doi: 10.12052/gdutxb.200009

一种提升机器人强化学习开发效率的训练模式研究

叶伟杰，高军礼，蒋　丰，郭　靖

（广东工业大学　自动化学院，广东　广州 510006）

摘要： 强化学习与深度学习结合的深度强化学习(Deep Reinforcement Learning, DRL)模型，目前被广泛应用于机器人控制领域。机器人强化学习需要在3D仿真环境中训练模型，然而在缺乏环境先验知识的情况下，在3D环境中进行试错学习会导致训练周期长、开发成本高的问题。因此提出一种贯通2D到3D的机器人强化学习训练模式，将计算量大、耗时多的工作部署到2D环境中，再把算法结果迁移到3D环境中进行测试。实验证明，这种训练模式能使基于个人电脑的机器人强化学习的开发效率提升5倍左右。

关键词： 深度强化学习；机器人控制；训练模式；开发效率

中图分类号： TP242.6　　**文献标志码：** A　　**文章编号：** 1007–7162(2020)05–0046–05

A Research on a Training Model to Improve the Development Efficiency of Robot Reinforcement Learning

Ye Wei-jie, Gao Jun-li, Jiang Feng, Guo Jing

(School of Automation, Guangdong University of Technology, Guangzhou 510006, China)

Abstract: Deep reinforcement learning (DRL) model combining reinforcement learning and deep learning is currently widely used in the field of robot control. Robot reinforcement learning needs to train the model in a 3D simulation environment. However, in the absence of prior environmental knowledge, trial and error learning in a 3D environment leads to long training cycles and high development costs. To solve this problem, a training mode from 2D to 3D is proposed. Time-consuming and computationally intensive work is completed in a 2D environment, and the results are transferred to a 3D environment for testing. Experiments show that this training mode can improve the development efficiency by about five times, so that personal computers can also do research related to robot reinforcement learning.

Key words: deep reinforcement learning; robot control; training mode; development efficiency

自从DeepMind的AlphaGo击败世界围棋冠军李世石[1]，强化学习开始成为主流研究方向之一，其中强化学习结合机器人控制具有良好的发展前景。强化学习与传统的机器人控制不同，它是通过与环境交互的方式学习指定任务且不依赖于精确的数学模型[2]。安全起见，机器人训练过程一般在3D仿真平台上进行。由于缺乏环境的先验知识，机器人要在3D环境中进行大量的交互学习，才能逼近真实的环境模型。该交互过程受限于仿真器的物理引擎，整个过程非常漫长，如在MuJoCo中实现机器人跑酷就需要64个CPU同时训练100 h[3]；Google用14个机械臂训练了2个月来实现抓取功能[4]。机器人强化学习训练时间太长，增加了实验验证、调参等工作时间成本。

针对机器人强化学习计算量大，训练时间长的问题，部分学者通过优化算法结构来提升训练速度，如DeepMind融合DQN(Deep Q-Learing Network)的各种变异算法来提升训练速度并在Atari游戏上取得了最高分数[5]；Schulman等[6]改良了梯度非负递增的数学式，解决了传统策略梯度算法数据利用率低下的问题。而有的学者则通过改变平台来应对该问题，如Wang等[7]把复杂的计算托管在云端，本机只做算法设计与编程；吴运雄等[8]则把小车跟踪和避障的研究放在2D环境中。还有部分学者如Stooke等[9]通过调整算法使其可以更好地并行利用CPU和GPU，从而提

收稿日期： 2020-01-09

基金项目： 国家自然科学基金资助项目(61803103)；国家留学基金资助项目(201908440537)

作者简介： 叶伟杰(1994–)，男，硕士研究生，主要研究方向为深度强化学习、路径规划，E-mail：380400483@qq.com

高硬件的利用率，这种深度强化学习的并行技术有效提高了训练速度。

这些方法从不同方面缓解计算资源的问题，减少了算法训练的时间，但仍然需要服务器级别的计算力。本文针对个人计算机在机器人强化学习上开发效率的局限，提出一种新的训练模式，即把计算量大、耗时的工作部署到2D环境中，把算法结果测试部署到3D环境中，通过这种2D到3D的训练模式，能有效提高机器人强化学习的开发效率。

针对机器人导航，文献[10-11]分别用笛卡尔坐标和极坐标描述位置信息，而强化学习通过回报函数决定机器人的学习目标。因此，本文设计了3种坐标系和2套回报函数，以对比实验的方式分析它们对机器人导航的影响，并在所提出的机器人强化学习训练模式中进行了快速验证，效果良好。

1　强化学习算法与仿真环境

1.1　强化学习与TD3算法

强化学习的学习过程如图1所示，在t时刻，智能体根据观测状态$\boldsymbol{s}_t$，选择动作$\boldsymbol{a}_t$与环境交互,获得奖励r_t和新的观测状态$\boldsymbol{s}_{t+1}$后，存储这次交互产生的数据$(\boldsymbol{s}_t,\boldsymbol{a}_t,r_t,\boldsymbol{s}_{t+1})$，并通过这些数据来优化策略。强化学习的目标是最大化累积奖励$R(\tau)=\sum_{t=0}^{\infty}\gamma^t r_t$，$\gamma^t$为折扣因子，在(0,1)间取值，表示未来回报的影响低于当前时刻的奖励。

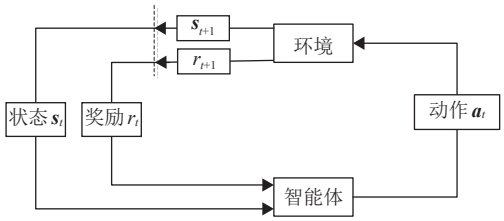


图 1　强化学习框架

Fig.1　Reinforcement learning framework

传统强化学习可以分为2类：基于策略与基于价值函数。基于策略的方法能处理连续输出的问题，但数据利用率低下；基于价值函数的方法恰相反，数据能重复利用却只能处理离散输出的问题。本文采用的TD3(Twin Delayed Deep Deterministic Policy Gradient)算法能很好地平衡传统算法的优缺点，如式(1)和式(2)所示。

$$\nabla_\phi \frac{1}{|\boldsymbol{B}|}\sum_{(\boldsymbol{s}_t,\boldsymbol{a}_t,r_t,\boldsymbol{s}_{t+1})\in\boldsymbol{B}}(\boldsymbol{Q}_\phi(\boldsymbol{s}_t,\boldsymbol{a}_t)-y(r_t,\boldsymbol{s}_{t+1}))^2 \tag{1}$$

$$\nabla_\theta \frac{1}{|\boldsymbol{B}|}\sum_{\boldsymbol{s}\in\boldsymbol{B}}\boldsymbol{Q}_\phi(\boldsymbol{s},\boldsymbol{\mu}_\theta(\boldsymbol{s})) \tag{2}$$

其中$\boldsymbol{Q}_\phi(\boldsymbol{s}_t,\boldsymbol{a}_t)$是价值函数，$\boldsymbol{\mu}_\theta(\boldsymbol{s})$是策略函数，它们由深度神经网络表示，网络结构如图2所示，每层由网络连接方式、维度和激活函数组成，$\boldsymbol{B}$则是每回合的采样数目。$\boldsymbol{Q}_\phi(\boldsymbol{s}_t,\boldsymbol{a}_t)$通过最小化贝尔曼均方差更新，$\boldsymbol{\mu}_\theta(\boldsymbol{s})$则通过$\boldsymbol{Q}_\phi(\boldsymbol{s}_t,\boldsymbol{a}_t)$更新，2个网络的学习率$l_r$相同。TD3结合传统算法优点的同时，也通过同时学习两个$\boldsymbol{Q}_\phi(\boldsymbol{s}_t,\boldsymbol{a}_t)$网络解决过估计的问题，算法具体细节参考文献[12]。TD3是目前深度强化学习中效果最优的算法之一，也是本文所采用的算法。

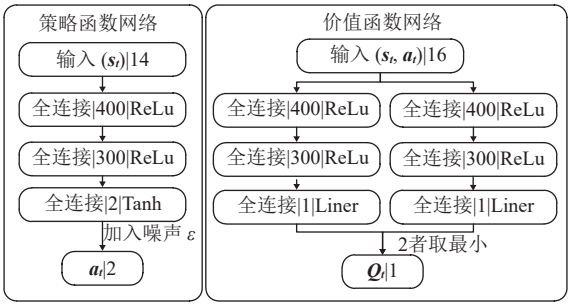
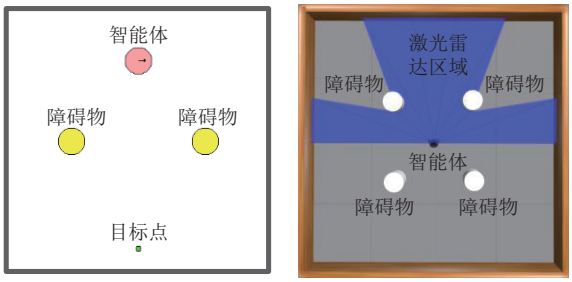


图 2　TD3的网络结构

Fig.2　The network structure for the TD3

1.2　仿真环境

2016年5月，OpenAI 开源了强化学习研究工具Gym[13]，它集成各种类型的测试环境，本文所采用的2D环境就是基于Gym中的经典控制环境。如图3(a)，该环境只有4面墙与2个黄色移动障碍物，Agent需要在最短的时间内避开障碍物到达目标点。障碍物位置为已知信息，以此代替传感器信息，这样既能保证轻量级计算，同时也兼顾了实际导航场景的基本特性，使之能有效迁移到3D环境。



(a) Gym　　(b) Gazebo

图 3　仿真环境

Fig.3　Simulation environment

对于3D环境，采用的是机器人操作系统ROS (Robot Operating System)自带的3D仿真器Gazebo[14]，它含有物理引擎和多种传感器插件。本文采用激光雷达作为传感器，设观测范围为0°~180°，机器人平台采用Turtlebot3，环境设置如下：在橙色围墙里有4个白色圆柱障碍物，Turtlebot3处于中央位置，蓝色

区域表示雷达的扫描范围，如图3(b)。2个场景的目标点都随机生成。

2 2D to 3D的训练模式

2.1 问题描述

机器人导航是指在给定目标后，要用尽量短的时间并无碰撞地到达指定地点，细分到每一步就是根据当前观测来修改自身的运动状态以满足任务要求，该过程如式(3)所示。

$$\boldsymbol{v}_t = f(\boldsymbol{s}_t) = f(\boldsymbol{x}_t, \boldsymbol{p}_t, \boldsymbol{v}_{t-1}) \tag{3}$$

其中，函数$f(*)$表示规划器，$\boldsymbol{x}_t$表示观测参数，在Gym中它表示障碍物的坐标，在Gazebo中它表示10维激光雷达数据，$\boldsymbol{p}_t$表示目标位置，$\boldsymbol{v}_{t-1}$表示上一时刻机器人的运动速度,由线速度和角速度组成。根据Turtlebot3的机械特性，本文将线速度固定为0.15 m/s,角速度则在(–1.5，1.5)之间任意取值，单位为 rad/s。

不同的输入状态与回报函数对强化学习有不同的影响，因此需要一个轻量级的计算平台快速验证实验猜想，另外还需要一个3D平台来测试功能。

2.2 2D平台测试

位置信息可以由不同的坐标系表示，虽然直观来说极坐标更方便表示物体间的空间关系，但在导航中还是会有不一样的选择，如文献[10-11]。因此本文设置3种坐标系来表示输入状态中的位置信息，分别为笛卡尔坐标系、极坐标系、混合坐标系。通过实验验证不同坐标系对导航的影响。

此外，强化学习中的回报函数直接反映了所要学习的目标，在导航场景中，本文的回报函数设置如式(4)所示。

$$r(\boldsymbol{s}_t, \boldsymbol{a}_t) = \begin{cases} r_a, & \text{if } D_t < D_g \\ r_c, & \text{if } D_t < D_{\min} \\ \alpha(D_g - D_t), & \text{otherwise} \end{cases} \tag{4}$$

除了到达目标给以正奖励，碰撞给以负奖励外，本文还把小车到目标点之间的距离取负作为当前奖励，以此驱使小车快速到达目标，防止原地打转。

借鉴于ROS在导航时会把地图中的障碍物做膨胀处理，本文给小车外围也加了一层膨胀层，障碍物在膨胀层外不受额外惩罚，但当障碍物进入膨胀层时会有额外的惩罚，如式(5)所示。其中α和β需要多次调参，保证导航效率与安全性的平衡。

$$r(\boldsymbol{s}_t, \boldsymbol{a}_t) = \begin{cases} 0, & \text{if } D_t \geqslant D_i \\ \beta\frac{D_t - D_i}{D_i}, & \text{otherwise} \end{cases} \tag{5}$$

上述的实验对比和调参等工作都在轻量级的Gym上完成。

本文所提到的参数说明均如表1所示。

表 1 实验中的所有变量说明

Table 1 Description of All Variables in the Experiments

变量	数值	定义
r_a	200	到达目标所获得的奖励
r_c	150	碰撞所得奖励
D_t/m	d_t	当前小车与目标间的距离
$D_{\min}$/m	$d_{\min}$	当前小车到最近障碍物的距离
D_g/m	0.15	判定是否到达目标的阈值
D_i/m	0.25	小车膨胀层距离
B	64	每回合的最小采样数
α	0.25	距离奖励的缩放比例
β	10	膨胀层奖励的缩放比例
γ	0.99	未来奖励的折扣因子
l_r	0.001	策略函数和价值函数神经网络的学习率

2.3 3D平台测试

把在Gym中得到算法结果迁移到Gazebo中，测试是否能在静态和动态环境中完成导航。该训练模式的实验流程为

[1]: 输入：智能体和目标位置
[2]: 输出：包含部署和回报的结果result
[3]: initiate TD3 参数
[4]: set Env={Gym, Gazebo}
[5]: set Methods={笛卡尔坐标, 极坐标, 混合坐标}
[6]: for m ∈ Methods do
[7]: step←0，reward←0
[8]: while step < 500 k do
[9]: agent与环境交互，采集数据，更新算法
[10]: step←step+1
[11]: reward +←reward
[12]: end while
[13]: store steps和rewards数据到result
[14]: end for
[15]: 选择收敛性最好的坐标系
[16]: 修改回报函数，加上膨胀层，重复[5]~[11]步
[17]: 将算法设计迁移到Gazebo中进行导航测试
[18]: return result

3 实验验证与分析

3.1 Gym与Gazebo的对比实验

3.1.1 2D to 3D的可行性验证

2D环境能取代3D环境做算法前期验证，是因为2D环境保留了3D环境的基本特性。为了验证2D环境所做实验和3D环境所得的结果一致，本文另外搭建了一个简化版的3D环境，如图4所示。该环境是图3(b)的简化版，只有一个静止的灰色障碍物和固定的初始点和目标点，蓝色区域依然表示激光扫描范围，在

这个环境中重复3种坐标对比实验，以及膨胀层对导航的影响实验。

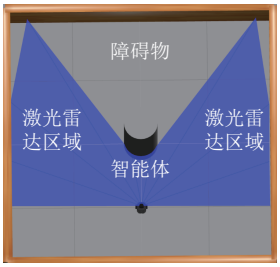


图 4　简化的Gazebo环境

Fig.4　Simplified Gazebo environment

3.1.2　3种坐标系对比实验

分别用笛卡尔坐标系、极坐标系和混合坐标系表示位置，回报函数设置如式(4)。在Gym中通过TD3算法训练Agent，整个过程持续5×10^5步，在简化的Gazebo环境中训练10^5步，记录回报变化趋势。训练结果如图5~图6所示，可得，在Gym和Gazebo环境中用极坐标系表示位置信息时收敛性都是最好。

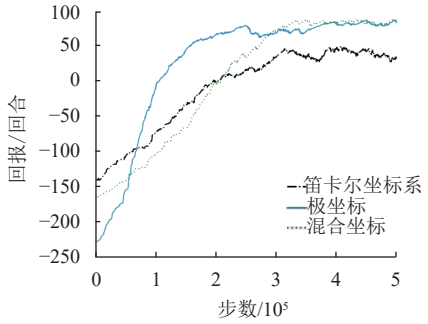


图 5　Gym中不同坐标系的收敛性

Fig.5　Convergence between different coordinates in Gym

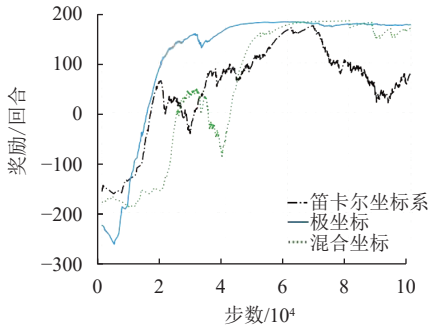
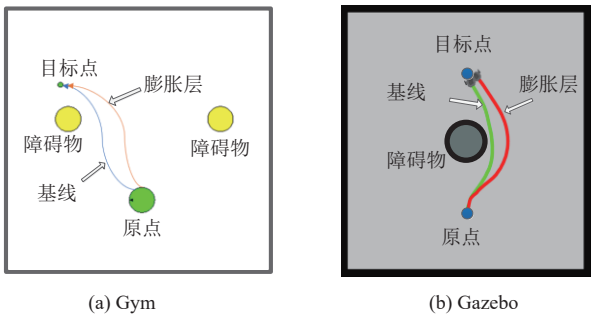


图 6　Gazebo中不同坐标系的收敛性

Fig.6　Convergence between different coordinates in Gazebo

3.1.3　膨胀层对比实验

基于以上结果，选用极坐标系表示位置信息，再在回报函数中增加膨胀层作为对比，其他设置同上。从图7可知，在Gym和Gazebo两个环境中，膨胀层的设置都会对小车的轨迹有显著影响。有膨胀层的路径会远离障碍物，而没有膨胀层的路径较为贴近障碍物。



(a) Gym　　(b) Gazebo

图 7　膨胀层对比实验

Fig.7　Expansion layer contrast experiment

从上面两个对比实验可知，Gym和Gazebo实验所得的结果基本一致，再根据表2分析两者算法训练时间，可知，即使是简化的Gazebo环境也比在Gym中的训练时间多，这证明了2D到3D的训练方式的可行性，既得到指导性的结果也节省了时间。

表 2　不同条件下的训练时间

Table 2　Training time under different conditions

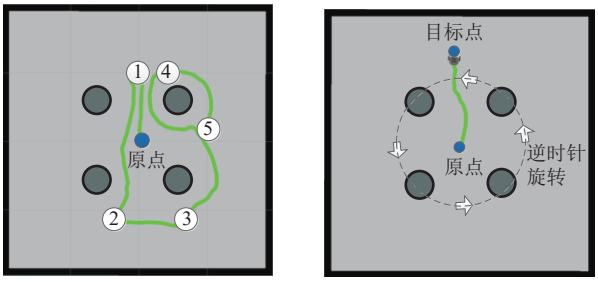
仿真环境	回报函数	坐标系	步数/10^5	时间/h
Gym	无膨胀层	笛卡尔坐标系	5	5.12
		极坐标系	5	4.46
		混合坐标系	5	4.51
	膨胀层	极坐标系	5	4.45
Gazebo	无膨胀层	笛卡尔坐标系	1	8.53
		极坐标系	1	8.48
		混合坐标系	1	8.37
	膨胀层	极坐标系	1	8.39

3.2　Gazebo最终测试

将极坐标系和膨胀层的模式应用到图3(b)的Gazebo环境上，每个回合目标点和初始点位置都是随机产生，训练5×10^5步，耗时约为25 h，在静态环境和动态环境中分别测试Turtlebot3的导航性能，运动轨迹如图8所示。图8(a)是静态场景，从起始点出发，随机生成5个目标点，黑色区域表示障碍物，绿色轨迹表明小车能依次避开障碍物且能应对任意的合理目标点；图8(b)黑色障碍物沿白色箭头方向逆时针旋转，绿色轨迹表明小车在观测到有障碍物靠近时依然能避开动态障碍物并到达终点。

3.3　结果分析

本文实验均在i5，1.8 GHz，8 G内存的笔记本中完成，从图5~图6可知，3种坐标系都能收敛，但极坐标的收敛性最好，这也符合直观认知。实际上网络结构和数据利用率低等算法自身问题对训练时间的影响更为直接，相比之下，在拥有服务器级别的实验场景中，选择哪种坐标系就没那么重要。但对于追求极致效率的情况，极坐标系还是首选。



(a) 静态场景 (b) 动态场景

图 8 Gazebo:2个场景的运动轨迹

Fig.8 Gazebo: motion trajectory of two scenes

从图7可知，没有膨胀层时，小车会优先考虑时间效率，导致出现沿着障碍物运动的危险路径；而膨胀层可以让小车在导航中更加注意与障碍物的距离，这提高了实际应用中的安全性。

从图8可知，将在Gym中得到的实验结果迁移到Gazebo中是完全可行的，而且适用于静态和动态环境。这主要得益于环境基本特征一致，并把三维雷达信息直接转换为Gym中确定的障碍物坐标，对观测信息既做了一定抽象也保留了基本的环境信息。

最后，实验表明在Gazebo中训练5×10^5步需要的时间约为25 h，而在Gym中则只需约5 h。假设把文中提到的实验方案放在Gazebo中验证，会增加大量时间成本，而且由于ROS的不稳定性，多次长时间的训练可能会遇到更多的问题。把实验猜想、验证、调参等工作都部署到轻量级的2D环境中，3D环境只作为最后的算法结果测试，那么基于个人电脑的机器人强化学习的开发效率能提高5倍左右。

4 结语

机器人强化学习在3D环境中训练，对计算量和训练时间都有较高要求。因此本文提出了一个从2D到3D的训练模式来加速开发过程，并通过实验成功实现静态和动态场景的导航功能。采用本文的训练模式做机器人强化学习的应用开发，比直接在3D环境中操作所需时间成本降低到原来的20%，使个人电脑也能胜任相关的研究工作。这证明在机器人强化学习的应用中，从2D到3D的训练模式具有较高的应用价值。

本文的导航场景只有单智能体，后续计划在本文训练模式的基础上，以多智能体协作为目标[15]，实现更复杂场景的导航功能。

参考文献：

[1] SILVER D, HUANG A, MADDISON C J, *et al*. Mastering the game of go with deep neural networks and tree search [J]. Nature, 2016, 529(7587): 484.

[2] SUTTON R S, BARTO A G. Introduction to reinforcement learning[M]. Cambridge: MIT press, 1998: 4-6.

[3] IRPAN A. Deep reinforcement learning doesn't work yet[EB/OL]. (2018-02-14)[2018-02-14]. https://www.alexirpan.com/2018/02/14/rl-hard.html.

[4] LEVINE S, PASTOR P, KRIZHEVSKY A, *et al*. Learning hand-eye coordination for robotic grasping with deep learning and large-scale data collection [J]. The International Journal of Robotics Research, 2018, 37(4-5): 421-436.

[5] HESSEL M, MODAYIL J, VAN HASSELT H, *et al*. Rainbow: Combining improvements in deep reinforcement learning[C]//Thirty-Second AAAI Conference on Artificial Intelligence. Orleans, Louisiana: AAAI, 2018.

[6] SCHULMAN J, WOLSKI F, DHARIWAL P, *et al*. Proximal policy optimization algorithms[J]. arXiv preprint arXiv, 2017: 1707.06347.

[7] WANG J, HU J, MIN G, *et al*. Computation offloading in multi-access edge computing using a deep sequential model based on reinforcement learning [J]. IEEE Communications Magazine, 2019, 57(5): 64-69.

[8] 吴运雄, 曾碧. 基于深度强化学习的移动机器人轨迹跟踪和动态避障[J]. 广东工业大学学报, 2018, 36(1): 42-50.
WU Y X, ZENG B. Trajectory tracking and dynamic obstacle avoidance of mobile robot based on deep reinforcement learning [J]. Journal of Guangdong University of Technology, 2018, 36(1): 42-50.

[9] STOOKE A, ABBEEL P. Accelerated methods for deep reinforcement learning[J]. arXiv preprint arXiv, 2018: 1803.02811.

[10] HENDERSON P, CHANG W D, SHKURTI F, *et al*. Benchmark environments for multitask learning in continuous domains[J]. arXiv preprint arXiv, 2017: 1708.04352.

[11] TAI L, PAOLO G, LIU M. Virtual-to-real deep reinforcement learning: Continuous control of mobile robots for mapless navigation[C]//2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS). [S.l.]:IEEE, 2017: 31-36.

[12] FUJIMOTO S, VAN HOOF H, MEGER D. Addressing function approximation error in actor-critic methods[J]. arXiv preprint arXiv, 2018: 1802.09477.

[13] BROCKMAN G, CHEUNG V, PETTERSSON L, *et al*. Openai gym[J]. arXiv preprint arXiv, 2016: 1606.01540.

[14] TAKAYA K, ASAI T, KROUMOV V, *et al*. Simulation environment for mobile robots testing using ROS and Gazebo[C]//2016 20th International Conference on System Theory, Control and Computing (ICSTCC). Sinaia: IEEE, 2016: 96-101.

[15] BUŞONIU L, BABUŠKA R, DE SCHUTTER B. Multi-agent reinforcement learning: An overview[M]. Berlin Heidelberg: Springer, 2010: 183-221.

（责任编辑：杨耀辉）