Proceedings of the 2017 IEEE
International Conference on Robotics and Biomimetics
December 5-8, 2017, Macau SAR, China

A Novel Swarm Robot Simulation Platform for Warehousing Logistics

Yandong Liu[1], Lujia Wang[1], Huaiyang Huang[2], Ming Liu[3] and Cheng-zhong Xu[1]

***Abstract*—Swarm robot systems are widely applied in the warehouse logistics, which effectively improve logistics efficiency. Task allocation is the core problem for swarm robot systems. Therefore, evaluation of the task allocation strategy has very important practical significance. However, current simulation platforms, such as GAZEBO, require a lot of setup work before testifying the task allocation strategy for various applications. Regarding warehousing logistics, a few special factors need to be considered: man-robot coexistence in working environment, the energy consumption of robots and collision avoidance among robots. Therefore, we propose a novel swarm robot simulation platform, called MultiBots, based on multi-agent pathfinding (MAPF) method and collision avoidance strategy, which can correctly evaluate the effectiveness of task allocation strategy. Moreover, we design the charging process to supplement the energy consumption of robots. The experimental results show that the proposed MultiBots satisfies the requirements of task allocation strategy evaluation in warehouse logistics scenarios. The MultiBots can be applied in testifying the efficiency of task allocation strategy for logistics systems.**

I. Introduction

The development of online business economy has put forward new demands to the logistics. Recently, multi-robot systems are adopted in logistics, such as a Kiva warehouse logistics system [1]. In addition, various task allocation strategies are applied in logistics systems to reduce the logistics cost [2] [3]. Therefore, suitable emulators are desired to develop for verifying the efficiency of logistics systems [4] [5]. Previous research in multi-agent simulator included TeamBots①, SWARM②, and NetLogo③, etc. However, these simulators are not suitable for the warehouse logistics scenarios. In this paper, we designed a simulation platform named MultiBots, which is a Python-based logistics system simulator.

The MultiBots mainly considers three factors: the multi-agent pathfinding (MAPF) method, the collision avoidance strategy, and the charging process. Because the application scenarios are in the warehouse, robots share their work spaces with humans. Therefore, the resulting predictability of their motion is necessary for the safety of the humans [6]. We adopt a scheme for MAPF, called Highways [7] [8], based on the ideas behind experience graphs [9] [10]. Because the MultiBots is a multi-agent system, the collision avoidance is a signification issue [11]. Highways solves the head-to-head collisions among robots. And the collision waiting strategy deals with other situations. Considering the charging process, because the system energy consumption is an important factor in evaluating the task allocation strategy.

The remainder of this paper is organized as follows. Section II introduces the related work about the multi-agent emulator. Section III describes our MultiBots including application scenarios, the MAPF method, the collision avoidance strategy, and the charging process. Section IV shows the implementation and analyses experimental results. The last section concludes this paper and provides our future work.

II. Related Work

Multi-agent is a hot issue in artificial intelligence and computational science. A simulation system is a valuable tool for studying multi-agent. Some simulation and experimental systems have already been developed. The Cellular Robotics System (CEBOT) can reconfigurate itself to optimal structure depending on the purpose and environment [12]. ACTor-based Robots and Equipments Synthetic System (ACTRESS) is an autonomous and distributed robot system composed of multi robotic elements which are provided with functions to decide with understanding the target of tasks, recognizing surrounding communicate with any other components [13]. A swarm-bot is comprised of autonomous mobile robots called s-bot which can either cat independently or self-assemble into a swarm-bot by using their grippers [14]. MASON is discrete-event multi-agent simulation toolkit in Java [15]. And serve as the basis for a broad range of multi-agent simulation tasks ranging from swarm robotics to machine learning to social complexity environments.

Related work above are used as multi-agent simulation toolkits which do not target specific areas. For unmanned autonomous vehicles, [16] presents a hardware-in-the-loop development simulation framework for multi-vehicle autonomous systems. For mobile robot localization, [17] proposes a multi-agent system using the SPADE MAS platform to improve

*This work was supported by the National Natural Science Foundation of China No. 61603376, awarded to Dr. Lujia Wang; Shenzhen Science, Technology and Innovation Comission (SZSTI) JCYJ20160401100022706 and the National Natural Science Foundation of China No. 61403325 awarded to Prof. Ming Liu; Science and Technology Planning Project of Guangdong Province (2015B010129011) and the National Natural Science Foundation of China No. U1401258, awarded to Prof. Cheng-zhong Xu.

[1]Yandong Liu, Lujia Wang and Cheng-zhong are with Center for Cloud Computing, Shenzhen Institutes of Advanced Technology, University of Chinese Academy of Sciences {yd.liu, lj.wang1, cz.xu}@siat.ac.cn

[2]Huaiyang Huang is with College of Mechanical Engineering, Zhejiang University {hyhuang1995@gmail.com}

[3]Ming Liu is with Electrical and Computer Engineering, Hong Kong University of Science and Technology and the City University of Hong Kong Shenzhen Research Institute {eelium@ust.hk}

①https://www.cs.cmu.edu/∼trb/TeamBots/

②http://www.swarm.org/wiki/Main_Page

③https://ccl.northwestern.edu/netlogo/

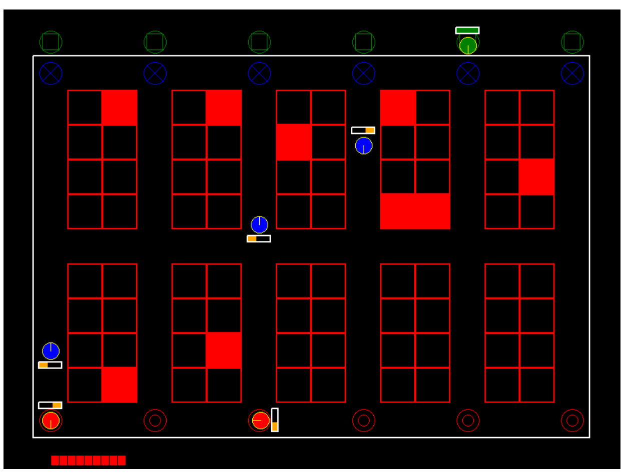
978-1-5386-3742-5/17/$31.00 © 2017 IEEE

the location of mobile robots in dynamic scenarios. For traffic, [18] presents the concept of an integrated multi-agent simulation platform to support the development and validation of autonomic cooperative car-to-car systems. For formation control, [19] provides two algorithms to tackle the multi-team formation problem. And two algorithms are evaluated in RMASBench (a rescue multi-agent benchmarking platform used in the RoboCup Rescue Simulation League).

For the warehouse logistics scenarios, several studies have designed the simulation model from a different perspective: [20] presents an agent-based simulation as a tool for decision-making about automatic warehouses management. [21] proposes towards a multi-agent logistics and commercial transport model. [22] provides advanced approaches for multi-robot coordination to resolve a fundamental problem which is for robots to make individual decisions so to optimize a system-wide objective function. However, the simulation process based on ROS and GAZEBO is complex and does not consider the energy consumption of the system. The above emulators do not verify the logistics task allocation strategy. The quality of the strategy decides the efficiency of logistics which has the economic value [23]. To fill such a gap, we design the MultiBots as Multi-Agent logistics system simulation platform.

III. The MultiBots System

A. The Warehouse Logistics scenarios



(a) The MultiBots

(b) Departure area (c) Unloading area (d) Charging area

Fig. 1: The simulation platform MultiBots and area marks

To achieve the logistics scenarios, we design a Multi-agent system simulation platform divided into charging area and working area. Simulator system uses a pixel as a unit length. As shown in Fig.1(a), an 800*550 white rectangle stands for the warehouse, which is the working area of the robot loading and unloading goods. There are six available departure zones, i.e., Fig.1(b). There are correspondingly six available unloading zones, i.e., Fig.1(c). There are eighty 50*50 red squares in the warehouse, divided into ten groups of 8 each, which denotes the storage area of the goods. And they are numbered from 0 to 79, as shown in Fig.2. When a red square is filled with black, it denotes there are no goods. On the contrary, the red means goods. Between the two groups, there is a 50 width passageway, which allows circle robots with a radius of 12.5 successfully passed. The straight line in the circle represents the head of the robot, and there is a life bar at the end of the robot. The three different filling colors in the circle represent the three different status of the robots. Blue means no load and the robot is on the way to pick up; Red means carrying goods and the robot is on the way to unload. Green means idle and the robot completes the task or goes back to charge. Above the warehouse, there are six available charging zones, i.e., Fig.1(d). The small red square located left below the warehouse denotes the goods taken. MutilBots parameters and their value are in the TABLE I.

TABLE I: MutilBots parameters and their value

Parameter	Value
Warehouse size	800*550
Goods size	50*50
The number of goods	80
Radius of the robot	12.5
Width of passageways	50

B. Multi-agent Pathfinding Highways

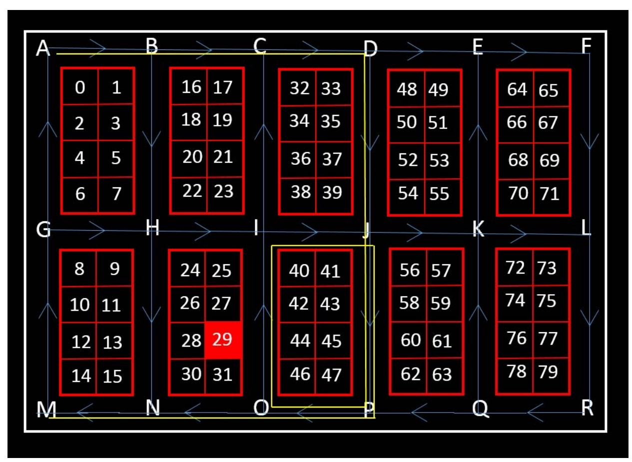


Fig. 2: User-provided highways
in the simulation platform MultiBots

Taking into account the application scenarios, in the warehouse, we use the Highways method to guarantee human safety and avoid head-to-head collisions among robots. Fig.2 shows a directed graph from ‘A’ to ‘R’, which robots must move along the narrow directed passageways between the storage locations based on Highways.

Algorithm 1 presents the whole framework of the Highways path planning procedure. First, we initialize the starting point

Algorithm 1 Highways algorithm

Initialize the letter list, the $StartLetter$ and $GoalLetter$
Initialize the path direction
Initialize the list $holder$ to store temporary paths
for the $LetterMark$ in the letter list **do**
    Set the $LetterMark$ coordinates
    Set $NextLetter$ of $LetterMark$
end for
Generate a list $graph$ to store the $NextLetter$
Set the temporary path:
  $temppath = [StartLetter]$ and $holder = [temppath]$
while $holder! = empty$ **do**
    if $temppath[LastLetter] = GoalLetter$ **then**
        return $temppath$
    end if
    for the $NextLetter$ of the $LastLetter$ **do**
        if the $NextLetter$ is not in the $temppath$ **then**
            Generate a newpath:
              $newpath$ = $temppath + NextLetter$
            Add the $newpath$ to the list $holder$
        end if
    end for
end while

and end point of the robots, the direction among letters. Second, we set the letter coordinate and their next letter of arrival. Third, we take out the first path in the queue of the road and check that its last letter is the goal. Last, if the result is not the goal, we will generate a new path by adding next letter. Loop the above process until the checked result is the aim.

For example, if the robot starts from 'A', loading the goods at the storage location filled with red in Fig.2, to 'M' unloading. It can not choose the nearest method, which faces the direction of the arrow. The right way is 'A→B→C→D→J→P→O', loading at the storage location, and move along the 'I→J→P→O→N→M', unloading at the end 'M', as shown by the yellow solid line in Fig.2.

C. A collision avoidance strategy

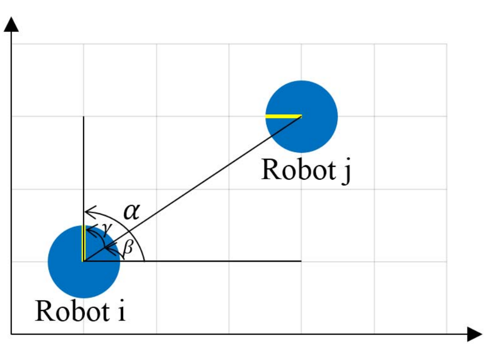
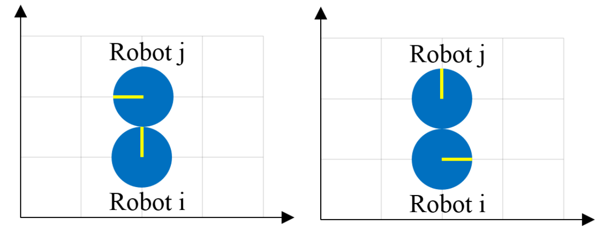


Fig. 3: The collision angle diagram



(a) The collision angle $\gamma = 0°$ (b) The collision angle $\gamma = 90°$

Fig. 4: Two kinds of collision angles

Highways can avoid the head-to-head collisions among robots. However, other situations are leading to crashes, such as rear-ending during loading and unloading, side collision at the crossroads. We propose collision waiting method to solve the above problems. When an accident occurs, the active robot stops working, and the passive robot normally works. We determine whether the collision happens through the following two conditions: distance satisfies the equation $\sqrt{(x_i - x_j)^2 + (y_i - y_j)^2} <= r_i + r_j$. $(x, y), r$ are respectively the coordinate and radius of the robots; angle satisfies the equation $|\alpha - \beta| < \theta$. In Fig.3, we set Robot i as the active collision robot and Robot j as the passive collision robot. α is the angle of Robot i. β is the angle between robot i and robot j. We call γ the collision angle, $\gamma = |\alpha - \beta|$. θ is set the threshold. In the case of satisfying the first condition, if γ is less than the θ, the collision happens. The set threshold smaller is, the probability of a collision is smaller. If the $\theta \to 0°$, only in the case of Fig.4(a) will be a collision. If the $\theta \to 90°$, in the case of Fig.4(b) will also be a collision.

D. The charging process

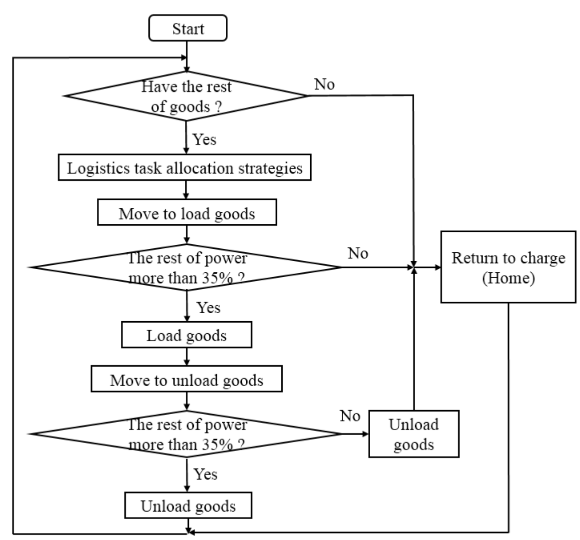


Fig. 5: Flowchart of the logistics charging process

Fig.5 shows the logistics charging process including loading and unloading, accompanied by the energy consumption of robots. When the remaining energy is less than 35%, robots without goods directly return to the charging area to charge, and others firstly unload the goods. There are other charging parameters in TABLE II. These parameters are set to ensure that the robot can return to the charging area when it needs to charge.

TABLE II: Charging parameters and their value

Parameter	Value
Full power	10
Threshold power	3.5
Consuming rate	0.005
Charging rate	0.05

IV. Implementation and Experimental Results

In this section, we describe the implementation of MultiBots and analyze experimental results. We provide three experimental indicators to access the performance of MultiBots:

- Task time T, it denotes the total time, which begins from the first robot into the warehouse to the last robot to leave off.
- The number of collisions S, it denotes the number of collisions happen among robots during the working.
- The number of charging C, it denotes the number of charging is during the working status of robots.

TABLE III: The location of goods

Group number	The location of goods
Group1	[11, 8, 39, 28, 53, 1, 69, 42, 79, 6]
Group2	[74, 50, 62, 15, 49, 34, 14, 72, 31, 37]
Group3	[68, 79, 56, 47, 48, 75, 23, 50, 76, 16]
Group4	[59, 6, 74, 2, 67, 49, 53, 28, 25, 79]
Group5	[14, 65, 41, 75, 22, 36, 54, 74, 9, 48]

TABLE IV: Other experimental parameters and their value

Parameter	Value
Starting positions	[A, B, C, D, E, F]
Unloading position	[M]
Loading time	10 time steps
Unloading time	20 time steps
Robot speed	4 pixels per time step
Collision angle	30°

1) Experiment Setup: To ensure the accuracy of the experiment, the research is conducted with six different number of robots, $N = 1, 2, 3, 4, 5, 6$, and each number has five diverse groups of goods, which the number of goods is ten. TABLE III shows the location of goods. The robots fetch in turn according to the order of the goods. In order to complete the experiment, other parameters are needed: the starting positions are [A, B, C, D, E, F] and the same unloading position is [M]; loading time and unloading time are 10 and 20 time steps respectively; the robot speed is 4 pixels per time step; the collision angle is 30°.

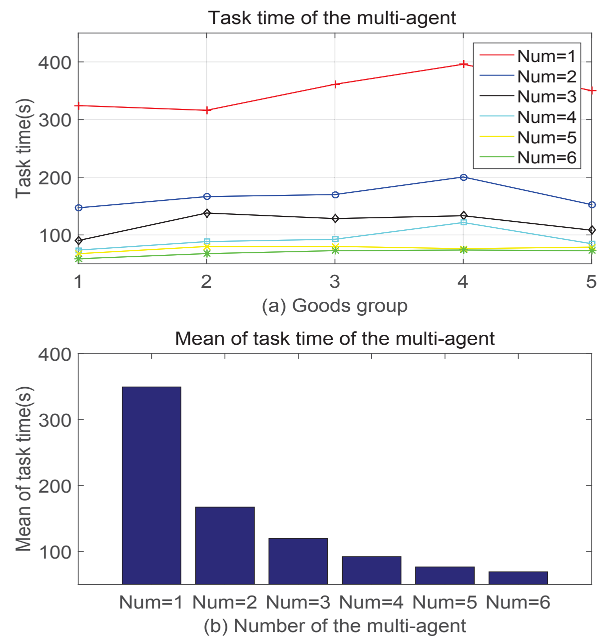


Fig. 6: The task time of six different number of robots

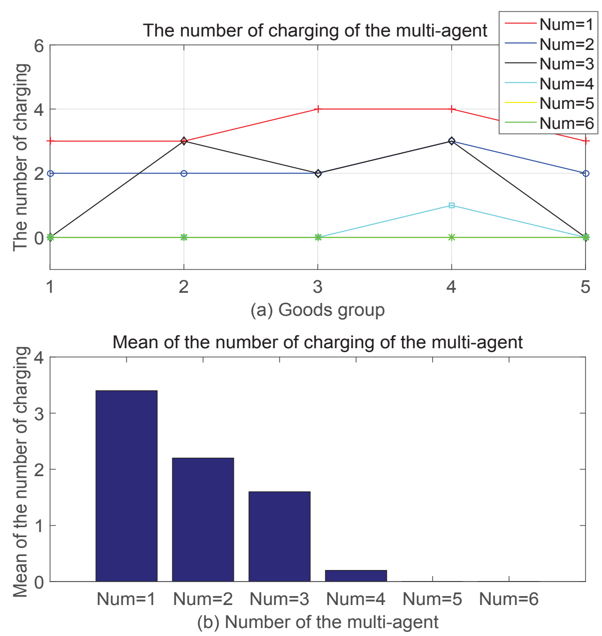


Fig. 7: The number of charging of six different number of robots

2) Experimental Results and Analysis: Fig.6, Fig.7, and Fig.8 respectively show three experimental indicators of six different number of robots. Part(a) is the specific distribution of indicators according to TABLE III. Part(b) is the mean of indicators of the five groups of goods.

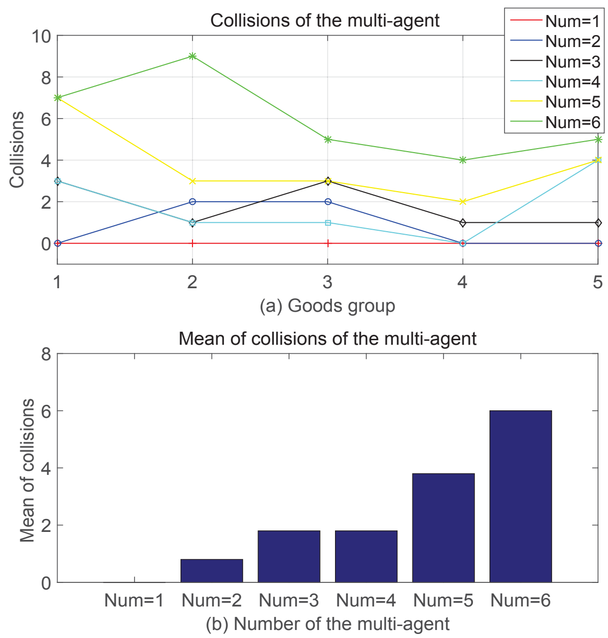


Fig. 8: The number of collisions of six different number of robots

Fig.6 shows the task time of six different number of robots. In general, for the different number of robots, the task time is completely different. As shown in Fig.6, the more the number of robots is, the less the task time spend. The same means less the amount of charging is, in Fig.7(b). The above description shows that the number of robots in the task has a fundamental impact on the task time and the number of charging. Fig.6(a) shows that the location of goods in the warehouse effects the task time. And this effect is based on the number of robots. For the number of charging is basically in line with this trend, in Fig.7(a). However, for the particular group, the impact of the location of goods is greater than the number of robots, e.g., group 2 in Fig.7(a).

As shown in Fig.8(b), the more the number of robots is, the higher the probability of collisions is. Fig.8(a) shows the distribution of crashes is complex, particularly when the number of robots is two and three. This is also confirmed from Fig.8(b). We conclude that when the difference in the number of robots is not great, the location of goods is the determinant of the number of collisions.

There is a link among the three experimental indicators. When the number of working robots is equal, the more the number of charging is, the longer the task time is. What is more, the impact of collisions on the task time is not visible. Because the setting of the parameters, loading time and unloading time, do not cause too much time to wait. Shortening the task time and reducing the number of collisions are the significant issues, which need to resolve in the future work.

V. Conclusions and Future Work

In this paper, we designed a multi-agent logistics simulation platform named MultiBots which can simulate the real warehouse logistics system and evaluate corresponding task allocation strategies. In addition, we adopted the Highways as the multi-agent path planning method and the collision waiting strategy to guarantee the proper operation of the MultiBots. The experimental results demonstrate that the Highways and the collision waiting strategy are useful. Moreover, the task allocation strategies can be testified in the MultiBots.

In the future, we will work on efficient multi-agents pathfinding method for the MultiBots. At the same time, taking into account the distance, the task time, the number of collisions, and other factors, we will propose efficient task allocation strategies to optimize the objective function $Cost_{total} = Cost(distance)+Cost(time)+Cost(collision)$, which will be testified in the MultiBots.

References

[1] Peter R Wurman, Raffaello D'Andrea, and Mick Mountz. Coordinating hundreds of cooperative, autonomous vehicles in warehouses. *AI magazine*, 29(1):9, 2008.

[2] Lujia Wang, Ming Liu, and Q. H Meng. An auction-based resource allocation strategy for joint-surveillance using networked multi-robot systems. In *IEEE International Conference on Information and Automation*, pages 424–429, 2013.

[3] M Tan, L Wang, D Tardioli, and Ming Liu. A resource allocation strategy in a robotic ad-hoc network. In *IEEE International Conference on Autonomous Robot Systems and Competitions*, pages 122–127, 2014.

[4] M. Gianni, P. Papadakis, F. Pirri, M. Liu, F. Pomerleau, F. Colas, K. Zimmermann, T. Svoboda, T. Petricek, and G. J. M. Kruijff. A unified framework for planning and execution-monitoring of mobile robots. In *AAAI Conference on Automated Action Planning for Autonomous Mobile Robots*, pages 39–44, 2011.

[5] Ming Liu, Lujia Wang, and R Siegwart. Dp-fusion: A generic framework for online multi sensor recognition. In *Multisensor Fusion and Integration for Intelligent Systems*, pages 7–12, 2012.

[6] Hang Ma, Sven Koenig, Nora Ayanian, Liron Cohen, Wolfgang Hönig, TK Kumar, Tansel Uras, Hong Xu, Craig Tovey, and Guni Sharon. Overview: Generalizations of multi-agent path finding to real-world scenarios. *arXiv preprint arXiv:1702.05515*, 2017.

[7] Liron Cohen and Sven Koenig. Bounded suboptimal multi-agent path finding using highways. In *IJCAI*, pages 3978–3979, 2016.

[8] Liron Cohen, T Uras, TKS Kumar, Hong Xu, Nora Ayanian, and Sven Koenig. Improved bounded-suboptimal multi-agent path finding solvers. In *International Joint Conference on Artificial Intelligence*, 2016.

[9] Mike Phillips, Benjamin J Cohen, Sachin Chitta, and Maxim Likhachev. E-graphs: Bootstrapping planning with experience graphs. In *Robotics: Science and Systems*, volume 5, 2012.

[10] M. Liu. Robotic online path planning on point cloud. *IEEE Transactions on Cybernetics*, 46(5):1217–1228, 2016.

[11] Thulasi Mylvaganam, Mario Sassano, and Alessandro Astolfi. A differential game approach to multi-agent collision avoidance. *IEEE Transactions on Automatic Control*, 2017.

[12] Toshio Fukuda and Yoshio Kawauchi. Cellular robotic system (cebot) as one of the realization of self-organizing intelligent universal manipulator. In *Robotics and Automation, 1990. Proceedings., 1990 IEEE International Conference on*, pages 662–667. IEEE, 1990.

[13] Deividi Moreira, Fernando Santos, Matheus Barbieri, Ingrid Nunes, and Ana L.C. Bazzan. Abstractme: Modularized environment modeling in agent-based simulations. In *Proceedings of the 16th Conference on Autonomous Agents and MultiAgent Systems*, AAMAS '17, pages 1802–1804, Richland, SC, 2017. International Foundation for Autonomous Agents and Multiagent Systems.

[14] Roderich Groß, Michael Bonani, Francesco Mondada, and Marco Dorigo. Autonomous self-assembly in swarm-bots. *IEEE transactions on robotics*, 22(6):1115–1130, 2006.

[15] Sean Luke, Claudio Cioffi-Revilla, Liviu Panait, and Keith Sullivan. Mason: A new multi-agent simulation toolkit. In *Proceedings of the 2004 swarmfest workshop*, volume 8, pages 316–327. Department of Computer Science and Center for Social Complexity, George Mason University Fairfax, VA, 2004.

[16] Luigi Pannocchi, Mauro Marinoni, and Giorgio Buttazzo. Hardware-in-the-loop development framework for multi-vehicle autonomous systems. In *Autonomous Robot Systems and Competitions (ICARSC), 2017 IEEE International Conference on*, pages 17–22. IEEE, 2017.

[17] Cristian Peñaranda, Vicente Julian, Javier Palanca, and Vicente Botti. A multi-agent system to improve mobile robot localization. In *International Conference on Hybrid Artificial Intelligence Systems*, pages 471–482. Springer, 2017.

[18] Martin Schaefer, Jiří Vokřínek, Daniele Pinotti, and Fabio Tango. Multi-agent traffic simulation for development and validation of autonomic car-to-car systems. In *Autonomic Road Transport Support Systems*, pages 165–180. Springer, 2016.

[19] Tenda Okimoto, Tony Ribeiro, Damien Bouchabou, and Katsumi Inoue. Mission oriented robust multi-team formation and its application to robot rescue simulation. In *IJCAI*, pages 454–460, 2016.

[20] Massimo Cossentino, Carmelo Lodato, Salvatore Lopes, and Patrizia Ribino. Multi agent simulation for decision making in warehouse management. In *Computer Science and Information Systems (FedCSIS), 2011 Federated Conference on*, pages 611–618. IEEE, 2011.

[21] Stefan Schroeder, Michael Zilske, Gernot Liedtke, and Kai Nagel. Towards a multi-agent logistics and commercial transport model: The transport service provider's view. *Procedia-Social and Behavioral Sciences*, 39:649–663, 2012.

[22] Alessandro Farinelli, Elena Zanotto, Enrico Pagello, et al. Advanced approaches for multi-robot coordination in logistic scenarios. *Robotics and Autonomous Systems*, 90:34–44, 2017.

[23] Lujia Wang, Ming Liu, and Q. H. Meng. A hierarchical auction-based mechanism for real-time resource allocation in cloud robotic systems. *IEEE Transactions on Cybernetics*, PP(99):1–12, 2016.