

基于深度强化学习算法的自主式水下航行器深度控制

王日中, 李慧平, 崔迪, 徐德民

(西北工业大学航海学院, 陕西 西安 710072)

摘要: 研究了基于深度强化学习算法的自主式水下航行器(AUV)深度控制问题。区别于传统的控制算法, 深度强化学习方法让航行器自主学习控制律, 避免人工建立精确模型和设计控制律。采用深度确定性策略梯度方法设计了 actor 与 critic 两种神经网络。actor 神经网络给出控制策略, critic 神经网络用于评估该策略, AUV 的深度控制可以通过训练这两个神经网络实现。在 OpenAI Gym 平台上仿真验证了算法的有效性。

关键词: 自主式水下航行器; 深度控制; 深度强化学习

中图分类号: TP242.6

文献标识码: A

doi: 10.11959/j.issn.2096-6652.202038

Depth control of autonomous underwater vehicle using deep reinforcement learning

WANG Rizhong, LI Huiping, CUI Di, XU Demin

School of Marine Science and Technology, Northwestern Polytechnical University, Xi'an 710072, China

Abstract: The depth control problem of autonomous underwater vehicle (AUV) by using deep reinforcement learning method was mainly studied. Different from the traditional control algorithm, the deep reinforcement learning method allows the AUV to learn the control law independently, avoiding the artificial establishment of accurate model and design control law. The deep deterministic policy gradient method was used to design two neural networks: actor and critic. Actor neural network enabled agents to make corresponding control actions. Critic neural network was used to estimate the action-value function in reinforcement learning. The AUV depth control was conducted by training of actor and critic neural networks. The effectiveness of the algorithm was proved by simulation on OpenAI Gym.

Key words: autonomous underwater vehicle, depth control, deep reinforcement learning

1 引言

随着机器人技术和传感器、控制、导航等相关技术的迅速发展, 自主式水下航行器(autonomous underwater vehicle, AUV)在水下探测、搜救、检测等领域得到了广泛的应用。传统的 AUV 运动控制需要建立一个精确的 AUV 模型, 控制质量与模

型精度密切相关。而 AUV 是一个典型的多变量、强耦合的非线性系统, 其参数和模型具有不确定性。这些不确定性给水下机器人的运动控制带来了巨大挑战。针对这种复杂环境下 AUV 的决策与控制问题, 机器学习(machine learning, ML)提供了一种有效的解决方案。作为一种最基本的 ML 方法, 监督学习的方法可以通过 AUV 运动数据训练神经

收稿日期: 2020-11-08; 修回日期: 2020-11-30

通信作者: 李慧平, lihuiping@nwpu.edu.cn

基金项目: 国家自然科学基金资助项目(No.61922068, No.61733014); 陕西省杰出青年科学基金资助项目(No.2019JC-14); 西北工业大学翱翔青年学者项目(No.20GH0201111)

Foundation Items: The National Natural Science Foundation of China (No.61922068, No.61733014), The Science Foundation for Distinguished Young Scholars of Shaanxi Province (No.2019JC-14), Northwestern Polytechnical University Aoxiang Youth Scholar Program (No.20GH0201111)

网络, 从而对模型误差和不确定性进行自适应补偿, 实现 AUV 的高精度运动控制^[1]。

强化学习是更有效的 ML 方法^[2]。强化学习算法不需要系统模型信息以及人为设计控制律, 而是通过设置奖励函数让机器人自己探索、学习有用的控制策略。对于一个不断与环境进行交互的机器人来说, 强化学习将系统实时状态和动作映射为奖励函数, 机器人依靠现有数据寻找可以产生最大回报的动作, 从而提高机器人在不确定场景中的智能性。例如, 在西洋双陆棋比赛中, 使用强化学习算法训练的机器人击败了世界冠军^[3], 而且, 强化学习算法训练的智能体在雅达利游戏公司出品的多种游戏中的表现超过了人类的水平^[4]。

2013 年, DeepMind 公司提出了深度 Q 学习网络 (deep Q-learning network, DQN) 算法, 这种将强化学习与深度神经网络结合的算法叫作深度强化学习 (deep reinforcement learning, DRL) 算法^[4]。在系统模型未知的情况下, 智能体通过设置的奖励函数来学习探索更好的控制策略, 并利用深度神经网络估计强化学习评价指标和拟合当前的控制策略。当控制器训练好时, 因为神经网络前向传播的计算速度极快, 能很快地实现输入状态和控制输入之间端到端的解算, 控制的实时性能得到保障。在 2016 年和 2017 年, AlphaGo 使用 DRL 算法连续两年击败围棋世界冠军, 引起了社会轰动^[5]。目前, DRL 算法发展迅速, 对于提高机器人的自主性和智能性具有十分重要的意义。本文旨在研究一种 DRL 算法来解决 AUV 的深度控制问题。

由于 AUV 是一个连续时间系统, 其控制行为有无限种选择, 而经典的强化学习 (reinforcement learning, RL) 算法主要针对状态或控制空间有限的离散情况, 因此, 无法直接控制 AUV 运动。为了解决这个问题, Silver D 等人^[6]提出确定性策略梯度 (deterministic policy gradient, DPG) 算法, 解决了在随机策略梯度下具有大动作空间的智能体采样数多的问题, 使得通过 RL 算法控制连续时间系统成为可能。在 2016 年国际学习表征会议 (ICLR) 上, Lillicrap T P 等人^[7]将深度神经网络与 DPG 结合, 提出了深度确定性策略梯度 (deep deterministic policy gradient, DDPG), 并将其应用于简单的连续系统控制。而苏黎世 ETH 机器人实验室成功地将 DRL 算法应用于腿部机器人的运动控制^[8]。

AUV 在执行任务时, 一般要求在固定的深度保

持稳定的运动, 因此深度控制是 AUV 的基本控制目标。本文的目的是探索一种更加先进的 AUV 深度控制方法, 解决 AUV 模型的不确定性和非线性给控制带来的难题, 并提高 AUV 的控制性能; 仅依靠训练得到的与周围环境的交互数据, 设计 DRL 算法, 从而实现 AUV 的深度控制。

2 AUV 的运动模型

由于 AUV 的深度控制主要是在垂直面实现的, 因此本文仅考虑其在铅垂平面内的运动方程。地面坐标系与体坐标系下的 AUV 如图 1 所示, oxy 与 $o_b x_b y_b$ 为地面坐标系, $o_b x_b y_b$ 为体坐标系; 水平舵角 δ_e 为控制输入; $[x, y, \theta]^T$ 为控制输出, 其中 $[x, y]^T$ 与 θ 分别为 AUV 在垂直面的位置与俯仰角。

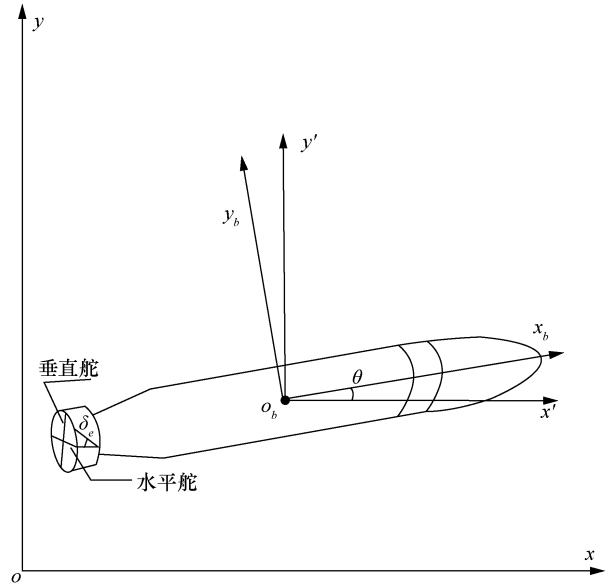


图1 地面坐标系与体坐标系下的 AUV

AUV 在垂直面的二维运动模型如下^[9]:

$$\begin{aligned}\dot{x} &= v_x \cos \theta - v_y \sin \theta \\ \dot{y} &= v_x \sin \theta + v_y \cos \theta \\ \dot{\theta} &= w_z\end{aligned}\quad (1)$$

$$\dot{v}_x = (T - C_{xs} \frac{1}{2} \rho v^2 S - \Delta G \sin \theta) / m_{11}$$

$$\dot{v}_y = (\eta(J_{zz} + \lambda_{66}) - \mu m_{26}) / (m_{22}(J_{zz} + \lambda_{66}) - m_{26}^2)$$

$$\dot{w}_z = (\mu m_{26} - \mu(J_{zz} + \lambda_{66})) / (m_{26}^2 - m_{22}(J_{zz} + \lambda_{66}))$$

其中, $\eta = -m v_x \omega_z + \frac{1}{2} \rho v^2 S C_y - \Delta G \cos \theta$, $C_y = C_y^\alpha \alpha +$

$$C_y^{\delta_e} \delta_e + C_y^{\varpi_z} \varpi_z, v = \sqrt{v_x^2 + v_y^2}, \mu = (-m x_c v_x \omega_z +$$

$$G(y_c \sin \theta - x_c \cos \theta) + \frac{1}{2} \rho v^2 S L m_z), m_{11} = m + \lambda_{11},$$

$m_{22} = m + \lambda_{22}$, $m_{26} = mx_c + \lambda_{26}$, $m_z = m_y^\alpha \alpha + m_y^{\delta_e} \delta_e + m_y^{\varpi_z} \varpi_z$ 。 m 是 AUV 的质量; G 是 AUV 的重力; λ_{11} 、 λ_{22} 、 λ_{26} 、 λ_{66} 是与流体动力学有关的附加质量; x_c 、 y_c 是 AUV 重心到浮力中心的距离在 x 轴和 y 轴上的分量; T 是由螺旋桨提供的推力; C_{xs} 是 AUV 的阻力系数, 该数值与 AUV 的最大横截面积 S 、速度 v 和水的密度 ρ 有关; ΔG 是 AUV 的负浮力, 即重力与浮力之差; J_z 是 AUV 绕 z 轴旋转的转动惯量; v_x 与 v_y 分别是 AUV 在 x 轴与 y 轴上的速度分量, w_z 是 AUV 的俯仰角速度。此外, 攻角 $\alpha = -\arctan(v_y / v_x)$; 无量纲角速度 $\varpi_z = w_z L / v$, L 是 AUV 的长度。 C_y^α 与 $C_y^{\delta_e}$ 分别是 AUV 的升力因数对攻角 α 、水平舵角 δ_e 的位置导数; $C_y^{\varpi_z}$ 是升力因数对角速度 ϖ_z 的旋转导数。相应地, m_y^α 、 $m_y^{\delta_e}$ 与 $m_y^{\varpi_z}$ 是偏航力矩因数对 α 、 δ_e 以及 ϖ_z 的导数。

对于给定的控制输入 δ_e , 通过式 (1) 中的运动模型即可求得 AUV 的状态 $\mathbf{s} = [x, y, z, v_x, v_y, w_z]^\top$ 。

3 基础知识

3.1 强化学习

强化学习是一种机器学习方法, 主要解决序列决策的问题。强化学习的基本框架如图 2 所示, 当智能体给出一个具体动作 a_t 时, 环境更新智能体的状态 s_t , 同时给智能体返回奖励 r_t 。这里, 当 $t \rightarrow \infty$ 时, 定义智能体的累积奖励为:

$$G = \sum_{t=0}^{\infty} \gamma^t r_t \quad (2)$$

其中, $\gamma \in (0, 1)$ 为折扣因子, 保证 G 是收敛的。

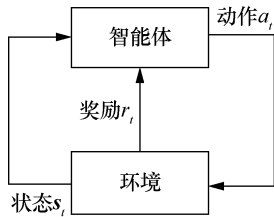


图2 强化学习的基本框架

强化学习理论推导是在马尔可夫决策模型下进行的, 如果一个系统满足马尔可夫性, 则系统将来时刻状态只与当前时刻状态有关, 与过去时刻状态无关。

$$p(\mathbf{s}_{t+1} | \mathbf{s}_t) = p(\mathbf{s}_{t+1} | \mathbf{s}_1, \mathbf{s}_2, \dots, \mathbf{s}_t) \quad (3)$$

马尔可夫决策过程是在马尔可夫过程中加入

了决策部分, 假设有智能体与环境从 1 到 T 时刻的交互序列:

$$\tau = \mathbf{s}_1, a_1, \mathbf{s}_2, a_2, \dots, \mathbf{s}_T, a_T \quad (4)$$

定义随机策略函数 $\pi_\theta(a_t | \mathbf{s}_t) = P[a_t | \mathbf{s}_t]$, 表示在状态 $\mathbf{s}_t$ 下智能体采用动作 a_t 的概率。设系统模型为 $p(\mathbf{s}_{t+1} | \mathbf{s}_t, a_t)$, 表示在当前状态下采用动作 a_t 到下一个状态的概率。其整个马尔可夫决策过程链为:

$$p_\theta(\tau) = p(\mathbf{s}_1) \prod_{t=1}^T \pi_\theta(a_t | \mathbf{s}_t) p(\mathbf{s}_{t+1} | \mathbf{s}_t, a_t) \quad (5)$$

在该交互序列下, 强化学习的目标是找到使回报最大的策略参数:

$$\theta^* = \arg \max_{\theta} E_{\tau \sim p_\theta(\tau)} \left[\sum_{t=1}^T \gamma^t r(\mathbf{s}_t, a_t) \right] \quad (6)$$

$$\text{其优化目标为 } J(\theta) = E_{\tau \sim p_\theta(\tau)} \left[\sum_{t=1}^T \gamma^t r(\mathbf{s}_t, a_t) \right]。$$

采用梯度下降法求最优参数:

$$\theta_{t+1} = \theta_t - \alpha \nabla_{\theta} J(\theta_t) \quad (7)$$

其中, α 为优化步长。

对优化目标求梯度:

$$\nabla_{\theta} J(\theta) = E_{\tau \sim p_\theta(\tau)} \left[\nabla_{\theta} \log p_\theta(\tau) \left(\sum_{t=1}^T \gamma^t r(\mathbf{s}_t, a_t) \right) \right] d\tau \quad (8)$$

将其化简为:

$$\nabla_{\theta} J(\theta) = E_{\tau \sim p_\theta(\tau)} \left[\left(\sum_{t=1}^T \nabla_{\theta} \log \pi_\theta(a_t | \mathbf{s}_t) \right) \left(\sum_{t=1}^T \gamma^t r(\mathbf{s}_t, a_t) \right) \right] \quad (9)$$

从式 (9) 可知, 对策略进行优化是对策略进行梯度求解。其中, 将式 (9) 后的半部分定义为强化学习状态值函数:

$$Q_\pi(\mathbf{s}_t, a_t) = E_{\tau \sim \pi_\theta(\tau)} \left[\sum_{t=1}^T \gamma^t r(\mathbf{s}_t, a_t) \right] \quad (10)$$

3.2 深度前馈神经网络

深度前馈神经网络是典型的深度学习模型。前馈神经网络定义了一个映射 $y = g(x; \zeta)$, 通过学习的方法选择最佳参数 ζ , 使得函数 $g(\cdot)$ 可以拟合函数 $y = f(x)$ 。

深度前馈神经网络的结构如图 3 所示。第一部分是神经网络的输入层, 中间的隐藏层通常有多层结构, 最后一部分是输出层, 用于输出函数值。 $\zeta_{ij}^{(1)}$ 与 $\zeta_{ij}^{(2)}$ 是神经网络的参数, 通过训练这些参数获得

函数的近似值。这个典型的神经网络是一个线性模型，需要对其进行去线性化处理，以便更好地近似非线性函数。常见的去线性化方法是在每个神经元的输出端添加非线性函数，使神经网络变为非线性的，添加的函数被称为激活函数。深度学习中常用的激活函数包括 ReLU 函数、sigmoid 函数和 tanh 函数。

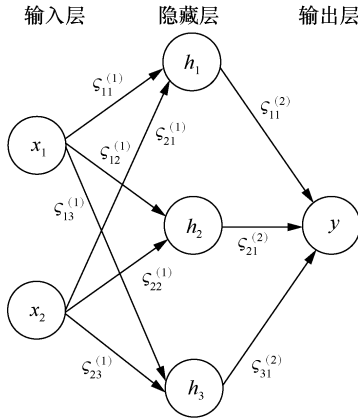


图3 深度前馈神经网络的结构

深度前馈神经网络得到的近似函数 $g(\cdot)$ 的精确度可用损失函数 $L(g(x;\varsigma), y)$ 表示，其中， $g(x;\varsigma)$ 为神经网络的估计值， y 为真实值。损失函数表示估计值和真实值之间的差异。深度学习中损失函数通常为交叉熵或其他特定函数。选定损失函数后，需要通过最小化损失函数来优化参数 ς ^[10]：

$$\varsigma^* = \arg \max_{\varsigma} L(g(x;\varsigma), y) \quad (11)$$

求解优化问题时常用的一种深度学习算法是 ADAM (adaptive moment estimation) 算法^[11]，如下所示。

算法1 ADAM 算法

- 步骤1：初始化步长参数 ε ；
- 步骤2：设定指数衰减速率 $\beta_1, \beta_2 \in [0, 1)$ ；
- 步骤3：初始化常数 $l=10^{-8}$ ；
- 步骤4：设定参数 $\varsigma, s=0$ 以及 $r=0, t=0$ ；
- 步骤5：计算梯度 $g = \nabla_{\varsigma} L(g(x;\varsigma), y)$ ；
- 步骤6： $t = t + 1$ ；
- 步骤7： $s = \beta_1 s + (1 - \beta_1) g$ ；
- 步骤8： $r = \beta_2 s + (1 - \beta_2) g \odot g$ ；
- 步骤9： $\hat{s} = s / (1 - \beta_1^t)$ ；
- 步骤10： $\hat{r} = r / (1 - \beta_2^t)$ ；
- 步骤11： $\varsigma = \varsigma - \varepsilon \hat{s} / (\sqrt{\hat{r}} + l)$ ；
- 步骤12：If $\|\varepsilon \hat{s} / (\sqrt{\hat{r}} + l)\| \leq 10^{-8}$

步骤13：输出 ς ；

步骤14：else

步骤15：返回步骤5；

步骤16：end

3.3 深度强化学习算法

在本文中，笔者利用 actor-critic 深度神经网络来实现强化学习算法^[2]，其中状态值函数 $Q_{\pi}(s_t, a_t)$ 与策略函数 $\pi(a_t | s_t)$ 分别用 critic 和 actor 神经网络进行估计。通过 critic 神经网络估计状态值函数 $Q_{\pi}(s_t, a_t)$ ，可以得到：

$$Q_{\pi}(s_t, a_t) \approx Q(s_t, a_t; \omega) \quad (12)$$

其中， ω 为 critic 神经网络参数。通过式 (12) 可以得到：

$$Q_{\pi}(s_t, a_t) = r_t + \gamma Q_{\pi}(s_{t+1}, a_{t+1}) \quad (13)$$

为了求得状态值函数 $Q_{\pi}(s_t, a_t)$ 在 t 时刻的最优估计，设计损失函数^[7]：

$$L = (Q_q - Q(s_t, a_t; \omega))^2 \quad (14)$$

根据式 (11)，将 Q_q 定义为：

$$Q_q = r_t + \gamma Q'(s_{t+1}, \pi'(s_{t+1}; \theta'); \omega') \quad (15)$$

其中， Q' 和 π' 为目标神经网络，其作用为稳定学习过程。目标神经网络参数 θ' 与 ω' 更新方式如下^[12]：

$$\begin{aligned} \theta' &\leftarrow \tau \theta + (1 - \tau) \theta' \\ \omega' &\leftarrow \tau \omega + (1 - \tau) \omega' \end{aligned} \quad (16)$$

其中， τ 的取值远小于 1，以保证目标神经网络的稳定性。

利用一组 N 个数据对 critic 神经网络进行训练，求得其平均损失函数为：

$$\bar{L} = \frac{1}{N} \sum_{t=1}^N (Q_q - Q(s_t, a_t; \omega))^2 \quad (17)$$

对其求梯度为：

$$\nabla_{\omega} \bar{L} = \nabla_{\omega} \left(\frac{1}{N} \sum_{t=1}^N (Q_q - Q(s_t, a_t; \omega))^2 \right) \quad (18)$$

通过 ADAM 算法对式 (18) 的梯度进行计算，利用学习到的参数 ω^* 对状态值函数 $Q_{\pi}(s_t, a_t)$ 进行估计。基于估计的状态值函数 $Q(s_t, a_t; \omega^*)$ 对策略进行评估。

与随机策略梯度不同，在确定性策略梯度中，

动作 a_t 在状态 s_t 确定时就确定了。利用 actor 神经网络 $\pi_\theta(s_t)$ 对确定性策略 $\pi(s_t)$ 进行估计^[13]，将最优策略求解转化为对 actor 神经网络参数 θ 的优化，以达到奖励函数最大化的目的，即

$$\theta^* = \arg \min_{\theta} E_{\tau \sim p_{\theta}(\tau)} \left[\sum_{t=1}^T \gamma^t r(s_t, a_t) \right] \quad (19)$$

其中：

$$p_{\theta}(\tau) = p(s_1) \prod_{t=1}^T \pi_{\theta}(s_t) p(s_{t+1} | s_t, a_t) \quad (20)$$

根据式 (19) 和式 (20)，可以得到优化目标：

$$J(\theta) = E_{\tau \sim \rho(\tau)} [Q(s_t, a_t) |_{a_t = \pi_{\theta}(s_t)}] \quad (21)$$

其中， $\rho(\tau)$ 为系统状态分布，计算式如下：

$$\rho(\tau) = p(s_1) \prod_{t=1}^T p(s_{t+1} | s_t, a_t) \quad (22)$$

对优化目标求梯度：

$$\nabla_{\theta} J(\theta) = E_{\tau \sim \rho(\tau)} [\nabla_a Q(s_t, a_t) |_{a_t = \pi_{\theta}(s_t)} \bullet \nabla_{\theta} \pi_{\theta}(s_t)] \quad (23)$$

该算法为离线迭代算法，通过 AUV 运动数据对其进行训练。利用一组 N 个数据对 actor 神经网络进行训练，利用蒙特卡洛方法对期望进行估计可以得到梯度：

$$\nabla_{\theta} J(\theta) = \frac{1}{N} \sum_{t=1}^N [\nabla_a Q(s_t, a_t) |_{a_t = \pi_{\theta}(s_t)} \bullet \nabla_{\theta} \pi_{\theta}(s_t)] \quad (24)$$

最后，利用 ADAM 算法对梯度进行计算，得到最优参数 θ^* 。

当对 critic 和 actor 神经网络进行训练时，一般假设训练数据是独立不相关的。但是机器人探索得到的数据是有相关性的，这会导致神经网络的训练效果不佳。可以通过建立一个回放记忆单元解决这一问题，将机器人探索得到的数据存入回放记忆单元中，在训练时随机从回放记忆单元中提取一组数据，以打破数据的相关性^[12]。DDPG 算法如下^[7]。

算法 2 DDPG 算法

步骤 1：初始化 critic 神经网络 $Q(s_t, a_t; \omega)$ 与 actor 神经网络 $\pi_{\theta}(s_t)$ ；

步骤 2：初始化目标神经网络参数 θ' 与 ω' ，令 $t=0$ ， $i=0$ ；

步骤 3：初始化回放记忆单元；

步骤 4：for $i \leq 2000$

步骤 5：初始化状态 s_0 ；

步骤 6：for $t \leq 1000$

步骤 7： $t = t + 1$ ；

步骤 8： $Q_t = r_t + \gamma Q'(s_{t+1}, \pi'(s_{t+1}; \theta'); \omega')$ ；

步骤 9： $L = (Q_t - Q(s_t, a_t; \omega))^2$ ；

步骤 10：计算平均损失函数 $\bar{L}$ 及其梯度 $\nabla_{\omega} \bar{L}$ ；

步骤 11：用 ADAM 算法更新 ω ；

步骤 12： $J(\theta) = E_{\tau \sim \rho(\tau)} [Q(s_t, a_t; \omega) |_{a_t = \pi_{\theta}(s_t)}]$ ；

步骤 13：对其求梯度 $\nabla_{\theta} J(\theta)$ ；

步骤 14：用 ADAM 算法更新 θ ；

步骤 15：更新目标神经网络；

步骤 16： $\theta' \leftarrow \tau \theta + (1 - \tau) \theta'$ ；

步骤 17： $\omega' \leftarrow \tau \omega + (1 - \tau) \omega'$ ；

步骤 18：end

步骤 19： $i = i + 1$ ；

步骤 20：end

4 基于 DRL 算法的 AUV 定深控制

利用 DRL 算法对 AUV 进行定深控制需要 AUV 对环境进行探索学习。考虑到在真实环境中 AUV 获取大量数据的费用是昂贵的，因此利用式 (1) 中的 AUV 模型来获取训练数据。

定义 $s_t = [x(t), y(t), z(t), v_x(t), v_y(t), w_z(t)]^T$ 为 AUV 在时刻 t 的状态； $\delta_e(t) = K(s_t)$ 为控制输入。对于 RL 算法来说，上述控制输入为当前时刻 AUV 的动作，即 $a_t = \delta_e(t)$ 。

定深控制的定义是通过控制输入将 AUV 保持在预定深度 y_d 运动。为了解决强化学习中的稀疏奖励和奖励延迟问题，根据 AUV 定深控制的运动特性设计了一个分段式的奖励函数。

在第一个阶段，设计了如下奖励函数：

$$R(s_t) = -c_y \|y(t) - y_d\| - c_{\theta} \|\theta(t)\| - c_{\omega} \|\omega_z(t)\| \quad (25)$$

其中， $c_y > 0$ 、 $c_{\theta} > 0$ 、 $c_{\omega} > 0$ ，均为常数。其目的是鼓励 AUV 下潜到预定深度，并防止其俯仰角过大。

当 AUV 到达预定深度 y_d 的一定范围内时，将奖励函数切换为：

$$R(s_t) = c_r \quad (26)$$

其中， c_r 是一个常数。其目的是鼓励 AUV 保持在该深度运行。综合上述分析，奖励函数为：

$$R(s_t) = \begin{cases} -c_y \|y(t) - y_d\| - c_\theta \|\theta(t)\| - c_\omega \|\omega_z(t)\| \\ (y(t) - y_d > 0.1) \\ c_r \\ (y(t) - y_d \leq 0.1) \end{cases} \quad (27)$$

将设计的奖励函数代入算法 2 中, 利用 critic 神经网络估计状态值函数:

$$Q_\pi(s_t, a_t) \approx Q(s_t, a_t; \omega) \quad (28)$$

通过训练 actor 神经网络得到最优策略 $\pi_\theta^*(s_t)$ 。此外, 为了让 AUV 在训练过程中更好地探索环境, 在动作中加入了随机噪声, 即:

$$a'_t = \pi_\theta(s_t) + v \quad (29)$$

其中, v 是 Ornstein-Uhlenbeck 随机噪声^[14]。

为了训练神经网络, 将下一时刻系统状态 s_{t+1} 、当前时刻状态 s_t 、动作 a_t 和奖励 r_t 组成一个元组 $\langle s_{t+1}, s_t, a_t, r_t \rangle$, 将其按时序存入状态缓冲区, 并用于训练神经网络。

基于深度强化学习的 AUV 定深控制系统如图 4 所示。其中, 用于估计状态值函数的 critic 神经网络的输入为 $\langle s_t, a_t, r_t \rangle$, 其输出为近似的状态值函数 $Q(s_t, a_t; \omega)$, 用于求解 actor 神经网络的损失函数。而训练好的 actor 神经网络的输入为 AUV 当前时刻的状态 s_t , 输出为定深控制动作。

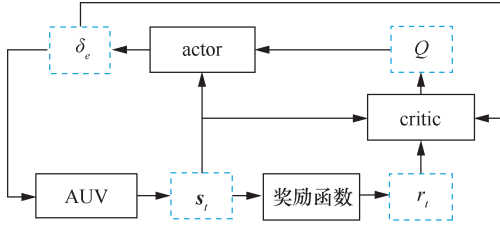


图4 基于深度强化学习的 AUV 定深控制系统

5 仿真实验

在本节中, 笔者用仿真实验验证算法的有效性。仿真平台为 OpenAI Gym 平台, 利用第 2 节的 AUV 运动模型建立仿真模型, 如图 5 所示。

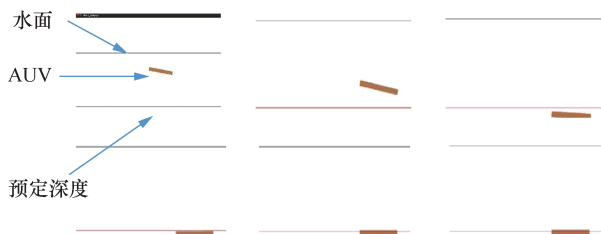


图5 OpenAI Gym 仿真平台中的 AUV 深度控制模型

初始化奖励函数参数 $c_y=1$ 、 $c_\theta=4$ 、 $c_\omega=2$ 、 $c_r=5$ 。通过 TensorFlow 搭建深度神经网络^[15], 其中 critic 神经网络由 5 个全连接层组成, 隐藏层每层节点数为 32 个, 除输出层外, 在其他 4 层加入 ReLU 激活函数^[16]。actor 神经网络由 4 个全连接层组成, 隐藏层每层节点数为 16 个, 除了输出层外, 在其他 3 层加入 ReLU 激活函数, 输出层使用 tanh 激活函数对输出进行 $[-1,1]$ 的归一化。

在训练初期, AUV 先进行自由探索, 将数据存入回放记忆单元中。当回放记忆单元存满后, 通过随机采样的方式抽取一组 64 个数据对神经网络进行训练。critic 神经网络训练的学习率为 0.001, actor 神经网络的学习率为 0.000 1。在对两个神经网络进行更新后, 通过式 (16) 对目标神经网络进行更新, 其中更新参数 $\tau=0.001$ 。最后, 利用训练后的 actor 神经网络与环境进行交互, 将得到的数据用于更新数据集。在整个学习过程中重复上述步骤, 直到奖励函数收敛。平均奖励值随训练幕数的变化如图 6 所示, 由图 6 可知, 在经过 1 000 幕的训练后, 每一幕的平均奖励函数得到了收敛。AUV 定深运动的轨迹如图 7 所示。

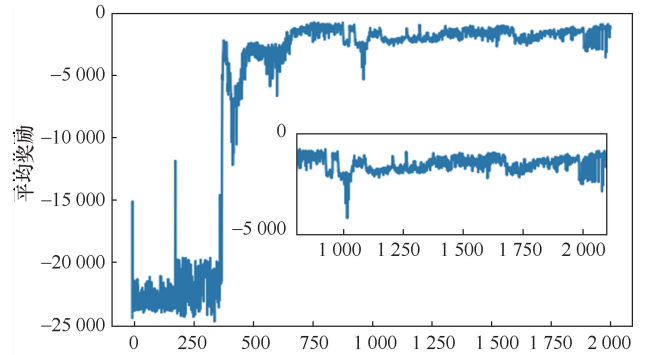


图6 平均奖励值随训练幕数的变化

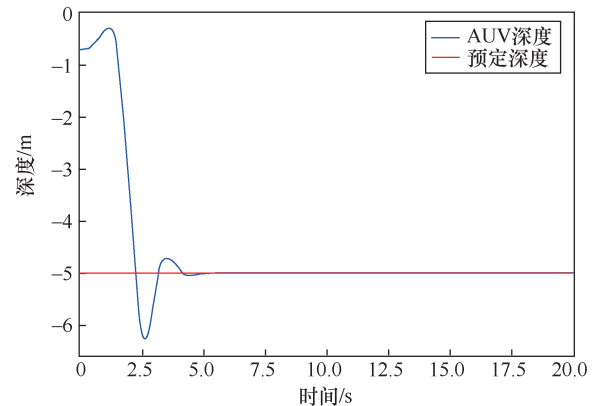


图7 AUV 定深运动的轨迹

假设 AUV 的预定深度 $y_d = -5$ m, 图 6 和图 7 显示, 经过 2 000 幕的训练后, 由 actor 神经网络控制 AUV 定深运动的轨迹。这表示经过训练的 actor 神经网络可以控制 AUV 高精度下潜到预定深度, 并保持在预定深度运动。如图 8 所示, 与另一种不需要模型信息的 PID 控制算法相比, DDPG 算法有更高的控制精度。

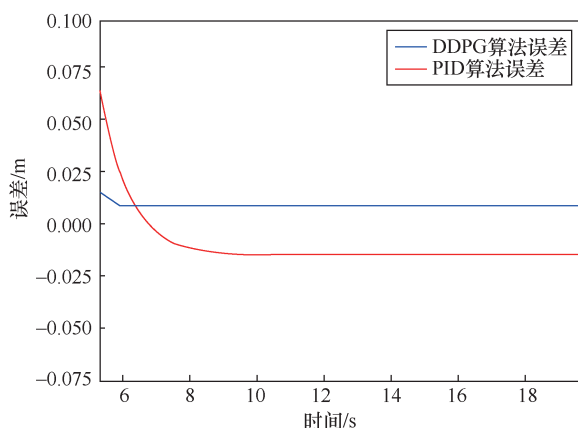


图8 AUV 定深控制效果比较

6 结束语

本文将 DRL 算法应用于 AUV 的定深控制, 不依赖于 AUV 的系统建模和控制律设计。通过设计合理的奖励函数引导 AUV 探索学习定深控制律。仿真实验证明了该方法的可行性, 比起传统无模型 PID 控制算法, 其具有更高的精度。在未来的研究中可以着重研究提高算法学习速率和学习效率的方法和向实际系统迁移的可能性。

参考文献:

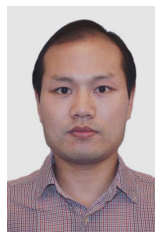
- [1] ZHANG L J, QI X, PANG Y J. Adaptive output feedback control based on DRFNN for AUV[J]. Ocean Engineering, 2009, 36(9-10): 716-722.
- [2] SUTTON R, BARTO A. Reinforcement learning: an introduction[M]. Cambridge: MIT Press, 1998.
- [3] TESAURRO G. TD-Gammon, a self-teaching backgammon program, achieves master-level play[J]. Neural Computation, 1994, 6(2): 215-219.
- [4] MNH V, KAVUKCUOGLU K, SILVER D, et al. Playing atari with deep reinforcement learning[J]. Computer Science, 2013.
- [5] 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): 484-489.
- [6] SILVER D, TECHNOLOGIES D, LEVER G, et al. Deterministic policy gradient algorithms[C]//International Conference on Machine Learning. New York: ACM Press, 2014.
- [7] LILLICRAP T P, HUNT J J, PRITZEL A, et al. Continuous control with deep reinforcement learning[J]. Computer Science, 2015, 6(6): A187.

- [8] HWANGBO J, LEE J, DOSOVITSKIY A, et al. Learning agile and dynamic motor skills for legged robots[J]. Science Robotics, 2019, 4(26).
- [9] 严卫生. 鱼雷航行力学[M]. 西安: 西北工业大学出版社, 2005.
- [10] GOODFELLOW I, BENGIO Y, COURVILLE A. Deep learning[M]. Cambridge: MIT Press, 2016.
- [11] KINGMA D, BA J. ADAM: a method for stochastic optimization[J]. Computer Science, 2014.
- [12] MNH V, KAVUKCUOGLU K, SILVER D, et al. Human-level control through deep reinforcement learning[J]. Nature, 2015, 518-529.
- [13] KONDA V, TSITSIKLIS J. Actor-critic algorithms[J]. In Advances in Neural Information Processing Systems, 2003: 1008-1014.
- [14] UHLENBECK G E, ORNSTEIN L S. On the theory of the Brownian motion[J]. Revista Latinoamericana De Microbiología, 1973, 15(1): 29.
- [15] ABADI M, BARHAM P, CHEN P, et al. TensorFlow: a system for large-scale machine learning[J]. Google Brain, 2016.
- [16] NAIR V, HINTON G. Rectified linear units improve restricted Boltzmann machines[C]//International Conference on Machine Learning. [S.l.:s.n.], 2010.

[作者简介]



王日中 (1995—), 男, 西北工业大学航海学院博士生, 主要研究方向为多机器人协同、强化学习。



李慧平 (1983—), 男, 博士, 西北工业大学航海学院教授, 博士生导师, 主要研究方向为水下机器人导航定位、智能决策与优化控制、模型预测控制、多机器人协同。



崔迪 (1995—), 女, 西北工业大学航海学院博士生, 主要研究方向为事件触发控制、模型预测控制。



徐德民 (1937—), 男, 中国工程院院士, 西北工业大学航海学院教授, 博士生导师, 主要研究方向为水下航行器总体设计、导航制导与控制。