is completed by learning agent teams. The following paragraphs introduce the general concepts of performance measures. Further details are described in the next section, with corresponding results and discussions.

Task performance: As shown in Fig. 1-(b), we are especially interested in the *Task performance* measured as an "L" configuration of the trained agent teams since it indicates the quality of the assembly task. The collision avoidance requirement is fulfilled by the collision punishment in the reward function and the episode termination condition (terminate once collide on walls or obstacles). Therefore, by querying two parameters, α and β shown in Fig. 2-(a), the *Task performance* can be presented and evaluated quantitively. As shown in Fig. 6, task results of learning teams are presented in a coordinate system whose x-axis is angle α, the y-axis is angle β, and one dot presents one learning team's task result. A perfect "L-shape" means $\alpha = 90^\circ$ and $\beta = 135^\circ$ (defined in equations (9) and (10)), represented by a red dot shown in Fig. 6. Other dots are the task results of the trained agent teams. The closer to the red dot the results are, the better quality of the final "L-shape" is.

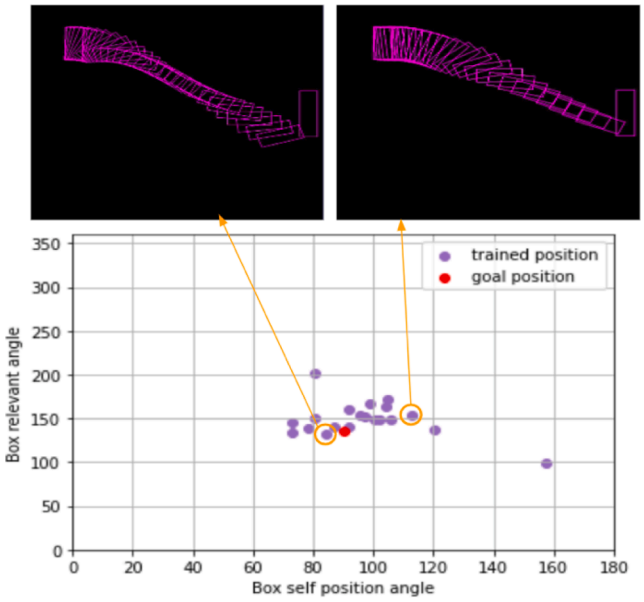


Fig. 6. Illustration of final configuration scatter plots in the form of angle α and angle β.

Furthermore, the statistical aspects can be assessed by calculating the *Euclidean Distance* between a learning team's performance and the goal in the space of $\vec{s} = <\alpha, \beta>$, which can be expressed as:

$$D_i = \sqrt{(\alpha_i - \alpha_{goal})^2 + (\beta_i - \beta_{goal})^2} \tag{22}$$

where α_i and β_i are obtained from the *i*th learning team's final configuration. and α_{goal} and β_{goal} are known from equations (9) and (10). The *Euclidean distance* measure assists in understanding the impact of reward shaping quantitatively and further processing the results. Smaller *distance* values indicate better final configurations, while zero *distance* means a perfect "L-shape".

5. Experiment results and discussion

5.1. Effect of reward shaping functions and signals

As illustrated in Section 4, the experiments are conducted with varying RS fields from P0 to P5 in the context of team sizes of 3, 5, 7, and 9, respectively. For every group of experiments, we run 20 training cases with 20 different random seeds. Considering that training results with 3, 5, 7, and 9 team sizes have shown a similar tendency, we select the 5-agent teams' results for presentation in this section, as shown in Fig. 7. The team size effect will be discussed in Section 5.3. Quantitively, the *mean* of *Euclidean distance* and the *standard deviation* introduced in Section 4.2 are presented in an upper-left box in each plot in Fig. 7.

No Reward Shaping: As shown in equation (16), the P0-field works as a benchmark in which the agent teams are trained without a reward shaping field. The training results with P0-field are shown in Fig. 7-(a), showing cases with a team size of 5. Compared with other results, the final configurations with P0-field suffer a significant variance. All data points distribute sparsely and are far from the target position (red dot), which means no team can produce a good "L-shape". It indicates that the agents can hardly find a good policy to push the box into the desired shape without additional guidance, even if the goal position has been

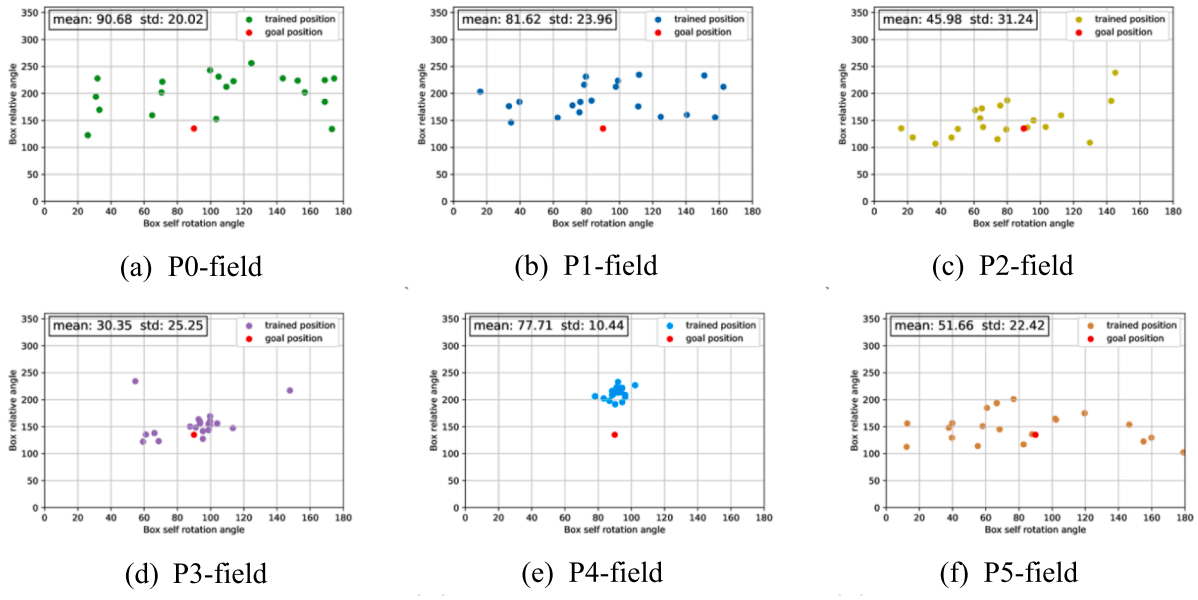


Fig. 7. Final configurations with P0 to P5 reward shaping fields for 5-agent teams.

imparted and rewarded with an enormous positive value through R_{goal} in the reward function. The results call for applying reward shaping fields to provide needed guidance for seeking the task performance of the trained agent teams.

Partial Reward Shaping: As shown in equations (17) (20) (21), P1-field, P4-field, and P5-field provide the reward shaping information on either angle α or angle β, but not both. These cases simulate situations where system designers may have an incomplete understanding of the problem or only a partial reward signal is observable by agents. By incorporating either α or β in the reward shaping field, the results show that even partial shaping information can improve the training quality and hence the task performance, but to different degrees. Fig. 7-(b), Fig. 7-(e), and Fig. 7-(f) are the training results with a team size of 5. They show that the data points become more concentrated in the target position (red dot) compared to the benchmark, which means the reward shaping fields helped the agent team learn better. Besides, it shows that different information has specific effects on tuning its own corresponding behavior; the field of α helps agents form a better box's rotational position, and the field of β guides agents to explore a good strategy to arrive at a good relevant position between the two boxes.

Convex Singular* vs *Concave Continuous Reward Shaping: Comparing Fig. 7-(b) and Fig. 7-(e), one can observe that different fields have different impacts on learning quality. Reward shaping fields constructed by a convex function with a singularity at the goal perform better on guiding agents than those by a concave and continuous function. The singularities in the reward shaping fields are especially effective in stimulating agents' learning and resulting in better training quality. The convex function leading to the sharper changing rate around the goal provides better gradient information and clearer signals for agents to capture the required "knowledge" for achieving the "L-shape" configuration. Gradient information is crucial for RL training, especially when reward shaping is involved. The results indicate that using *convex functions* in reward shaping fields is more effective for training than *concave functions*.

Complete Reward Shaping: The P2-field and P3-field simulate the situations when complete information is available for reward shaping, such as degrees of freedom of a desired shape and signals that are needed to configure that shape. By incorporating more information, different reward shaping fields can be composed. P2-field applies a concave function without singularity on the angle α and a convex function with singularity on the angle β, while P3-field applies two convex functions with singularities on both angles α and β. By comparing the performance results of the trained agent teams in Fig. 7-(c) with Fig. 7-(d), one can see that P3-field excels over the P2-field on training quality greatly. In Fig. 7-(d), a majority of training cases gather around the target point (red dot), indicating that P3-field is much more effective in guiding agents to learn how to complete the assembly task. It proves again that the gradient information does impact the shaping significantly, and *convex functions* have better effects than *concave functions*.

The above results show that the final configuration of the "L-shape" assembly task is highly influenced by different reward shaping fields. A better design of the reward shaping field can improve the precision of the final assembly configuration.

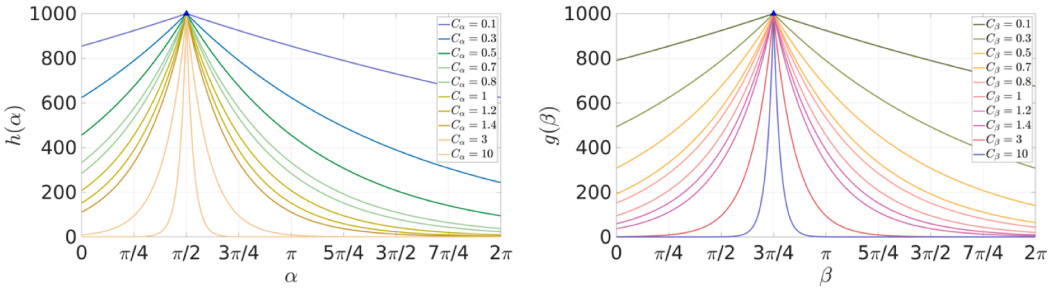
5.2. Impact of reward shaping gradients

As indicated in Fig. 7 and the discussion in Section 5.1, reward shaping fields constructed by *convex functions* with *singularities* can be more effective for guiding agent teams in learning for assembly tasks. To go further, we focus on P3-field and investigate how the reward shaping gradients impact the training quality by measuring the following metrics:

- *Best performance*—i.e., the closest learning team's result to the goal position.
- *Average performance*—i.e., the *mean* of all learning teams' results.
- *Majority performance*—i.e., the *mean* learning performance of the main results cluster.
- *Deviation* from the *mean*—i.e., the standard deviation of the learning teams' results from the average performance.
- *Outliers*—i.e., the learning teams' results that are significantly deviated from the majority of teams.

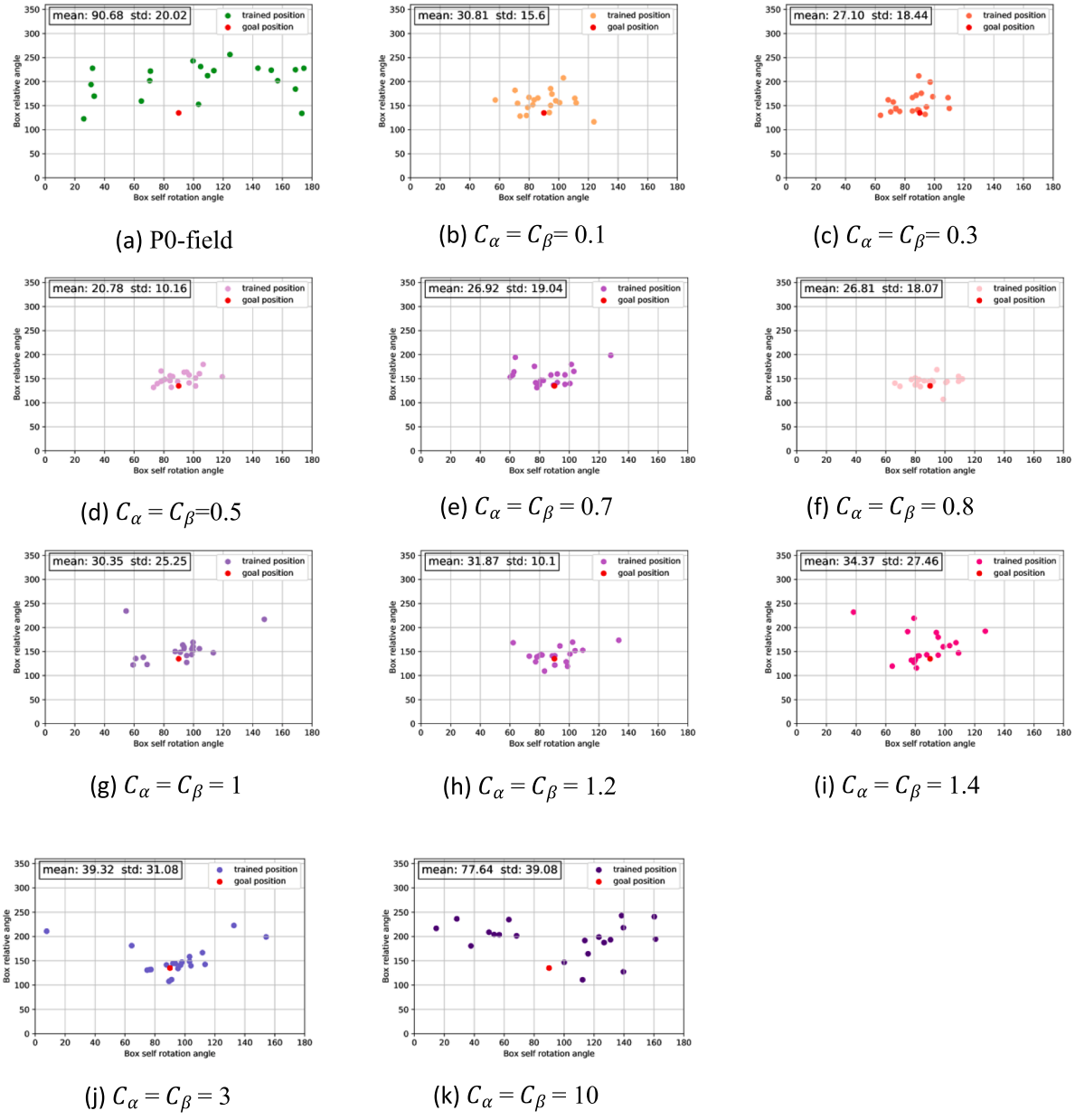
Fig. 8 shows a set of convex functions applied to reward shaping fields. They have different gradients controlled by the coefficients C_α (Fig. 8-(a)) and C_β (Fig. 8-(b)) in equations (14) and (15). One can see that the bigger the coefficients are, the steeper the fields; hence, stricter rules are applied in the training process of the agent teams.

Fig. 9 shows the final configurations of the results trained by the P3 reward shaping field for 5-agent teams. The coordinates and the results plotted in the figures follow the same convention as described in Section 4.2. The *mean (average performance)* and *standard deviation* (*deviation)*



(a) α-fields (b) β-fields.

Fig. 8. Convex shaping reward functions with different gradients.



(a) P0-field (b) $C_\alpha = C_\beta = 0.1$ (c) $C_\alpha = C_\beta = 0.3$

(d) $C_\alpha = C_\beta = 0.5$ (e) $C_\alpha = C_\beta = 0.7$ (f) $C_\alpha = C_\beta = 0.8$

(g) $C_\alpha = C_\beta = 1$ (h) $C_\alpha = C_\beta = 1.2$ (i) $C_\alpha = C_\beta = 1.4$

(j) $C_\alpha = C_\beta = 3$ (k) $C_\alpha = C_\beta = 10$

Fig. 9. Final configurations for 5-agent teams with P3-fields of various gradients C_α & C_β.

can be found in the upper-left blue box in each plot in Fig. 9, the same as in Section 5.1. Similarly, considering that training results with 3, 5, 7, and 9 team sizes have shown a similar tendency, only 5-agent teams' results are presented here, and the training results with 3-, 7- and 9-agent teams are completed and will be discussed in Section 5.3.

While Fig. 9 provides straightforward views of the agent teams' task

performance under different settings, some critical metrics of training results can be quantitively summarized in boxplots, as shown in Fig. 10. Fig. 10 describes the distribution of final configuration data with 5-agent teams (shown in Fig. 9), where x-axes present coefficient values (same C_α and C_β) in the reward shaping fields varying from 0.1 to 10, and y-axes are the statistical indicators. In each box plot, the orange line represents the median *Euclidean distance* of one group of training (20 cases), and the box area (*majority performance*) shows the data range from the first quartile (Q1) to the third quartile (Q3). Flier points shown in Fig. 10 are those out of the 1.5 * IQR (Q3-Q1) (*outliers*).

Effective Range: As shown in Fig. 9, the task results of the trained teams follow a pattern that the proportion of cases with the higher *best performance* and *majority performance* and lower majority *deviation* rises as the coefficients increase, but at the cost of more *outliers*.

At the same time, there exist thresholds of the coefficients. The training suffers the risk of having very low *best*, *average,* and *majority performance* and high *deviation* when coefficients are out of that range and below or over the thresholds. In Fig. 9-(k), the data points are distributed very sparsely, indicating that nearly no teams can learn well when the coefficients are set to 10. From a statistical perspective, Fig. 10 shows the same tendency. When the coefficients are out of the effective range, the data distributions deviate from the goal extremely.

Mild Reward Shaping: Fig. 9-(b), Fig. 9-(c) and Fig. 9-(d) show the results trained by a mild or flatter reward shaping field whose coefficients are 0.1, 0.3 and 0.5. All the cases are distributed around the target shape (good *average performance*), but the high *deviation* means only a few of them can perform perfectly.

It indicates that flatter reward shaping fields can benefit learning by lowering the barrier of catching the system preference but limit the teams' potential to perform perfectly. A mild reward-shaping field leads to somehow 'okay' results but far from 'perfect'.

Sharper Reward Shaping: As reward shaping fields become sharper (i.e., coefficient values are bigger), the distributions concentrate more on the central point, meaning the agent teams can learn better and create close to perfect configurations with higher *best performance*, better *majority performance*, and lower *deviation* of the majority teams.

Fig. 11 focuses on tracking the performance of the majority of teams (*majority performance)* that are the main data clusters. In Fig. 11, the orange error bars present the *mean* (*average performance*) and *deviation* of the majority of teams among 20 trials, while the blue error bars present all teams as a comparison. One can observe that the majority of teams behave much better than the whole teams, with smaller *mean* and *deviation* when the coefficients increase. It indicates that the shaper reward shaping fields encourage pioneering teams to outperform others and become more capable of performing given tasks.

However, as the gradient increases, overfitting happens, and more *outliers* appear, gradually leading to larger *deviations*, as indicated in

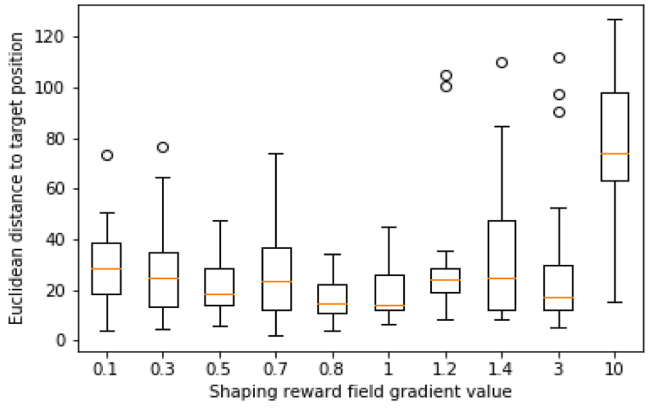


Fig. 10. Euclidean distance boxplot of training results with P3-fields of various gradients for 5-agent teams.

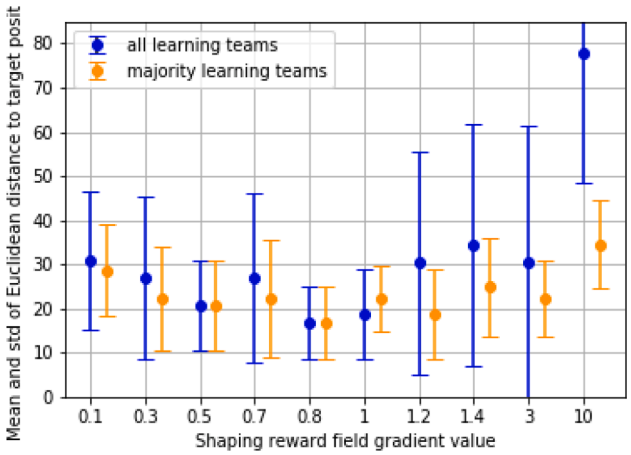


Fig. 11. Mean and standard deviation of Euclidean distance with different reward shaping fields for 5-agent teams.

Fig. 9-(h), Fig. 9-(i), and Fig. 9-(j). The blue error bars for 1.2, 1.4, and 3 coefficients in Fig. 11 show the same tendency.

The results have revealed that sharper reward shaping is desirable but may risk overfitting for training failures. Therefore, engineers can regulate the risk by tuning the gradients of the reward shaping field, pointing to the scale of the rules' strictness of reward shaping field analysis and design.

5.3. Interaction with team size

An organization's performance is influenced greatly by its size, which also applies to self-organizing systems. Usually, small teams have higher flexibility and spend lower costs on team coordination and task performance, but their ability and capacity are limited. Contrastively, large teams, have a larger potential to implement more complex strategies, but at the cost of higher requirements on the information integration in a complex system for teams' efficient training. In this subsection, the effect of team size on reward shaping with different gradients is assessed.

Based on the simulation results of P3-field with different gradients (0.1, 0.3, 0.5, 0.7, 0.8, 1, 1.2, 1.4, and 3) in the context of team sizes of 3, 5, 7, and 9, the statistics of *Euclidean distance* to the goal position of the task performance including *standard deviations* are examined and presented in Fig. 12, which illustrates how different reward shaping fields interact with team size. Considering that the gradient "10″ failed to provide effective training for all teams, as shown in Fig. 9, we neglect it in this "team-size" study. In Fig. 12, the x-axis represents gradient values, and for each value, there are task results of four trained teams with the size of 3, 5, 7, and 9, respectively. The y-axis indicates the *mean* of *Euclidean distance* of 20 training trials and their *standard deviations.* Consistent with Section 4.2, zero *Euclidean distance* means a perfect "L".

In Fig. 12, it can be seen that larger teams favour the reward shaping fields with bigger gradients, while smaller teams perform better under reward shaping with lower and middle-range gradients. In 3-agent groups and 5-agent groups, they exhibit better *means* and smaller *deviations* when the coefficients are in the range of 0.3 to 0.5, and their team performance excels the larger teams. When the coefficients become larger (such as 1.2 and 1.4), the sharp fields begin to hurt the teamwork, resulting in larger *mean* and bigger *deviations*. On the other hand, for 7-agent teams and 9-agent teams, the performance gradually improves as the field gradients increase. While too mild fields cannot lead to efficient learning, the large teams' potentials are simulated after the gradients arrive at 0.7. Their performance turns to exceed the small teams with smaller *mean* and *deviation*. That implies that more complex organizations demand stricter rules to acquire effective learning.

Furthermore, it is intriguing to see that for a team of a given size

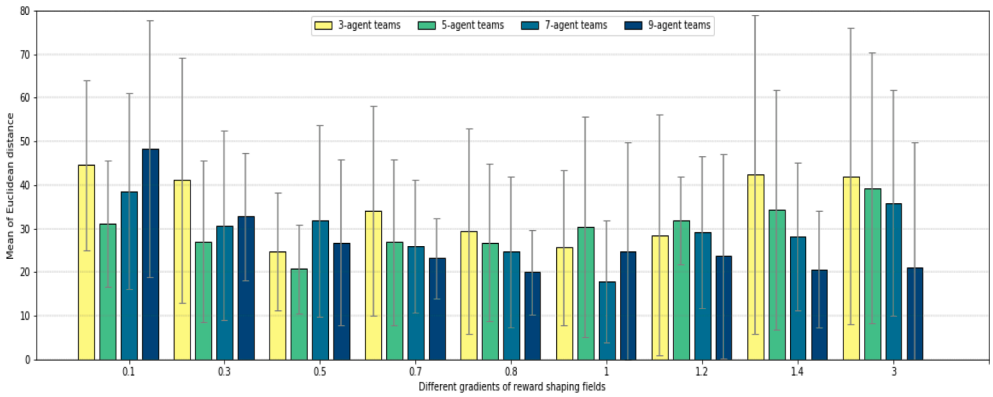


Fig. 12. Team size comparison.

there exists a "sweet spot" gradient that leads to the best training outcome, such as better *average performance* (i.e., closer to the goal configuration) and much smaller *deviations* of the agent teams in acquiring the assembly knowledge. From Fig. 12, one can observe that the 'sweep spots' of 3-agent teams and 5-agent teams appear at $C_\alpha = C_\beta = 0.5$, while 7-agent teams have their best performances around $C_\alpha = C_\beta = 1$, and 9-agent teams reach the best output when the gradients are around 1.4. The trend shows that the 'sweet spots' get larger as the team size increases. That provides hints of designing an effective multiagent RL on proper reward shaping field selection corresponding to the size of the agent teams. Furthermore, 3-agent teams' performance is not as good as other teams in the "L-shape" assembly task, indicating that a certain number of agents is needed for a complex task to release teams' potential and make an optimal policy practical.

6. Conclusion and future work

Self-organizing systems are desirable for performing complex tasks with changing situations as long as the agents in the system possess sufficient knowledge. The approaches of applying task fields and social fields aim to use simple agents that self-organize in artificial fields with complex mathematical forms. MARL opens ways to attain more sophisticated and knowledgeable agents through training, but its success depends on how the reward functions are designed given the task context. In this paper, the impact of reward shaping in MARL is investigated in the context of "L-shape" assembly tasks. Based on the experiment results and ensuing discussions, the following conclusions can be drawn.

- Reward shaping fields can be highly effective in guiding agents' learning process when the proper reward shaping functions and important reward signals are included in the fields. Even partial reward shaping can improve task performance to a certain extent.
- Convex functions with singularity can offer effective guidance for agents to learn how to achieve the best task performance. The task goals should be framed as singular points in reward shaping functions. Convex functions are preferred to provide effective gradients than concave functions for agent teams to learn.
- The gradient of reward shaping fields is an essential factor for the successful training of agent teams. Sharper fields benefit teams to reach excellent task performance but at the cost of risking *overfitting* and bigger *deviation,* while milder gradients generate overall normal results with fewer *outliers.*
- The effective range of gradient changes depending on the size of agent teams. In general, large teams tend to favor steeper gradients, and smaller teams call for milder gradients. For a given agent team of any size, there is an effective range and even a "sweet spot" of the gradient value that leads to the most effective training and best task performance.

It is worth mentioning that this study is limited by the "L-shape" assembly tasks and the reward shaping functions studied. The long-term goal of this research is to develop mappings between specific engineering tasks and the reward shaping fields for training agent teams. Our ongoing work extends this study to more complex assembly tasks with more components, component shapes, and more complex configurations.

Declaration of Competing Interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Data availability

Data will be made available on request.

Acknowledgments

This paper is based on the work supported in part by the Autonomous Ship Consortium (ASC) with members of BEMAC Corporation, ClassNK, MTI Co. Ltd., Nihon Shipyard Co. (NSY), Tokyo KEIKI Inc., and National Maritime Research Institute of Japan (NMRI). The authors are grateful for their support and collaboration on this research.

References

[1] D. Vuksanović, J. Ugarak, D. Korčok, Industry 4.0: The Future Concepts and New Visions of Factory of the Future Development, Sinteza (2016) 293–328. https://doi.org/10.15308/sinteza-2016-293-298.
[2] A.K. Inkulu, et al., Challenges and opportunities in human robot collaboration context of Industry 4.0-a state of the art review, Ind. Robot: Int. J. 49 (2) (2022) 226–239.
[3] A. Khamis, A. Hussein, A. Elmogy, Multi-robot task allocation: A review of the state-of-the-art, Cooper. Robots Sensor Networks 2015 (2015) 31–51.
[4] M. Knudsen, J. Kaivo-Oja, Collaborative robots: Frontiers of current literature, J. Intell. Syst.: Theory Appl. 3 (2) (2020) 13–20.
[5] J.A. Marvel, R. Bostelman, J. Falco, Multi-robot assembly strategies and metrics, ACM Comput. Surv. (CSUR) 51 (1) (2018) 1–32.
[6] Y. Jin, C. Chen, Cellular self-organizing systems: A field-based behavior regulation approach, AI EDAM 28 (2) (2014) 115–128.
[7] J. Berg, et al., Action recognition in assembly for human-robot-cooperation using hidden Markov models, Proc. CIRP 76 (2018) 205–210.
[8] H. Ji, Y. Jin, Designing Self-Organizing Systems With Deep Multi-Agent Reinforcement Learning, in: International Design Engineering Technical Conferences and Computers and Information in Engineering Conference, vol. 59278. American Society of Mechanical Engineers, 2019, August, p. V007T06A019.
[9] W. Banzhaf, Self-organizing Systems, Encyclopedia Complex. Syst. Sci. 14 (2009) 589.
[10] Y. Jin, C. Chen, Field-based behavior regulation for self-organization in cellular systems, in: Design Computing and Cognition'12, Springer, Dordrecht, 2014, pp. 605–623.

[11] N. Khani, J. Humann, Y. Jin, Effect of social structuring in self-organizing systems, J. Mech. Des. 138 (4) (2016), 041101.
[12] N. Shaker, Intrinsically motivated reinforcement learning: A promising framework for procedural content generation, in: 2016 IEEE, 2016, September.
[13] V.V. Prasad, et al., A novel computative strategic planning projections algorithm (CSPPA) to generate oblique directional interference matrix for different applications in computer-aided design, Comput. Ind. 141 (2022), 103703.
[14] G.A. Kumar, et al., A novel Geometric feasibility method to perform assembly sequence planning through oblique orientations, Eng. Sci. Technol., Int. J. 26 (2022), 100994.
[15] A.K. Gulivindala, et al., A heuristic method with a novel stability concept to perform parallel assembly sequence planning by subassembly detection, Assembly Autom. (2020).
[16] M. Oikawa, et al., Reinforcement learning for robotic assembly using non-diagonal stiffness matrix, IEEE Rob. Autom. Lett. 6 (2) (2021) 2737–2744.
[17] A.Y. Ng, D. Harada, S. Russell, Policy invariance under reward transformations: Theory and application to reward shaping, in: Icml, Vol. 99, 1999, June, pp. 278-287.
[18] S. Proper, K. Tumer, Modeling difference rewards for multiagent learning, in: AAMAS, 2012, June, pp. 1397-1398.
[19] E. Wiewiora, G.W. Cottrell, C. Elkan, Principled methods for advising reinforcement learning agents, in: Proceedings of the 20th International Conference on Machine Learning (ICML-03), 2003, pp. 792-799.
[20] M. Grzes, D. Kudenko, Plan-based reward shaping for reinforcement learning, in: 2008 4th International IEEE Conference Intelligent Systems, vol. 2. IEEE, 2008, September, pp. 10-22.
[21] S. Devlin, D. Kudenko, Plan-based reward shaping for multiagent reinforcement learning, Knowledge Eng. Rev. 31 (1) (2016) 44–58.
[22] B. Badnava, N. Mozayani, A new potential-based reward shaping for reinforcement learning agent. arXiv preprint arXiv:1902.06239, 2019.
[23] T. Brys, A. Harutyunyan, H.B. Suay, S. Chernova, M.E. Taylor, A. Nowé, Reinforcement learning from demonstration through shaping. Twenty-fourth international joint conference on artificial intelligence, 2015.
[24] P. Mannion, S. Devlin, J. Duggan, E. Howley, Reward shaping for knowledge-based multi-objective multiagent reinforcement learning, Knowledge Eng. Rev. 33 (2018).
[25] A.K. Agogino, K. Tumer, Unifying temporal and structural credit assignment problems, in: AAMAS, vol. 4, 2004, July, pp. 980-987.
[26] A.K. Agogino, K. Tumer, Analyzing and visualizing multiagent rewards in dynamic and stochastic domains, Auton. Agent. Multi-Agent Syst. 17 (2) (2008) 320–338.
[27] S. Devlin, L. Yliniemi, D. Kudenko, K. Tumer, Potential-based difference rewards for multiagent reinforcement learning, in: Proceedings of the 2014 international conference on Autonomous agents and multiagent systems, 2014, May, pp. 165-172.
[28] B. Marthi, Automatic shaping and decomposition of reward functions, in: Proceedings of the 24th International Conference on Machine learning, 2007, June, pp. 601-608.
[29] R.S. Sutton, A.G. Barto, Reinforcement learning: An introduction, MIT press, 2018.
[30] S. Devlin, D. Kudenko, M. Grześ, An empirical study of potential-based reward shaping and advice in complex, multiagent systems, Adv. Complex Syst. 14 (02) (2011) 251–278.
[31] C.J.C.H. Watkins, Learning from delayed rewards, 1989.
[32] V. Mnih, K. Kavukcuoglu, D. Silver, A. Graves, I. Antonoglou, D. Wierstra, M. Riedmiller, Playing atari with deep reinforcement learning. arXiv preprint arXiv: 1312.5602, 2013.
[33] V. Mnih, K. Kavukcuoglu, D. Silver, A.A. Rusu, J. Veness, M.G. Bellemare, et al., Human-level control through deep reinforcement learning, Nature 518 (7540) (2015) 529–533.
[34] H. Ji, Y. Jin, Evaluating the learning and performance characteristics of self-organizing systems with different task features, AI EDAM (2021) 1–19.
[35] T. Rashid, M. Samvelyan, C. Schroeder, G. Farquhar, J. Foerster, S. Whiteson, Qmix: Monotonic value function factorisation for deep multiagent reinforcement learning, in: International Conference on Machine Learning, PMLR, 2018, July, pp. 4295-4304.
[36] Y. Wang, C.W. De Silva, Multi-robot box-pushing: Single-agent q-learning vs. team q-learning, in: 2006 IEEE/RSJ International Conference on Intelligent Robots and Systems. IEEE, 2006, October, pp. 3694-3699.
[37] H. Ji, Y. Jin, Knowledge Acquisition of Self-Organizing Systems With Deep Multiagent Reinforcement Learning, J. Comput. Inf. Sci. Eng. 22 (2) (2022).
[38] M. Grześ, D. Kudenko, Online learning of shaping rewards in reinforcement learning, Neural Networks 23 (4) (2010) 541–550.
[39] P. Shinners, Pygame – Python Game Development, 2011. Retrieved from http ://www.pygame.org.
[40] V. Blomqvist, Pymunk: A easy-to-use pythonic rigid body 2d physics library (version 5.6.0). Opgehaal van, 2007. https://www.pymunk.org.
[41] T. Brys, Reinforcement Learning with Heuristic Information (Doctoral dissertation, PhD thesis, PhD thesis, Vrije Universitet Brussel), 2016.