

本科毕业论文（设计）

基于强化学习的多移动机器人
分布式避障导航

**REINFORCEMENT LEARNING BASED
DISTRIBUTED MULTI-ROBOT NAVIGATION
WITH COLLISION AVOIDANCE**

吴新裕

哈尔滨工业大学

2023 年 6 月

密级：公开

本科毕业论文（设计）

基于强化学习的多移动机器人 分布式避障导航

本 科 生：吴新裕

学 号：190320321

指 导 教 师：梅杰教授

专 业：自动化

学 院：深圳校区机电工程学院

答 辩 日 期：2023 年 6 月

学 校：哈尔滨工业大学

摘 要

移动机器人具有载荷能力强，能量效率高的特点，是现代社会进行运输的重要工具，也是机械臂、雷达等多种载荷的重要载体，对社会生产力具有重要影响。分布式多移动机器人系统是完成大规模任务的重要实现手段，可应用在地图探索、智能交通、配送分发等多种应用场景中。分布式多移动机器人系统的基石是避障导航技术，鲁棒性强的避障是系统稳定运行的关键和安全性的保障。强化学习是探索复杂任务的利器，基于学习的算法能够拥有强鲁棒性和长远决策能力，能够对任务进行高层次的抽象。基于强化学习的多移动机器人分布式避障导航具有重要研究意义。

本论文首先建立了多移动机器人避障导航问题的数学模型和单个机器人的运动学模型。建立数学模型后，本论文设计了环境编码和相应的神经网络结构和强化学习算法。本论文的环境编码基于 RVO 演变而来，增强了 RVO 表示法对形状的描述能力，神经网络采用计算高效的全连接网络，用补全来处理边长输入。在奖励函数部分，本论文采用与速度指令高度相关的奖励函数，直接高效的指导神经网络输出。在然后本论文阐述了仿真环境的随机构建和交互结算的实现方法。本论文采用 `mojuco` 物理引擎作为仿真环境而不是简单的积分器，避免了速度不连续，迫使学习算法探索更加符合物理实际的避碰方案，得到了更长的规划能力。

最终，本论文通过设计强化学习系统，实现了多移动机器人在障碍物密集场景下的避障，在 15 个测试场景，最高 30 个机器人同时运行的情况下，算法拥有优秀的指标表现。算法可完全分布式部署，能够快速扩大系统规模而避免集中式算法在规模扩大时的缺点。算法能够在小型机载电脑上部署，能够适应低控制频率等性能和能效要求，且对不同机器人动力学模型具有良好适应能力。在与其他论文的横向对比中，本论文的算法相较与三个基于学习的算法和一个非学习算法均具有明显优势。

关键词：多移动机器人；分布式；避障；强化学习

Abstract

Mobile robots have the advantages of strong payload capacity and high energy efficiency, and they are important tools for transportation in modern society. They are also essential carriers for various payloads, such as manipulators, radars, and so on, and they have a significant impact on social productivity. Distributed multi-mobile robot systems (MMRS) are an important means to accomplish large-scale tasks, and they can be applied in various scenarios, such as map exploration, intelligent transportation, distribution and delivery. The foundation of MMRS is obstacle avoidance navigation technology, which is crucial for ensuring the stability and safety of the system. Reinforcement learning (RL) is a powerful tool for exploring complex tasks. RL-based algorithms can exhibit strong robustness and long-term decision-making ability, and they can abstract tasks at a high level. Therefore, RL-based MMRS distributed obstacle avoidance navigation has important research significance.

In this paper, we first establish the mathematical model of the MMRS obstacle avoidance navigation problem and the kinematic model of a single robot. Based on the mathematical model, we design the environment encoding scheme and the corresponding neural network structure and RL algorithm. Our environment encoding scheme is evolved from RVO (Reciprocal Velocity Obstacles), which enhances the ability of RVO representation to describe shapes. Our neural network adopts a computationally efficient fully connected network, which uses completion to handle variable-length input. For the reward function design, we adopt a reward function that is highly correlated with the velocity command, which directly and efficiently guides the neural network output. Then, we describe the implementation methods of random construction and interaction settlement of the simulation environment. We use mujoco physical engine as the simulation environment instead of a simple integrator, which avoids velocity discontinuity and forces the RL algorithm to explore more physically realistic collision avoidance schemes, resulting in longer planning ability.

Finally, we implement and evaluate our RL system for MMRS obstacle avoidance in obstacle-dense scenarios. In 15 test scenarios, with up to 30 robots running simultaneously, our algorithm shows excellent performance indicators. Our algorithm can be fully distributed deployed, which enables fast scaling up of the system size and avoids the drawbacks of centralized algorithms when expanding the scale. Our algorithm can be deployed on small onboard computers, which can adapt to

performance and energy efficiency requirements such as low control frequency, and it has good adaptability to different robot dynamics models. In comparison with other papers, our algorithm has obvious advantages over three learning-based algorithms and one non-learning algorithm.

Keywords: Multi-Robot, Distributed, Collision Avoidance, Reinforcement Learning

目 录

摘 要	I
ABSTRACT	II
目 录	IV
第 1 章 绪 论	1
1.1 课题背景及研究的目的和意义	1
1.2 非学习式多机器人避障算法研究现状	2
1.3 强化学习研究现状	2
1.4 基于强化学习的多移动机器人避障算法研究现状	3
1.5 本文的主要研究内容	3
第 2 章 数学模型的建立	5
2.1 引言	5
2.2 马尔可夫决策模型的建立	5
2.3 机器人运动模型的建立	6
2.4 机器人坐标系与世界坐标系	7
2.5 本章小结	8
第 3 章 多智能体强化学习算法架构设计	9
3.1 引言	9
3.2 机器人自身状态的表示	9
3.3 环境的表示	9
3.3.1 RVOp 表示法	9
3.3.1 障碍物和其他机器人的统一编码	10
3.4 多智能体强化学习的算法与网络结构设计	10
3.5 针对多智能体避障强化学习奖励的设置	12
3.6 本章小结	13
第 4 章 多智能体系统仿真环境构建与交互	14
4.1 引言	14
4.2 仿真环境的构建	14
4.2.1 模型定义文件的生成	14
4.2.2 障碍物的参数表示	16

4.2.3 障碍物轮廓的生成	16
4.3 与仿真环境的交互	17
4.3.1 控制量转换与传入	17
4.3.2 数据读出与处理	18
4.3.3 视觉渲染	20
4.4 本章小结	20
第 5 章 训练与验证	21
5.1 引言	21
5.2 强化学习算法选择	21
5.3 多级训练机制和学习率衰减	22
5.3 指标确立	23
5.4 测试场景	23
5.4.1 训练场景	24
5.4.2 对照场景	25
5.4.3 差异场景	27
5.5 指标表现	30
5.5.1 训练场景中的指标表现	30
5.5.2 对照场景中的指标表现	30
5.5.3 差异场景中的指标表现	30
5.5.4 跨论文横向对比	31
5.5.5 控制频率的影响	33
5.5.6 机器人动力学的影响	33
5.6 本章小结	34
结 论	35
参考文献	36
哈尔滨工业大学本科毕业论文（设计）	40
原创性声明和使用权限	40

第 1 章 绪 论

1.1 课题背景及研究的目的和意义

当今社会，经济活动规模越来越大，对流程自动化的要求越来越高，大型自动化系统便受到越来越多的关注。在仓储管理、自动配送、大型搜救、大尺度地图构建、智能工厂等应用场景，多移动机器人系统有着广阔的应用空间。利用多智能体系统，我们能解决单体无法实现的复杂大型任务，将智能自主系统的能力进一步扩展，使其能够服务体形日益庞大的国民经济。多移动机器人系统有着高负载能力、高能量效率的优势，配合智能的控制算法，能够做到高感知、高鲁棒、低成本、易扩展，是提供大规模优质廉价的大众服务的理想载体。得益于广阔的应用场景、巨大的应用市场，对多移动机器人系统的研究也越来越多，避障导航便是核心之一。

避障导航是多移动机器人系统的核心功能，效率高、鲁棒性强的避障导航能力是多移动机器人系统完成复杂任务的基石。避障导航也是多移动机器人系统良好鲁棒性的基础，适应力强的避障导航能力决定了多移动机器人多复杂应用场景的适应能力。在密集、狭窄、复杂室外环境、具有人际交互的环境中，适应能力强的避障导航能力尤为重要，例如在密集化作业的仓储、包含室外环境的产业园区物流配送等场景，高适应能力的避障导航技术都是不可或缺的核心技术。

基于学习的算法，因其高泛化能力，能够有更强的适应性，是面对复杂应用环境的理想解决方案。基于学习的算法也更加适用于端到端的解决方案，能够从原始的传感器数据中抽象出复杂的有用特征，是提高多移动机器人系统感知能力的理想方法。

分布式设计是多移动机器人系统扩张规模，提高鲁棒性的关键。通过规避集中式控制，可以降低通信成本，分摊计算成本，避免系统规模扩张时的复杂度指数爆炸问题，对成本敏感、规模庞大、实时性要求高的真实工业系统尤为重要。分布式设计省去中心节点，也避免了中心节点故障威胁整个系统的危险，在各单体故障时系统结构不发生质的改变，也更容易处理单体故障，拥有更好的鲁棒性。

综上，基于学习的分布式避障导航算法对于大规模移动机器人系统具有重要意义。

1.2 非学习式多机器人避障算法研究现状^{[1] [28]}

一些方法为每个机器人设计一个一定时间内可行的轨迹，以实现无碰撞导航，然后在下一个时刻重新规划新的轨迹^[1]，但为获得更准确的结果需要更加细致的时间划分，进而需要更多的计算资源。传统的反应式控制器直接以低成本算法计算最优速度，例如人工势场(APF)的方法^[3]，利用势场的概念（包括人工吸引力和排斥力）来寻找无碰撞和时间高效的速度^[4]，但容易陷入局部最优。还有缓冲 Voronoi 单元(BVC)方法^{[5][6]}，和控制屏障函数(CBF)^[7]方法等。虽然这些反应式方法在计算上很高效，但是机器人的动力学并没有完全建模，而且机器人的运动通常只考虑一步以内的约束。

基于速度障碍(VO)的方法及其扩展^{[31][32][33][34][35]}广泛用于动态碰撞避免，通过计算机器人的碰撞速度集，并实时确定每个机器人的速度，以避免到达这些碰撞速度并集，基于速度障碍(VO)的方法有很高的计算效率。最优互惠避障(ORCA)^{[34][35]}及其衍生算法已经广泛应用于人群模拟和多智能体系统中。ORCA 提供了一个充分条件，使得多个独立的移动机器人或智能体在一个未来的一个短时间间隔内避免与其他智能体发生碰撞，并且可以很容易地扩展到包含许多智能体的大规模系统中。ORCA 及其衍生算法^{[8][32]}使用启发式方法构建了一个参数化的避障模型，但这些模型有大量的纷杂的参数，很难被调整到令人满意的性能。此外，这些非学习方法很难适应实际场景中无处不在的不确定性，因为它们假设每个机器人对周围智能体的位置、速度和形状有完美的感知。此外，ORCA 的原始公式是基于全向的完全可控的机器人的，而实际场景中的机器人通常是差速的非完全可控的。为了在最常见的差动驱动机器人上部署 ORCA，已有研究提出了几种方法来处理非完全可控的机器人。ORCA-DD^[35]将每个智能体的有效半径翻倍，以确保在差分约束下为机器人提供无碰撞和平滑的路径，但是由于机器人的有效半径被人为增大，它在狭窄通道和拥挤环境中表现不佳。NH-ORCA^[9]使差分驱动机器人能够在 ϵ 误差内跟随期望速度指令，只需要根据 ϵ 值稍微增加机器人的有效半径而非增大两倍，因此在碰撞避免性能方面优于 ORCA-DD。

1.3 强化学习研究现状

强化学习是一种特殊的机器学习算法，目标是让智能体在环境中执行动作以获得最大的累计奖励。强化学习和有监督学习不同，没有直接的监督信号，只有延迟和随机的反馈。强化学习需要不断地执行动作，观察效果，积累经验，

形成模型。强化学习应用广泛，是通向强人工智能/通用人工智能的核心技术之一。例如，在游戏与博弈中，强化学习算法可以控制星际争霸、Atari 游戏等。

随着人工智能技术的兴起，强化学习受到越来越多的关注，也帮助人们解决了许多令人惊艳的复杂问题。强化学习最经典的算法 VPG^[1] 是其他算法的基石。TRPO^[11] 算法通过引入 KL-Divergence 约束，使算法更加稳定。PPO^[12] 算法利用正则或者剪裁方法，实现与 TRPO 相似的 KL-Divergence 限制效果，但将问题转化为一阶问题，更容易实现，效率更高。DQN^{[13][14][15]} 和 DDPG^{[16][17]} 算法是 off-policy 的算法，拥有更高的采样效率。TD3^[18] 算法采用两个 Q 函数，避免了 DDPG 中对 Q 函数过高估计的问题。SAC^{[19][20][21]} 算法采用最大熵学习策略，有更好的探索能力，在仿真和实物^[21]上都取得了很好的效果。模仿学习、逆强化学习等技术也在被积极研究中。延迟奖励^[22]等问题也越来越受到关注，正在被不断改进。

1.4 基于强化学习的多移动机器人避障算法研究现状

多移动机器人系统正在被积极研究中，尤其是以强化学习为基础的解决方案正在被不断推出。对环境的表示方法、对其他机器人的表示方法、通信内容等关键点都有研究。在对可变长障碍物输入的处理中，使用一定数目的最近邻^[23]机器人的信息可以规避可变长输入问题并降低计算成本；使用 LSTM^[24]来处理变长输入可以获得更加完整的信息，但训练、计算的开销大；使用池化技术^[25]也可以处理变长输入产生定长编码，但需谨慎选择编码长度，要求可加性也限制了编码所代表的特征；使用栅格图^[26]的方式来记录其他机器人也可以处理边长问题，利用卷积网络实现参数共享降低计算量，但面临精度与效率的矛盾；使用 RVO^[1]来表示其他机器人也被认为比单纯位置速度表示法更加高效。在通信内容选择上，现有研究基本上都选择了当前时刻的速度和位置^[24]，但也有采用定长的速度位置序列^[25]来更好的预测未来轨迹的方案，配合 MPC，可以实现只用深度学习而不用强化学习来完成避障导航。在对静态障碍物的感知上，现有研究普遍采用 180°2D 激光雷达^{[27][28]}来完成，并用一维卷积网络来处理信息，实现端到端学习。在训练方法上，普遍采用 PPO 算法，算法成熟稳定，但属于 on-policy 方法，采样利用率较低。也有专为 MAS 做出改进的强化学习算法，如基于 DDPG 的 MADDPG^[29]。

1.5 本文的主要研究内容

随着科技进步，人们希望将越来越复杂，越来越庞大的问题自动化，多移动

机器人系统就是其中重要的一环，在智能仓储，无人物流、大面积搜救、大面积地图构建等领域都有广泛应用。避障导航是多移动机器人系统完成复杂任务的基石，也是多移动机器人系统安全性、稳定性的关键。高效、稳定、泛化能力强的避障导航技术是多移动机器人系统的核心技术，也是本文的研究方向。本文主要从以下几个方面展开工作：

（1）将多移动机器人系统数学模型化，定义去中心化的部分可观测马尔可夫决策模型(Dec-POMDP)，定义导航目标、定义静态和动态障碍约束、碰撞约束、定义指令到位置的运动学约束等，并将避障导航目标公式化为时间的期望。

（2）整体设计为每个智能体获得局部观测，通过学习得到的策略，输出自身的速度指令，具体设计包括：设计合适的环境描述方法，尽可能凸显重要环境特征；设计合理的通信内容，使得各智能体能够合理的协作；设计合适的奖励函数，以充分体现理论目标并加快算法学习速度；将合适的强化学习算法和模型结合，得到训练效率高、泛化性能强的策略。

（3）通过仿真环境产生数据，用学习算法进行训练。确立合适的技术指标。通过仿真来验证技术指标优劣和算法对不同场景的适应性，通过物理仿真引擎来验证算法的泛化能力和对现实物理场景的适应能力。对比其他论文的结果，以评价模型在横向对比中的表现。

第 2 章 数学模型的建立

2.1 引言

为研究多移动机器人避障导航问题，首先需要建立问题的数学模型，包含马尔可夫决策模型和机器人运动模型。马尔可夫决策模型用于描述强化学习的环境交互过程，是强化学习的基础，而机器人运动模型用于描述仿真环境中的移动机器人运动学模型，是机器人控制与交互的基础。

2.2 马尔可夫决策模型的建立

部分可观的马尔可夫模型由一个六元组描述：\$(S, A, T, R, Z, O)\$。

1. S 为所有可能的状态的集合：\$s_1, s_2, s_3, \dots \in S\$
2. A 为所有可能的动作的集合：\$a_1, a_2, a_3, \dots \in A\$
3. T 为状态动作转移概率：\$T: p(s_i|a, s_j)\$
4. R 为状态动作奖励函数：\$R: r(s, a)\$
5. Z 为所有可能的观测集合：\$o_1, o_2, o_3, \dots \in Z\$
6. O 为条件观测概率函数：\$O: p(o|s)\$

分布式部分可观的马尔可夫决策过程^[30]由部分可观的马尔可夫模型衍生而来，分布式的部分可观的马尔可夫决策过程由一个九元组描述：\$(D, S, A, T, R, Z, O, h, b_0)\$：

1. D 为所有智能体的集合，\$D = \{1, 2, 3, 4, \dots, n\}\$
2. S 为所有可能的状态的集合：\$s_1, s_2, s_3, \dots \in S\$
3. A 为所有可能的联合动作集合：\$a_1, a_2, a_3, \dots \in A, a_i = \{a_{1,i}, a_{2,i}, a_{3,i}, \dots, a_{n,i}\}\$
4. T 为状态动作转移概率函数：\$T: p(s_i|a, s_j)\$
5. R 为状态联合动作奖励函数：\$R: r(s, a)\$
6. Z 为所有可能的联合观测集合：\$o_1, o_2, o_3, \dots \in Z, o_i = \{o_{1,i}, o_{2,i}, o_{3,i}, \dots, o_{n,i}\}\$
7. O 为联合条件观测概率函数：\$O: p(o|s)\$
8. h 为智能体的视界，智能体与环境进行交互的步数
9. b_0 为智能体的初始信念，\$b_0 \in S\$

在决策过程中，每个智能体只能得知自身的动作和观测而不是联合动作和联合观测，即智能体 d_i 在时间 t 时，只能得知部分历史动作观测信息：

$\{(a_{i,0}, o_{i,0}), (a_{i,1}, o_{i,2}), (a_{i,3}, o_{i,3}) \dots (a_{i,t}, o_{i,t})\}$, 从而做出下一步动作 $a_{i,t+1}$ 。而系统的转移则依靠联合动作, 决策过程见图 2-1。

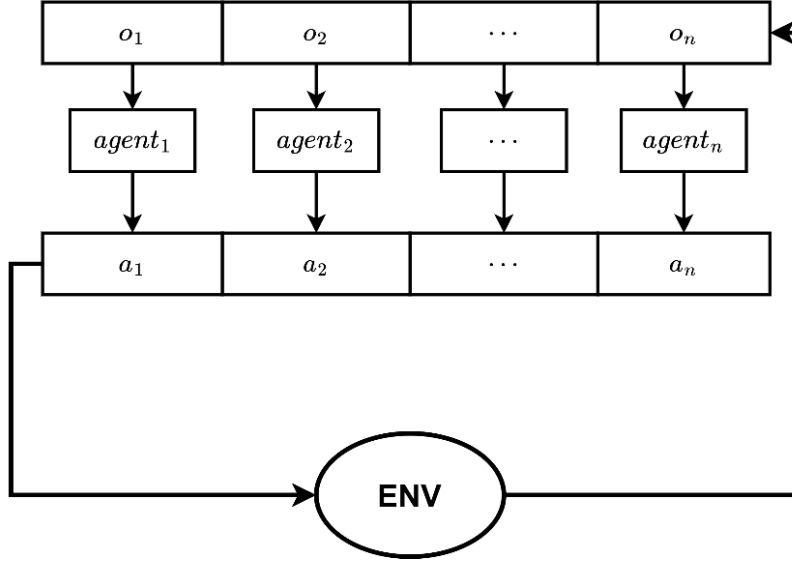


图2-1 分布式马尔可夫决策

2.3 机器人运动模型的建立

所采用的机器人模型为两主动轮一万向轮的差速机器人模型, 用 (x, y) 表示机器人中心点坐标, ϕ 表示机器人坐标系相对于世界坐标系的转角, ω 表示机器人角速度, v 表示机器人线速度, w_r 表示右轮转动角速度, w_l 表示左轮转动角速度, 示意图见图 2-2, 模型具有如下运动学描述:

$$\dot{x} = \frac{R}{2}(w_r + w_l) \cos \phi \quad (2-1)$$

$$\dot{y} = \frac{R}{2}(w_r + w_l) \sin \phi \quad (2-2)$$

$$\omega = \dot{\phi} = \frac{R}{L}(w_r - w_l) \quad (2-3)$$

$$v = \sqrt{\dot{x}^2 + \dot{y}^2} = \frac{R}{2}(w_r + w_l) \quad (2-4)$$

若已知 v, ω , 则可推导:

$$w_r = \frac{2v + \omega L}{2R} \quad (2-5)$$

$$w_l = \frac{2v - \omega L}{2R} \quad (2-6)$$

其中 R 为主动轮半径, L 为两主动轮间距, 两轮的驱动速度为最终的控制量。

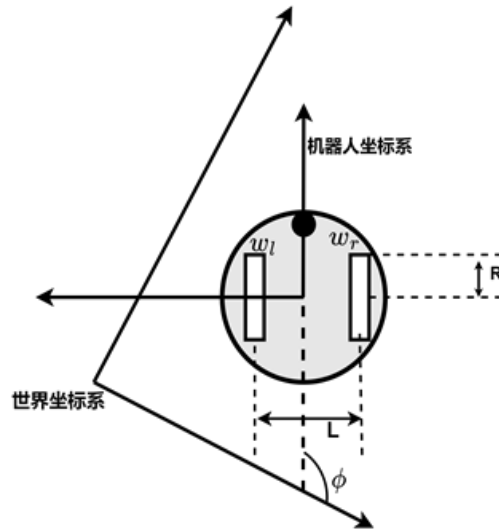


图2-2 机器人模型

2.4 机器人坐标系与世界坐标系

机器人自身坐标系 L_r 以机器人中心为原点, 以机器人正方向为 x 轴正方向。

机器人对环境的观测将在机器人自身坐标系中表示, 见图 2-3。以机器人 2 为例, 不仅仅是 v_1, v_2, v_n 将在机器人坐标系中表示, 自身目标 $targ_2$ 和其他机器人的目标 $targ_1$ 和动作 a_2 也将在机器人自身坐标系中表示。

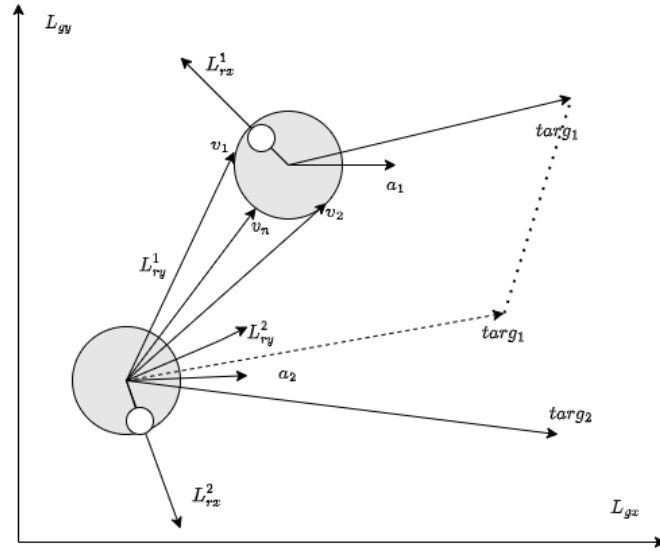


图2-3 机器人坐标系

完全在机器人自身坐标系中表示输入输出是必要的，这将避免训练场景引起的偏置，比如训练场景为以 y 轴为中心左右对换，若以世界坐标系表示输入输出，则模型对世界坐标系的 x, y 方向将产生不同的偏好。

2.5 本章小结

本章建立了分布式部分可观马尔可夫决策模型和机器人运动模型用于描述强化学习的环境交互过程和仿真环境中的移动机器人运动学模型，为后续强化学习算法和速度指令跟踪算法奠定理论基础。

第 3 章 多智能体强化学习算法架构设计

3.1 引言

建立数学模型后，需要设计多智能体强化学习算法架构，从而利用强化学习得到符合指标的控制算法。此部分主要包含状态编码表示设计、强化学习算法设计、网络结构设计、强化学习奖励函数设计。

3.2 机器人自身状态的表示

智能体自身状态可以用一个四元浮点数组表示： $O_{self} = [targ_x, targ_y, v, \omega]$ ，其中 $v_t = [targ_x, targ_y]$ 为机器人目标向量在机器人自身坐标系 L 中的表示。 v 为机器人线速度。 ω 为机器人角速度。

用 $v_T = [Targ_x, Targ_y]$ 表示机器人目标位置在世界坐标系 G 中的表示则可知：

$$\begin{bmatrix} targ_x \\ targ_y \end{bmatrix} = \begin{bmatrix} \cos \phi & \sin \phi \\ -\sin \phi & \cos \phi \end{bmatrix} \begin{bmatrix} Targ_x - x \\ Targ_y - y \end{bmatrix} \quad (3-1)$$

3.3 环境的表示

3.3.1 RVOp 表示法

Reciprocal Velocity Obstacle Plus (RVOp)由互惠速度障碍 Reciprocal Velocity Obstacle^[31] (RVO)衍生而来。

RVOp 由一个八元浮点数组表示： $[v_{px}, v_{py}, v_{1x}, v_{1y}, v_{2x}, v_{2y}, v_{nx}, v_{ny}]$ ，其中前六个数为 RVO，最后两个数不属于 RVO。RVO 由三个向量构成，RVOp 即在 RVO 的基础上增加一个向量。RVO^[31] 表示法被 HRVO^{[32][33]}、ORCA^{[34][35]} 等算法采用。

在图 3-1^[31]中， $v_p = [v_{px}, v_{py}]$ 为 A 机器人中心指向图中深色锥形顶点的向量， $v_1 = [v_{1x}, v_{1y}]$ 和 $v_2 = [v_{2x}, v_{2y}]$ 为图中深色锥形的两边向量，这三个向量构成 RVO。 $v_n = [v_{nx}, v_{ny}]$ 为机器人到障碍物或其他机器人轮廓最近点的向量，此部分不属于 RVO。RVOp 中的四个向量均为在机器人自身坐标系 L 中表示的。

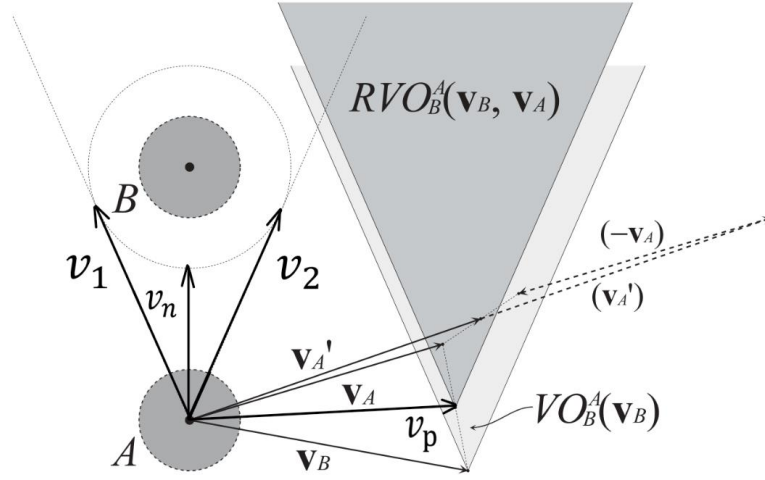


图3-1 RVO示意图

3.3.1 障碍物和其他机器人的统一编码

有了 RVO_p 后，即可对环境中的障碍物和其他机器人进行统一编码。每个障碍物或其他机器人均使用一个十元浮点数编码： $O_{sur} = [RVO_p, O_{targ}]$ ，其中 RVO_p 为八个浮点数， O_{targ} 为两个浮点数。对于其他机器人， O_{targ} 即为对方的目标向量 v_t 在自身坐标系 L 中的表示。对于静态障碍物， O_{targ} 为零向量 $[0,0]$ 。

每个机器人只能得到自身附近一定半径 d_{max} 内的信息，而非全局信息，且最多观测到 O_{max} 个 O_{sur} ，当超出数目时， $v_{ni} = O_{suri}[6:8]$ 的二范数小的 O_{sur} 被优先观测到，即优先观测较近的其他机器人和障碍物。

3.4 多智能体强化学习的算法与网络结构设计

算法采用 SAC 算法^{[19][20][21]}，网络设计有两种，均通过 μ layer 和 σ layer 两个线性层分别产生二维高斯分布的均值和方差，最终动作 $a = [v_x, v_y]$ 由高斯分布采样得到，为连续动作空间。

第一种策略网络采用 BiGRU^[36]网络处理边长的观测，见图 3-2，其中 $obs_i =$

$[O_{self}, O_{suri}]$, 按照距离远近排序, 距离越近, $v_{ni} = O_{suri}[6:8]$ 的二范数越小, 排序越靠前。

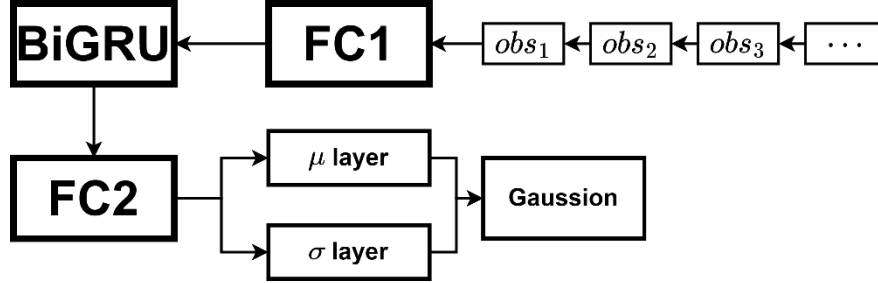


图3-2 BiGRU 策略网络

其对应的动作价值网络见图 3-3:

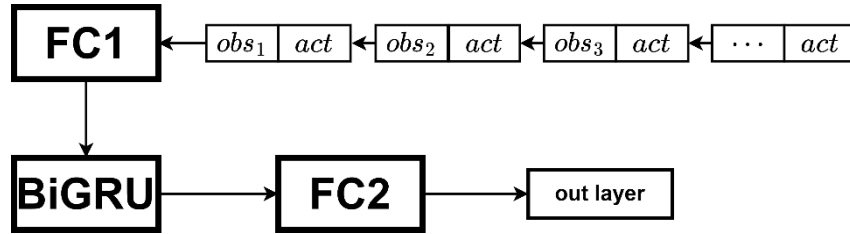


图3-3 BiGRU 价值网络

第二种策略网络不采用 BiGRU 网络, 输入为 O_{self} 和一系列 O_{sur} 的串联, 见图 3-4。其中 O_{sur} , 按照距离远近排序, 距离越近, $v_{ni} = O_{suri}[6:8]$ 的二范数越小, 排序越靠前。如果没有 O_{max} 个观测, 则使用一个特定的浮点数 f_{null} 将输入补齐至一致长度, 最终输入到全链接网络中。

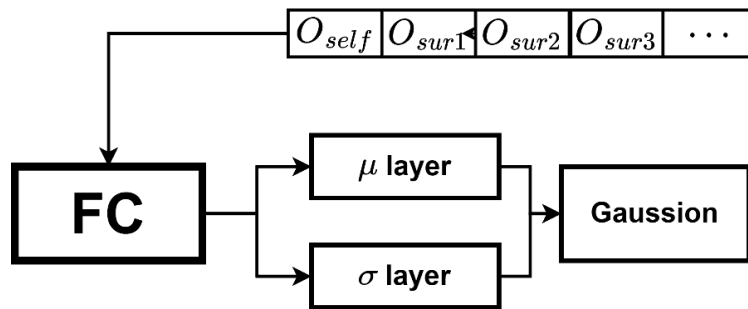


图3-4 非 BiGRU 策略网络

对应的动作价值网络见图 3-5:

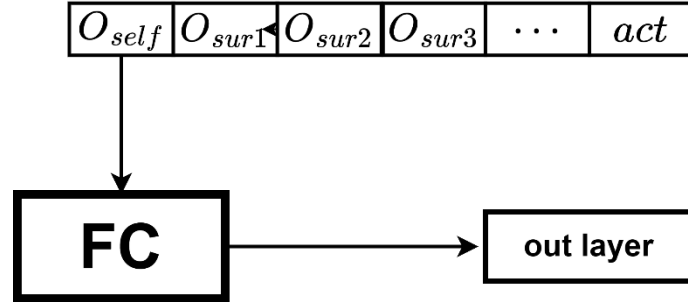


图3-5 非 BiGRU 价值网络

3.5 针对多智能体避障强化学习奖励的设置

奖励项主要由六部分构成，时间惩罚，目标奖励、两个碰撞惩罚、两个靠近惩罚。

时间惩罚：

$$r_{time} = t_b < 0 \quad (3-3)$$

目标奖励：

$$r_{targ} = a \frac{v \cdot v_{targ}}{v_{max}|v_{targ}|} \in (-a, a) \quad (3-4)$$

其中 v 为机器人速度， v_{targ} 为指向目标的速度， v_{max} 为机器人最大速度。

对静态障碍物的碰撞惩罚：

$$r_{colo} = -b \frac{v \cdot v_n}{v_{max}|v_n|} - c \in (-b - c, b - c) \quad (3-5)$$

其中 v_n 即为 RVOp 中的最后一个向量。

对其他机器人的碰撞惩罚：

$$r_{colr} = -d \frac{(v_1 - v_2) \cdot v'}{v_{max}|v'|} - f \in (-2d - f, 2d + f) \quad (3-6)$$

其中 $v_1 - v_2$ 为自身速度减去对方速度， v' 为自身指向对方的向量。

对静态障碍物的靠近惩罚：

$$r_{nearo} = -g \left(\frac{v \cdot v_n}{v_{max}|v_n|} + 1 \right) e^{-\frac{|v_n|}{\eta R}} \in (-2g, 0) \quad (3-7)$$

其中 R 为机器人半径。

对其他机器人的靠近惩罚：

$$r_{nearr} = -h \left(\frac{(v_1 - v_2) \cdot v'}{v_{max}|v'|} + 2 \right) e^{-\frac{|v_n|}{\mu R}} \quad (3-8)$$

其中 R 为机器人半径。

公式中的 $a, b, c, d, e, f, g, \eta, h, \mu, t_b$ 为可调参数,总奖励 r_{all} 为各项相加:

$$r_{all} = r_{time} + r_{targ} + r_{colo} + r_{colr} + r_{nearo} + r_{nearr} \quad (3-9)$$

3.6 本章小结

本章说明了多智能体强化学习算法的设计,包含状态编码设计、算法与网络的选择、奖励的设计。本章为算法总体结构设计,为算法的最终实现奠定基础。

第 4 章 多智能体系统仿真环境构建与交互

4.1 引言

强化学习算法需要可交互的环境用于学习迭代，本课题构建的仿真环境基于 mujoco 物理引擎，其环境描述文件由一个 xml 文本文件定义，本课题使用 python 代码生成所需的 xml 代码，从而实现随机场景的构建。对障碍物及其轮廓的参数表示也尤为重要，为仿真引擎生成的 xml 描述文件必须和 python 程序中的参数化表示一一对应。

与仿真环境的交互重点在将神经网络的速度指令映射到底层的驱动轮转速，需要考虑限幅等约束。数据读出与处理是生成神经网络输入的必要步骤，其中对任意凸多边形生成 RVOp 表示编码涉及大量复杂几何运算，为保证运行效率，底层由 C++ 实现，再接入 python 主程序。基于 GLFW 的视觉渲染使得过程可视。

4.2 仿真环境的构建

仿真环境的构建依托 mujoco 物理引擎进行，主要包含三部分工作：模型定义文件的生成、障碍物在程序中的参数表示、由障碍物生成障碍物轮廓。

4.2.1 模型定义文件的生成

mujoco 模型用一个 xml 文本文件定义，为在仿真训练时生成随机环境，需要自动生成 xml 模型定义文件。xml 模型定义文件主要包含以下几个部分：

1. 全局设置
2. 地面的 body 和 geom
3. 用于渲染的光源
4. 各个机器人的 body、geom、pos、joint、name 和 actuator
5. 各个障碍物的 body、geom、pos

name 是为了在程序运行时进行数据索引而定义的唯一标记，即一个 ID。body 被 mujoco 作为一种容器，用来容纳 geom、joint，构成模型定义的主体架构。geom 即几何刚体，mujoco 提供几种基本类型的凸刚体，比如圆柱，球体，长方体，椭球体等，每一个 geom 可以定义碰撞掩膜用于控制碰撞检测，定义摩擦参数用于接触力学仿真以及 rgba 四通道颜色用于渲染等。joint 用于赋予 body 自由度，mujoco 提供的基础 joint 如 free joint、hinge joint、ball joint，每

个 joint 可以定义限幅、摩擦损失、阻尼等关节特性。 actuator 即为驱动器， mujoco 中所有 actuator 都可以由 general actuator 设置不同的参数产生，并且 mujoco 提供集中常用驱动器的别名，如力驱动器，速度驱动器等。 mujoco 相关内容详情请参考官方文档。

本项目建模的机器人为一个两轮驱动的圆形差速机器人，见图 4-1，主要由外壳、两个驱动轮及其对应的转动关节和速度驱动器、两个球形万向轮及其对应的球关节构成。

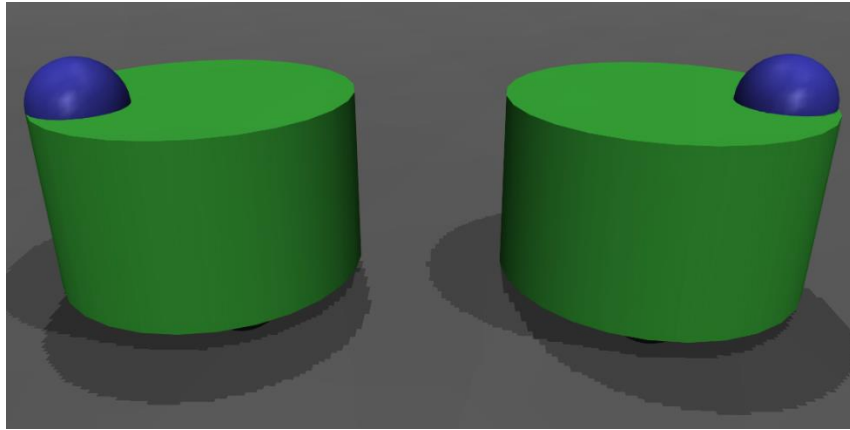


图4-1 碟形差速机器人

静态障碍物有两种类型，圆柱体和多边形柱体，圆柱体由 mujoco 直接提供，而多边形柱体需要由多个纤薄长方体拼接而成，见图 4-2。

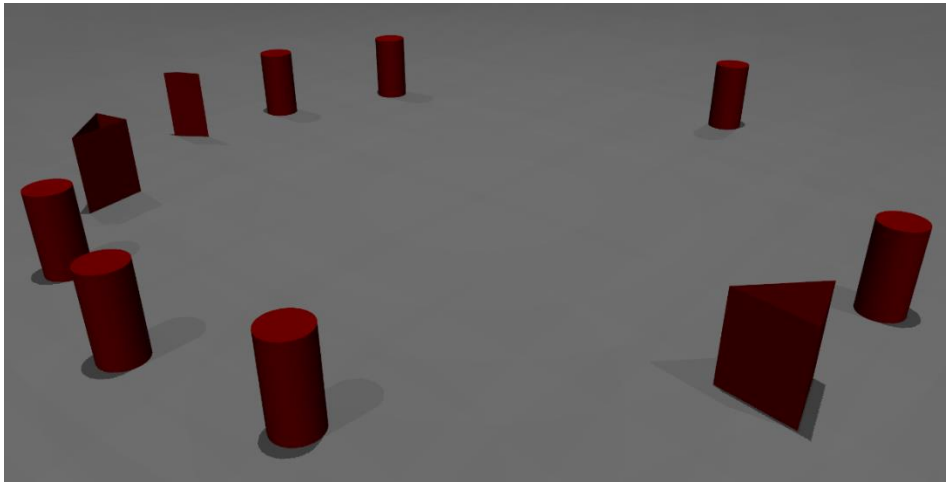


图4-2 障碍物

4.2.2 障碍物的参数表示

生成 xml 模型定义文件的同时，需要保存模型中的障碍物参数，以供后续交互时解算观测与状态。

圆柱障碍物由三元浮点数组表示 $obstacle_{cld} = [c_x, c_y, r]$ ，分别表示圆柱体在水平面上的中心点和圆柱体的半径。

多边形棱柱障碍物由多元浮点数组表示 $obstacle_{plg} = [c_x, c_y, r, v_{1x}, v_{1y}, v_{2x}, v_{2y}, \dots \dots]$ ，用 $proj_{plg}$ 表示多边形棱柱在水平面上的投影形，即一个凸多边形，其中 $[c_x, c_y]$ 为 $proj_{plg}$ 的外接圆中心， r 为 $proj_{plg}$ 的外接圆半径， $[v_{ix}, v_{iy}]$ 为 $proj_{plg}$ 的各顶点坐标，顶点按逆时针方向排序。

环境中的所有静态障碍物可由一个集合表示： $OBS = \{obstacle_{cld_1}, obstacle_{cld_2}, \dots, obstacle_{plg_1}, obstacle_{plg_2} \dots\}$ 。

4.2.3 障碍物轮廓的生成

静态障碍物需要按照机器人半径进行膨胀才能用于解算 RVOp 等观测值，膨胀后的轮廓可以由一系列的圆弧和线段表示。

圆弧可由一个 9 元数组表示， $arc = [c_x, c_y, r, v_{1x}, v_{1y}, v_{2x}, v_{2y}, v_{mx}, v_{my}]$ ，其中 $[c_x, c_y]$ 为圆弧中心， r 为圆弧半径， $[v_{1x}, v_{1y}, v_{2x}, v_{2y}]$ 为圆弧的两个顶点， $[v_{mx}, v_{my}]$ 为圆弧的中点。

线段可以由一个 6 元数组表示， $lineseg = [v_{nx}, v_{ny}, v_{1x}, v_{1y}, v_{2x}, v_{2y}]$ ，其中 $[v_{nx}, v_{ny}]$ 表示线段的切线方向，即指向障碍物内侧的方向， $[v_{1x}, v_{1y}, v_{2x}, v_{2y}]$ 为线段的两个顶点。

每个障碍物的轮廓均可由一系列的线段和圆弧构成，比如一个圆柱体障碍物的轮廓 $contour_{cyl} = [arc_1, arc_2]$ ，即两个半圆弧构成。一个三棱柱障碍物的轮廓 $contour_{tri} = [lineseg_1, arc_1, lineseg_2, arc_2, lineseg_3, arc_3]$ ，即三条线段和三段圆弧构成。轮廓示意图见图 4-3 和图 4-4。

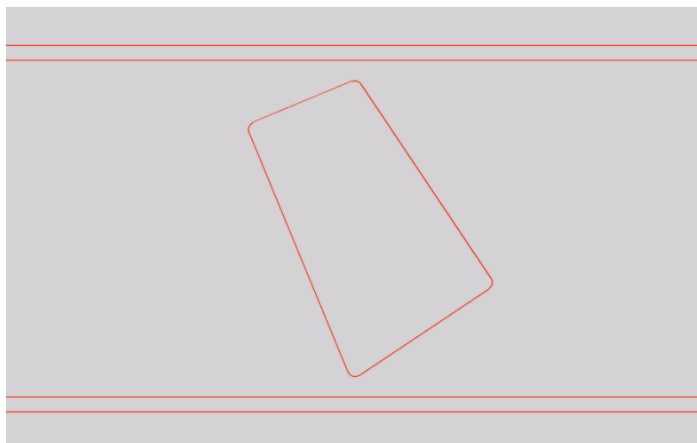


图4-3 障碍物轮廓1

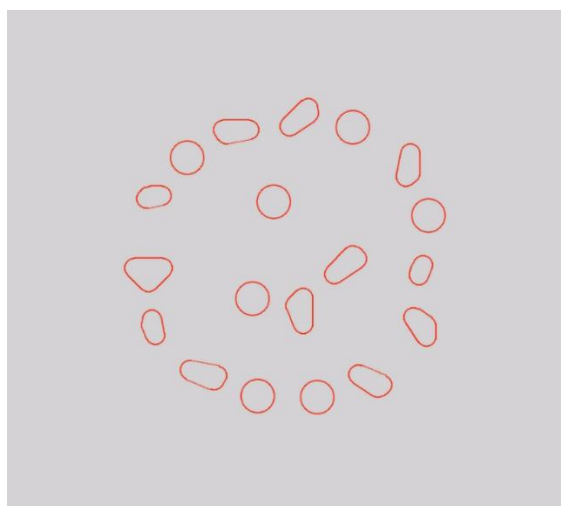


图4-4 障碍物轮廓2

4.3 与仿真环境的交互

mujoco 底层为 C 函数代码，提供 python binding。交互主要分为三个部分，控制量转换与传入，数据读出与解算和视觉渲染。

4.3.1 控制量转换与传入

mujoco 提供 `mujoco.MjData.ctrl` 指向控制量数组，在调用 `mj_step` 仿真迭代之前对 `mujoco.MjData.ctrl` 控制量数组进行赋值即可传递关节控制量，即本项目模型两个主动轮的速度驱动器的目标转速 $[w_l, w_r]$ 。但其单位并非标准国际单位，需先进行固定比例缩放。

神经网络输出量为在机器人自身坐标系L下速度指令 $a = [v_x, v_y]$ ，而 mujoco 控制量为关节转速，需要进行转换。

1. 首先将 $a = [v_x, v_y]$ 转换为世界坐标系下的控制量 $v = [v_{gx}, v_{gy}]$

$$v' = a' \begin{bmatrix} \cos \phi & -\sin \phi \\ \sin \phi & \cos \phi \end{bmatrix} \quad (4-1)$$

2. 再将 $v = [v_{gx}, v_{gy}]$ 转换为差速控制量 $a_{vw} = [v, \omega]$ ，其中 v 为线速度， ω 为角速度。

$$v = \sqrt{v_{gx}^2 + v_{gy}^2} \cos \langle \varphi, \phi \rangle \quad (4-2)$$

其中 φ 为速度 $v = [v_{gx}, v_{gy}]$ 与 x 轴的夹角， ϕ 为机器人坐标系相对世界坐标系的转角， $\langle \varphi, \phi \rangle$ 为两者的交角。

$$\omega = \frac{\langle \varphi, \phi \rangle}{\tau} \quad (4-3)$$

其中 τ 为可调参数，用于调节转向速度。

3. 最后，将 $a_{vw} = [v, \omega]$ 转换为两个差速轮转速。

$$w_r = \frac{2v + \omega L}{2R} \quad (4-4)$$

$$w_l = \frac{2v - \omega L}{2R} \quad (4-5)$$

最终控制量为 $[w_r, w_l]$ 的一个固定比例缩放，使得传入 mujoco 后转速为标准单位 rad/s。

4.3.2 数据读出与处理

mujoco 在仿真过程中的关节数据可由 mujoco.MjData.joint 读出，代表机器人位姿的关节为每个机器人与 worldbody 之间的 freejoint，位姿在 mujoco 中使用一个 7 元浮点数组表示 $config = [x, y, z, q_r, q_1, q_2, q_3]$ ，前三元 $[x, y, z]$ 为关节平移量，后四元 $[q_r, q_1, q_2, q_3]$ 为关节位姿的四元数表示，并已被归一化。机器人只具有 yaw 角旋转，而没有 roll 和 pitch 角的旋转，可由四元数知机器人绕 z 轴旋转角 ϕ 。

$$\phi = \arctan(2q_r q_3, 1 - 2q_3^2) \quad (4-6)$$

读出机器人位姿后，可由位姿和环境创建时保存的障碍物轮廓信息进行

RVOp 解算，此过程较为复杂，表示为以下函数：

$$RVOp = RVOpcalculate(x, y, z, \phi, contours) \quad (4-7)$$

其中 *contours* 即为由线段和圆弧描述的障碍物轮廓。为提高效率，*RVOpcalculate* 部分的代码实现由 C++ 完成，编译成动态链接库后再接入 python。RVOp 的解算结果见示意图 4-5 和 4-6。

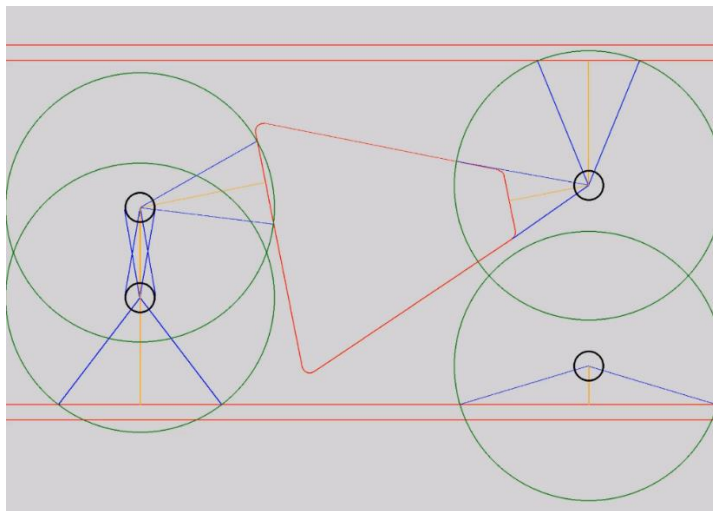


图4-5 RVOp解算示意图1

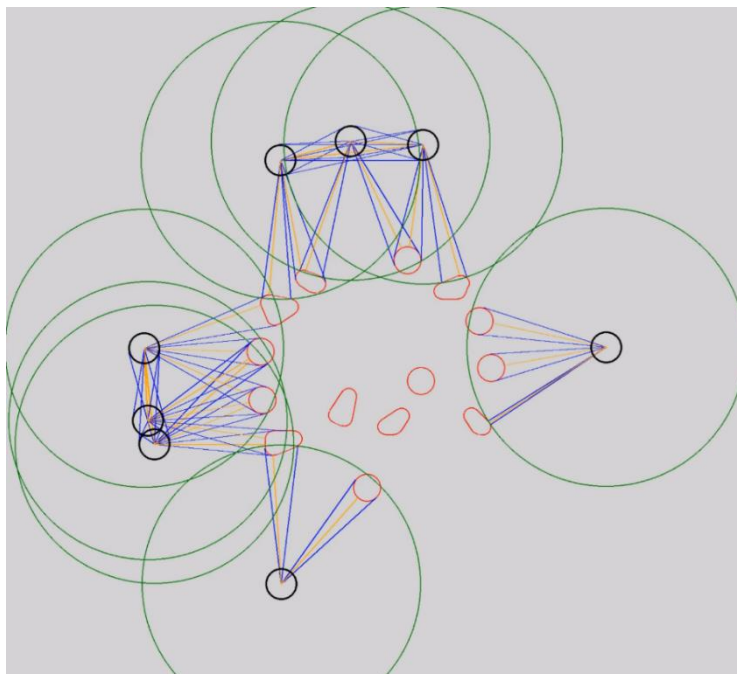


图4-6 RVOp解算示意图2

4.3.3 视觉渲染

mujoco 视觉渲染使用 OpenGL 完成，主要步骤可简略概括为以下几步：

1. 初始化 GLFW 库，设置 GLFW.window 的可见性，buffer 类型等。
2. 绑定 mjvScene，构建 mjvContext
3. 调用 OpenGL 对 Scene 进行渲染
4. 读出 buffer 中的 rgb 通道为一个图片数组
5. 通过 pyav 音视频库对单帧图片数组进行视频编码并保存至文件。

以上为离屏渲染（off screen rendering）的主要步骤，on screen rendering 步骤有所不同。

4.4 本章小结

本章说明了如何在程序实现层面上构建仿真环境包含为 mujoco 生成 xml 文本文件，为 python 主程序解算对应的障碍物及其轮廓，并实现 python 代码与 mujoco 物理引擎之间的仿真交互，包含观测量的解算、控制量的转换。

第 5 章 训练与验证

5.1 引言

强化学习算法主体采用 SAC 算法，配合为任务特殊设计的多级训练进行仿真训练。训练得到模型参数化，本课题主要从四个指标验证算法有效性，分别为成功率，到达率，平均速度，多余路程。测试场景中既有训练时使用的场景，也有训练中从未出现的全新场景，以测试模型泛化能力。本课题还进行控制频率测试，验证算法实时性，以及动力学测试，以验证算法对不同具有不同动力学特征的移动机器人的适应能力。本课题还对比了其他论文的结果，以评价模型在横向对比中的表现。

5.2 强化学习算法选择

强化学习算法主要由使用 pytorch 库构建，实现的算法为 SAC 算法，主要由一个策略网络 π_ϕ 、两个价值网络 Q_θ 、两个 target 价值网络 $Q_{\hat{\theta}}$ 、一个随训练实时更新的标量参数 α 和一个经验池构成，每一步的更新大致可以概括为以下几步：

1. 从经验池中抽取 batch
2. 利用 batch 对价值网络做梯度更新

$$\begin{aligned} \nabla_{\theta} J_Q(\theta) = \nabla_{\theta} \frac{1}{2} (Q_{\theta}(s_t, a_t) - (r(s_t, a_t) + \gamma Q_{\hat{\theta}}(s_{t+1}, a_{t+1}) \\ - \alpha \log(\pi_{\phi}(a_{t+1}|s_{t+1})))) \end{aligned} \quad (5-1)$$

3. 利用 batch 对策略网络做梯度更新

$$\nabla_{\phi} J_{\pi}(\phi) = \nabla_{\phi} (\alpha \log(\pi_{\phi}(a_t|s_t)) - Q_{\theta}(s_t, a_t)) \quad (5-2)$$

4. 用价值网络的参数对 target 价值网络的参数做更新

$$\hat{\theta} = \tau \theta + (1 - \tau) \hat{\theta} \quad (5-3)$$

5. 根据动作不确定性对参数 alpha 做更新

$$\nabla_{\alpha} J(\alpha) = -\alpha \log \pi_{\phi}(a_t|s_t) - \alpha \hat{H} \quad (5-4)$$

将公式代入图 5-1^[17]中的具体算法流程即可进行训练：

Algorithm 1 Soft Actor-Critic

Input: θ_1, θ_2, ϕ $\bar{\theta}_1 \leftarrow \theta_1, \bar{\theta}_2 \leftarrow \theta_2$ $\mathcal{D} \leftarrow \emptyset$ for each iteration do for each environment step do $\mathbf{a}_t \sim \pi_\phi(\mathbf{a}_t \mathbf{s}_t)$ $\mathbf{s}_{t+1} \sim p(\mathbf{s}_{t+1} \mathbf{s}_t, \mathbf{a}_t)$ $\mathcal{D} \leftarrow \mathcal{D} \cup \{(\mathbf{s}_t, \mathbf{a}_t, r(\mathbf{s}_t, \mathbf{a}_t), \mathbf{s}_{t+1})\}$ end for for each gradient step do $\theta_i \leftarrow \theta_i - \lambda_Q \hat{\nabla}_{\theta_i} J_Q(\theta_i)$ for $i \in \{1, 2\}$ $\phi \leftarrow \phi - \lambda_\pi \hat{\nabla}_\phi J_\pi(\phi)$ $\alpha \leftarrow \alpha - \lambda \hat{\nabla}_\alpha J(\alpha)$ $\bar{\theta}_i \leftarrow \tau \theta_i + (1 - \tau) \bar{\theta}_i$ for $i \in \{1, 2\}$ end for end for Output: θ_1, θ_2, ϕ	▷ Initial parameters ▷ Initialize target network weights ▷ Initialize an empty replay pool ▷ Sample action from the policy ▷ Sample transition from the environment ▷ Store the transition in the replay pool ▷ Update the Q-function parameters ▷ Update policy weights ▷ Adjust temperature ▷ Update target network weights ▷ Optimized parameters
--	--

图5-1 SAC算法

5.3 多级训练机制和学习率衰减

通过逐级递增的难度设置，可以帮助强化学习训练更好的探索，提高训练效果。本课题采用三级难度递增的训练机制，每一级 5 个训练场景。典型场景如图 5-2 示意。

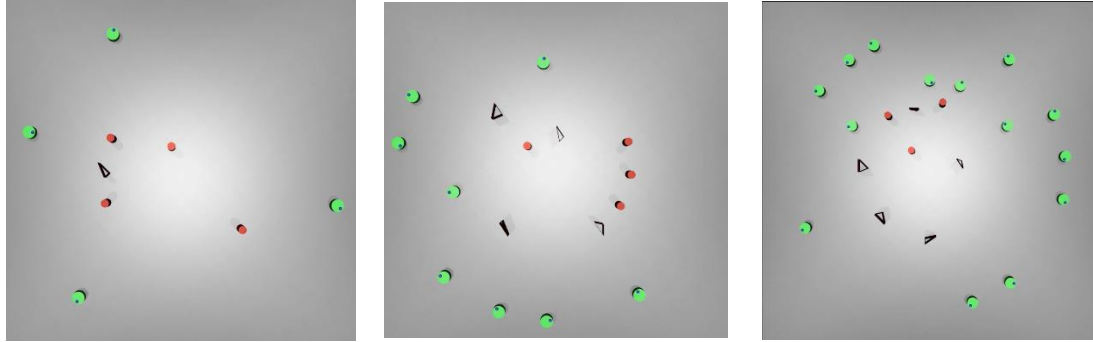


图5-2 三级训练场景典型对比

训练曲线如下图所示,ave ret 为奖励的滑动平均,pi loss 为策略迭代的误差, q loss 为动作价值函数的迭代误差。训练过程中的误差的两个明显尖峰为切换场景难度造成。

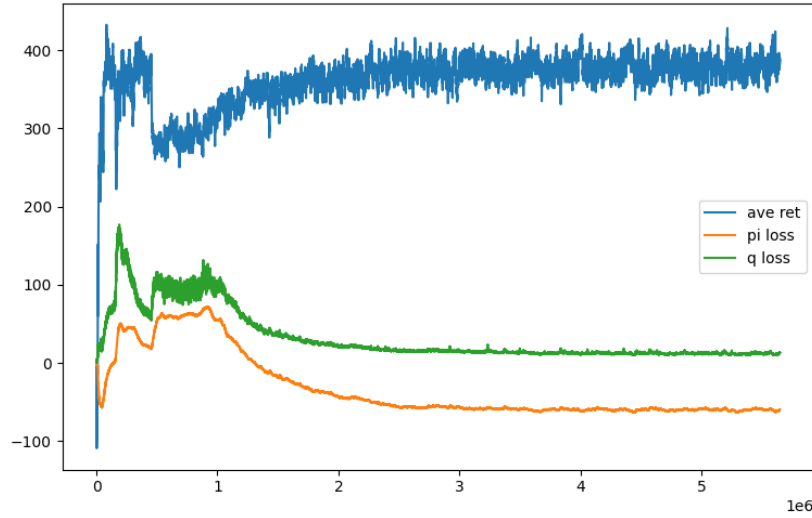


图5-3 训练曲线

学习率衰减对学习的作用是普遍的，通常在模型训练到达高原，误差指标无法下降时，减小学习率具有积极作用。

本课题使用四个参数调节学习率衰减，分别为 Dc_s, Dc_f, Dc_m, Dc_t ，在 Dc_s 训练步后，若平均奖励在 Dc_t 次迭代后依然没有上升，则将学习率乘以 Dc_f ，但最小学习率不小于 Dc_m 。

5.3 指标确立

成功率指的是在规定时间内无碰撞到达终点的机器人占有所有机器人的百分比。到达率指的是在规定时间内到达终点的机器人占有所有机器人的百分比，产生碰撞后分开，然后在规定时间内到达终点也计入到达率。平均速度指计入成功率的机器人的平均线速度，机器人最大线速度为 1m/s。多余路程指的是计入成功率的机器人的曲线路径长度除以直线路径的比值。

5.4 测试场景

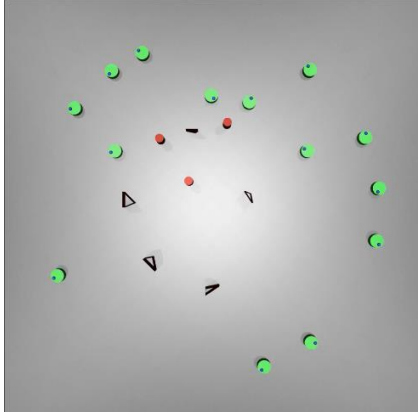
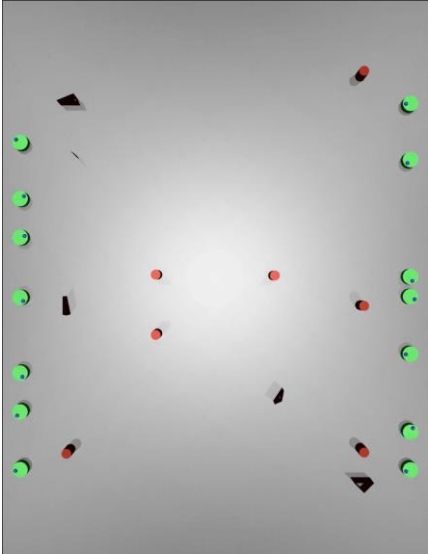
实验场景主要有机器人、小障碍物、大障碍物组成。小障碍物包括圆柱体、三棱柱、四棱柱。大障碍物包括圆柱体、三棱柱、四棱柱以及直墙围挡。小障碍物的位置和形状随机生成，大障碍物外接圆中心点固定，形状随机生成。机器人的目标点有三种方式确定，分别为 circle、line、spin。设机器人初始点位为 (x_0, y_0) ，目标点位为 (x_t, y_t) 。

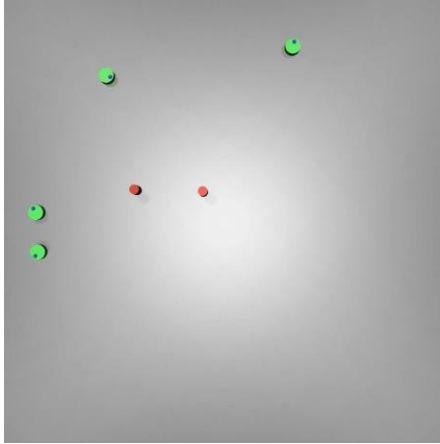
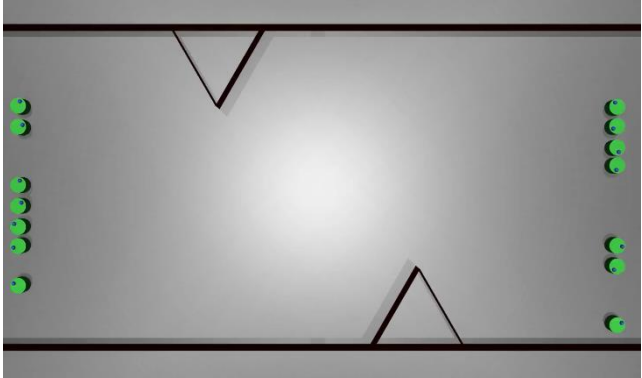
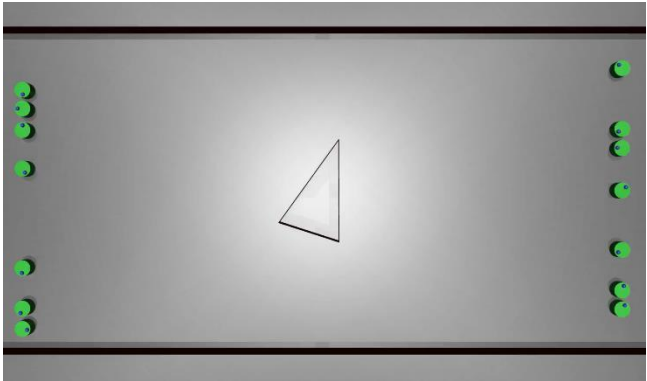
1. 若采用 circle 目标点，则 $(x_t, y_t) = (-x_0, -y_0)$ ，即走对角线。
2. 若采用 line 目标点，则 $(x_t, y_t) = (-x_t, y_t)$ ，即左右镜像互换。
3. 若采用 spin 目标点，则 $(x_t, y_t) = (x_t, y_t) \begin{pmatrix} 0 & 1 \\ -1 & 0 \end{pmatrix} = (-y_t, x_t)$ ，即围绕场景中心逆时针旋转 90 度。

5.4.1 训练场景

训练场景中的小障碍物仅包含圆柱体和三棱柱，不含四棱柱。

表 5-1 训练场景说明

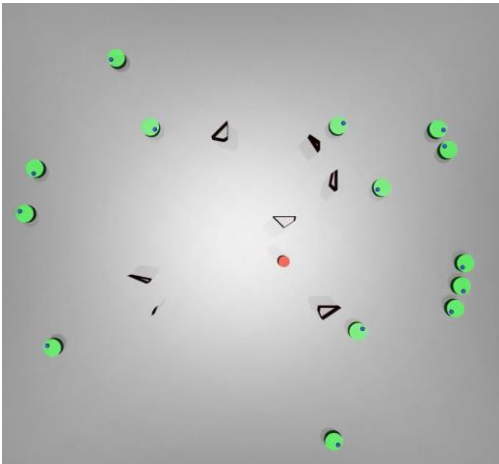
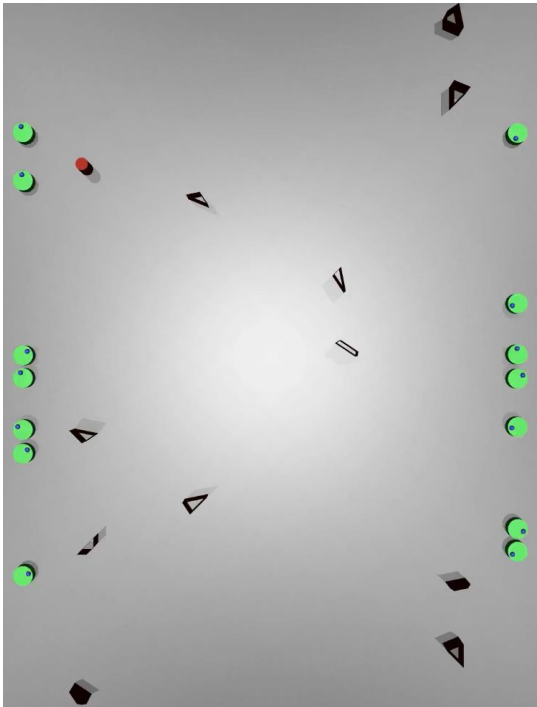
场景号	机器人	小障碍	大障碍	目标点	示意图
2-0	14	8	0	circle	
2-1	14	12	0	line	

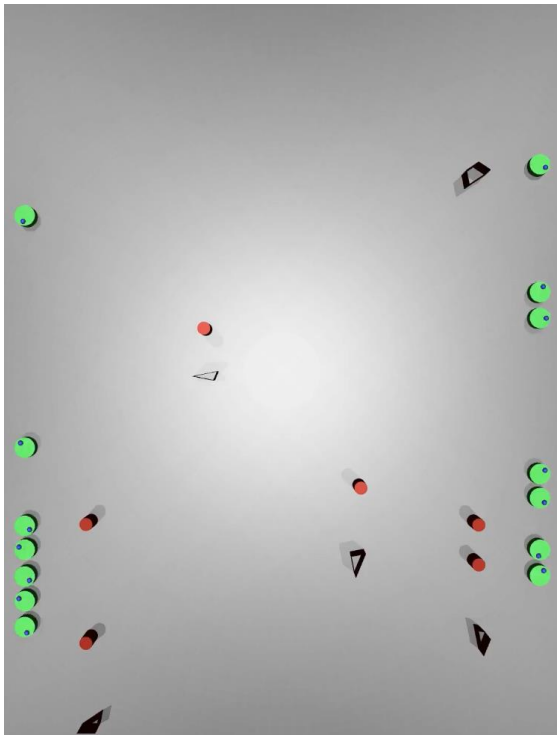
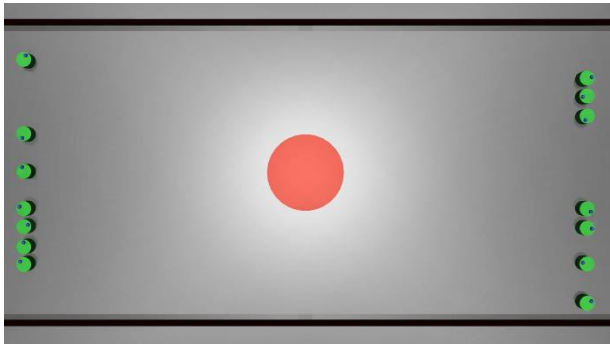
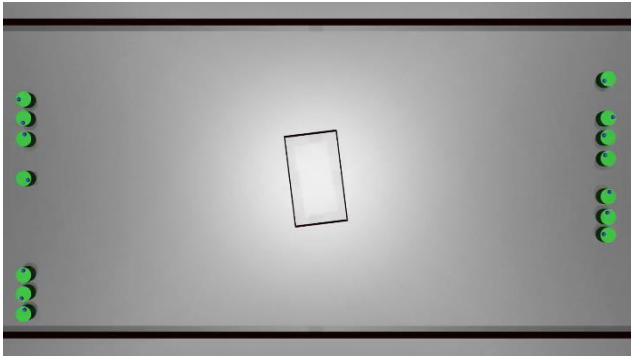
2-2	4	2	0	circle	
2-3	14	0	4	line	
2-4	14	0	3	line	

5.4.2 对照场景

对照场景中的小障碍物包含圆柱体和三棱柱、四棱柱，其中四棱柱不在训练场景中出现。对照场景与训练场景结构基本一致，机器人数量、障碍物数量也基本一致，主要检验在场景整体结构不变的情况下，模型对不同形状障碍物的适应能力。

表 5-2 对照场景说明

场景号	机器人	小障碍	大障碍	目标点	示意图
5-0	14	8	0	circle	
5-1	14	12	0	line	

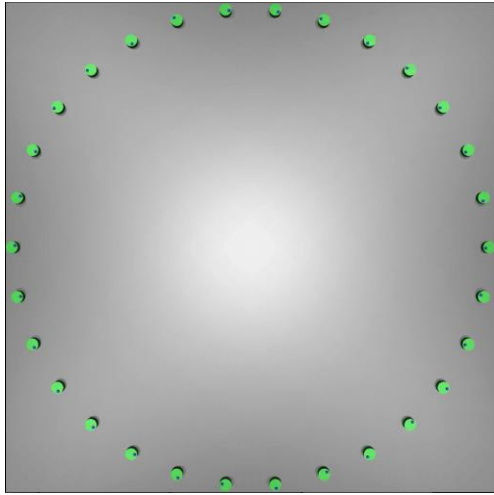
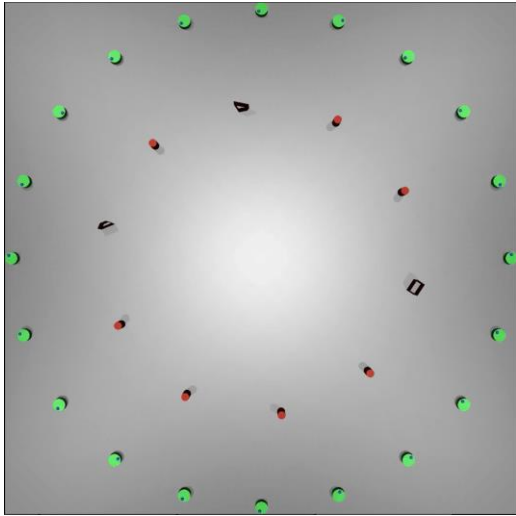
5-2	14	12	0	circle	
5-3	14	0	3	line	
5-4	14	0	3	line	

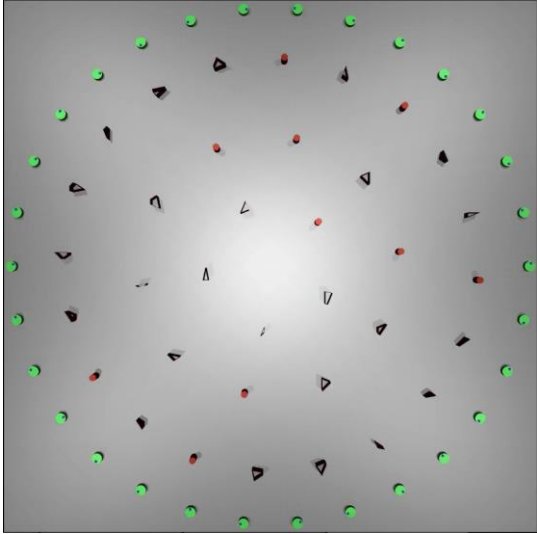
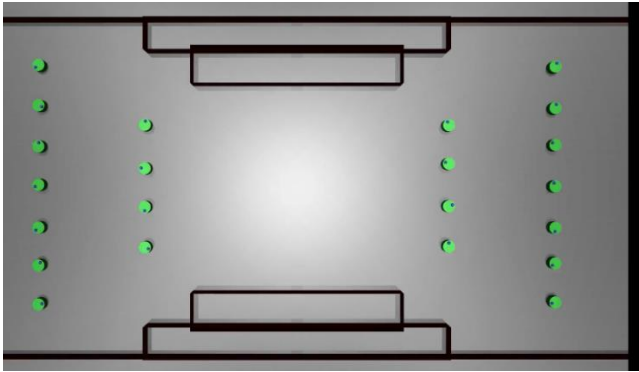
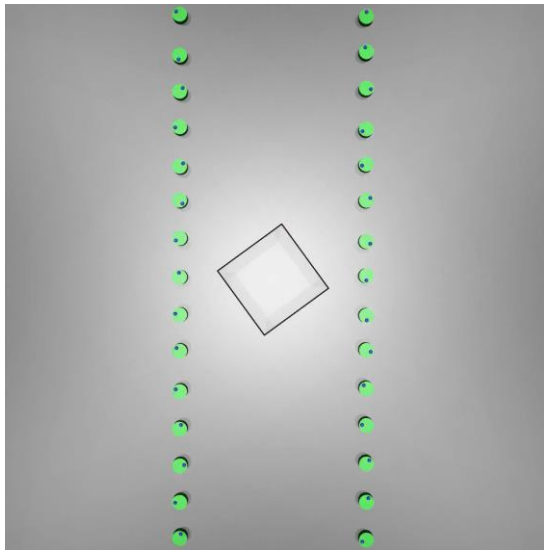
5.4.3 差异场景

差异场景中的小障碍物包含圆柱体和三棱柱、四棱柱，其中四棱柱不在训练

场景中出现。差异场景与训练场景存在结构不一致或机器人数量、障碍物数量显著提高，主要检验模型对差异巨大的场景的泛化能力。

表 5-3 差异场景说明

场 景 号	机 器 人	小 障 碍	大 障 碍	目 标 点	示意图
6-0	30	0	0	circle	
6-1	20	10	0	circle	

6-2	30	35	0	line	
6-3	22	0	6	line	
6-4	30	0	1	spin	

5.5 指标表现

5.5.1 训练场景中的指标表现

训练场景上的性能表现是分析其他场景和泛化性能的基准，各性能指标由 100 轮随机试验平均而来。

表 5-4 训练场景测试结果

指标	场景 2-0	场景 2-1	场景 2-2	场景 2-3	场景 2-4
成功率	0.9821	0.99	1.0	0.9893	0.9914
到达率	0.9993	0.9971	1.0	0.9993	1.0
平均速度	0.8381	0.8951	0.8475	0.8767	0.8791
多余路程	1.257	1.052	1.059	1.040	1.030

5.5.2 对照场景中的指标表现

对照场景加入四棱柱障碍，主要检验在整体结构不变的情况下，模型对不同形状障碍物的泛化能力，各性能指标由 100 轮随机试验平均而来。

表 5-5 对照场景测试结果

指标	场景 5-0	场景 5-1	场景 5-2	场景 5-3	场景 5-4
成功率	0.9899	0.9893	0.9764	0.9979	0.9879
到达率	0.9979	0.9986	0.9943	1.0	0.9993
平均速度	0.8384	0.8945	0.8425	0.8821	0.8798
多余路程	1.258	1.053	1.141	1.031	1.032

5.5.3 差异场景中的指标表现

差异场景与训练场景存在极大差异，主要检验模型对大差异环境的泛化能力，各性能指标由 100 轮随机试验平均而来。

表 5-6 差异场景测试结果

指标	场景 6-0	场景 6-1	场景 6-2	场景 6-3	场景 6-4
成功率	1.0	0.9915	0.9950	0.9995	0.9903
到达率	1.0	0.9940	0.9970	1.0	0.9997
平均速度	0.8136	0.8288	0.8505	0.8838	0.8267
多余路程	1.165	1.098	1.022	1.031	1.049

5.5.4 跨论文横向对比

对比论文来自 IEEE ROBOTICS AND AUTOMATION LETTERS，发表日期为 2022 年七月，标题为 Reinforcement Learned Distributed Multi-Robot Navigation With Reciprocal Velocity Obstacle Shaped Rewards^[1]，此论文中的算法被命名为 RL-RVO。

RL-RVO 论文的训练场景为圆形场景，使用一阶积分器作为仿真环境，神经网络输出与本课题一致，均为速度指令，但没有动力学约束，速度指令立即生效，积分后得到位置。

三个测试场景分别具有 10、16、20 个机器人，不具有静态障碍物，见图 6-1，机器人最大速度 1.5m/s，大于本课题中的 1m/s。

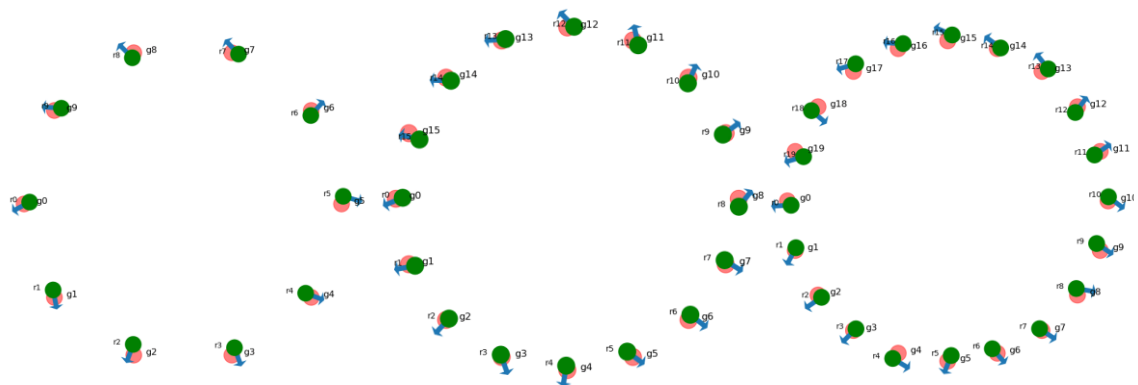


图5-4 RL-RVO使用的场景

RL-RVO 论文中给出的实验结果为

表 5-7 RL-RVO的测试结果

指标	10 机器人场景	16 机器人场景	20 机器人场景
成功率	0.99	0.93	0.9
平均速度	0.98	0.83	0.76
等效平均速度	0.653	0.553	0.507

本课题场景 6-0 中有 30 个机器人、同样没有障碍物，同样为圆形位置互换，难度显然大于 RL-RVO 论文中所有测试场景，成功率 1.0，平均速度 0.8136 m/s，具有明显优势。

本课题场景 6-1 中有 20 个机器人、10 个障碍物，同样为圆形位置互换，难度显然大于 RL-RVO 论文中所有机器人场景，成功率 0.9915，平均速度 0.8288 m/s，具有明显优势。

相比 RL-RVO 论文中的场景，本课题所使用的场景机器人更多，且具有障碍物，并且动力学更加复杂，速度指令成为实际速度的延时显著提高。在明显难度更高的环境中，本课题所用模型依然有比 RL-RVO 更好的成功率和平均速度。

在 RL-RVO 论文中，作者还对比了 SARL 算法、GA3C-CADRL 算法和 NH-ORCA 算法，实验结果如下：

表 5-8 SARL、GA3C-CADRL、NH-ORCA的测试结果

指标	10 机器人场景	16 机器人场景	20 机器人场景
成功率	0.89\0.82\1.0	0.76\0.61\0.89	0.71\0.34\0.80
平均速度	0.92\0.97\0.94	0.90\0.86\0.72	0.80\0.75\0.70
等效平均速度	0.613\0.647\0.627	0.60\0.573\0.48	0.533\0.50\0.467

在明显难度更高的环境中，本课题所用模型依然有比 SARL 算法、GA3C-CADRL 算法和 NH-ORCA 算法都更好的成功率和平均速度。

从计算资源要求角度，RL-RVO 算法使用 BiGRU 循环神经网络，对算力要求较高，SARL 算法、GA3C-CADRL 算法比 RL-RVO 算法的算力要求还要高一个数量级，而本课题采用的模型为简单的多层感知机，即 ReLU 函数激活的全连接网络，模型参数为 3.3M，并行度高，算力要求低。

5.5.5 控制频率的影响

本课题采用的模型为简单的多层感知机，模型参数为 3.3M，并行度高，算力要求低，在搭载 Intel® Core™ i7-9750H CPU @ 2.60GHz 和 GeForce GTX 1650 Mobile / Max-Q 的笔记本电脑上，使用 GPU，模型运行频率可高达 2300Hz，显存占用约 700MB，使用 CPU，模型运行频率依然可达 1800Hz，内存占用约 250MB。在性能更低，对功耗要求更高的小型机载电脑上，运行频率往往有严格限制，以下使用 25Hz 和 10Hz 控制频率对模型进行评估，检验算法的实时性，数据来自 100 次随机试验的平均。

表 5-9 不同控制频率测试结果

指标	场景 6-0	场景 6-1	场景 6-2	场景 6-3	场景 6-4
成功率	1.0\0.9697	0.992\0.996	0.995\0.992	1.0\1.0	0.990\0.993
到达率	1.0\0.970	0.994\0.997	0.997\0.992	1.0\1.0	1.0\0.999
平均速度	0.814\0.694	0.829\0.724	0.851\0.766	0.884\0.794	0.827\0.728
多余路程	1.165\1.134	1.098\1.094	1.022\1.018	1.031\1.038	1.049\1.048

可以看到，将控制频率从 25Hz 下调至 10Hz 基本未对模型的成功率和到达率构成影响，也并未改变多余路程，对机器人整体轨迹没有明显改变，但平均速度略有下降。

平均速度的下降来主要源于速度指令间的变化更大。在更小的控制频率下，控制间隔变长，速度指令间的变化更大，底层的速度指令跟踪算法将更多的驱动轮转速用于差速转向，而驱动轮转速被速度指令跟踪算法限幅以保证与现实场景的吻合，因此能够提供给直线行驶的驱动轮速度减小，导致平均速度下降。

5.5.6 机器人动力学的影响

通过改变机器人的动力学响应曲线，可以对比模型对不同动力学的泛化能力。见图 6-2 为线速度的阶跃响应和角速度的阶跃响应，可以看到 mujoco 使用的复杂接触学模型带来了强非线性，当线速度和角速度同时响应时，动力学模型将非常复杂且具有强非线性。

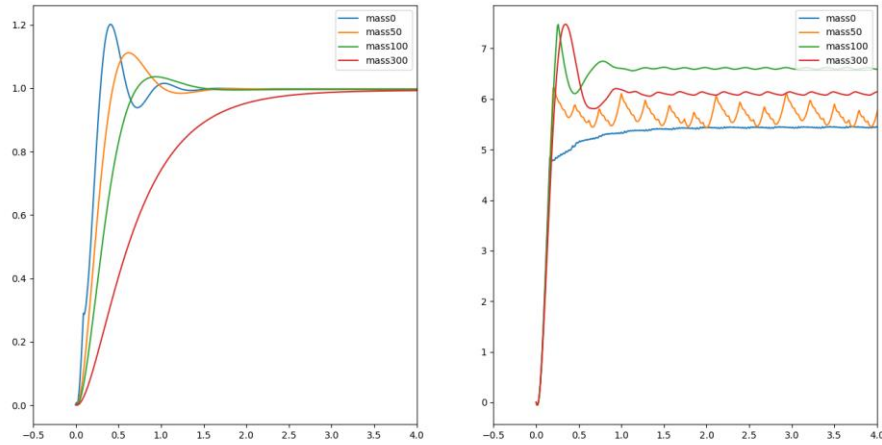


图5-5 四种不同动力学的线速度和角速度响应曲线

以下在场景 6-3 和 6-4 中测试不同动力学对应的性能指标。

表 5-10 不同动力学测试结果

指标	mass300	mass100	mass50	mass0
成功率	0.993\0.99	0.978\0.987	1.0\0.991	1.0\0.990
到达率	0.993\0.992	0.978\0.994	1.0\1.0	1.0\1.0
平均速度	0.7127\0.6571	0.6897\0.6035	0.8368\0.7813	0.8838
多余路程	1.028\1.062	1.031\1.057	1.036\1.064	1.031

可以看到，模型对不同动力学具有较好泛化能力。

5.6 本章小结

本章阐述了强化学习训练的实现，通过 SAC 算法配合多级训练和学习率衰减，获得稳定的训练过程，得到表现良好的训练结果。

本章通过大量实验和对比，阐述了模型的性能指标与泛化能力。在训练场景、对照场景、差异场景共 15 个不同场景中，关键性能指标均有优秀表现，在与其他论文的横向对比中，本课题所用算法相对与其他四个算法均有显著优势。通过 25Hz 和 10Hz 控制频率的对比实验，验证了算法的实时性。在不同机器人动力学对比试验中，验证了模型对不同底层动力学变化的适应能力。

结 论

本文提出了以 RVO_p 为核心的障碍物表示方法，对障碍物的形状、速度与意图具有良好且简明的刻画能力；以速度为核心的强化学习奖励函数，能够良好的指导强化学习训练过程，对多机避障任务具有良好的刻画能力。本文设计的多级训练过程，能够良好的由易到难地推进训练过程。本文将 off-policy 的 SAC 算法引入多机避障，替代其他论文中使用的 on-policy 的 PPO 算法，得到了良好的结果。在 15 个场景实验和横向论文对比实验中，本文的算法有良好的性能表现，相较于 RL-RVO 算法、SARL 算法、GA3C-CADRL 算法和 NH-ORCA 算法具有显著优势。通过不同控制频率和机器人动力学模型的对比测试，验证了算法的实时性和鲁棒性。

在本文的基础上，可再进行多项进一步研究。本文设计的 BiGRU 网络计算效率低，训练难度大，非 BiGRU 网络计算效率高但对输入的处理又略显笨拙，通过更巧妙的设计或许可以有更加高效简洁的解决方案。基于 mujoco 的强化学习环境实现相较于基于简单积分器的实现具有易拓展的特性，可方便的加入各式传感器，从而实现端到端算法的探索。

参考文献

- [1] R. Han, Shengduo Chen, Shuaijun Wang, Zeqing Zhang, Rui Gao, Qi Hao, Jia Pan. Reinforcement Learned Distributed Multi-Robot Navigation With Reciprocal Velocity Obstacle Shaped Rewards[J]. in IEEE Robotics and Automation Letters, vol. 7, no. 3, pp. 5896-5903, July 2022, doi: 10.1109/LRA.2022.3161699.
- [2] S. Kousik, S. Vaskov, F. Bu, M. Johnson-Roberson, and R. Vasudevan. Bridging the gap between safety and real-time performance in receding horizon trajectory design for mobile robots[C]. Int. J. Robot. Res., vol. 39, no. 12, pp. 1419–1469, 2020
- [3] Y. Yongjie and Z. Yan. Collision avoidance planning in multi-robot based on improved artificial potential field and rules[C]. in Proc. IEEE Int. Conf. Robot. Biomimetics , 2009, pp. 1026–1031
- [4] M. Guerra, D. Efimov, G. Zheng, and W. Perruquetti. Avoiding local minima in the potential field method using input-to-state stability. Control Eng[C]. Pract., vol. 55, pp. 174–184, 2016.
- [5] D. Zhou, Z. Wang, S. Bandyopadhyay, and M. Schwager. Fast, on-line collision avoidance for dynamic vehicles using buffered voronoi cells [J]. IEEE Robot. Autom. Lett., vol. 2, no. 2, pp. 1047–1054, Apr. 2017.
- [6] H. Zhu and J. Alonso-Mora. B-uavc: Buffered uncertainty-aware voronoi cells for probabilistic multi-robot collision avoidance[C]. in Proc. Int. Symp. Multi. Robot. Multi. Agent. Syst., 2019, pp. 162–168.
- [7] L. Wang, A. D. Ames, and M. Egerstedt. Safety barrier certificates for collisions-free multirobot systems[J]. IEEE Trans. Robot., vol. 33, no. 3, pp. 661–674, Jun. 2017.
- [8] D. Bareiss and J. van den Berg. Generalized reciprocal collision avoidance[J]. The International Journal of Robotics Research, vol. 34, no. 12, pp. 1501–1514, 2015.
- [9] Alonso-Mora, A. Breitenmoser, M. Rufli, P. Beardsley, and R. Siegwart. Optimal reciprocal collision avoidance for multiple non-holonomic robots[J]. in Distributed Autonomous Robotic Systems. Springer, 2013, pp. 203–216.
- [10] R. S. Sutton, D. McAllester, S. Singh, Y. Mansour. Policy Gradient Methods for Reinforcement Learning with Function Approximation[J]. In Neural Information Processing Systems (NIPS), 1999.
- [11] J. Schulman, S. Levine, P. Moritz, M. Jordan, P. Abbeel. Trust Region Policy

- Optimization[J]. arXiv preprint arXiv:1502.05477v5, 2017.
- [12] J. Schulman, F. Wolski, P. Dhariwal, A. Radford, O. Klimov. Proximal Policy Optimization Algorithms[J]. arXiv preprint arXiv:1707.06347v2, 2017.
 - [13] V. Mnih, K. Kavukcuoglu, D. Silver, A. A. Rusu, J. Veness, M. G. Bellemare, A. Graves, M. Riedmiller, A. K. Fidjeland, G. Ostrovski, S. Petersen, C. Beattie, A. Sadik, I. Antonoglou, H. King, D. Kumaran, D. Wierstra, S. Legg, D. Hassabis. Human-level control through deep reinforcement learning[J]. In Nature: doi:10.1038/nature14236, (2015).
 - [14] H. v. Hasselt, A. Guez, D. Silver. Deep Reinforcement Learning with Double Q-learning[J]. arXiv preprint arXiv:1509.06461v3, 2015.
 - [15] T. Schaul, J. Quan, I. Antonoglou, D. Silver. Prioritized Experience Replay[J]. arXiv preprint arXiv:1511.05952v4, 2016.
 - [16] D. Silver, G. Lever, N. Heess, T. Degris, D. Wierstra, M. Riedmiller. Deterministic Policy Gradient Algorithms[C]. In International Conference on Machine Learning (ICML), 2014.
 - [17] T. P. Lillicrap, J. J. Hunt, A. Pritzel, N. Heess, T. Erez, Y. Tassa, D. Silver, D. Wierstra. Continuous Control With Deep Reinforcement Learning[J]. arXiv preprint arXiv:1509.02971v6, 2019.
 - [18] S. Fujimoto, H. v. Hoof, D. Meger. Addressing Function Approximation Error in Actor-Critic Methods[J]. arXiv preprint arXiv:1802.09477v3, 2018.
 - [19] T. Haarnoja, A. Zhou, P. Abbeel, S. Levine. Soft Actor-Critic: Off-Policy Maximum Entropy Deep Reinforcement Learning with a Stochastic Actor[J]. arXiv preprint arXiv:1801.01290v2, 2018.
 - [20] T. Haarnoja, A. Zhou, K. Hartikainen, G. Tucker, S. Ha, J. Tan, V. Kumar, H. Zhu, A. Gupta, P. Abbeel, S. Levine. Soft Actor-Critic Algorithms and Applications[J]. arXiv preprint arXiv:1812.05905v, 2019.
 - [21] T. Haarnoja, S. Ha, A. Zhou, J. Tan, G. Tucker, S. Levine. Learning to Walk via Deep Reinforcement Learning[J]. arXiv preprint arXiv:1812.11103v3, 2019.
 - [22] J. A. Arjona-Medina, M. Gillhofer, M. Widrich, T. Unterthiner, J. Brandstetter, S. Hochreiter. RUDDER: Return Decomposition for Delayed Rewards[J]. arXiv preprint arXiv:1806.07857v3, 2019.
 - [23] R. Ourari, K. Cui, A. Elshamhory, H. Koepl. Nearest-Neighbor-based Collision Avoidance for Quadrotors via Reinforcement Learning[C]. In IEEE International Conference on Robotics and Automation (ICRA) : DOI: 10.1109/ICRA46639.2022.9812221. (2022)
 - [24] C. C. Bai, P. Yan, W. Pan, J. F. Guo. Learning-Based Multi-Robot Formation

- Control With Obstacle Avoidance[C]. In IEEE Transactions On Intelligent Transportation Systems, Vol. 23, No. 8, August 2022.
- [25] H. Zhu, F. M. Claramunt, B. Brito, J. Alonso-Mora. Learning Interaction-Aware Trajectory Predictions for Decentralized Multi-Robot Motion Planning in Dynamic Environments[C]. In IEEE ROBOTICS AND AUTOMATION LETTERS, VOL. 6, NO. 2, APRIL 2021.
- [26] Q.Y. Tan, T. X. Fan, J. Pan, D. Manocha. DeepMNavigate: Deep Reinforced Multi-Robot Navigation Unifying Local & Global Collision Avoidance[C]. In 2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS): DOI: 10.1109/IROS45743.2020.9341805, (2020)
- [27] P. X. Long, T. X. Fan, X. Y. Liao, W. X. Liu, H. Zhang, J. Pan. Towards Optimally Decentralized Multi-Robot Collision Avoidance via Deep Reinforcement Learning [C]. In 2018 IEEE International Conference on Robotics and Automation (ICRA), (2018).
- [28] T. X. Fan, P. X. Long, W. X. Liu, J. Pan. Fully Distributed Multi-Robot Collision Avoidance via Deep Reinforcement Learning for Safe and Efficient Navigation in Complex Scenarios[J]. arXiv preprint arXiv:1808.03841v1, (2018).
- [29] R. Lowe, Y. Wu, A. Tamar, J. Harb, P. Abbeel, I. Mordatch. Multi-Agent Actor-Critic for Mixed Cooperative-Competitive Environments[J]. arXiv preprint arXiv:1706.02275v4, (2020).
- [30] F. A. Oliehoek, M. T. J. Spaan, & N. Vlassis (2008). Optimal and Approximate Q-value Functions for Decentralized POMDPs[J]. Journal of Artificial Intelligence Research, 32, 289–353.
- [31] J. van den Berg, Ming Lin and D. Manocha. Reciprocal Velocity Obstacles for real-time multi-agent navigation[C]. 2008 IEEE International Conference on Robotics and Automation, Pasadena, CA, 2008, pp. 1928-1935, doi: 10.1109/ROBOT.2008.4543489.
- [32] J. Snape, J. v. d. Berg, S. J. Guy and D. Manocha. The Hybrid Reciprocal Velocity Obstacle[C]. in IEEE Transactions on Robotics, vol. 27, no. 4, pp. 696-706, Aug. 2011, doi: 10.1109/TRO.2011.2120810.
- [33] J. Snape, J. van den Berg, S. J. Guy and D. Manocha. Dependent navigation of multiple mobile robots with hybrid reciprocal velocity obstacles[C]. 2009 IEEE/RSJ International Conference on Intelligent Robots and Systems, St. Louis, MO, USA, 2009, pp. 5917-5922, doi: 10.1109/IROS.2009.5354821.
- [34] Van Den Berg, J., Guy, S. J., Lin, M., & Manocha, D. (2011). Reciprocal n-body collision avoidance[C]. In Robotics Research - The 14th International

- Symposium ISRR (STAR ed., pp. 3-19). (Springer Tracts in Advanced Robotics; Vol. 70, No. STAR). https://doi.org/10.1007/978-3-642-19457-3_1.
- [35] Jamie Snape, Jur van den Berg, Stephen J. Guy, and Dinesh Manocha. Smooth and Collision-Free Navigation for Multiple Robots Under Differential-Drive Constraints[C]. IEEE RSJ Int. Conf. Intelligent Robots and Systems, Taipei, Taiwan, 2010.
- [36] Chung, J., Gulcehre, C., Cho, K. & Bengio, Y. (2014). Empirical Evaluation of Gated Recurrent Neural Networks on Sequence Modeling[C]. (cite arxiv:1412.3555Comment: Presented in NIPS 2014 Deep Learning and Representation Learning Workshop).

哈尔滨工业大学本科毕业论文（设计）

原创性声明和使用权限

本科毕业论文（设计）原创性声明

本人郑重声明：此处所提交的本科毕业论文（设计）《基于强化学习的多移动机器人分布式避障导航》，是本人在导师指导下，在哈尔滨工业大学攻读学士学位期间独立进行研究工作所取得的成果，且毕业论文（设计）中除已标注引用文献的部分外不包含他人完成或已发表的研究成果。对本毕业论文（设计）的研究工作做出重要贡献的个人和集体，均已在文中以明确方式注明。

作者签名： 日期： 年 月 日

本科毕业论文（设计）使用权限

本科毕业论文（设计）是本科生在哈尔滨工业大学攻读学士学位期间完成的成果，知识产权归属哈尔滨工业大学。本科毕业论文（设计）的使用权限如下：

（1）学校可以采用影印、缩印或其他复制手段保存本科生上交的毕业论文（设计），并向有关部门报送本科毕业论文（设计）；（2）根据需要，学校可以将本科毕业论文（设计）部分或全部内容编入有关数据库进行检索和提供相应阅览服务；（3）本科生毕业后发表与此毕业论文（设计）研究成果相关的学术论文和其他成果时，应征得导师同意，且第一署名单位为哈尔滨工业大学。

保密论文在保密期内遵守有关保密规定，解密后适用于此使用权限规定。
本人知悉本科毕业论文（设计）的使用权限，并将遵守有关规定。

作者签名： 日期： 年 月 日

导师签名： 日期： 年 月 日

致 谢

在此，我要向我的导师梅杰教授表达我最衷心的感谢。他对本人的悉心指导、答疑解惑使我得以完成本论文。他在治学态度和思想品德上，更是给予了我深刻的教育和影响。他严谨而不失亲切，博学而不失谦逊，创新而不失务实。他的言传身教将使我终身受益。

感谢我的母校哈尔滨工业大学，尤其是机电工程与自动化学院所有的老师们。他们用严谨的治学态度和精湛的专业知识，为我打下了坚实的基础。他们用热情的教学风格和亲切的师生关系，为我营造了良好的学习氛围。

在此，我要感谢我的父母，他们给予了我无私的爱和无尽的支持，让我拥有了最温暖的家庭。虽然他们总是与我相隔千里，但他们总是用理解和关爱，为我打造了最坚实的港湾。他们在我遇到困难时，给我鼓励 and 信心；在我取得成绩时，给我赞扬和骄傲；在我迷茫时，给我指引和启示。他们是最爱我的人，也是我最信赖的人。

感谢我的师兄师姐，他们用友好的合作和互相学习，为我提供了最宝贵的经验。他们用耐心的解答和及时的反馈，为我提供了最有效的帮助。他们用真诚的交流和相互尊重，为我提供了最美好的友情。

感谢我的同学朋友，他们用共同的目标和相同的兴趣，为我提供了最有趣的伙伴。他们用轻松的聊天和欢乐的笑声，为我提供了最放松的娱乐。他们用坦诚的分享和互相支持，为我提供了最温馨的陪伴。

感谢开源社区，尤其是 `pytorch`、`vscode`、`git`、`ubuntu` 等项目，他们用先进的技术和优秀的贡献者，为我提供了最强大的工具。他们用开放的理念和活跃的社区，为我提供了最广阔的平台。他们用创新的精神和协作的方式，为世界提供了最大的启发。