结合 LSTM 的强化学习动态环境路径规划算法

武 曲,张 义,郭 坤,王 玺

(青岛理工大学 信息与控制工程学院,山东 青岛 266520)
E-mail:zhangyi_626@126.com

摘 要: 在路径规划领域已经涌现出了诸多的优秀的经典算法,但这些传统方法往往基于静态环境,对于动态可变环境缺乏处理能力.本文提出一种结合 LSTM 强化学习动态环境路径规划算法.首先,本文以环境图像作为输入,最大限度了保证了原始的信息来源.而后构建了自动编码器用来对环境图像进行特征降维,降低了整体模型的复杂程度.最后采用深度强化学习算法 DDPG 进行路径规划,其中 Actor 部分采用 LSTM 的网络构建,使 Actor 在决策时可以参考前序信息,做到有预测的避开动态障碍.最后通过实验证明了本文算法的可行性和高效性.

关 键 词: 自动编码器;LSTM;DDPG;强化学习;动态路径规划
中图分类号: TP391 **文献标识码:** A **文 章 编 号:** 1000-1220(2021)02-0334-06

LSTM Combined with Reinforcement Learning Dynamic Environment Path Planning Algorithm

WU Qu, ZHANG Yi, GUO Kun, WANG Xi
(School of Information and Control Engineering, Qingdao University of Technology, Qingdao 266520, China)

Abstract: Many excellent classical algorithms have emerged in the field of path planning, but these traditional methods are often based on static environment and lack processing power for dynamic variable environment. This paper proposes a path planning algorithm for dynamic environment based on LSTM reinforcement learning. First of all, this paper takes the environment image as the input to ensure the original information source to the maximum extent. Then an Autoencoder is built to reduce the dimension of environment image, which reduces the complexity of the whole model. At last, the deep reinforcement learning algorithm DDPG is used for path planning, and the Actor part uses LSTM network, so that the Actor can refer to the prior information and make decisions with the prediction of environment change. Finally, the feasibility and efficiency of the proposed algorithm are proved by experiments.

Key words: autoencoder; LSTM; DDPG; reinforcement learning; dynamic path planning

1 引 言

路径规划是人工智能领域的一个重要研究方向,在各个领域得到了广泛的应用.迄今已经有许多经典的路径规划算法被提出.

Dijkstra 算法是一种很早就被提出的路径规划算法[1],它将环境抽象为一个图问题,利用广度优先搜索策略遍历图,直到找到最短路径.A * 算法是 Dijkstra 算法[2]的改进.在原有算法的基础上增加了启发式函数,并定义了一种当区域与扩展点之间的一种度量作为扩展优先级,在进行路径扩展时会优先扩展优先级高的节点.但当该方法用于处理多维复杂问题时,无论是把环境抽象为图模型还是对图模型求解都将变得很复杂.势场法[3]把规划空间看作物理学中的场,把智能体看作一种粒子.障碍物对粒子产生排斥力,目标对粒子产生引力.两者的合力即为智能体的最终运动的方向.这种方法实时性较好,产生的路径通常十分平滑,适合于机械臂一类的应用,缺点是在合力为 0 的位置智能体容易陷入局部最优解.

近年来,随机人工智能的兴起,很多基于人工智能的路径规划方法被提出,Chen 等[4]提出了一种双向神经网络来解决未知环境下的路径规划问题.Wu 等[5]将路径规划任务转化为环境分类任务,使用 CNN 来进行路径规划.Yu 等[6]提出了一种基于神经网络的鲁棒控制方案,并结合自适应补偿器和自适应控制增益来实现具有避障能力的编队控制.

强化学习是一类应用在未知环境的算法,作为机器学习的 3 大分支之一,不同于监督学习和无监督学习,强化学习无需提供数据,所有的学习资料都将从环境中获取.智能体通过不断的探索环境,根据不同的动作产生的不同的反馈进行模型的学习,最终智能体将能以最优策略在指定环境中完成任务.

自 V. Mnih 等提出 DQN[7]以来,深度强化学习不断取得突破性进展,也有一些研究者尝试通过深度强化学习解决路径规划问题.Piotr Mirowski 等[8]以多模态感知信息作为输入,通过强化学习进行决策来完成网格空间中的导航任务.Panov 等[9]使用神经 Q-Learning 算法来完成网格环境下的路

收稿日期:2020-08-03 收修改稿日期:2020-09-25 基金项目:山东省自然科学基金项目(ZR2017BF043)资助.作者简介:武 曲,女,1982 年生,博士,副教授,CCF 会员,研究方向为强化学习、深度强化学习;张 义,男,1995 年生,硕士研究生,研究方向为深度强化学习;郭 坤,女,1995 年生,硕士研究生,研究方向为深度强化学习;王 玺,女,1996 年生,硕士研究生,研究方向为深度强化学习.

径规划任务. Lei 等[10]采用 CNN 和 DDQN 进行动态环境下的路径规划. Lv 等[11]提出了一种改进的基于 DQN 的学习策略,在学习的初始阶段,创建一个体验价值评价网络,当发生路径漫游现象时,利用并行探索结构考虑对漫游点之外的其他点的探索,提高体验池的广度.

尽管上述方法在各自的领域都取得了不错的效果,但是他们实现路径规划仍存在一些不足之处. 他们大多数只是在静态环境中进行路径规划,缺乏处理动态场景的能力;动作空间或状态空间是离散的,这与连续的现实环境是不符合的,而且在某些情况下,离散动作得出的最优解还可以被连续动作进一步优化;上述方法实现的路径规划多是从固定起点到固定终点的路径规划,这相当于模型只学习到了一个局部最优解,并不能完成整个环境的路径规划,这对指导现实应用具有很大的局限性.

为了实现全局动态环境下的路径规划任务,本文提出了一种结合了 LSTM 的路径规划算法. 本文算法以环境图像作为输入,通过预训练的自动编码器进行降维提取特征. 在训练模型时,以连续 4 帧图片降维后的特征信息作为输入,通过 LSTM 构建的 DDPG 模型进行路径规划,利用 LSTM 处理时序数据的特性,实现了在动作选择时进行有预测的规避环境中的危险区域的动态路径规划.

2　相关工作

2.1　自动编码器

自动编码器(Autoencoder)可以看做是利用深度学习的对数据进行降维的一种方式,通过一系列的神经网络计算将高维数据压缩到低维,再以对称的方式将数据复原,其结构图如图 1 所示.

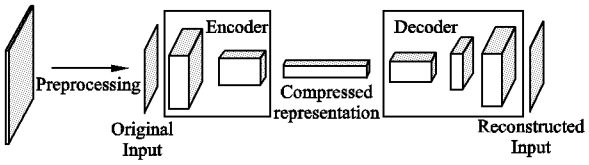


图 1　自动编码器

Fig. 1　Autoencoder

当编码器模型收敛后即可认为中间的低维数据为压缩后的降维数据,再对低维数据进行其他操作,即可在保证达到与原数据相同效果的同时,亦能极大地降低操作过程的复杂度.

2.2　LSTM

循环神经网络(Recurrent Neural Network,RNN)是一种处理时序数据的神经网络,RNN 以一条时序数据为输入,其结构单元如图 2(a)所示,在一个计算单元的计算中,输入部分除当前时刻数据 x_t 之外,还有一项 h_{t-1},该数据是由之前的 t-1 个时刻的数据传导计算而得,同样地,RNN 在 t 时刻的输出,除了 y_t 之外,还会生成一项 h_t,而 h_t 则是包含了前 t 个时刻的信息,h_t 将被传送到 t+1 时刻参与到 t+1 时刻的输出的计算过程中.

RNN 的这种结构设计,使得 RNN 网络具有了预测的能力. 但是,在经典的 RNN 网络中,隐藏单元 h_t 所携带的信息是所有前 t 个时刻的信息,这样的结构产生了两个问题:有些前序时刻信息对当前时刻的输出而言并没有价值,参与到当前时刻输出的计算过程中反而会造成误差;大量的前序信息参与当前时刻输出的处理过程将增加计算的负担,该问题在序列较长时将会变得尤为突出.

长短期记忆(Long Short-Term Memory,LSTM)是一种改进的 RNN,该网络结构在产生当前时刻的输出时又增加了一项遗忘门的设计,通过一个状态参量 c 来实现遗忘功能,LSTM 的结构但愿如图 2(b)所示.

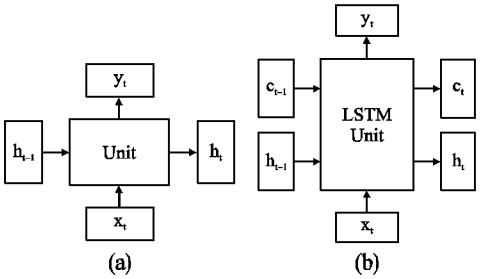


图 2　RNN 结构单元

Fig. 2　RNN unit

在 LSTM 的当前时刻,计算单元首先通过 x_t 和 h_{t-1} 计算出一个中间结果,而后通过状态参量 c_{t-1} 参与构建的遗忘门进行选择,最终输出 y_t、h_t 以及 c_t. LSTM 的设计方式以一种更有效的方式利用了前序信息,同时也减少了中间过程携带的数据量,相对于经典 RNN 具有更好的效果.

2.3　DDPG

2.3.1　马尔科夫决策

强化学习过程普遍遵循马尔科夫决策过程(Markov Decision Process,MDP). MDP 由一个 <S,A,P,R> 的四元组组成,其中 S(State)为状态空间,表示智能体在环境中可能存在的状态描述的集合. A(Action)为动作空间,表示智能体在环境中可能采取的动作描述的集合. P(Policy)为转移策略,处在某个状态的智能体将依 P 进行动作选择,进而从一个状态转移到另一个状态. R(Reward)为回报,表示智能体在某个状态下采取某个动作而从环境中获得的回报值. 强化学习的目标即为一个求取最佳策略 P,在环境中进行执行一系列的动作,使智能体以最佳的回合回报完成给定任务.

2.3.2　Actor-Critic

Actor-Critic[12]是 Vijay R. Konda 和 John N. Tsitsiklis 提出的一种应用在马尔科夫决策过程中的算法,该算法由两部分构成,用来生成决策动作了 Actor 部分和用来对动作进行评价的 Critic 部分,Actor 是动作生成器,以当前状态作为输入,输出一个当前状态下的要执行的动作. Critic 则是一个评价器,即值函数生成器,以当前状态和 Actor 生成的动作为输入,生成一个价值量,该量用以衡量 Actor 生成的动作的优劣.

在训练过程中,模型按式(1)所示对探索过程中产生的数据进行处理.

$$Q_\tau^\theta(x,u)=\sum_{j=1}^{m} r^j \phi_\theta^j(x,u) \tag{1}$$

使 Critic 模型学会为 Actor 生成的动作进行评估,Actor 则向着 Critic 评价高的方向学习.

2.3.3 Policy Grident

策略梯度(Policy Grident,PG)是由 Richard S. Sutton 等人提出的一种独立与价值函数的、根据期望回报进行策略更新的强化学习方式[13],PG 采用回合更新的方式,在得到一条完成回合序列之后,对于序列中的状态的值函数定义如式(3)所示.

$$V_{\kappa}^{\pi}(s_t)=\sum_{t=1}^{\infty}\gamma^{t-1}r_t \tag{2}$$

多个回合后,s_t 的值应表示为多个回合的期望值,其定义如式(3)所示.

$$V^{\pi}(s_t)=E\{\sum_{t=1}^{\infty}\gamma^{t-1}r_t \mid s_0,\pi\} \tag{3}$$

在 PG 方法中,策略 π 按式(4)所示进行参数更新.

$$\theta\leftarrow\theta+\alpha\nabla_{\theta}\log\pi_{\theta}(s_t,a_t)V_t \tag{4}$$

2.3.4 DQN

深度 Q 网络(Deep Q Network,DQN)是深度强化学习的一个重要算法,它通过神经网络来构造状态价值函数,直接生成 Q 值,解决了传统 Q-Learning 方法维度受限、无法处理未参与训练的状态数据的问题;通过 off policy 的策略解决了强化学习数据的强相关性导致的很难应用深度学习方法处理的问题. DQN 由两个结构相同,时间差分的网络构成,通过式(5)所示的算法进行网络参数的更新,由 DQN 开始,围绕深度强化学习不断涌现出许多优秀的研究成果.

$$\begin{cases}y=r+\gamma\max_{a},\bar{Q}^{*}(s',a'),\\ L(\theta)=E_{s,a,r,s},[(Q^{*}(s,a|\theta)-y)^{2}]\end{cases} \tag{5}$$

2.3.5 DDPG

DDPG[14](Deep Deterministic Policy Gradient)算法结合了 AC、PG、DQN 中的诸多特点,率先将深度强化学习扩展到连续空间领域. DDPG 整体采用 Actor-Critic 的框架结构,DDPG 中的 Actor 和 Critic 两部分都由神经网络来构建,两部分的网络各自采用 DQN 的设计思路,分别为是两个时间差分的网络. 在 Critic 更新时,采用策略梯度的更新方式、与传统的策略梯度不同的是,DDPG 采用一种确定性策略进行动作选择.

3 结合 LSTM 的强化学习动态环境路径规划算法

在很多路径规划研究中,通常为智能体设置扫描射线,以此来观察周围的环境,智能体需要对当前周围的不同类型的实体进行扫描,然后构建包含到这些物体距离的向量,提供给模型进行动作选择. 使用扫描射线的方式虽然可以尽可能的使得智能体获取周围的信息,但是仍然不可避免地会信息遗漏,针对这种情况,本文采用图像为模型提供输入. 图像虽然极大地保留了环境的真实数据,但是同样存在着维度过大,模型难收敛的问题. 自动编码器是一种采用深度学习对数据进行降维的方式,本文在处理图像数据时,首先采用预训练的编码器对图像数据进行了降维.

3.1 预训练图像编码器

为了降低高维图像对模型收敛增加的复杂度问题,本文设计了图像编码器对图像数据进行特征降维,本文构建的图像编码器结构如图 3 所示.

编码器首先对图片进行预处理,包括通过常规方法降低图片尺寸和灰度化,然后对得到的灰色图片进行编码和解码过程,通过解码后的图像与编码器的图像的差值作为损失来拟合编码器的参数. 表 1 所示为本文设计的编码器参数表,本文编码器由 5 层组成,前 2 层为编码部分,后 3 层为解码部分.

表 1 编码器参数

Table 1 Parameters of autoencoder

Layer	Neure	Parameters	Data size (after process)
0	/	/	1 * 200 * 100
1	Conv	c = 16; k = 5; s = 5; p = 0;	16 * 40 * 20
	Relu	/	16 * 40 * 20
	Pool	k = 2; s = 2; p = 0	16 * 20 * 10
2	Conv	c = 16; k = 3; s = 5; p = 0;	8 * 20 * 10
	Relu	/	8 * 20 * 10
	Pool	k = 2; s = 2; p = 0	8 * 10 * 5
3	ConvTrans	c = 16; k = 2; s = 2; p = 0;	16 * 20 * 10
	Relu	/	16 * 20 * 10
4	ConvTrans	c = 8; k = 2; s = 2; p = 0;	8 * 40 * 20
	Relu	/	8 * 40 * 20
1	ConvTrans	c = 1; k = 5; s = 5; p = 0;	1 * 200 * 100
	Tanh	/	1 * 200 * 100

* c 代表深度、k 代表卷积核尺寸、s 代表步长、p 代表填充

3.2 结合 LSTM 的 DDPG

本文算法的主要目标是更好的避开动态危险区域,根据到动态危险区域的距离来进行规避诚然是一种可行方式,但是这种被动的响应方式对整体的路径规划是不利的,它仍然避免不了智能体需要探索对应区域才能进行规避,这造成规划路线上增加了一些额外的长度. 如果模型能预测环境的变化趋势,就可以避开某些未来不能通过的区域,避免一些没有结果的探索工作,直接规划出一条最佳的可行路径. 本文利用

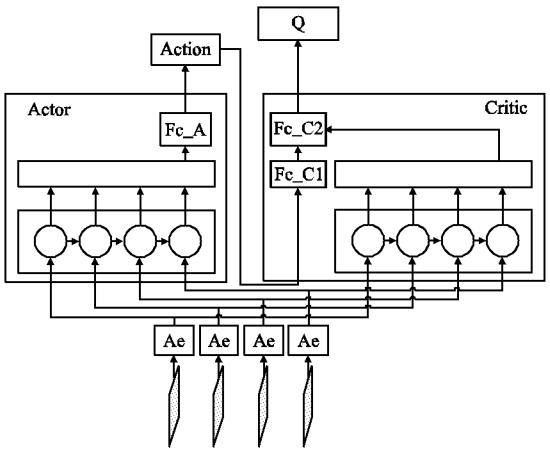


图 3 LSTM DDPG 结构图

Fig. 3 Structure of LSTM DDPG

了循环神经网络实现了这一设想,循环神经网络是一种用来处理时序数据的神经网络,会结合前序信息来生成当前时刻的输出,当前时刻的输出参考了之前时刻信息的变化趋势,所以循环神经网络是一种具有预测功能的网络. LSTM 的 RNN

的一种改进,解决了经典 RNN 无差别携带前序信息带来的弊端. DDPG 是一个在连续动作上有很好表现的强化学习算法,本文将 LSTM 融合到 DDPG 的框架中,构建了如图 3 所示的 LSTM-DDPG 算法.

其中 Actor 网络由 3 层构成,分别是两层 LSTM 和 1 层全连接层,对于 LSTM 设置 input size 为 400,隐藏层单元为 64,之后接一个全连接层以 64 维数据为输入计算生成 2 维的动作输出. Critic 网络首先对输入的环境数据和动作数据做分别处理,其中环境数据利用 LSTM 进行处理,网络设置与 Actor 中的 LSTM 部分设置相同;对于 Actor 产生的 Action,用一个全连接层将 2 维输入映射到 20 维;然后将上面两步的输出拼接一个向量传递给下一层的全连接层,由这个全连接层计算生成对 Action 的评价 Q 值.

3.3　动作空间

本文模拟人类的动作行为方式设计了智能体的动作空间,采用连续的动作空间设计,将动作空间设计为两个维度 (δ,l),其中 δ 表示智能体的转动角度,取值范围设定为 $(-180,180)$,其中当 $\delta<0$ 时,智能体向左转动相应角度,当 $\delta>0$ 时,智能体向右转动相应角度. l 表示智能体执行动作的位移大小,取值范围为 $(-0.7,0.7)$,其中 $l<0$ 当时,表示智能体后退相应距离,$l>0$ 时,表示智能体前进相应距离.

3.4　环境回报

在强化学习中,智能体通过在环境获得的累计回报来修正策略函数的参数,因此,环境回报的设定对策略函数能否收敛到理想的状态而言至关重要. 为了验证本文方法处理动态环境的能力,本文除了设计墙体这种单纯的静态障碍之外,还设计一种危险区域,智能体接触该区域即死亡,回合结束,视为一次失败的路径规划. 结合现实经验和多次试验结果作为参考,本文进行了以下环境回报的设定.

3.4.1　决策回报

在一条路径生成的过程中,智能体通过一系列的动作选择在不同状态间切换,为了能保证智能体能以最少的状态切换次数即为了使智能体尽可能规划出一条更短的路径,智能体每执行一步动作,为智能体设置 -1 的回报,即 $r_{step}=1$.

3.4.2　碰壁回报

本文在环境中设置了墙体,用来圈围边界和构建智能体前进的障碍. 对于智能体而言“撞墙”的行为是无意义的,不但增加了动作执行次数,也不会增加位移,因此对于智能体撞墙这种行为应该给予一定的负回报,在本文中设置 $r_{wall}=-1$.

3.4.3　遇险回报

本文设置了动态变化的危险区域来对提出的算法进行验证,该区域设置在智能体和目标位置之间,其体积会随着时间动态变化,对于智能体而言该区域的效果为在智能体接触到该区域时,智能体即死亡,回合结束,路径规划任务失败,因此应该对涉足该区域的智能体以最低的回报来使智能体远离该区域,在本文中设置 $r_{danger}=-50$.

3.4.4　目标回报

目标区域是路径规划任务的最终目标,应该给予其全局最大的回报,引导智能体向着最终目标进行路径规划. 在本文中设置 $r_{target}=200$.

综上,设置环境回报如式(6)所示.

$$Reward=\begin{cases} r_{target} & \text{if } s\in S_{target} \\ r_{danger} & \text{if } s\in S_{danger} \\ r_{step}+r_{wall} & \text{if } s'\in S_{wall} \\ r_{step} & \text{else} \end{cases} \tag{6}$$

4　实验及结果分析

本文通过 Unity-3D 引擎构进行了强化学习环境的搭建,实验所用的软硬件配置如下:CPU i7-8750H,内存 24G,显卡 GTX1060,显存 6G,软件环境 Unity2019.4.2f1,深度学习框架使用 Pytorch.

4.1　实验环境搭建

在 Unity 工具中构建如图 4 所示的环境.

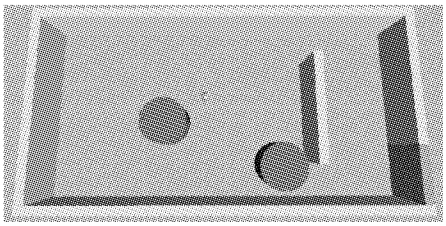


图 4　实验环境

Fig. 4　Experimental environment

该环境由面积为 40×20 矩形区域构成,在地面平面建立坐标系,以矩形区域中点为坐标原点,分别以向右和向上为 x 轴,y 轴的正方向. 其中中部较小的圆形个体为智能体,半径为 0.5,在每个回合训练开始时智能体将会随机生成在环境中的任意位置. 两处黑色圆形区域为危险区域,智能体碰撞到该区域即死亡,回合结束. 该区域为动态变化区域,两处危险区域各自由初始半径为 0.5 的规格随智能体决策次数的增加而扩大,其半径依 0.3 单位/次的速度增加,此处之所以设置危险区域依智能体决策次而变化,是因为执行一个回合的具体时间会因计算机处在不同状态而有所差异,从而造成训练结果不稳定. 左边危险区域的底面圆心坐标为 $(-6.5,-1.5)$,右边危险区域的底面圆心坐标为 $(6,-6)$. 图中的灰色条形实体为墙体,该区域为静态障碍. 左右两面边界墙的中线分别为 $x=\pm 20.5$,上下两面边界墙的中线分别为 $y=\pm 10.5$,内部的障碍墙的中心线为 $x=10$,墙的长度为 12. 图中右下角的深灰色区域为安全出口,智能体到达此处视为路径规划成功的标志.

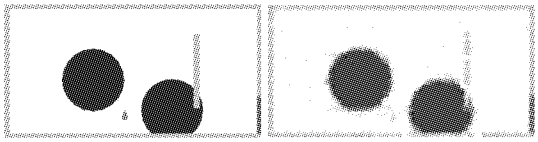
根据上述设定,随着智能体决策次数的增加,障碍墙下方的通道将会被危险区域封堵,智能体只能选择从上方的通道绕行到达终点. 另外,为了避免智能体在训练前期探索环境的阶段不停地在环境中往返而不能结束一个回合,设定智能体单个回合的最大步数为 200.

4.2　图形编码器训练结果

首先,通过随机动作的方式令智能体在环境中探索,获得不同状态下的环境截图,为了减小模型训练的难度,在训练时将环境地面设置为白色,并为智能体设置添加一个箭头用来指示方向. 截取的原始图像大小为 1200×600.

在本文实验中,共截取 1 万张环境图像用来训练编码器.

在正式训练之前，为了降低模型的处理难度，首先使用OpenCV模块下的函数将截图初步降维到200×100，再对图片进行灰度化处理，处理后如图5(a)所示. 本文使用小批量



(a) (b)

图5 “编码-解码”过程前后的图片

Fig. 5 Images before and after encode-decode

梯度下降的方式训练自动编码器，设置学习率为0.01，经过1000轮训练之后，模型趋于收敛. 提取训练好的模型，对一张环境截图进行编码解码过程，得到如图5所示“编码-解码”过程前后的两张图片对比，可以看到降维后的数据被比较完整的复原了，说明本文构建额的编码器成功的完成了图像数据降维的工作，训练的编码器可以应用到后续的任务中.

4.3 LSTM DDPG 实验结果

通过上一步的编码器，环境图像被压缩到了400维的大小. 通过连续4帧图像编码后的数据构成时序数据作为LSTM-DDPG算法的输入数据. 实验设置Actor学习率为0.001，Critic学习率为0.001，回报衰减设置为0.95，采用小批量梯度下降的方式进行模型训练，批次大小设置为128；模型收敛后，收集到的训练过程中的数据变化如图6所示.

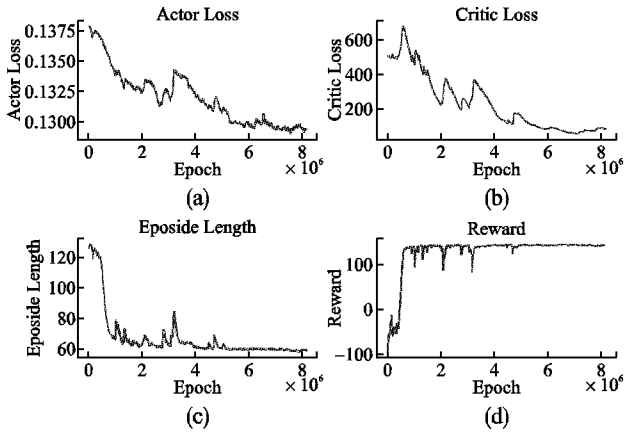
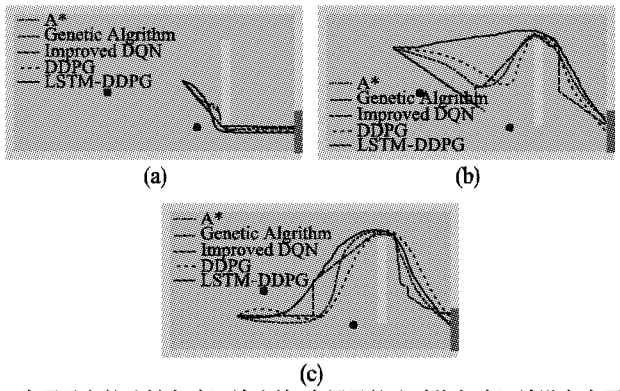


图6 LSTM DDPG模型训练数据

Fig. 6 Training data of LSTM DDPG

其中图6(a)为Actor部分的损失变化曲线，图6(b)为Critic部分的损失变化曲线，观察两图可以发现，模型的两个部分都可以收敛，说明本文设计的模型是合理的，具有可行性. 图6(c)为平均回合步数(/1000步)的变化，图6(d)为平均回合回报(/1000步)的变化，结合两图可以发现，在训练的前期，算法模型还不能进行正确的路径规划，动作选择多为随机动作，智能体在环境中执行较多的步数才能结束一个回合，结合图6(d)可以发现，此时智能体结束一个回合多因为陷入危险区域或达到回合步数上限而结束. 在训练后期，算法模型逐渐收敛，平均回合步数和平均回报都趋于稳定，回报稳定在140上下，回合步数稳定在60步左右，这基本上可以说明智能体可以在不碰到墙壁和危险区域的情况下到达目标位置，进一步说明了本文的算法是可行的.

本文除了通过上述方式验证了提出算法的可行性，还在相同的环境下，设计了与经典的A*算法、遗传算法以及文献[11](Improved DQN)和文献[14](DDPG)中的深度强化学习方法实现效果的对比实验. 对比实验分别以环境中的3处为起点测试3种算法的路径规划能力，这3个点分别是(5,0)、(−10,5)和(−10,−5)，图7给出了LSTM-DDPG与其他4种算法的规划路径结果对比.



*为了避免轨迹被危险区域遮挡，在展示轨迹时将危险区域设定为了初始化状态，其中(b)和(c)中A*算法产生的路径在中途停止是因为接触危险区域而结束.

图7 LSTM-DDPG与其他算法规划路径对比

Fig. 7 Comparison of LSTM-DDPG with other algorithms in the results of path planning

其中图7(a)表示以(−10,5)为起点时5种路径规划算法所规划的路径，图7(b)表示以(−10,−5)为起点时5种路径规划算法所规划的路径. 表2所示是LSTM-DDPG同其他4种算法进行路径规划的相关数据.

表2 LSTM-DDPG与其他算法的路径规划对比

Table 2 Comparison of LSTM-DDPG with other algorithms in path planning

Origin	Arithmetic	Length	Steps	Epoch reward
(5,0)	A*	18.67	26	174
	Genetic Algrithm	20.37	33	162
	Improved DQN	19.17	27	173
	DDPG	20.22	31	163
	LSTM-DDPG	19.85	29	171
(−10,5)	A*	Fail	/	/
	Genetic Algrithm	42.34	78	122
	Improved DQN	50.3	89	111
	DDPG	43.25	76	124
	LSTM-DDPG	38.72	61	139
(−10,−5)	A*	Fail	/	/
	Genetic Algrithm	46.61	73	127
	Improved DQN	48.42	86	114
	DDPG	46.13	74	126
	LSTM-DDPG	43.02	69	131

通过就表2中的数据进行横向对比，发现本文提出的算法在同等条件下拥有较好的表现. 在以离目标点比较近的(0,5)点为起点时，A*算法取得了最好的表现，可以看到A*

算法所规划的轨迹为直线,是以距离最短,回报值最佳. 离散动作的强化学习算法 Improved DQN 的取得了次之的效果. 遗传算法和 DDPG 在躲避动态危险时产生了撞墙的动作,LSTM-DDPG 算法所规划的路径虽然并非最短,但是相对较为平滑,也没有产生撞墙的行为. 在选择较远处的点为起点,其中与目标之间的环境更复杂时,A* 算法的表现不佳,不能完成路径规划任务,这是因为 A* 算法在进行路径规划时只能以初始环境为参考进行规划,可以看出 A* 算法缺乏处理动态环境的能力. 同样是离散动作的强化学习算法 Improved DQN 虽然完成了路径规划,但是在面对危险区域时没有预测能力,又因为可供选择的动作有限,规划出的路径不如连续动作的算法所规划的路径效果好. 在连续动作的算法中,相较于遗传算法和 DDPG,可以看到本文算法生成的轨迹更加平滑,路径更短,回报更高,这是因为遗传算法和 DDPG 虽然具有处理动态环境的能力,但是也只是被动的应对变化的环境,规划的路径中增加了对某些区域的探索. 而本文的算法具有预测环境变化的能力,该特性在图 7(b)有较为明显的体现,可以看到其他算法会向右下方的通道进行探索,本文算法则预测到了右下方的通道将会被封堵,直接选择从右上方通过到达目标地点,减少了探索过程的路径长度,使规划的总路径最短. 综上,本文的算法在动态路径规划任务中能够取得较好的表现.

5　总　结

本文针对传统的路径规划算法多基于静态环境;缺乏对动态环境的处理能力的问题,提出了一种结合 LSTM 的强化学习路径规划算法. 本文的方法以环境图像作为输入,首先构造了能够压缩图像特征的编码器,在尽可能完整地保留环境图像原始信息的前提下,降低图像的特征维度,进而从整体上降低了路径规划任务的复杂程度. 本文基于在连续动作空间上具有良好表现的 DDPG 算法,在 DDPG 算法中结合了 LSTM 结构,利用 LSTM 能够处理时序数据的特性,使其在生成动作时能够有选择的参考之前时刻的信息,做出基于对环境预测的动作输出,预先规避环境中可能发生的危险. 最后通过实验与经典路径规划算法和其他强化学习算法进行性能对比,证明了本文算法对动态环境的预测能力以及路径规划的高效性.

References:

[1] Dijkstra E W. A note on two problems in connexion with graphs [J]. Numerische Mathematik, 1959, 1(1): 269-271.

[2] Hart P E, Nilsson N J, Raphael B. A formal basis for the heuristic determination of minimum cost paths [J]. IEEE Transactions on Systems Science and Cybernetics, 1968, 4(2): 100-107.

[3] Khatib O. Real-time obstacle avoidance for manipulators and mobile robots [J]. Autonomous Robot Vehicles, Springer, 1986, 5(1): 90-98.

[4] Chen Y W, Chiu W Y. Optimal robot path planning system by using a neural network-based approach[C]//International Automatic Control Conference, IEEE, 2015: 85-90.

[5] Wu P, Cao Y, He Y, et al. Vision-based robot path planning with deep learning[C]//International Conference on Computer Vision Systems, Springer, 2017: 101-111.

[6] Yu J, Ji J, Miao Z, et al. Neural network-based region reaching formation control for multi-robot systems in obstacle environment [J]. Neurocomputing, 2019, 333(10): 11-21.

[7] Mnih V, Kavukcuoglu K, Silver D, et al. Playing atari with deep reinforcement learning[J]. arXiv preprint arXiv: 1312. 5602, 2013.

[8] Mirowski P, Pascanu R, Viola F, et al. Learning to navigate in complex environments [J]. arXiv preprint arXiv: 1611. 03673, 2016.

[9] Panov A I, Yakovlev K S, Suvorov R. Grid path planning with deep reinforcement learning: preliminary results [J]. Procedia Computer Science, 2018, 123(1): 347-353.

[10] Lei X, Zhang Z, Dong P. Dynamic path planning of unknown environment based on deep reinforcement learning[J]. Journal of Robotics, 2018, 2018(9): 1-10.

[11] Lv L, Zhang S, Ding D, et al. Path planning via an improved DQN-based learning policy[J]. IEEE Access, 2019, 7(5): 67319-67330.

[12] Konda V R, Tsitsiklis J N. Actor-critic algorithms[C]//Advances in Neural Information Processing Systems, 2000: 1008-1014.

[13] Sutton R S, McAllester D A, Singh S P, et al. Policy gradient methods for reinforcement learning with function approximation[C]// Advances in Neural Information Processing Systems, 2000: 1057-1063.

[14] Lillicrap T P, Hunt J J, Pritzel A, et al. Continuous control with deep reinforcement learning [J]. arXiv preprint arXiv: 1509. 02971, 2015.