DOI：10.13973/j.cnki.robot.210265

一种基于功用性图的目标推抓技能自监督学习方法

吴培良[1,2]，刘瑞军[1]，毛秉毅[1,2]，史浩洋[1]，陈雯柏[3]，高国伟[3]

（1. 燕山大学信息科学与工程学院，河北 秦皇岛 066004；
2. 河北省计算机虚拟技术与系统集成重点实验室，河北 秦皇岛 066004；
3. 北京信息科技大学自动化学院，北京 100192）

摘 要：提出了一种基于功用性图的目标推抓技能自监督学习方法。首先，给出了杂乱环境下面向目标推抓任务的机器人技能自监督学习问题描述，将工作空间中机器人推抓操作的决策过程定义为一个全新的马尔可夫决策过程（MDP），分别训练视觉机制模块与动作机制模块。其次，在视觉机制模块中融合自适应参数与分组拆分注意力模块设计了特征提取网络 RGSA-Net，可由输入网络的原始状态图像生成功用性图，为目标推抓操作提供良好的前提。然后，在动作机制模块中搭建了基于演员－评论家（actor-critic）框架的深度强化学习自监督训练框架 DQAC，机器人根据功用性图执行动作后利用该框架进行动作评判，更好地实现了推、抓之间的协同。最后，进行了实验对比与分析，验证了本文方法的有效性。

关键词：推抓技能学习；功用性图；自监督学习；自适应参数；拆分注意力机制

中图分类号：TP242 **文献标识码**：A **文章编号**：1002-0446(2022)-04-0385-14

A Self-supervised Learning Method of Target Pushing-Grasping Skills Based on Affordance Map

WU Peiliang[1,2], LIU Ruijun[1], MAO Bingyi[1,2], SHI Haoyang[1], CHEN Wenbai[3], GAO Guowei[3]

(1. *School of Information Science and Engineering*, *Yanshan University*, *Qinhuangdao* 066004, *China*;
2. *The Key Laboratory for Computer Virtual Technology and System Integration of Hebei Province*, *Qinhuangdao* 066004, *China*;
3. *School of Automation*, *Beijing Information Science & Technology University*, *Beijing* 100192, *China*)

Abstract: A self-supervised learning method of target pushing-grasping skills based on affordance map is presented. Firstly, the self-supervised learning problem is described for robot to learn target pushing-grasping skills in cluttered environment. The decision process of robot pushing and grasping operation in workspace is defined as a new Markov decision process (MDP), in which the vision mechanism module and action mechanism module are trained separately. Secondly, the adaptive parameters and group split attention module are fused in the vision mechanism module to design the feature extraction network RGSA-Net, which can generate the affordance map from the original state image of the input network, and provide a good premise for the target pushing-grasping operation. Then, a deep reinforcement learning based self-supervised training framework DQAC based on actor-critic framework is built in the action mechanism module. After the robot performs the action according to the affordance map, the DQAC framework is used to evaluate the action, and thus better cooperation between pushing and grasping is realized. Finally, experimental comparison and analysis are carried out to verify the effectiveness of the proposed method.

Keywords: pushing-grasping skill learning; affordance map; self-supervised learning; adaptive parameter; split attention mechanism

1 引言（Introduction）

近年来，机器学习方法快速发展，使得机器人可以实现对未知物体的自主抓取操作，但机器人抓取杂乱场景下的目标物体仍面临不小的挑战。一些研究在 Conell Grasping、Jacquard 等抓取数据集上通过深度学习方法进行有监督的训练以获取最优抓取框[1-4]、判定物体的抓取点或抓取姿势，进而完成机器人对物体的抓取任务，但应用大量数据集导致训练时间过长。强化学习的兴起，使得机器人可以自监督地与环境进行信息交互，进而完成所设定

基金项目：国家重点研发计划（2018YFB1308300）；国家自然科学基金区域联合基金（U20A20167）；北京市自然科学基金（4202026）；河北省自然科学基金（F202103079）.

通信作者：毛秉毅，ysdxmby@163.com 收稿/录用/修回：2021-06-18/2021-08-20/2021-08-24

的任务。传统强化学习方法为机器臂抓取、机器人堆叠和整理、四足机器人行走以及机器人导航等高维度的控制问题提供了较好的解决方案。

数据驱动的方法在许多方面都取得了成功，受此鼓舞，深度神经网络也被用在杂乱场景下的物体姿态估计。机器人通过对抓取数据集训练实现了对目标物体的抓取[1-4]。Morrison 等[1] 提出生成式抓取卷积神经网络（GG-CNN），对图像中每个像素的抓取质量和姿势给出得分。Redmon 等[2] 使用单阶段回归法找到可抓取的矩形检测框。Levine 等[3] 拟合了末端夹具的运动轨迹进而预测抓取成功的概率，并探索了多个机器人并行化训练的方法。Schmidt 等[4] 提出了一种数据驱动、自下而上的学习方法，使用深度卷积神经网络（DCNN）来实现机器人对新颖对象的抓取。Mahler 等[5] 使用 Dex-Net 2.0 数据集训练抓取质量卷积神经网络（GQ-CNN）模型，根据深度相机给出的 3D 点云信息，找到鲁棒性最高的吸附抓取点。Kumra 等[6] 提出了一种新颖的生成式残差网络（GR-ConvNet）模型，用于反足机器人从场景的 n 通道图像中生成最佳抓取点，成功完成对未知目标的抓取。Lou 等[7] 提出了一种基于 3 维像素的深度卷积神经网络（3D-CNN），可在能实现可达性感知的无限制工作空间中生成可行的 6 自由度机器人抓取姿态。Shao[8] 等基于 ResNet[9] 和 U-Net 网络训练了一种端到端的神经网络，在不需要进行识别和姿态估计的非结构化环境中预测抓取区域。深度学习中使用的数据集大多由人工标记抓取点构成，由于存在多种抓握方式，因此标记并非易事；而且，受语义偏见的影响，实验中使用的数据量较低，这也导致机器人倾向于过拟合[10]。

基于深度强化学习算法的自监督抓取策略为实现自动化抓取提供了可能，该类方法在抓取对象形状、位姿未知的情况下，通过策略迭代与奖励值反馈进行自监督学习。Shukla 等[11] 将形状规则的刚性目标抓取的问题分解为位置和方向学习，提出了一种 GDQN（抓取深度 Q 网络）深度强化学习算法，用于姿态估计，并验证了该网络的有效性，但其对于目标物体要求较高且需要部分人工标注。Kalashnikov 等[12] 提出了一种大规模分布式优化和深度 Q 学习拟合的离线策略训练方法，实现了机器人的动态操作，但其硬件要求过高导致复现较难。Sarantopoulos 等[13] 提出了一种模块化的强化学习方法，使用连续动作将目标对象从周围的杂波中完全分离出来。通过动作原语和特征选择，将先验知识有效地融入到学习中，提高了样本效率，一定程度上解决了训练效率低的问题。Quillen 等[14] 探索了一种基于视觉的机器人抓取深度强化学习算法，根据异策略（off-policy）强化学习算法评估无模型的机器人抓取任务。Hou 等[15] 在机器人抓取之前，通过监督学习对 Q 网络进行预训练，验证了该方法能够在早期阶段显著加速自监督学习，并且与工作空间中对象的稀疏性几乎无关。Deng 等[16] 基于 DQN（深度 Q 网络）设计了一种结合吸盘与夹持器的新型机械手的主动探索算法，在杂波环境中完成了抓取任务。Xie 等[17] 提出带失败概率的贝叶斯逆强化学习（BIRLF）算法，该算法从策略最优性条件导出半空间，在贝叶斯逆强化学习（BIRL）框架下合并失败的经验并回传给机器人，适用于工作空间更复杂的环境。但其需要对规则物体进行前期人工标注，工作量较大。Johannink 等[18] 直接在现实世界中训练智能体，将机械臂控制难题分解为常规反馈控制方法解决的部分和用 RL（强化学习）法解决的残差部分，有效地解决实际控制问题。Mohammadi 等[19] 在改进的 DDPG（深度确定性策略梯度）基础上提出了一种在线连续深度强化学习的方法，用于完成混乱环境中的抓取任务。Ni 等[20] 针对边缘物体抓取，将抓取质量函数的变化与遗忘机制相结合来训练推抓动作，此外，还设置了双重体验重播，以增加边界上的搜索。文 [21-22] 都在 DQN 算法的基础上提出了新的改进，Gui 等[21] 提出了一种知识诱导的深度 Q 学习模型（KI-DQN），在采取抓取动作之前先将目标物体推向墙壁，将该问题看作马尔可夫决策过程，主动利用环境优势来抓取物体。Joshi 等[22] 构建双重 DQN 框架抓取网络来输出抓取概率，提高了抓取成功率，但其需要收集海量训练数据，还要在多个物理机器人上进行训练，大大增加了训练成本。Zhang 等[23] 基于 DQN 提出了一种堆叠抓取网络（GSN），在研究目标物体的抓取问题时进一步讨论了物体放置问题，但其仅考虑了单个物体的场景，具有一定的局限性。Zeng 等[24] 通过卷积神经网络预先计算每个动作的置信度 Q 值，然后通过环境奖励反馈进行梯度回传，通过推、抓之间的协同完成抓取任务，但其使用传统 DQN 算法训练动作，使得推、抓之间的协同性较差。Yang 等[25] 在文 [24] 的基础上通过语义分割得到目标掩码，并使用二分类网络训练推、抓之间的协同，重新排列目标周围的干扰物体，实现了对目标物体的抓取，但其仍然存在抓取点定位不准确、推抓协同效率低等问题。

针对杂乱环境下面向目标物的抓取任务，本文提出一种基于功用性图检测的目标抓取技能自监督学习方法，主要贡献包括：

(1) 给出了杂乱环境下面向目标抓取的问题描述，并将其表示为一个全新的马尔可夫决策过程（MDP）。其中状态信息包括 t 时刻的图像状态与特征状态，对于 2 种状态分别给出奖励函数集合。

(2) 基于自适应参数调节与分组拆分注意力机制，提出一种新型特征提取网络 RGSA-Net，它可根据 RGB-D 摄像机采集到的状态得到精准推抓的功用性图（affordance map）。同时，通过自适应参数调节防止梯度弥散。

(3) 基于演员－评论家框架的思想，在动作机制模块中设计 DQAC 算法。该算法将基于值函数与策略梯度的算法的优势结合起来，用于训练机器人推、抓之间的协同。

2　问题描述与求解框架（Problem description and solution framework）

2.1　推动和抓取任务描述及抽象化

杂乱环境的特点是在一个场景下一些形状未知的物体呈无规律堆叠，而且目标物体被严重遮挡。因此，该环境中面向目标的抓取尤为困难。功用性图定义了某一像素的推抓置信度，颜色越深表示采取动作的置信度越高，反之置信度越低。本文研究机器人在图 1 所示的杂乱环境下面向目标的推抓问题，采用 RGB-D 相机作为机器人的外部传感器，利用功用性图判断该场景中是否存在适于抓取的位置和角度。若存在则直接抓取；若不存在，则采取若干次推动动作，增大目标物体周围的空间供机器人抓取。

本文将机器人面向目标的推抓技能学习问题定义为一个马尔可夫决策过程（MDP），该过程由七元组 $(S, S^{\mathrm{c}}, A, R^{\mathrm{q}}, R^{\mathrm{c}}, \boldsymbol{P}, \gamma)$ 组成。参数 S 表示图像状态集，集合中以图像的形式存储每一时刻相机所采集到的环境状态；参数 S^{c} 表示特征状态集，集合中以功用性图的形式存储每一时刻状态 S 经特征提取网络处理后生成的特征状态；参数 A 为有限动作集合，包括推动与抓取动作；参数 R^{q} 为针对特征设计的奖励函数，集合中存储每一时刻的 $\boldsymbol{Q}$ 值以及奖励值；参数 R^{c} 为根据动作以及环境反馈给出的奖励函数集合；参数 $\boldsymbol{P}(\boldsymbol{s}_{t+1}^{\mathrm{c}}|\boldsymbol{s}_t^{\mathrm{c}}, a_t)$ 为根据当前时刻状态与动作得到的下一时刻状态的转移概率矩阵；参数 $\gamma \in (0, 1]$ 为折扣因子，用于平衡即时奖励以提高奖励期望。机器人在任意时刻 t 接收到图像状态 $\boldsymbol{s}_t$ 与特征状态 $\boldsymbol{s}_t^{\mathrm{c}}$，并基于深度强化学习策略以概率 p 选择行为 a_t，进入新的状态 $\boldsymbol{s}_{t+1}$ 和 $\boldsymbol{s}_{t+1}^{\mathrm{c}}$ 并得到相应的奖励 $r^{\mathrm{q}}(\boldsymbol{s}_t, a_t)$ 和 $r^{\mathrm{c}}(\boldsymbol{s}_t^{\mathrm{c}}, a_t)$。

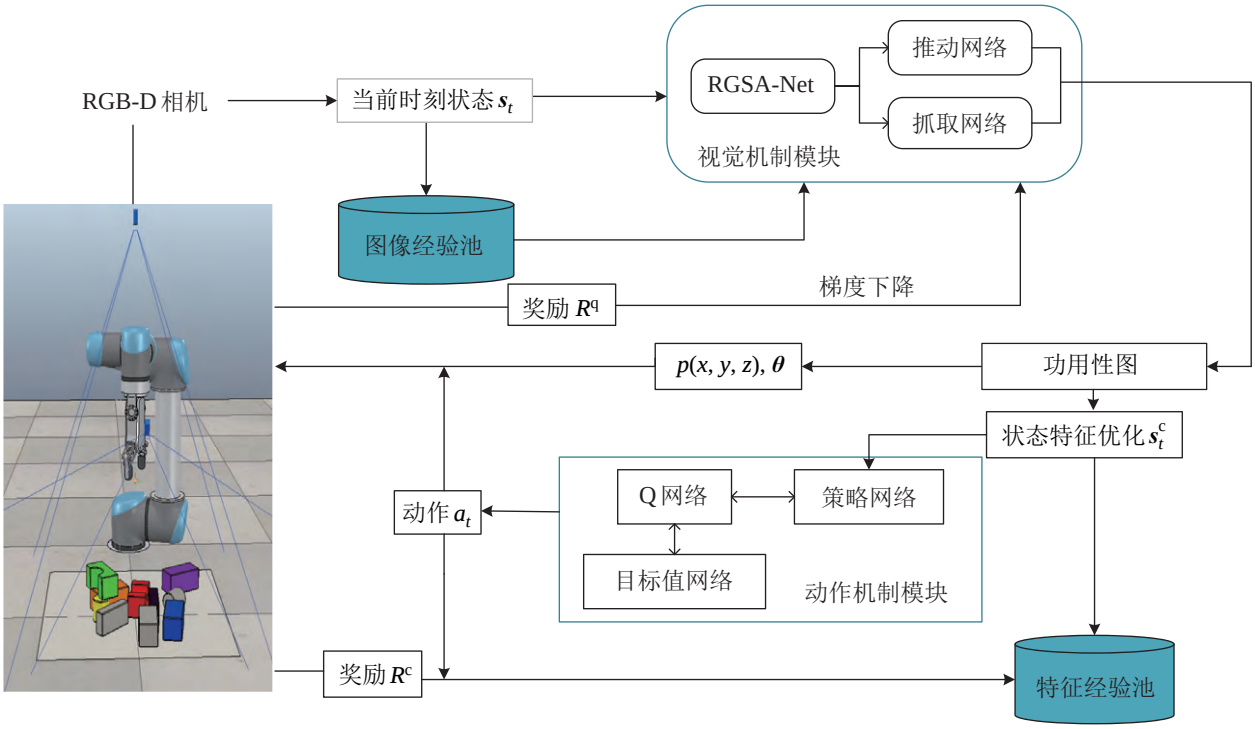

图 1　机器人目标推抓技能自监督学习系统

Fig.1　Robot self-supervised learning system for target pushing-grasping skills

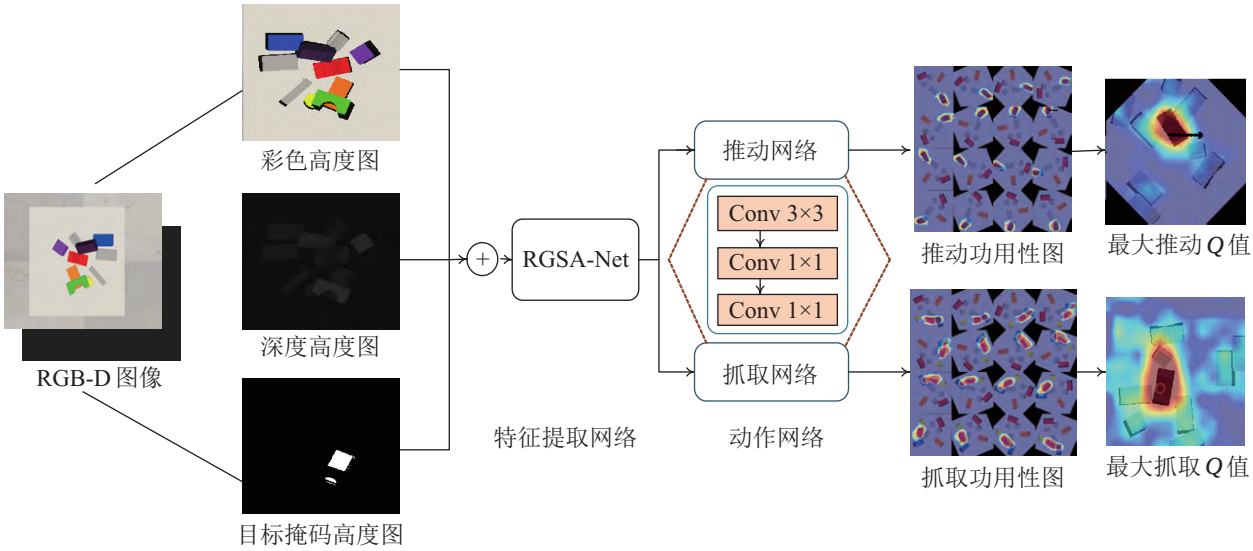

图 2 视觉机制模块执行流程

Fig.2 Implementation process of the vision mechanism module

2.2 问题求解的总体框架

基于深度强化学习设计了一种目标推抓技能自监督学习方法。如图 1 所示，整个框架分为视觉机制模块和动作机制模块。首先将机器人上方的 RGB-D 相机获取的工作台图像信息作为当前时刻的状态 $\boldsymbol{s}_t$ 放到图像经验池（SRB）和视觉机制模块中。图像经验池用于打破相邻时间内图像的相关性。视觉机制模块由两部分构成，一部分是 RGSA-Net 特征提取网络，另一部分是由推抓网络构成的动作网络。该模块用于生成推抓功用性图，整合推抓操作的位置 $p(x,y,z)$ 和角度信息 $\boldsymbol{\theta}$。利用环境给出的功用性图奖励 R^{c} 进行随机梯度下降来训练网络参数。将功用性图作为当前时刻的特征状态 $\boldsymbol{s}_t^{\mathrm{c}}$ 送入特征经验池（FRB）与动作机制模块中。特征经验池用于打破相邻时间内特征的相关性。动作机制模块将传统的演员－评论家框架进行拆分，将评价网络进一步分解成值网络（Q 网络）与目标值网络（Target Q 网络），用于训练机器人推、抓动作之间的协同，利用环境给出的动作奖励 R^{c} 更新参数。

视觉机制模块的执行流程如图 2 所示。首先在相机坐标系中将深度信息投影到 3D 点云中，创建 RGB-D 高度图，经预处理后得到状态高度图（heightmap），将其旋转 16 次后输入到特征提取网络 RGSA-Net 进行特征提取。然后将提取到的特征作为推抓动作网络的输入，分别得到推动与抓取动作的功用性图，进而求得在执行动作时所需的置信度最大的像素点与角度。最后根据手眼标定结果将相机坐标系信息转化为机器人坐标系信息，在工作空间中得到接触点位置与角度来执行动作。

动作机制模块基于演员－评论家框架的思想，其策略网络输入为当前状态值，输出为决策动作，目标是训练最优的策略 $\boldsymbol{\pi}_{\boldsymbol{\theta}}(\boldsymbol{s}^{\mathrm{c}},a)$ 以达到最高奖励，即在状态 $\boldsymbol{s}_t^{\mathrm{c}}$ 下执行动作 a_t 可得到最优的参数 $\boldsymbol{J}(\boldsymbol{\theta})$：

$$\boldsymbol{J}(\boldsymbol{\theta})=\sum_{a\in A}\boldsymbol{\pi}_{\boldsymbol{\theta}}(\boldsymbol{s}^{\mathrm{c}},a)r^{\mathrm{c}}(\boldsymbol{s}_t^{\mathrm{c}},a_t) \tag{1}$$

其评价网络基于值函数，负责评估动作网络的表现，并指导该网络下一阶段的输出动作，即评判动作执行的好坏，其目标函数梯度更新计算公式如下：

$$\nabla\boldsymbol{J}(\boldsymbol{\theta})=\sum_{a\in A}\boldsymbol{\pi}_{\boldsymbol{\theta}}(\boldsymbol{s}^{\mathrm{c}},a)\nabla_{\boldsymbol{\theta}}\,\mathrm{lb}\,\boldsymbol{\pi}_{\boldsymbol{\theta}}(\boldsymbol{s}_t^{\mathrm{c}},a)r^{\mathrm{c}}(\boldsymbol{s}_t^{\mathrm{c}},a_t) \tag{2}$$

3 方法（Method）

3.1 推抓功用性图特征提取网络 RGSA-Net

为了生成准确的推抓功用性图，本文在视觉机制模块中以深度残差网络（ResNet18[9]）作为主骨干，设计了一种基于分组卷积和自适应拆分注意力机制的特征提取网络 RGSA-Net。该网络将预处理后得到的 3 种高度图（彩色高度图、深度高度图、目标掩码高度图）旋转 16 次后作为输入，得到所有像素点期望 Q 值并生成推抓功用性图，采取贪婪策略选取最大的 Q 值像素点转化为动作执行位置点 $p(x,y,z)$，同时得到动作执行的角度 $\boldsymbol{\theta}$。

3.1.1 自适应参数

深度神经网络虽然在各个领域取得了很大进展，但是经常面临梯度消失或梯度弥散等问题，研究者通常使用 3 种方法解决上述问题[26]：1) 设置

初始化；2) 使用 BatchNorm 或 LayerNorm 等算法实现正则化；3) 残差连接方法。而这些方法同样存在着设计痕迹过重、计算开销大等问题。对于一个深度为 L 的模型，传统残差网络第 l 层输出为

$$\boldsymbol{z}^{(l)}=\mathcal{H}\left(\boldsymbol{x}^{(l-1)}\right)=\boldsymbol{x}^{(l-1)}+\mathcal{F}\left(\boldsymbol{x}^{(l-1)}\right) \tag{3}$$

本文在残差连接前增加一个初始值为 0 的自适应参数，该参数可以使得模型更好地接收梯度信号，加快网络的收敛速度，其基本形式如下：

$$z^{(l)}=\mathcal{H}\left(\boldsymbol{x}^{(l-1)}\right)=\boldsymbol{x}^{(l-1)}+\alpha_i\times\mathcal{F}\left(\boldsymbol{x}^{(l-1)}\right) \tag{4}$$

残差网络可以解决传统神经网络中梯度弥散的问题，但用其拟合恒等变换并不容易。恒等变换的主要目标是构造一种天然的恒等映射，即：

$$\boldsymbol{x}^{(l)}=\boldsymbol{z}^{(l)} \tag{5}$$

等价于令残差部分 $\mathcal{F}(\boldsymbol{x}^{(l-1)})\to 0$。

对于任意 2 个层数 $l_2>l_1$，可将式 (3) 递归展开：

$$\boldsymbol{x}^{(l_2)}=\boldsymbol{x}^{(l_1)}+\sum_{i=l_1}^{l_2-1}\alpha_i\times\mathcal{F}(\boldsymbol{x}^{(i)}) \tag{6}$$

最终损失 $\boldsymbol{\varepsilon}$ 对于低层输出的梯度可展开为

$$\frac{\partial\boldsymbol{\varepsilon}}{\partial\boldsymbol{x}^{(l_1)}}=\frac{\partial\boldsymbol{\varepsilon}}{\partial\boldsymbol{x}^{(l_2)}}+\frac{\partial\boldsymbol{\varepsilon}}{\partial\boldsymbol{x}^{(l_2)}}\frac{\partial}{\partial\boldsymbol{x}^{(l_1)}}\sum_{i=l_1}^{l_2-1}\alpha_i\times\mathcal{F}(\boldsymbol{x}^{(i)}) \tag{7}$$

式中前一项表示反向传播时错误信号不经过任何中间权重矩阵变换直接传播到低层，可以很好地缓解梯度弥散问题。

自适应参数 α_i 的训练如下所示。假设深度神经网络模型输出由式 (8) 给出：

$$\boldsymbol{x}_L=(1+\boldsymbol{\alpha}\boldsymbol{\omega})^L\boldsymbol{x}_0 \tag{8}$$

其中权重参数 $\boldsymbol{\omega}$ 的训练迭代过程为

$$\boldsymbol{\omega}\leftarrow\boldsymbol{\omega}-\lambda L\boldsymbol{x}_0(1+\boldsymbol{\alpha}\boldsymbol{\omega})^{L-1}\partial C \tag{9}$$

式中 C 为损失函数，λ 为学习率。由式 (9) 可以看出：当 $\boldsymbol{\alpha}=\boldsymbol{0}$ 时，第 1 轮更新没有更新参数 $\boldsymbol{\omega}$。参数 $\boldsymbol{\alpha}$ 的训练过程如下：

$$\boldsymbol{\alpha}\leftarrow\boldsymbol{\alpha}-\lambda L\boldsymbol{x}_0\boldsymbol{\omega}\partial C=-\lambda L\boldsymbol{x}_0\boldsymbol{\omega}\partial C,\quad \boldsymbol{\alpha}=\boldsymbol{0} \tag{10}$$

从第 2 轮开始，权重参数 $\boldsymbol{\omega}$ 的更新方式为

$$\lambda L\boldsymbol{x}_0(1+\boldsymbol{\alpha}\boldsymbol{\omega})^{L-1}\partial C=\lambda L\boldsymbol{x}_0(1-\lambda L\boldsymbol{x}_0\boldsymbol{\omega}^2\boldsymbol{\alpha}C)^{L-1}\partial C \tag{11}$$

由式 (11) 可以看出，如果损失函数 C 是合理的，当前的更新不会导致很强的梯度振荡（$\lambda L\boldsymbol{x}_0\times\boldsymbol{\omega}^2\boldsymbol{\alpha}C\approx 0$），保证了深度神经网络模型更新时网络的稳定性。自适应参数 α_i 在 RGSA-Net 网络中的更新过程如图 3 所示。

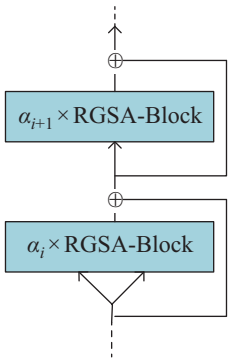

图 3　自适应参数更新过程

Fig.3　The update process of adaptive parameter

3.1.2　基于分组卷积的 RGSA-Block

分组卷积已经被证实可以在效能不变的情况下大大减少参数量[27]。若初始输入深度神经网络的特征向量维度为 $C_1\times H\times W$，则将该特征在通道维度上分成 g 个基数组 $\{\boldsymbol{G}_1,\boldsymbol{G}_2,\cdots,\boldsymbol{G}_g\}$，每组特征维度为 $(C_1/g)\times H\times W$。若普通卷积对应的卷积核大小为 $K\times K\times C_1$，则分组卷积核的维度为 $K\times K\times C_1/g$。将 g 组特征在通道维度拼接后仍可得到维度为 $C_2\times H\times W$ 的输出特征。此时深度神经网络的参数量可减少为标准卷积操作的 $1/g$，且经过不同路径卷积得到的特征图之间的耦合性较低，利用所关注的不同特征可以得到互为补充的特征图。

将 RGB-D 相机采集到的图片数据经过一种语义分割的深度全卷积神经网络结构 SegNet[28] 作预处理后，得到了彩色高度图（RGB-height）、深度高度图（depth-height）、目标掩码高度图（mask-height）。将完成特征提取后的 3 种数据在通道维度融合，得到维度为 $1\times 5\times 640\times 640$ 的特征。深度强化学习框架每次训练输入网络的样本数量为 1 会导致特征归一化失败，因此本文在通道维度增加相同维度的数据并进行特征拼接。最终 RGSA-Net 网络的输入特征维度为 $2\times 5\times 640\times 640$，输出特征维度为 $1\times 1024\times 20\times 20$。

如图 4 所示，本文的 RGSA-Block 基本块先将输入特征分为 2 个候选集（$K=2$），每一集合又分成 2 组（$G=2$）进行训练，每一组特征在经过 2 个不同卷积块后馈入到拆分注意力模块进行特征融合。卷积块由卷积层、归一化层、ReLU（线性整流）激活函数依次构成，区别在于卷积核的大小分

别为 3×3 和 5×5。2 个候选集的输出特征经过拼接操作后由 1×1 的卷积块进行维度调整以适应网络输入，乘上自适应参数 α_i 后与原始特征向量相加，作为最终的输出特征。

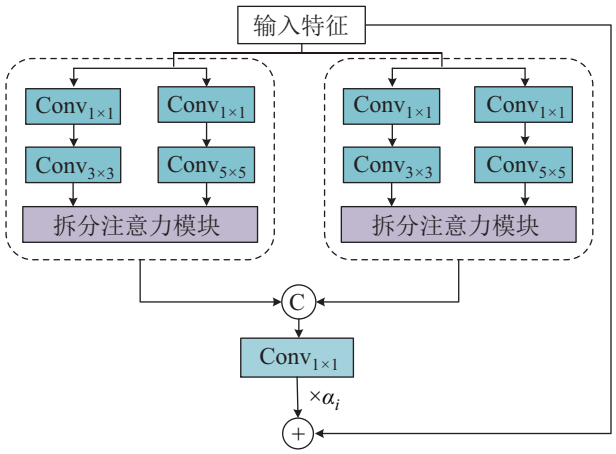

图 4 RGSA-Block 网络结构

Fig.4 The network structure of RGSA-Block

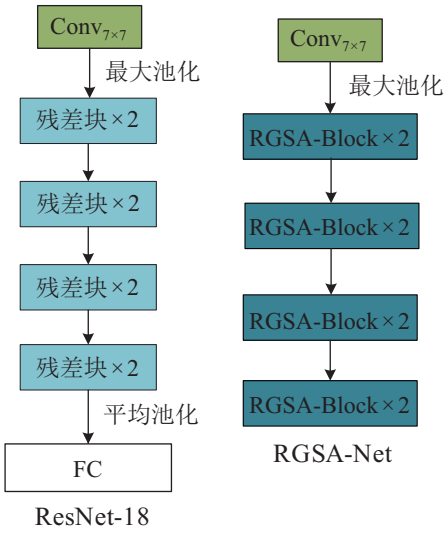

图 5 RGSA-Net 网络结构

Fig.5 The network structure of RGSA-Net

如图 5 所示，将 RGSA-Block 基本块嵌入到残差网络 ResNet-18 后得到了 RGSA-Net。

3.1.3 基于拆分注意力机制的 Split-Attention 模块

在计算能力有限的情况下，机器人需要更加注意目标物体周围的情况[29]。使用具有注意力机制的深度神经网络，可以使机器人更快地定位目标物体，提高对目标物体的注意力程度，从而探索更多的环境状态以获得目标信息。如图 6 所示，本文采用基于通道的拆分注意力机制[29]，对于输入 RGSA-Net 网络的彩色高度图、深度高度图、目标掩码高度图 3 种状态高度图在通道方面给予不同的权重，以建模 5 个通道的重要程度。

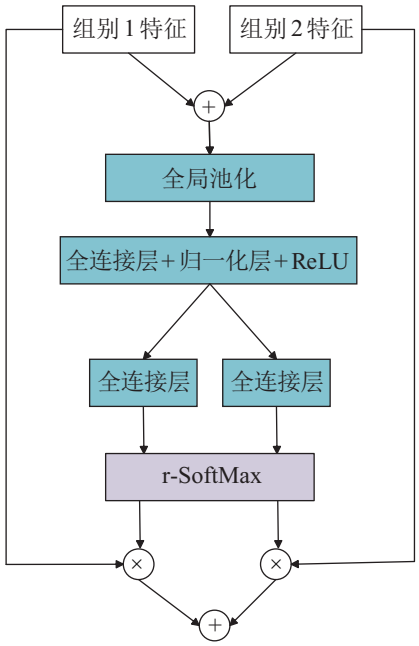

图 6 拆分注意力网络结构

Fig.6 The structure of split-attention network

图 6 中，设每个输入特征的维度为 $W\times H\times C$，其中 $C=\dfrac{C_1}{K}$，特征融合计算公式为

$$\hat{\boldsymbol{T}}_k=\sum_{j=r(k-1)+1}^{rk}\boldsymbol{T}_j \tag{12}$$

其中 $r=2$ 为基数，将融合后的特征进行全局平均池化，计算公式如下：

$$\boldsymbol{s}_k^{\mathrm{c}}=\frac{1}{H\times W}\sum_{i=1}^{H}\sum_{j=1}^{W}\hat{\boldsymbol{T}}_k^{\mathrm{c}}(i,j) \tag{13}$$

至此，将得到的单图平均后的中间特征经过卷积块处理后进行通道拆分，拆分成通道数为 3、1、1 的 3 种特征后，再分别经过全连接层和 r-SoftMax 层进行特征提取，r-SoftMax 层参数计算如下：

$$a_i^k(d)=\begin{cases}\dfrac{\exp(F_i^{\mathrm{c}}(\boldsymbol{s}_k))}{\sum_{j=0}^{r}\exp(F_i^{\mathrm{c}}(\boldsymbol{s}_k))}, & r>1\\[2ex] \dfrac{1}{1+\exp(-F_i^{\mathrm{c}}(\boldsymbol{s}_k))}, & r=1\end{cases} \tag{14}$$

将经过 r-SoftMax 层处理的 3 种特征在通道维度进行特征融合，得到通道数为 5 的输出特征，并将其输入到下一个网络模块中。

3.2 面向机器人推抓技能学习的 DQAC 算法

训练面向目标物体的推、抓动作之间的协同是机器人抓取成功的关键。在保证抓取成功率的同时尽可能地减少机器人运动次数，就需要建立起良好的评估机制，充分训练推、抓之间的协同策略。

3.2.1　DQAC 算法思想

本文的动作机制模块是基于演员－评论家框架[30]设计的一种自监督学习算法 DQAC，该算法由策略网络和值网络构成。策略网络建立模型 $\boldsymbol{\pi}(\boldsymbol{s},a;\boldsymbol{\theta})$ 将动作选择策略参数化，根据状态的分布概率选择离散动作。值网络评判当前动作以迭代更新策略参数值，使得策略模型的累积回报不断增加，从而得到最优的协同动作策略。

3.2.2　DQAC 算法设计

熵是信息论中的重要概念，用于表示信息的不确定程度，熵值越大，则信息的不确定程度越大。

$$\boldsymbol{H}_{\mathrm{e}}(\boldsymbol{X})=-\sum_{1}^{n}\boldsymbol{p}(x_i)\,\mathrm{lb}\,\boldsymbol{p}(x_i) \tag{15}$$

DQAC 算法是针对离散动作空间异策略改进的强化学习算法，使用信息熵最大化随机策略来提高机器人对环境的探索能力，从而实现目标策略的优化。

$$\boldsymbol{\pi}^{*}=\arg\max_{\boldsymbol{\pi}}E_{(\boldsymbol{s},a)\sim\rho_{\boldsymbol{\pi}}}\left(\sum_{t}\boldsymbol{r}(\boldsymbol{s}_t^{\mathrm{c}},a_t)+\boldsymbol{\beta}\boldsymbol{H}_{\mathrm{e}}(\boldsymbol{\pi}(\cdot|\boldsymbol{s}_t^{\mathrm{c}}))\right) \tag{16}$$

式中，目标函数值 $\boldsymbol{\pi}^{*}$ 为最优策略，$\boldsymbol{H}_{\mathrm{e}}(\boldsymbol{\pi}(\boldsymbol{s}_t^{\mathrm{c}}))$ 表示在状态为 $\boldsymbol{s}_t^{\mathrm{c}}$ 时采取不同策略得到的不同熵值。熵值越大表示机器人对环境的探索策略越随机，同时采用软更新方式进行策略迭代。

DQAC 算法的环境奖励值由功用性奖励 r_{g} 和动作奖励 $r_{\mathrm{g}}^{\mathrm{c}}$ 两部分构成。将推动动作奖励定义为：当推动动作导致环境结构发生变化的像素值超过阈值时 $r_{\mathrm{p}}=0.5$，$r_{\mathrm{p}}^{\mathrm{c}}=0.5$；未超过阈值时 $r_{\mathrm{p}}=0$，$r_{\mathrm{p}}^{\mathrm{c}}=-1$。本文将阈值设置为 30 个像素。将抓取奖励定义为：抓取目标成功时，$r_{\mathrm{g}}=1$，$r_{\mathrm{g}}^{\mathrm{c}}=1$；抓取目标失败时，$r_{\mathrm{g}}=0$，$r_{\mathrm{g}}^{\mathrm{c}}=-1$。

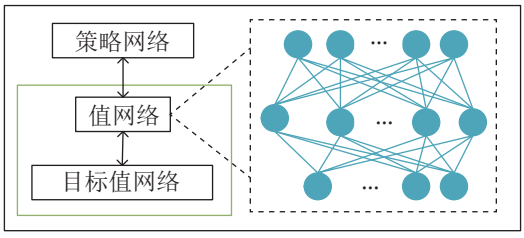

图 7　动作机制模块网络结构

Fig.7　The structure of the action mechanism module network

如图 7 所示，动作机制模块由策略网络（actor）、值网络（critic）和目标值网络（target critic）组成。每个网络由结构相同的多层感知机（MLP）网络构成。该网络由输入、输出及中间层构成，相邻 2 层之间的所有神经元均互相连接，而同层神经元之间无连接。

动作机制模块的作用是根据 RGSA-Net 网络提取的特征作出决策。具体的流程为：将从环境中提取的特征状态送入到策略网络中，中间经过不同神经元的前向传播直到输出层神经元。使用 SoftMax 函数计算出不同动作的概率值，使用非线性激活函数 ReLU 防止梯度消失。值网络（Q 网络）由 2 个结构相同的 MLP 网络构成，用于拟合计算某一时刻的机器人特征状态与动作决策；目标值网络（Target Q 网络）以固定周期复制 Q 网络的参数，采用软更新策略进行更新，用于预测环境中下一时刻的状态以及动作 Q 值。

考虑机器人在杂乱环境中采取推动和抓取 2 种离散动作，DQAC 算法根据特征状态 $\boldsymbol{s}_t^{\mathrm{c}}$ 对策略 $\boldsymbol{\pi}$ 进行拟合以减小目标估计方差。由贝尔曼方程得出计算离散动作对应值函数公式：

$$\boldsymbol{V}_{\boldsymbol{\pi}}(\boldsymbol{s})=\boldsymbol{\pi}(\boldsymbol{s}_t^{\mathrm{c}})\left(\boldsymbol{Q}(\boldsymbol{s}_t^{\mathrm{c}},a_t)-\boldsymbol{\beta}\,\mathrm{lb}\,\boldsymbol{\pi}(\boldsymbol{s}_t^{\mathrm{c}})\right) \tag{17}$$

式中，$\boldsymbol{\beta}$ 为温度参数，用于自适应调节值函数的奖励。

目标值网络的输出计算公式如下：

$$\boldsymbol{y}_t=\boldsymbol{r}_t^{\mathrm{c}}(\boldsymbol{s}_t^{\mathrm{c}},\boldsymbol{s}_{t+1}^{\mathrm{c}};a_t)+\gamma\boldsymbol{V}_{\boldsymbol{\pi}}(\boldsymbol{s}_t^{\mathrm{c}}) \tag{18}$$

该网络使用均方误差损失函数计算目标值与现实值之间的差值 $\boldsymbol{\delta}$ 并进行梯度下降处理。均方误差损失函数计算公式如下：

$$\boldsymbol{\delta}=(\boldsymbol{y}_t-\boldsymbol{Q}(\boldsymbol{s}_t^{\mathrm{c}},a_t))^2 \tag{19}$$

通过减小熵值损失来减小估计值的损失方差，据此，设计了如下的温度熵损失目标函数：

$$\boldsymbol{J}(a)=\boldsymbol{\pi}_t\left((\boldsymbol{s}_t^{\mathrm{c}})^{\mathrm{T}}\right)\left[-\boldsymbol{\beta}\left(\mathrm{lb}\,\boldsymbol{\pi}_t(\boldsymbol{s}_t^{\mathrm{c}})+\boldsymbol{H}_{\mathrm{e}}\right)\right] \tag{20}$$

在目标值网络中加入动作值函数期望和熵值计算来更新策略，从而对离散动作空间输出准确的概率分布以减小误差。因此，设计策略损失目标函数如下：

$$\boldsymbol{J}_{\boldsymbol{\pi}}(\phi)=E_{\boldsymbol{s}_t^{\mathrm{c}}\sim D}\left[\boldsymbol{\pi}_t\left((\boldsymbol{s}_t^{\mathrm{c}})^{\mathrm{T}}\right)\left(\boldsymbol{\beta}\,\mathrm{lb}\,\boldsymbol{\pi}_{\phi}(\boldsymbol{s}_t^{\mathrm{c}})-\boldsymbol{Q}_{\boldsymbol{\theta}}(\boldsymbol{s}_t^{\mathrm{c}},a_t)\right)\right] \tag{21}$$

综上所述，DQAC 推抓技能自监督学习算法描述如下。

算法 1 DQAC 推抓技能学习算法

输入： 当前状态 $\boldsymbol{s}$

输出： $\boldsymbol{Q},\boldsymbol{\pi}^*$

初始化视觉网络参数 ω 和动作策略网络 $\boldsymbol{\pi}(\boldsymbol{\varphi})$，值网络 $\boldsymbol{Q}_{u_1},\boldsymbol{Q}_{u_2}$，目标值网络 $\overline{\boldsymbol{Q}}_{u_1},\overline{\boldsymbol{Q}}_{u_2}$

$u_1 \leftarrow \overline{u}_1$，$u_2 \leftarrow \overline{u}_2$

初始化图像经验池 D 和特征经验池 D^{c}，定义 M

迭代次数 $< M$：

计算置信度 $\boldsymbol{Q}$ 值：

$$\boldsymbol{Q}_{t+1}=\boldsymbol{R}_{a_t}(\boldsymbol{s}_t,\boldsymbol{s}_{t+1})+\gamma\max_a(\boldsymbol{s}_{t+1},a;\boldsymbol{\omega})$$

根据功用性图最大 $\boldsymbol{Q}$ 值，得到机器人工作点位置 $p(x,y,z)$ 和旋转角度 $\boldsymbol{\theta}$

初始化特征状态信息 $\boldsymbol{s}^{\mathrm{c}}$

$a_t \sim \boldsymbol{\pi}_\phi(a_t|\boldsymbol{s}_t^{\mathrm{c}})$ #根据策略选择执行动作

选择执行动作 a_t

$\boldsymbol{s}_{t+1}^{\mathrm{c}} \sim p(\boldsymbol{s}_{t+1}^{\mathrm{c}}|\boldsymbol{s}_t^{\mathrm{c}},a_t)$ #转移概率更新下一特征状态

环境反馈奖励值 r,r^{c}

策略函数计算：

$$\boldsymbol{\pi}^*=\arg\max_{\boldsymbol{\pi}} E_{(\boldsymbol{s},a)\sim\rho_{\boldsymbol{\pi}}}\left(\sum_t R(\boldsymbol{s}_t^{\mathrm{c}},a_t)+\boldsymbol{\beta}\boldsymbol{H}_{\mathrm{e}}(\boldsymbol{\pi}(\cdot|\boldsymbol{s}_t^{\mathrm{c}}))\right)$$

计算损失函数，优化目标，更新网络参数

$\hat{\boldsymbol{\varphi}} \leftarrow \boldsymbol{\varphi}-\lambda_{\boldsymbol{\pi}}\nabla_\phi \boldsymbol{J}_{\boldsymbol{\pi}}(\boldsymbol{\varphi})$ #更新策略网络参数

$\hat{\boldsymbol{\mu}} \leftarrow \boldsymbol{\mu}-\lambda_Q\nabla_\mu \boldsymbol{J}(\mu)$ #更新值网络参数

$\hat{\boldsymbol{\beta}} \leftarrow \boldsymbol{\beta}-\lambda\nabla_{\boldsymbol{\beta}} \boldsymbol{J}(\boldsymbol{\beta})$ #更新温度参数

$\hat{\boldsymbol{Q}} \leftarrow \tau\hat{\boldsymbol{Q}}+(1-\tau)\boldsymbol{Q}$ #目标网络参数进行软更新

$\boldsymbol{s}_{t+1}=\boldsymbol{s}_t$，$\boldsymbol{s}_{t+1}^{\mathrm{c}}=\boldsymbol{s}_t^{\mathrm{c}}$

将元组存储到图像经验池和特征经验池中

下一步训练，在 2 个经验池分别采样 $(\boldsymbol{s}_t,\boldsymbol{s}_{t+1},a,r)$ 和 $(\boldsymbol{s}_t^{\mathrm{c}},\boldsymbol{s}_{t+1}^{\mathrm{c}},a,r^{\mathrm{c}})$

直到迭代次数 $> M$

4 实验与结果（Experiment and results）

4.1 仿真环境搭建

为了验证机器人采用本算法抓取目标物体的性能，使用 V-REP 3.5.0[31] 动力学仿真软件模拟机器人在未知环境下对目标物体的抓取。该软件内部的运动学模块可准确地模拟真实机器人的运动轨迹，同时还具有重力等物理引擎，可模拟真实物体属性。使用 RGB-D 相机采集工作空间状态信息。该相机由被动 RGB 相机和主动深度传感器组成，除了提供 RGB 图像外还提供每个像素的深度信息，可将每个像素的深度值快速转换为点云信息用于 3D 感知。

建立了图 8 所示的仿真实验环境。建立装有 RG2 夹具的 UR5 机械臂模型，并在工作空间正上方与斜上方 45° 的位置安装 RGB-D 相机，该相机会在每次机械臂执行完动作后进行图像采集，提供完整的且大小为 640×480 的深度信息。

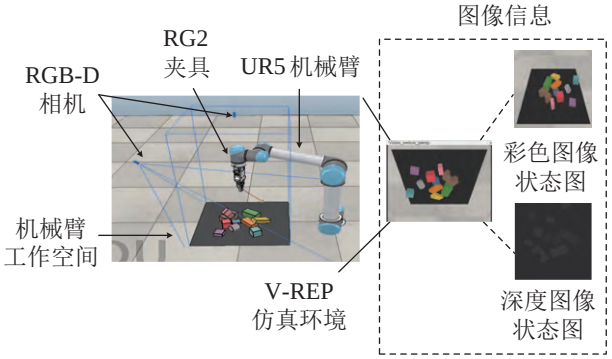

图 8 V-REP 仿真环境与图像信息

Fig.8 V-REP simulation environment and image information

仿真硬件配置为 3.6 GHz Intel Core i9-9900 k CPU 和 NVIDIA 2070S GPU，操作系统为 Ubuntu 18.04 LTS，V-REP 的版本为 3.5.0 的教育版，采用 0.4 版本的 PyTorch 框架来训练网络模型。

4.2 训练阶段

在训练阶段，待抓取的目标物体的颜色、形状及位置随机，机器人通过不断试错寻找最优策略来实现未知环境下对目标物体的成功抓取。该阶段采用特征提取网络 RGSA-Net 生成推动和抓取功用性图，采用 DQAC 框架选取最优动作，前期采用初始探索概率为 0.50 的 ε 贪婪策略进行探索，且概率随训练回合数增加而减小，并采用最大贪婪策略与动作分类策略探索推、抓动作之间的协同，自监督训练的回合数 M 取 2500。如图 1 所示，在仿真环境中杂乱放置了 m 个随机目标块和 n 个不同形状的基本块，机器人通过协调推抓动作实现对目标物体的成功抓取。

为了验证本方法的有效性，将其与其他 4 种方法在训练阶段的抓取性能进行了对比。

(1) RAND：一种不经过监督训练而采取随机像素点抓取的方法；

(2) Grasping-only[24]：一种贪婪的确定性抓取策略，它使用单个 FCN（全卷积网络）进行抓取，该网络使用二分类（来自试错）的监督。此策略下的机器人仅执行抓取动作；

(3) VPG[24]：提出面向目标的抓取任务，使用基于 DQN 的强化学习框架训练推、抓之间的协同实现目标抓取；对于给定目标物体，使用 2 个动作全卷积网络映射动作 Q 值。

(4) GIT[25]：一种深度强化学习方法，使用目标分割网络提取特征来增强机器人感知，基于 DQN

二分类器进行机器人推动与抓取训练。

4.2.1　性能展示

训练阶段执行动作次数的最大阈值设置为 30，当动作数超过阈值或整个工作区域无目标物体时，重置抓取环境开启新一轮的抓取训练，该过程中随机指定目标物块。使用不同算法对机器人进行 2500 次的训练，不同算法的耗时如表 1 所示。图 9 绘制了机器人在不同方法下的抓取成功率。

表 1　不同算法 2500 次训练耗时

Tab.1　Time consumption of different algorithms in 2500 training

方法	训练耗时 /h
RAND	4.45
Grasping-only	5.58
VPG	6.33
GIT	5.72
RGSA-Net	**6.15**
DQAC	**5.84**
RGSA-Net + DQAC	**6.24**

4.2.2　性能分析

从表 1 和图 9 可见，在训练过程中，RAND 算法训练耗时最短，但其面对目标抓取任务时忽略环境而采取随机策略选择动作，使得抓取成功率极低。Grasping-only 算法中加入了卷积神经网络处理视觉输入，训练耗时较 RAND 算法有所增加，但其仅采用抓取动作，忽略了杂乱环境对目标任务的影响，使得其抓取成功率偏低。VPG 方法中首次将推动动作加入动作集，但其仅采用传统 DQN 进行训练，并未很好地考虑二者的协同作用，且所用网络较为繁琐导致训练耗时增加，在所有算法中耗时最长，抓取成功率在 60% 左右。GIT 算法使用简单的二分类器对动作进行拟合预测以提高动作协同效率，同时简化协同训练网络，使得训练耗时有所降低，抓取成功率在 60%～70% 之间。

本文算法在前期对视觉输入进行处理时使用了分组拆分注意力 RGSA 模块，该模块较传统残差块具有较深的网络层次，使得训练时长有所增加，但训练网络参数时融合自适应参数方法，通过加快梯度下降速度来提高网络收敛速度，最终耗时比 GIT 算法稍长。本文的 DQAC 框架在传统演员－评论家算法的基础上将动作网络拆分成两部分，更好地训练了推、抓之间的协同，其网络结构较为简单，耗时与 GIT 算法相差不大。但 DQAC 框架采用传统的视觉机制模块，前期对于目标定位的准确程度不足。本文将视觉机制模块与动作机制模块进行结合后，训练时长较 DQAC 框架有所增加，但其效果达到最佳，受环境干扰较小，抓取成功率在 90% 左右，总体训练性能优于其他方法。

视觉效果对比如图 10 所示，由本文算法得到的推抓功用性图能更好地定位到目标物块，推、抓中心均恰好在目标物块之上。而使用 GIT 算法产生

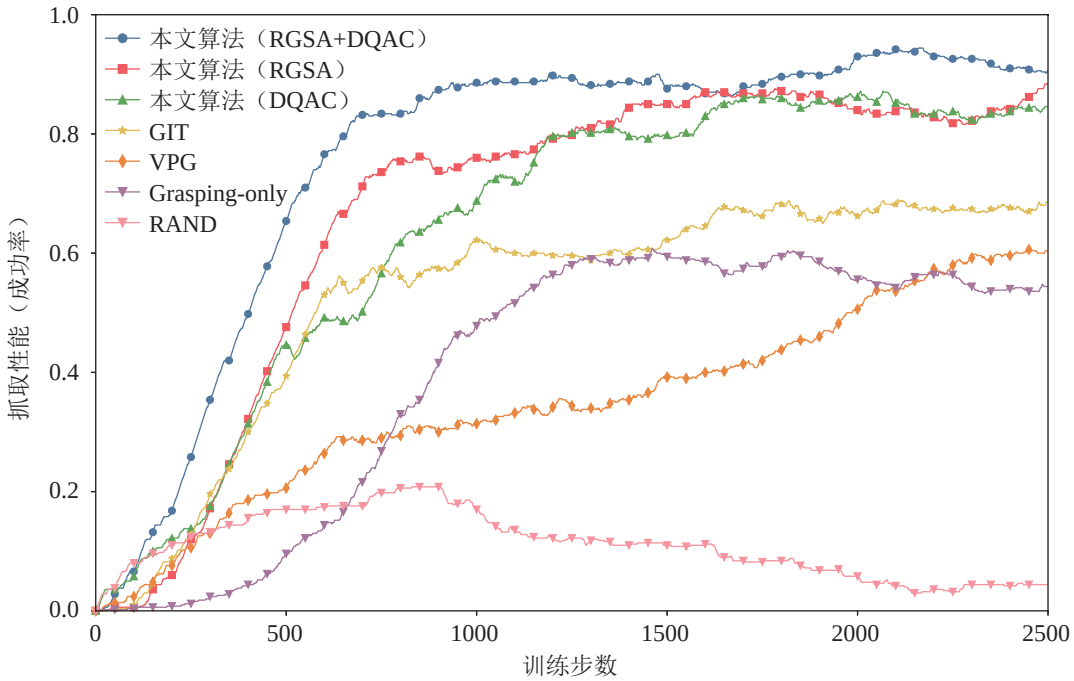

图 9　抓取成功率对比

Fig.9　Comparison of grasp success ratio

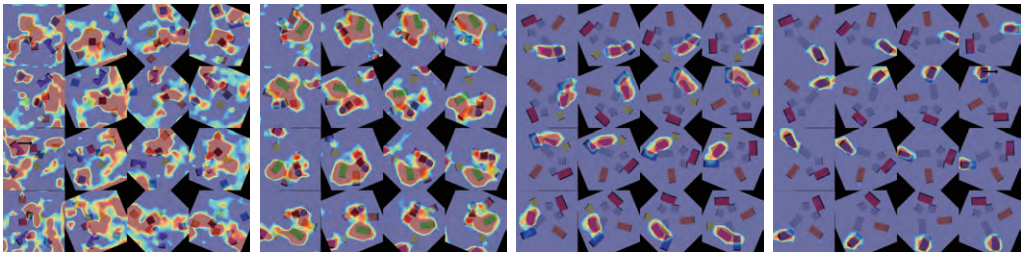

图 10 GIT 与 RGSA 算法视觉效果对比

Fig.10 Comparison of visual effects between GIT and RGSA algorithms

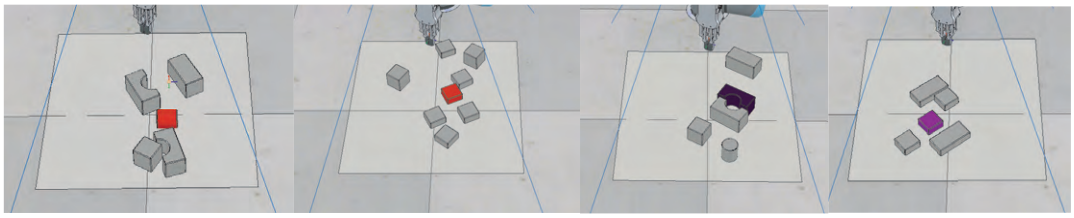

图 11 松散放置物块情形的 4 种测试案例

Fig.11 4 test cases with loosely placed blocks

的推抓功用性图某一区域热度较高，无法实现对目标的精准定位。这进一步验证了视觉机制模块对机器人最终抓取效果的重要作用。

4.3 测试阶段

为了检验本算法在不同实验场景中的有效性，在仿真中设置了 3 组对比实验进行测试，分别为松散放置物块情形、紧密放置物块情形和完全遮挡物块情形。

与 RAND、Grasping-only、VPG、GIT 四种方法进行对比来验证本方法的有效性。对每个测试案例设置 30 轮实验，每轮的动作执行次数阈值设置为 n 次。若机器人在 n 次内实现对目标的抓取，则记为一轮成功的抓取案例，阈值选定与环境有关。定义抓取成功率为

$$\frac{\#\text{成功抓取目标轮数}}{\#\text{实验总轮数}}$$

定义平均运动次数为

$$\frac{\sum_{1}^{n}\text{每一轮成功运动数（推动 + 抓取）}}{n\text{（重复测试实验次数）}}$$

该指标数值越小，说明实验成功率越高。

4.3.1 松散放置物块情形

由于训练过程的实验场景均是随机摆放实验场景，故本文只设置了 4 个松散放置物块情形的测试案例。如图 11 所示，每一个测试案例中设置一个带特殊颜色的目标物块，其余为干扰物块，用来模拟该算法在松散放置物块环境中的实验效果。

松散放置物块情形的 4 种测试案例较为简单，可以通过 RGB-D 相机直接获取目标物块的完整特征，无法验证推、抓技能的协同作用，故将执行动作阈值设为 1，即一次抓取就成功则计入成功案例，否则记为失败，用以检验视觉机制模块对抓取效果的影响。实验结果对比如表 2 所示。

表 2 松散放置物块案例的平均表现

Tab.2 Average performance in the cases of loosely placed blocks

方法	平均成功率 /%	平均运动次数
RAND	39.5	4.325 ± 0.41
Grasping-only	52.0	3.170 ± 0.30
VPG	72.0	3.901 ± 0.03
GIT	89.5	2.645 ± 0.15
RGSA-Net	**95.3**	**1.413 ± 0.22**
DQAC	**93.1**	**1.751 ± 0.80**
RGSA-Net + DQAC	**97.4**	**1.212 ± 0.26**

4.3.2 紧密放置物块情形

对于紧密放置物块的实验环境，设计了 8 个不同形状物块的抓取测试案例来验证训练模型的性能，各测试案例中设置一个特殊颜色的目标物块，被其他物块紧紧包围，用来模拟推、抓之间的协同训练效果。机器人通过动作集合 $a=\{a_{\mathrm{p}},a_{\mathrm{g}}\}$ 执行有限次动作，其中 a_{p} 表示机器人面向目标执行推动

动作，改变环境状态提供足够的抓取空间；a_g 表示抓取动作。推动和抓取动作总数的阈值设为 5，即推动和抓取动作在 5 次以内时，表明推抓成功。根据环境状态选择动作并得到环境的反馈训练网络参数，最终实现成功抓取。测试案例如图 12 所示。

紧密放置物块情形突出推动动作对最终抓取效果的辅助作用，即动作机制模块对抓取效果的影响。图 13 和图 14 分别展示了本方法和其他 4 种方法在 8 个测试案例中的表现，由于每个测试案例的目标场景是不同的，故本文的改进方法表现不同。表 3 展示了本文方法与其他 4 种方法的平均抓取成功率与平均运动次数对比结果。

表 3　紧密放置物块案例的平均表现

Tab.3　Average performance in the cases of closely placed blocks

方法	平均成功率 /%	平均运动次数
RAND	17.5	4.775 ± 0.60
Grasping-only	35.0	4.325 ± 0.98
VPG	70.0	4.025 ± 0.83
GIT	87.5	3.675 ± 0.90
RGSA-Net	**90.3**	**3.575 ± 0.80**
DQAC	**90.0**	**3.506 ± 0.75**
RGSA-Net + DQAC	**91.1**	**3.436 ± 0.85**

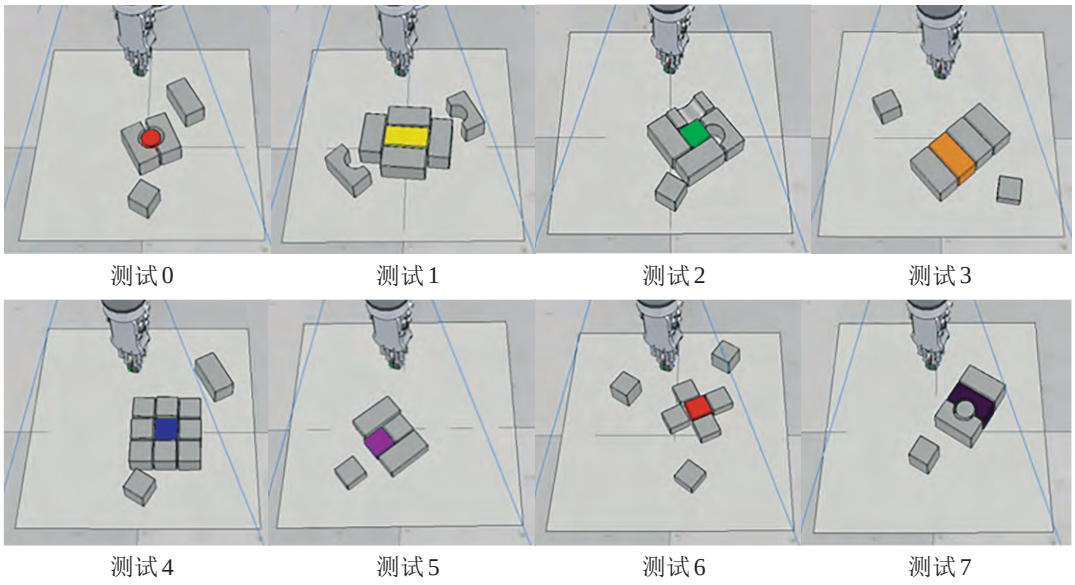

图 12　紧密放置物块情形的 8 种测试案例

Fig.12　8 test cases with closely placed blocks

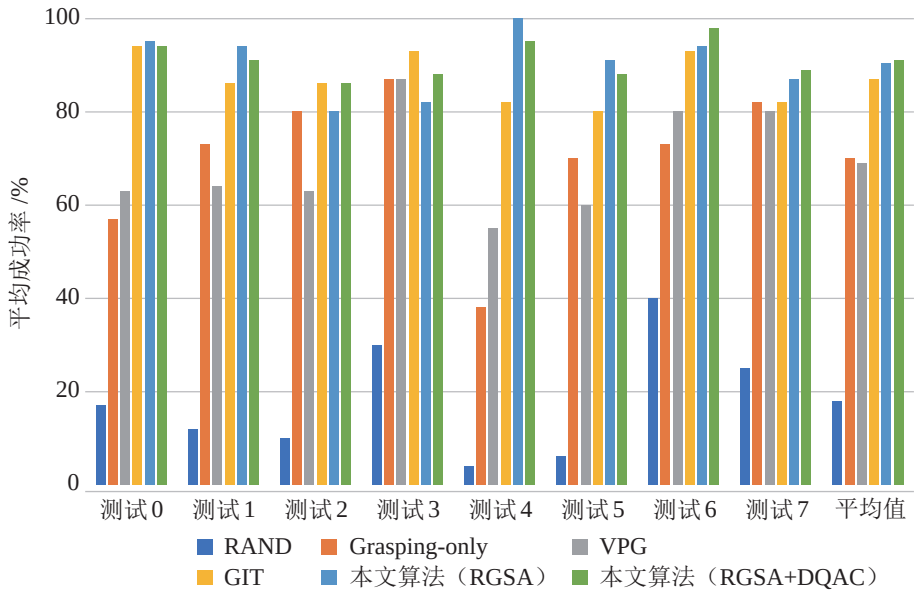

图 13　紧密放置物块案例的平均成功率

Fig.13　Average success ratio in the cases of closely placed blocks

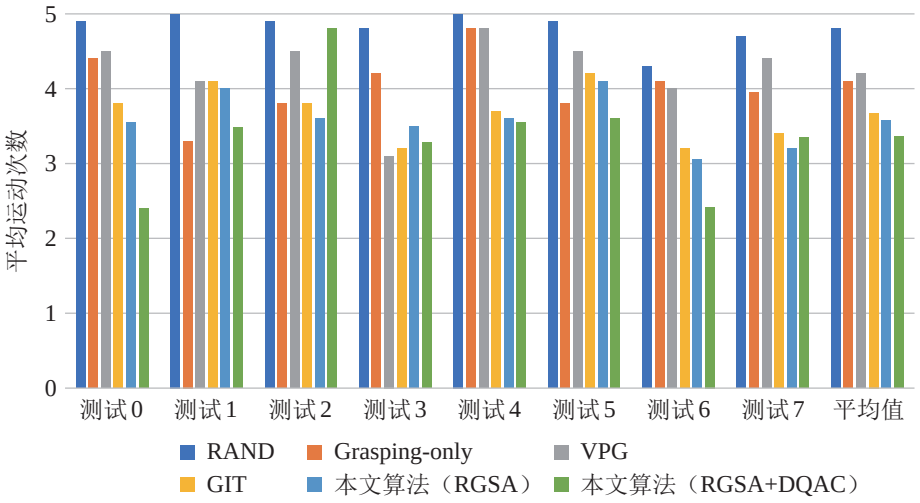

图 14 紧密放置物块案例的平均运动次数

Fig.14 Average motion times in the cases of closely placed blocks

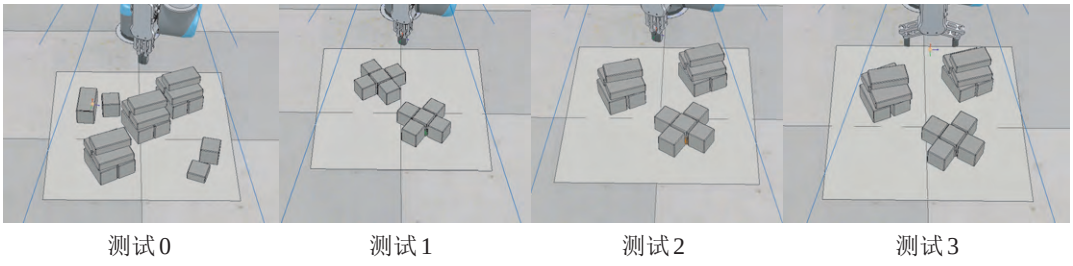

图 15 完全遮挡物块情形的 4 种测试案例

Fig.15 4 test cases with completely occluded blocks

4.3.3 完全遮挡物块情形

对于完全遮挡物块的实验场景，设置了 4 种不同的测试案例。如图 15 所示，每个场景中的目标物块都是不可见的，用来模拟前期无法生成高度图时，机器人对于环境中目标物块的探索能力。

在该场景下，由于目标物块不可见，故本文算法在获取到当前状态的深度信息后，优先选择较高的堆叠部分使用 ε 贪婪策略进行探索，使目标物块特征显现，该场景转化成紧密放置物块情形。由于该过程为探索过程，机器人仅采用推动动作，故将推动动作阈值设置为 $n_{\text{push}} = n_{\text{clutter}} - 1$，其中 n_{clutter} 表示工作空间中干扰块的数量。该过程检验了视觉机制模块与动作机制模块的协同作用效果。实验结果对比如表 4 所示。

表 4 完全遮挡物块案例的平均表现

Tab.4 Average performance in the cases of completely occluded blocks

方法	平均成功率 /%	平均运动次数
RAND	47.5	9.832 ± 1.14
Grasping-only	50.1	8.603 ± 0.78
VPG	78.5	7.645 ± 1.33
GIT	89.2	7.175 ± 1.28
RGSA-Net	**90.7**	**5.705 ± 1.04**
DQAC	**91.3**	**5.516 ± 1.75**
RGSA-Net + DQAC	**93.8**	**5.436 ± 1.24**

4.4 结果分析

松散放置物块情形模拟了真实场景中随机放置目标物块的抓取情况，用于检验视觉机制模块对最终抓取效果的影响。由表 2 可以看出，加入 RGSA-Net 视觉机制模块后，算法以 1.41 的平均移动次数实现了 95.3% 的抓取成功率。而使用传统视觉机制的 GIT 算法仅实现了 89.5% 的抓取成功率，即使加入动作机制模块的 DQAC 算法仍无法很好地完成视觉特征提取，仅实现了 92.9% 的抓取成功率。而本文的 RGSA-Net 算法具有较好的视觉特征效果。

紧密放置物块情形模拟了真实世界中目标物块被紧紧包围的场景，此时由于没有足够的空间供机械手完成抓取操作，故可以检验动作机制模块 DQAC 对抓取效果的影响。图 13 和图 14 是不同对比方法在 8 个紧密放置物块测试案例中的性能指标。其中，RAND 算法与 Grasping-only 算法的

平均成功率在 10%～35% 之间，但平均运动次数在 4.3 次以上。VPG 算法仅将推动动作引入抓取任务而未考虑二者之间的相互作用，其平均成功率为 60%～75% 之间，平均运动次数为 4.0 次，进一步验证了推动动作在抓取过程中的重要性。GIT 算法基于 DQN 二分类器进一步训练推动与抓取之间的协同，平均成功率在 85% 以上，平均运动次数为 3.6 次左右。而本文采取基于演员－评论家框架的 DQAC 算法，以平均 3.4 次的动作实现了 91.1% 的成功率，性能优于其他基线方法。

完全遮挡物块情形模拟真实世界中目标物块完全不可见，需要机器人探索环境、寻找物块并成功抓取的场景。该过程用于检验视觉机制模块与动作机制模块的协同作用效果。由表 4 数据可得，RAND 和 Grasping-only 两种方法对抓取目标任务的效率比较低，抓取成功率均在 50% 左右。RAND 算法倾向于随机抓取目标物体而忽略了目标周围物体的密集程度，故实验效果较差。Grasping-only 算法还未引入非抓取算法的协同作用，仅采用抓取动作改变环境结构，当目标物体被完全遮挡时往往会陷入局部最优导致任务失败。VPG 算法使用基于值函数的 DQN 算法计算出的 Q 值，忽略了目标物块先验知识的影响，有较多错误的抓取动作和冗余的推动动作，故平均成功率仅为 78%，平均运动次数为 7.645。GIT 算法使用动作二分类器来协调机器人的推动与抓取动作，但在完全遮挡任务前期，高度图不完整，GIT 算法无法通过良好的视觉感知获取目标物块周围的密集程度，故平均成功率为 89.2%，平均运动次数为 7.175。本文提出的基于 RGSA-Net 和 DQAC 算法的机器人自监督学习方法，使用拆分注意力模块提高对目标物块的关注度，使用 DQAC 算法进行动作评判及反馈，增强了对杂波环境的感知能力，完成任务的平均成功率达到了 93.8%，平均运动次数为 5.436，性能达到最优。

5　结论（Conclusion）

众所周知，在服务机器人共融式宜人化进化过程中，自主认知与操作杂乱场景下的工具是其必备的技能。本文提出一种基于功用性图的目标推抓技能自监督学习方法，克服了抓取成功率低的问题，提高了机器人在复杂环境中对目标抓取技能的学习性能。

提出了基于 RGSA-Net 特征提取网络的视觉机制模块，生成了准确的推抓功用性图；提出了基于 DQAC 自监督学习方法的动作机制模块，实现了推、抓之间的协同。在训练阶段提高了抓取性能，在测试阶段提高了 3 组测试案例的抓取成功率，减少了平均移动次数，在仿真环境中设计实验验证了本文方法的有效性。接下来，将搭建实物实验环境并进一步验证本文方法的有效性。

参考文献（References）

[1] Morrison D, Corke P, Leitner J. Multi-view picking: Next-best-view reaching for improved grasping in clutter[C]//International Conference on Robotics and Automation. Piscataway, USA: IEEE, 2019: 8762-8768.

[2] Redmon J, Angelova A. Real-time grasp detection using convolutional neural networks[C]//IEEE International Conference on Robotics and Automation. Piscataway, USA: IEEE, 2015: 1316-1322.

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

[4] Schmidt P, Vahrenkamp N, Wachter M, et al. Grasping of unknown objects using deep convolutional neural networks based on depth images[C]//IEEE International Conference on Robotics and Automation. Piscataway, USA: IEEE, 2018: 6831-6838.

[5] Mahler J, Liang J, Niyaz S, et al. Dex-Net 2.0: Deep learning to plan robust grasps with synthetic point clouds and analytic grasp metrics[C]//Robotics: Science and Systems. Cambridge, USA: MIT, 2017. DOI: 10.15607/rss.2017.xiii.058.

[6] Kumra S, Joshi S, Sahin F. Antipodal robotic grasping using generative residual convolutional neural network[C]//IEEE/RSJ International Conference on Intelligent Robots and Systems. Piscataway, USA: IEEE, 2020: 9626-9633.

[7] Lou X, Yang Y, Choi C. Learning to generate 6-DoF grasp poses with reachability awareness[C]//IEEE International Conference on Robotics and Automation. Piscataway, USA: IEEE, 2020: 1532-1538.

[8] Shao Q Q, Hu J, Wang W M, et al. Suction grasp region prediction using self-supervised learning for object picking in dense clutter[C]//IEEE 5th International Conference on Mechatronics System and Robots. Piscataway, USA: IEEE, 2019: 7-12.

[9] He K M, Zhang X Y, Ren S Q, et al. Deep residual learning for image recognition[C]//IEEE Conference on Computer Vision and Pattern Recognition. Piscataway, USA: IEEE, 2016: 770-778.

[10] Pinto L, Gupta A. Supersizing self-supervision: Learning to grasp from 50K tries and 700 robot hours[C]//IEEE International Conference on Robotics and Automation. Piscataway, USA: IEEE, 2016: 3406-3413.

[11] Shukla P, Kumar H, Nandi G C. Robotic grasp manipulation using evolutionary computing and deep reinforcement learning[J]. Intelligent Service Robotics, 2021, 14(1): 61-77.

[12] Kalashnikov D, Irpan A, Pastor P, et al. Learning dexterous in-hand manipulation[J]. International Journal of Robotics Research, 2018, 39(1): 3-20.

[13] Sarantopoulos I, Kiatos M, Doulgeri Z, et al. Total singulation with modular reinforcement learning[J]. IEEE Robotics and Automation Letters, 2021, 6(2): 4117-4124.

[14] Quillen D, Jang E, Nachum O, et al. Deep reinforcement learning for vision-based robotic grasping: A simulated comparative evaluation of off-policy methods[C]//IEEE International Conference on Robotics and Automation. Piscataway, USA: IEEE, 2018: 6284-6291.

[15] Hou Y X, Li J, Fang Z H, et al. An initialization method of deep Q-network for learning acceleration of robotic grasp[C]// IEEE International Conference on Networking, Sensing and Control. Piscataway, USA: IEEE, 2020. DOI: 10.1109/ICNSC48988.2020.9238061.

[16] Deng Y H, Guo X F, Wei Y X, et al. Deep reinforcement learning for robotic pushing and picking in cluttered environment [C]//IEEE/RSJ International Conference on Intelligent Robots and Systems. Piscataway, USA: IEEE, 2019: 619-626.

[17] Xie X, Li C Y, Zhang C, et al. Learning virtual grasp with failed demonstrations via Bayesian inverse reinforcement learning[C] //IEEE/RSJ International Conference on Intelligent Robots and Systems. Piscataway, USA: IEEE, 2019: 1812-1817.

[18] Johannink T, Bahl S, Nair A, et al. Residual reinforcement learning for robot control[C]//IEEE International Conference on Robotics and Automation. Piscataway, USA: IEEE, 2019: 6023-6029.

[19] Mohammadi H B, Zamani M A, Kerzel M, et al. Mixed-reality deep reinforcement learning for a reach-to-grasp task[C]//28th International Conference on Artificial Neural Networks. Cham, Switzerland: Springer, 2019: 611-623.

[20] Ni P Y, Zhang W G, Zhang H R, et al. Learning efficient push and grasp policy in a totebox from simulation[J]. Advanced Robotics, 2020, 34(13): 873-887.

[21] Gui B X, Qian K, Chen S H, et al. Knowledge induced deep Q-network for robot push and grasp manipulation skills learning [C]//Chinese Automation Congress. Piscataway, USA: IEEE, 2020: 4078-4083.

[22] Joshi S, Kumra S, Sahin F. Robotic grasping using deep reinforcement learning[C]//IEEE 16th International Conference on Automation Science and Engineering. Piscataway, USA: IEEE, 2020: 1461-1466.

[23] Zhang J H, Zhang W, Song R, et al. Grasp for stacking via deep reinforcement learning[C]//IEEE International Conference on Robotics and Automation. Piscataway, USA: IEEE, 2020: 2543-2549.

[24] Zeng A, Song S, Welker S, et al. Learning synergies between pushing and grasping with self-supervised deep reinforcement learning[C]//IEEE/RSJ International Conference on Intelligent Robots and Systems. Piscataway, USA: IEEE, 2018: 4238-4245.

[25] Yang Y, Liang H, Choi C. A deep learning approach to grasping the invisible[J]. IEEE Robotics and Automation Letters, 2020, 5(2): 2232-2239.

[26] Gong L Y, He D, Li Z H, et al. Efficient training of BERT by progressively stacking[C]//36th International Conference on Machine Learning. 2019: 2337-2346.

[27] Krizhevsky A, Sutskever I, Hinton G E. ImageNet classification with deep convolutional neural networks[J]. Communications of the ACM, 2017, 60(6): 84-90.

[28] Badrinarayanan V, Kendall A, Cipolla R. SegNet: A deep convolutional encoder-decoder architecture for image segmentation [J]. IEEE Transactions on Pattern Analysis and Machine Intelligence, 2017, 39(12): 2481-2495.

[29] Li D Q, Hu X Q, Wang S Q, et al. Hyperspectral images ground object recognition based on split attention[C]//IEEE 2nd International Conference on Big Data, Artificial Intelligence and Internet of Things Engineering. Piscataway, USA: IEEE, 2021: 324-330.

[30] Barto A G, Sutton R S, Anderson C W. Neuronlike adaptive elements that can solve difficult learning control problems[J]. IEEE Transactions on Systems, Man, and Cybernetics, 1983, 13(5): 834-846.

[31] Rohmer E, Singh S P N, Freese M. V-REP: A versatile and scalable robot simulation framework[C]//IEEE/RSJ International Conference on Intelligent Robots and Systems. Piscataway, USA: IEEE, 2013: 1321-1326.

作者简介：

吴培良（1981 –），男，博士，教授。研究领域：家庭服务机器人工具认知，机器人抓取，强化学习。

刘瑞军（1996 –），女，硕士生。研究领域：家庭服务机器人工具操作技能学习，强化学习。

毛秉毅（1964 –），男，博士。研究领域：家庭服务机器人功用性认知。