第43卷 第1期 电子科技大学学报 Vol.43 No.1
2014年1月 Journal of University of Electronic Science and Technology of China Jan. 2014

Multi-Robot Learning Using PSO Combined with CBR Algorithm

LIU Qiang [1], MA Jia-chen [1,2], XIE Wei [2], and MA Li-yong [2]

(1.School of Astronautics, Harbin Institute of Technology Harbin 150001;

2. School of Information and Electrical Engineering, Harbin Institute of Technology(Weihai) Weihai Shandong 264209)

Abstract Case-based reasoning (CBR) which stores old problems and solution information as cases can solve new problems of the particle swarm optimization (PSO) with its long-term memory during the learning phase for multiple robots in an unknown environment. The PSO components which offer trainings to the robot in specially-designed simulation environments to deliver basic behaviors enhance their robustness and adaptivity. The CBR components which selects solution from the case base evolved for basic behaviors rank them according to their performance in the new complex enviroment and introduce them to a PSO algorithm's initial population, hence speeding up the learning process. Behavior-based multi-robot formation control task is employed as a platform to assess the effectiveness of this approach with robot simulation software MissionLab. Simulation and experimental results show that the CBR-injected PSO algorithm can quickly obtain optimal control parameters and multi-robot formation performs better in unknown environment.

Key words behavior-based; case-based reasoning; MissionLab; multi-robot formation; particle swarm optimization

混合粒子群优化算法和案例推理方法的多机器人学习

刘 强[1]，马家辰[1, 2]，谢 玮[2]，马立勇[2]

(1. 哈尔滨工业大学航天学院 哈尔滨 150001; 2. 哈尔滨工业大学(威海)信息与电气工程学院 山东 威海 264209)

【摘要】以未知环境下多机器人学习为研究平台，因案例推理方法可存储以前的问题和解信息，用该方法的长期记忆特性可帮助粒子群优化算法更好地解决新的问题。在特定的仿真环境里，粒子群优化算法可训练机器人的几个基本行为，经过学习使机器人具有更好的鲁棒性和自适应学习能力。根据机器人不同行为在复杂环境下的性能指标，CBR可从案例库中选择特定的行为，并将其参数传送到粒子群优化算法的初始解库，从而加速整体的学习过程。利用机器人仿真软件MissionLab，采用基于行为的多机器人编队任务，用来测试该算法的有效性。仿真和实验结果表明，案例推理方法和粒子群优化算法相结合，使机器人获得更优的控制参数，同时在未知环境下的多机器人编队具有更好的性能。

关 键 词 基于行为的方法; 案例推理; MissionLab仿真平台; 多机器人编队; 粒子群优化算法

中图分类号 TP273 文献标志码 A doi:10.3969/j.issn.1001-0548.2014.01.023

1 Introduction

A multi-robot system consists of a group of robots, which are organized into a multi-agent architecture so as to collaboratively perform a common task. Over the past decade, multi-robot systems have get more and more attention in robot research field because of its special capabilities like cooperative behavior, robustness, parallel operation, and scalability[1-3].

Formation control is one of the most important research subjects in multi-robot systems. It applies to many areas such as geographical exploration, rescue operations, surveillance, mine sweeping, and transportation. A lot of approaches have been proposed recently, such as behavior-based control, LQ control, visual servoing control, Lyapunov-based control, input and output feedback linearization control, graph theory, and nonlinear control[4]. In this paper, a behavior-based

Received date: 2012 – 05 – 29; Reviced date: 2013 – 01 – 15

收稿日期：2012 – 05 – 29; 修回日期：2013 – 01 – 15

Foundation item: Supported by Shandong Province Natural Science Foundation (ZR2011FM005)

基金项目：山东省自然科学基金(ZR2011FM005)

Biography: LIU Qiang was born in 1984, and his research interests include in intelligent robot and multi-robot system.

作者简介：刘强(1984 –)，男，博士生，主要从事智能机器人和多机器人系统方面的研究.

approach is adopted, which integrates the case-based reasoning (CBR) algorithm with particle swarm optimization (PSO) algorithm.

Reactive control systems have removed the apparent defaults of the deliberative system. Behavior-based robot reacts to stimulation from the local states of the world. These behaviors are closely related to the effectors that implement the behaviors of the robot[5]. The subsumption architecture proposed by Ref. [6] and the motor schema-based architecture proposed by Ref. [7] are both classical architectures. In this paper, each robot is designed to be the motor schema-based architecture and multi-robot formation control is adopted as a task. The behavior-based reactive system requires the selection and structuring of the control parameters, which underlies the behaviors of the robot. Although simpler than modeling a complex and dynamic environment, selecting parameters controlling robot behaviors can be complicated.

PSO is a a random and parallel search algorithm that searches from a population of points[8]. To date, PSO-based machine-learning system can only acquire optimized outcome which has been deprived of former experiences. This leads to its inability in improving their performance persistently[9]. However, a lot of application areas are more suitable for case-based storage of historical experience[10-11]. The CBR algorithm can offer solutions to many problems in knowledge engineering such as knowledge elicitation, encoding and maintenance, and it is often viewed as a low-risk way[12].

This paper describes the application of combining CBR and PSO algorithms in solving the problem of optimizing control parameters of multi-robot formation navigation. The PSO algorithm provides an unsupervised learning method which greatly reduces the effort of the designer in configuring a navigation system. The CBR algorithm can save good parameters from PSO and inject appropriate cases into the initial population of the PSO algorithm, which not only speeds up convergence but also provides higher quality solutions. Our approach is to train robots of behavior-based reactive control of multi-robot formation in various types of environments, thus creating a set of optimized cases (control parameters) which can be used in similar environments, including those that are not presented in the learning phase.

2 Review of CBR and PSO algorithm

2.1 Particle swarm optimization algorithm

Ref. [13] first introduced PSO algorithm[13]. By adopting idea of swarms in the nature such as birds and fish, the PSO algorithm was proposed. PSO has particles driven from natural swarms with communications based on evolutionary computations. A taxonomy of the PSO algorithm was presented in Ref. [14] which classified the elements of the PSO algorithm into four main groups: variables, particles, swarm and process[15].

The particles of PSO that show the solution candidates start their movement from stochastic positions in a search area. In each iteration, particles update their position according to:

$$\mathrm{prtpos}_j^i = \mathrm{prtpos}_j^{-1} + \mathrm{prtvel}_j^i \tag{1}$$

$$\mathrm{prtvel}_j^i - x[\omega \mathrm{prtvel}_j^{i-1} + c_1 r_1 (\mathrm{pbest}_j^{i-1} + \mathrm{prtpos}_j^i) \times c_2 r_2 (\mathrm{gbest}^{i-1} - \mathrm{prtpos}_j^{i-1})] \tag{2}$$

in which prtpos_j^i = the position of j^{th} particle in i^{th} iteration, prtvel_j^i = the velocity position of j^{th} particle in i^{th} iteration, pbest_i^j = the best position of j^{th} particle till i^{th} iteration, gbest^i = the best position of the swarm, so:

$$x = \frac{2}{\left|2 - j - \sqrt{j^2 - 4j}\right|} \qquad j > 4$$

The particles shift from their current position to another in accordance with the above updating equation and their movement is influenced by a fitness function that evaluates the quality of each solution. As shown in Equ. (2), PSO has several dependent parameters. Factors c_1 and c_2 are capable of balancing the effect of self-knowledge and social knowledge when the particle moves toward the target, and they are usually set to 2, though good results have been also produced with c_1 = c_2 = 4[16]. r_1 and r_2 are random numbers varying from 0 to 1, at different iteration, and χ is a constriction factor to limit the velocity[15].

ω is used to regulate the global search behavior.

It can be set to a large value at the beginning of the searching process and it dramatically decreases through any predefined scheme during the optimization. It ranges from 0.2 to 0.4. Several benefits can be obtained from dynamic adjustment. Firstly, it facilitates the convergence to an optimal solution. Secondly, it controls the impact of previous part velocities on current velocities, which can adjust the tradeoff between the capability of swarms in local and global exploration. Fig. 1 illustrates a schematic view of how the position of a particle in two successive iterations is updated[15]. The detailed description of PSO can be found in Ref. [13].

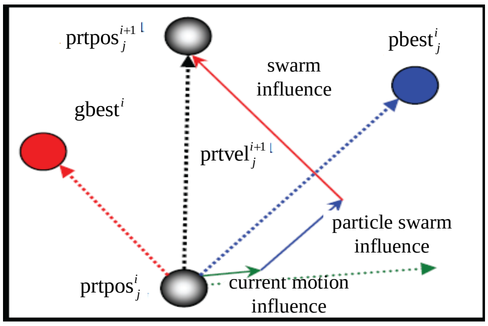

Fig. 1　depiction of a particle's position update in PSO

2.2　Case-based reasoning

Ref. [17] gives a detailed description of the Learning CBR module. In this section, only a high level overview of the module is given. The overall structure of the CBR unit is similar to the non-learning case-based reasoning system[18].

The following can be founded in Ref. [19]. The feature identification sub-module of CBR unit captures the sensor data and goal information. The spatial feature vector representing the relevant spatial characteristics of the environment and the temporal features vector representing relevant temporal characteristics can be computed. The purpose of these two vectors is to select a best matching case.

Case selection can be completed in three steps. Initially, all the cases will be selected and weighted. Euclidean distances between their spatial feature vectors and environmental spatial feature vector will be computed. It defines spatial similarities of cases with the environment. Euclidean distances between their spatial feature vectors and environmental spatial feature vector will be computed, which defines spatial similarities of cases with the environment. The case which has the highest spatial similarity of the best spatially matching case changes into the best spatially matching case. But all the cases with a spatial similarity within certain limits from the similarity of the best spatially matching case are selected for the following phase selection procedure. The resultant is called a spatially matching case. At the second phase of selection, all the spatially matching cases are seeked and weighted. Euclidean distances between their temporal feature vectors and the environmental temporal feature vector are computed. It defines temporal similarities of cases with the environment. The case that has the highest temporal similarities is called the best temporally matching case. Then all the cases with a temporal similarity within certain limits from the similarity of the best temporally matching case are selected for the next stage selection process. At the final selection phase, a case from the set of spatially and temporally matching cases is selected randomly[19].

3　Behavior-based multi-robot formation control

There are two parts in designing a behavior- based reactive control architecture: structure and a set of control values. The tasks that the robot must accomplish determine the structure, which constraints the collection of behaviors that the robot can present. After the structure of the system was defined, the system was tuned by adjusting the parameters that control the behaviors[20].

In this paper, five input vectors to characterize the environment around the robot in multi-robot formation are adopted, which discriminate among different environment configurations. Obstacle-density provides a measure of the occupied areas that impede navigation. Absolute-motion measures the activity of the system and relative-motion represents the change in motion activity. Space between robots denotes the distance between robots and motion-towards-goal specifies how much progress the system has actually made towards the goal. These input vectors are continually updated in accordance with the information received from the sensors of robot[21].

For achieving formation control, two primary aspects deserve our attention: how to control each robot to stay in formation and how to produce avoidance collision trajectory. Here several motor schemas, such as move_to_goal, keep_formation, avoid_static_ obstacle and avoid_robot, are adopted to realize the overall behavior for a robot to move to a designated location without clashing with obstacles, other robots and staying in formation.

Eight output vectors representing the schema parameter values are used to adapt to the formation control module[21]. These parameters are as follows: obstacle_gain, obstacle_sphere, obstacle_safety_ margin, avoid_robot_gain, avoid_robot_sphere, avoid_ robot_min_range and move_to_goal_gain. These values are set periodically according to the new case which best matches the current environment. The new values remains unchanged until the following setting period.

The gain parameters are the multiplicative weights of the corresponding schemas. Obstacle_ sphere controls the distance within which the robot reacts to obstacles with the avoid_static_obstacle schema. Obstacle_safety_margin controls the distance at which inter-robot collisions are assumed to occur. Avoid_robot_sphere controls the distance beyond which other robots are not considered by avoid_robot schema. Avoid_robot_min_range controls the distance within which the repulsion from another robot is set a maximum value. Therefore, a case in a library is a set of values for the above parameters.

4 Control parameters selection with CBR-injected PSO

The following is based on Ref. [22]. In this paper, we define experience *E* with respect to behavior-based multi-robot formation control task *T* and performance measure *P* if its performance of *T*, as measured by *P*, improves with experience *E*.

Using above notion, *T* is denoted as the task with solutions found in the search space which is defined by the set *S*. Casting our definition of the task in terms of search is provided to the PSO by objective function *O* corresponding to the task *T*, which maps candidate solutions to the set of real numbers. That is:

$$\langle O:\{S \to R\}\rangle \tag{3}$$

where *R* is the set of real numbers indicating the range of mapping from *S* to *R*. The PSO algorithm intends to maximize the fitness function which is an objective function that maps objective function values to the set of nonnegative real numbers $R \geqslant 0$. Here *F* is a collection of maps from points in *S* to $R \geqslant 0$:

$$\langle F:\{S \to R \geqslant 0\}\rangle \tag{4}$$

CBR combined with PSO gets experience by adding novel cases (control parameters) from PSO. Performance can be tested by the time taken to solve task P^t and by the quality to solve P^q.

When robot encounters new conditions, the CBR module searches for similar input vectors and their associated output vectors (control parameters). Note that CBR research has shown that setting up a problem similarity mechanism is important[23].

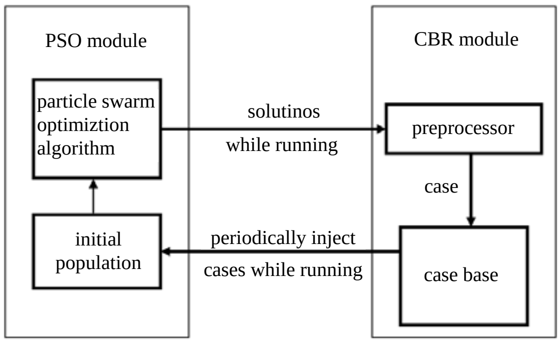

Fig. 2 conceptual view of CBR-injected PSO

Once similar cases (input vectors) are found, some solutions (control parameters) will flow into the initial population of the PSO, which are called case initialization. The rest of the population are initialized randomly so as to maintain diversity and the PSO searches from the entire combined population. Fig.2 shows a conceptual figure of CBR combined with PSO. As the figure shows that while the PSO runs on a new input vectors of surrounding, nice individuals of the population are stored into the case base after preprocessor. Next, when work on a new input vectors of surrounding starts, appropriate cases are picked out from the case base and used to populate a small fraction of the initial population. In this paper, a case is a member of the population (a set of candidate control parameters) together with ancillary information including its fitness and the timestep at which this case was generated[24]. During the PSO search, whenever the fitness of the best individual in the population increases, the new best individual is stored in the case base[22].

After CBR combined with PSO runs, the CBR module periodically pours some solutions (control parameters) similar to the current best member of the PSO population into the current population in order to displace the worst counterparts. The PSO unit keeps searching with this combined population.

5 Simulation and experiment

Simulation were achieved by Georgia Tech's robot simulation software MissionLab[25]. MissionLab runs on Unix machines (SunOS or Linux) by using the X_{11} graphical windows system, a powerful set of software tools for developing and testing behaviors for single or a group of robots. Code generated by MissionLab can directly control commercial robots. ATRV-Jr/Urban Robot (iRobot), AmigoBot/Pioneer AT/Pioneer 2DX (ActivMedia, Inc.), and Nomad-150/200 (Nomadic Technologies, Inc.) are among those robots which MissionLab has supported[23].

The simulation environment is a 1 000 by 1 000 meters two dimensional field in which various sizes of circular obstacles can be scattered. Each simulated robot is a separately running C program that interacts with the simulation environment via a Unix socket. The simulation displays the environment graphically and maintains world state information which transmits to the robot as requested. In the following simulation, diamond formation of four robots as a team is implemented by adopting the unit-center-referenced approach. The diamond formation as whole moves from strating point which begins inside a rectangle area to destination point goal.

Simulation 1 is designed as follows. Four robots in diamond formation are commanded to travel between two points: point “begin” inside a rectangle square and point “goal”, which is 800 m apart. Obstacles are placed randomly so 3% of the total area is covered with obstacles 2 to 10 meters in diameter. Based on the CBR-injected PSO algorithm, the near-optimal behavior parameters of robots' reactive control are obtained.

At the beginning of run, the libraries of CBR do not contain any cases and are created as the robot proceeds with its mission. As a result, the performance P^t (final time of all robots arriving destination for one trial) and P^q (average length of all robots running for one trial) of the robots in training runs are poor, as shown in Fig.3 and Fig.4. The search for optimal parameterization has just started in these runs and thus the robot's behavior is very noisy. In contrast, after about sixty training runs in this environment, the robot gradually learns more optimal parameterizations.

To test the performance of CBR combined with the PSO algorithm, multi-robot formation in unknown enviroment taken into form after learning process. Simulation 2 is designed to verify it, where a more complexible unknown environment for robots, which has 15% of the total area covered with obstacles 1 to 20m in diameter, is adopted by MissionLab. Simulation results show that the robots' trajectory in the final run is far better that each robot can better adjust its combined direction of behaviors to arrive at destination while avoiding obstacles and collisions with other robots, and keeping in diamond formation as quickly as possible (P^t =1 436 ms, P^q =917 m).

As mentioned above, Missionlab is a flexible application that at run time a researcher may choose between a simulated run and a run on physical robots. The same behavioral control code is used both in simulation and controlling the robots. The experimental platform for the results reported here are Nomad 150 robots, which are three-wheeled holonomic robots epuipped with a separatedly steerable turret and 16 ultrasonic range sensors for hazard detection and are controlled by on-board laptop computers running Linux. They communicate over a wireless network supporting Unix Sockets via TCP/IP.

Experiments were conducted in the Mobile Robot Laboratory to demonstrate formation performance on mobile robots and validate the quantitative results from simulation experiments. Experimental enviroment is a test area of approximately 20 by 10 meters which is a smaller test area versus simulation enviroment. The robots formation were commanded to navigate from West to East across the room. Sixty runs were conducted for diamond formation by the unit-center-referenced approach. These robots estimate their position with their shaft encoders. In order to communicate with the formation’s unit-center, each robot broadcasts its position to the other over a wireless network. The performance P^t and P^q of Nomad 150 robots’ formation in the learning phase is analogous to the simulation results, as shown in Fig.5 and Fig.6. After learning process is finished and more obstacles are added, Nomad 150 robots’ formation can obtain a nice performance (P^t =12.3 s, P^q =23.6 m).

This experiment tests and verifies the simulation result that the formation of robots with on-line learning capability including CBR and PSO demonstrates good flexibility and adaptivity in unkown environments, while effectively completing a formation task.

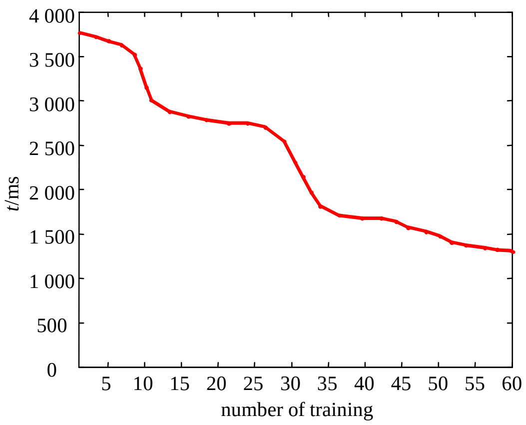

Fig. 3 final time P^t of all robots arriving destination for one trial

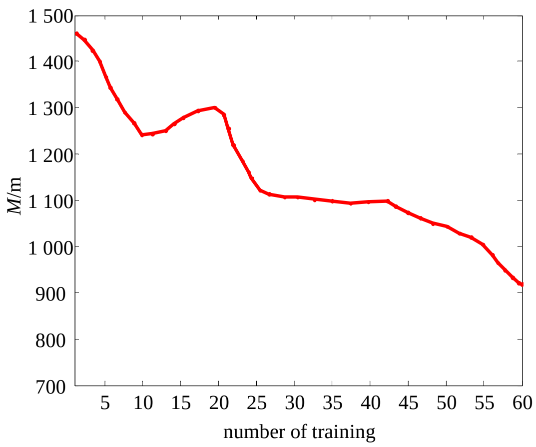

Fig. 4 average length P^q of all robots running for one trial

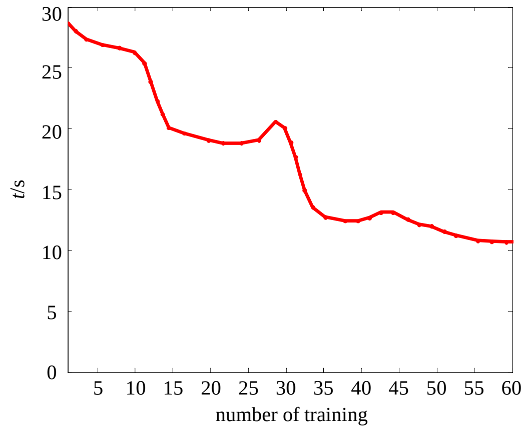

Fig. 5 performance P^t of Nomad 150 robots formation in experiment

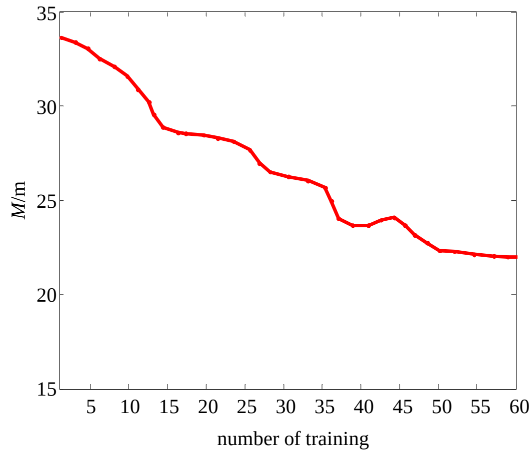

Fig. 6 performance P^q of Nomad 150 robots formation in experiment

6 Conclusions

In this paper, the PSO algorithm and CBR algorithm are combined to function as formation navigation for a team of robots. Four behaviors are presented and the parameters which control these behaviors are set autonomously with a CBR-injected PSO approach. The approach leads to a better performance of the robot in comparison to a non-adaptive system. After the combination of CBR combined and PSO, the process of library of CBR configuration becomes full-automatic through training. There is no need to set any configuration of behavioral parameters, such as creating an initial case-based reasoning library. The more missions are accomplished, the better the parameterization becomes, and the better robots’ performance we will get.

References

[1] ARAI T, PAGELLO E, PARKER L E. Editorial: Advances

in multi-robot systems[J]. IEEE Transactions on Robotics and Automation, 2002, 18(5): 655-661.
[2] CAO Y U, FUKUNAGA A S, KAHNG A B, et al. Cooperative mobile robotics: Antecedents and directions[J]. Autonomous Robots, 1997(4): 1-23.
[3] LIU J, WU J. Multi-agent robotic systems[M]. Boca Raton, FL, USA: CRC Press, 1993.
[4] CERNEGA D, SOLEA R. Hybrid control structure for multi-robot formation[M]. Berlin: Springer Heidelberg, 2010.
[5] KAELBLING L P. An architecture for intelligent reactive systems[J]. Reasoning about Actions and Plans, 1987, 1(2): 395-410.
[6] BROOKS R. A robust layered control system for a mobile robot[J]. IEEE Journal of Robotics and Automation, 1986, 2(1): 14-23.
[7] ARKIN R C. Motor schema-based mobile robot navigation[J]. The International Journal of Robotics Research, 1989, 8(4): 92-112.
[8] AGUIRRE A H, ZAVALA A M, DIHARCE E V, et al. COPSO: Constrained optimization via PSO algorithm[R]. 1-07-04/22-02-2007. Mexico: Center for Research in Mathematics (CIMAT), 2007.
[9] BARRERA A J, PENA J, HINCAPIÉ R. Subpopulation best rotation: a modification on PSO[J]. Revista Facultad de Ingeniería Universidad de Antioquia, 2007(40): 118-122.
[10] SYCARA K, NAVINCHANDRA D. Retrieval strategies in a case-based design system[J]. Artificial Intelligence in Engineering Design, 1992, 2(7): 145-160.
[11] GOEL A, CHANDRASEKARAN B. Case-based design: a task analysis[J]. Artificial Intelligence Approaches to Engineering Design, 1992(2): 165-184.
[12] WATSON I, MARIR F. Case-based reasoning: a review[J]. Knowledge Engineering Review, 1994, 9(4): 327-354.
[13] KENNEDY J. Particle swarm optimization[M]. Berlin: Springer, 2010: 760-766.
[14] SEDIGHIZADEH D, MASEHIAN E. A new taxonomy for particle swarm optimization (PSO)[C]//Proc 10th International Conference on Automation Technology. Tainan, Taiwan, China: National Cheng Kung University, 2009.
[15] MASEHIAN E, SEDIGHIZADEH D. A multi-objective PSO-based algorithm for robot path planning[C]//Industrial Technology (ICIT). Busan, Korea: [s.n.], 2010: 465-470.
[16] SHI Y, EBERHART R C. Particle swarm optimization with fuzzy adaptive inertia weight[C]//Evolutionary Computation. Berlin: Springer Heidelberg, 2001: 101-106.
[17] LIKHACHEV M, ARKIN R C. Spatio-temporal case-based reasoning for behavioral selection[C]//Robotics and Automation. Piscataway, USA: IEEE, 2001: 1627-1634.
[18] ALTHOFF K D. Case-based reasoning[J]. Handbook on Software Engineering and Knowledge Engineering, 2001(1): 549-587.
[19] LIKHACHEV M, KAESS M, ARKIN R C. Learning behavioral parameterization using spatio-temporal case-based reasoning[C]//Robotics and Automation, Proceedings ICRA'02. Piscataway, USA: IEEE, 2002: 1282-1289.
[20] RAM A, BOONE G, ARKIN R, et al. Using genetic algorithms to learn reactive control parameters for autonomous robotic navigation[J]. Adaptive Behavior, 1994, 2(3): 277-305.
[21] RAM A, SANTAMARIA J C. A multistrategy case-based and reinforcement learning approach to self-improving reactive control systems for autonomous robotic navigation[C]//Proceeding of the Second International Workshop on Multistrategy Learning. Fairfax, Virginia, USA: George Mason University, 1993: 17-34.
[22] LOUIS S J, MCDONNEL J. Learning with case-injected genetic algorithms[J]. IEEE Transactions on Evolutionary Computation, 2004, 8(4): 316-328.
[23] BALCH T, ARKIN R C. Behavior-based formation control for multirobot teams[J]. IEEE Transactions on Robotics and Automation, 1998, 14(6): 926-939.
[24] LOUIS S, MCGRAW G, WYCKOFF R O. Case-based reasoning assisted explanation of genetic algorithm results[J]. Journal of Experimental & Theoretical Artificial Intelligence, 1993, 5(1): 21-37.
[25] MACKENZIE D C, ARKIN R C, CAMERON J M. Multiagent mission specification and execution[J]. Autonomous Robots, 1997, 4(1): 29-52.

编 辑 黄 莘