海上火箭发射回收船用
辅助作业行走机器人系统研制

技术研究报告

上海交通大学

2024 年 4 月 1 日

目 录

1 研究目标与研究内容 3
2 系统研制过程 5
2.1 系统流程图 5
2.2 系统模块介绍 5
3 国内外相关技术的研究现状 7
4 关键技术创新 10
4.1 实物与仿真结合的训练环境构建 12
4.2 四足机器人的运动控制 15
4.3 实时视频传输 18
4.4 单调的鲁棒策略优化算法 20
4.4.1 基于域随机化的强化学习训练仿真环境介绍 20
4.4.2 对策略与环境交互过程中最差性能和平均性能的分析 21
4.4.3 单调的鲁棒策略优化 23
4.5 域随机化中控制策略优化的方差减少研究 24
4.5.1 强化学习策略梯度算法中的基于基准的方差减少方法 24
4.5.2 域随机化中最优的依赖于环境和状态的基准 25
4.5.3 基于聚类的方差减少算法框架 25
4.5.4 算法部分实验展示 26
4.6 深度元强化学习中的双重鲁棒增强的重标记算法 27
4.6.1 基准元强化学习算法介绍 27
4.6.2 双重鲁棒保证的值估计器 28
4.6.3 双重鲁棒增强的重标记算法 28
4.7 基于 D*算法的四足机器人路径规划与避障 29
5 知识产权情况 33
5.1 学术论文列表 33
5.2 授权发明专利列表 33
5.3 申请发明专利列表 34
5.4 申请软件著作权列表 34
参考文献 35

1 研究目标与研究内容

项目研究目标为：针对发射平台和回收平台有人作业的安全性问题，开展研制一套适应卫星发射与回收平台环境，能够执行巡逻检查任务的智能辅助机器人，并集成到海上卫星发射指挥控制系统研制。辅助作业行走机器人系统采用鲁棒的深度强化学习方法，以及领域随机化技术，实现从模拟到真实环境的迁移，解决发射回收作业平台的复杂环境下，机器人的半自主、全自主的鲁棒控制问题，从而具备远程控制下半自主巡检作业、自主巡检作业能力。针对发射回收平台作业环境的主要特点，包括甲板风大且不稳定（横摇幅度大）、甲板障碍物多、地形复杂、舱室通道狭窄等，突破弱通信环境下鲁棒控制技术，形成系统，主要研究任务包括：

（1）实物与仿真结合的训练环境构建

主要拟引入鲁棒的深度强化学习方法，结合领域随机化技术实现从模拟到真实的迁移，解决不稳定的发射回收作业平台上，机器人半自主、全自主的鲁棒控制问题。拟研究发射、回收平台的甲板风大且不稳定（横摇幅度）、甲板障碍物多、地形复杂、舱室通道狭窄的问题，研究弱通信环境下的机器人控制任务。

（2）甲板及舱室机器人的鲁棒运动控制

自主导航首先要求机器人在船舱内具备诸如前进、上下楼梯等活动能力，拟研究基于强化学习算法，联合训练感知模块和控制模块，实现机器人的运动控制。针对甲板风大且存在横摇的不稳定因素，研究基于鲁棒深度强化学习的发射回收辅助机器人的鲁棒运动控制算法。

（3）甲板及舱室环境机器人有人操作下的自动避障控制

针对甲板障碍物多，在行进中易发生碰撞导致机器人跌倒的问题，开展机器人避障算法的研究，研究基于目标检测、SLAM 方法和深度强化学习的发射回收辅助机器人的辅助避障控制算法。

（4）甲板及舱室环境机器人的无人自主导航

针对弱通信环境下有人控制存在通信中断的问题，研究机器人自主导航、包括激光雷达与视觉融合地图生成、克服立体障碍的路径规划、依靠机器人自身运算能力的视觉导航，研制甲板及舱室环境机器人自主导航软件。

（5）智能辅助机器人系统集成

将前述研究内容进行系统集成，形成海上火箭发射回收船用辅助作业行走机器人原型系统一套，并集成到海上卫星发射指挥控制系统研制。

具体技术要求为：

（1）机器人可以在海风及甲板摇晃等外界干扰下实现行走功能；

（2）机器人可以在行进过程中自动躲避甲板障碍物；

（3）机器人可以根据指令控制摄像头完成检视任务。

具体技术指标包括：

（1）机器人可负载重量不小于 3 千克；

（2）抗甲板横摇幅度不小于±5°；

（3）可爬阶梯高度差不小于 20 厘米；

（4）典型发射甲板环境下避障成功率不低于 90%。

（5）采集图像视频数据可达到 4K 分辨率，并实现 20 倍变焦。

2 系统研制过程

2.1 系统流程图

本项目的系统流程图如图 2.1.1 所示。

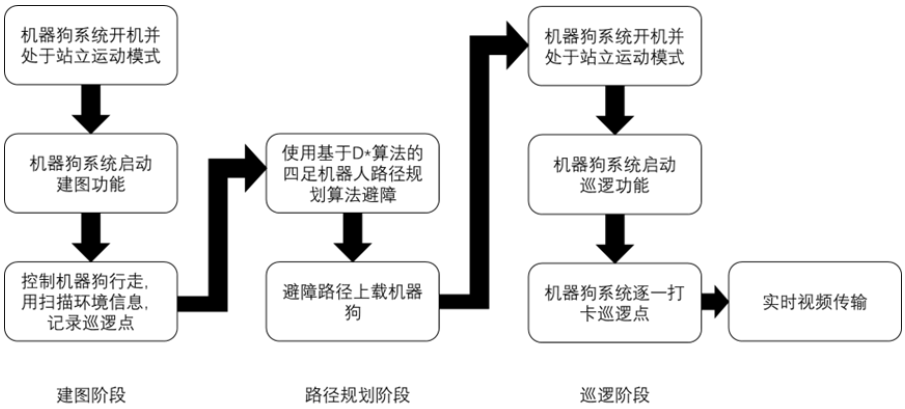


图 2.1.1 系统流程图

1）首先是建图阶段，将测试用机，宇树科技 AlienGo 型号四足机器人以平趴姿态放置于平整地面。安装好测试用机的电池，确定电池不会松动，先短按电源键，再长按电源键直至听到风扇声，此时可视为四足机器人成功开机，等待 60 秒。通过蓝牙控制手柄让机器人处于站立运动模式。将控制用机连接四足机器人发出的热点信号。运行命令开始对船上环境进行建图。此时需四足机器人开始建图时的位置、朝向。通过蓝牙控制手柄控制四足机器人走遍发射船，扫描环境信息，并记录巡逻点，进行完整建图。

2）之后是路径规划阶段，利用基于 D*算法的四足机器人路径规划算法进行避障路径规划，并将避障的路径上载四足机器人。

3）最后是巡逻阶段，将机器人开机并处于站立模式，将四足机器人恢复到建图开始时的位置、朝向。运行命令启动四足机器人的巡逻功能，四足机器人将逐一打卡巡逻点。同时，四足机器人搭载的摄像头将把拍摄的视频远程传输到控制船。

2.2 系统模块介绍

本项目系统模块图如图 2.2.1 所示。

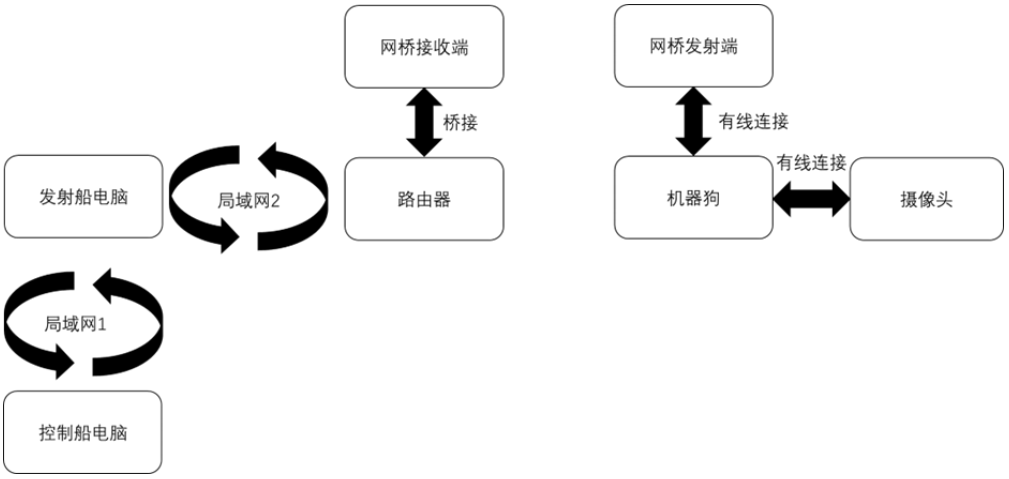


图 2.2.1 系统模块图

系统由控制船电脑、发射船电脑、网桥发射端、网桥接收端、路由器、四足机器人、摄像头组成。

系统中各个设备主要通过两个局域网通信。其中局域网 1 负责发射船电脑和控制船电脑之间的通信，局域网 2 用于发射船电脑与四足机器人和摄像头等相关设备通信。考虑到甲板空旷，为了增强信号，利用网桥传播局域网 2。

控制船电脑发出的指令，通过局域网 1 传递到发射船电脑。发射船电脑将指令发送到局域网 2，通过网桥传给甲板上巡逻的四足机器人。四足机器人做出相应的行为。此时，连接在四足机器人上的网络摄像头将拍摄到的视频通过局域网 2 传递给网桥，通过网桥传递到发射船电脑，再通过局域网 1 传递回控制船电脑，实现远程实时监控。

3 国内外相关技术的研究现状

传统的四足机器人运动控制方法基于控制理论和有限状态机。MIT Cheetah 3 型号机器人利用模型预测控制算法（model predictive control）能实现四足机器人的高速行走功能和无外部传感器情况爬楼梯功能[1-2]。ANYmal 型号机器人利用基于倒立摆模型（inverted pendulum model）的算法，能在有障碍的环境实现指定的步态[3]。强化学习通过将机器人运动控制问题建模成一个马尔可夫决策过程，合理设置该过程的奖励函数，并用深度神经网络来拟合运动控制策略函数[4-18]。近年来，随着物理引擎技术的成熟，强化学习的训练往往是在仿真器中进行，在控制策略与仿真器交互得到充足的数据后，就能通过相应的策略函数优化算法迭代控制策略，在仿真器中训练得到表现优良的机器人运动控制策略。

但深度强化学习算法容易在仿真器过拟合，一旦迁移到真实环境中或者任务场景发生较大的变化，往往性能损失非常严重。当部署到真实环境时，目前的研究主要采用系统识别、域随机化和元学习的方法来实现仿真器到真实环境的策略迁移。

系统识别（system identification）[19]的方法利用环境物理参数的信息辅助运动控制策略做出决策。当运动控制策略被运用到真实环境中时，它能通过推断模块来推测环境物理参数，或者直接使用进化算法（evolutionary algorithms）来搜索能使运动控制方法奖励函数最大的环境参数。事实上，精确预测环境物理参数是困难且非必要的，往往可以用低维度的隐变量来表示环境物理参数的信息，如图 3.1.1 所示。

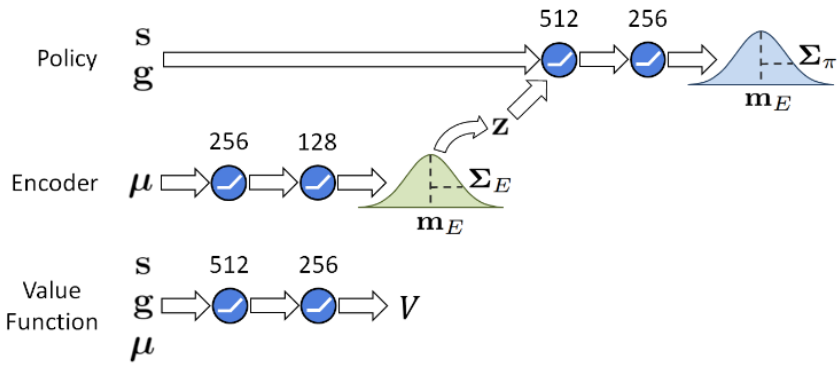


图 3.1.1 系统识别方法利用环境物理参数信息微调策略[19]

在现实场景测试运动控制策略时，需要寻找合适的隐变量来表示当前环境的特征。当前研究主要采用策略梯度下降方法（policy gradient methods）、贝叶斯优化（Bayesian optimization）或者随机搜素（random search）。

域随机化（domain randomization）[19]是另一类解决方案。这类方法在用强化学习训练时，对环境参数做了随机化，并给传感器采集到的观测值添加噪声，使得学到的运动控制策略在较大的环境参数范围内都比传统的方法更有鲁棒性。但是域随机化算法往往过于保守，不是较优策略。

另外一类流行的解决方法是元强化学习（Meta Reinforcement Learning）[20-28]。元学习（Meta Learning）是元认知的一个分支，涉及对自己的学习和学习过程的学习。传统的强化学习算法能根据一个特定的任务，给出解决该任务的策略，而元强化学习的训练阶段同时需要学习策略和自适应两个模块。元强化学习在一个任务的分布上学习一个整体的策略，和根据不同任务调整策略的自适应模块。当测试阶段遇上新的任务，甚至是训练时未见过的任务时（如在仿真器中训练完，部署到真实环境中时），元强化学习的自适应模块，通过策略与新任务的环境交互得到的数据对策略做针对性的调整。虽然元强化学习已经在真实四足机器人上取得了一定的成果，但当仿真器中训练的策略部署到真实环境上时，策略自适应模块仍需要机器人与真实环境做大量的交互采集数据来调整策略。而事实上，当运动控制策略部署到真实四足机器人上时，常常没有充足的时间来完成这些数据的采集，且如果真实环境是高低不平的岩石表面，在采集数据时很可能造成四足机器人的损伤。因此理想情况下，元强化学习的训练、自适应模块的学习都要在仿真器中完成，且自适应模块的采样效率要充分高。

路径规划算法是四足机器人控制问题中的控制或决策部分，如何在各种场景下迅速、准确的规划出一条高效路径且使其具备应对场景动态变化的能力是路径规划算法应当解决的问题。Dijkstra 算法[29]采用了一种贪心模式，其解决的是有向图中单个节点到另一节点的最短路径问题，其每次迭代时选择的下一个节点是当前节点最近的子节点，在每一次迭代过程中，都要对起始节点到所有遍历到的点之间的最短路径进行更新。A*算法[30]是启发式搜索算法，启发式搜索即在搜索过程中建立启发式搜索规则，以此来衡量实时搜索位置和目标位置的距离关系，使搜索方向优先朝向目标点所处位置的方向，最终达到提高搜索

效率的效果。D*算法[31]是一种基于 A*算法的反向增量式搜索算法。反向即算法从目标点开始向起点逐步搜索，增量式搜索即算法在搜索过程中会计算每一个节点的距离度量信息，在动态环境中若出现障碍物无法继续沿预先路径搜索，算法会根据原先已经得到的每个点的距离度量信息在当前状态点进行路径再规划，无需从目标点进行重新规划。本项目在传统 D*的基础上开发了适用于四足机器人的路径规划与避障算法。

4 关键技术创新

本项目的主要技术要点包括：

1) 实物与仿真结合的训练环境构建。控制策略算法的训练可能存在导致机器人损坏的情况，因此在仿真器中进行训练是一种安全的做法。本项目利用物理仿真引擎 Pybullet 实现了四足机器人运动模拟器构建，模拟甲板及舱室环境下的载荷四足机器人。并基于深度强化学习 Proximal Policy Optimization 算法训练仿生四足机器人的运动控制策略。

2) 四足机器人的运动控制。通过 Raibert 启发落脚点理论估计四足机器人运动时足端的运动轨迹；简化四足机器人力学模型，列写、求解了四足机器人运动的动力学方程；完成了通过控制关节力矩控制四足机器人运动的框架。

3) 四足机器人巡视过程中的实时视频传输。根据视频质量需求，选择了能够 20 倍变焦、4K 分辨率、且使用 H.264 编码的 SH-HK590A 型号网络摄像头。并利用 RSTP 协议拉取视频流，完成了一个局域网下的远程视频传输系统。

4) 单调的鲁棒策略优化算法。由于源训练环境和目标部署环境存在动态模型差异，现有的深度强化学习算法倾向于在训练环境上过拟合。尽管在训练阶段应用领域随机化技术，在仿真器中随机化参数生成的足够多样性的环境集合上优化策略的期望性能可以提升策略的平均性能，最差环境性能在优化过程中仍然被忽略且没有任何性能保证。因为平均性能和最差性能对于强化学习泛化同样重要，在本项目中，我们提出策略优化方法来同时提升策略的平均和最差性能。我们理论上推导出了任一策略的与期望性能相关的最差性能的下界。基于这个下界，我们构建了优化问题来联合优化策略和采样分布，同时证明了通过迭代解决优化问题，最差性能能够单调提升。然后我们设计出一个实用的算法，名叫单调的鲁棒策略优化（MRPO）。在多个机器人控制任务上的实验评估结果证实了 MRPO 大体上能够在源训练环境上同时提升平均和最差性能，并且在一些训练中不可见的测试环境中，使学到的策略具有更好的泛化能力。

6) 域随机化中控制策略优化的方差减少研究。通过在环境集合上引入随机性，域随机化技术为深度强化学习训练引入了足够丰富的多样性，因此提升了基于强化学习控制策略的泛化能力。然而引入域随机化的训练方式，为策略优化过程中策略梯度的估计带来了额外的样本随机性，加剧了其本身由于样本采样而梯

度估计方差就很大的问题。而使用传统策略梯度方法中基于依赖状态的基准减少梯度方差的方法，无法解决这额外的方差，因此在域随机化的训练过程中造成算法采样效率低下。在本项目中，在域随机化我们理论上推导了无偏的依赖状态和环境的最优基准，在理论上分析了其相较于固定基准和仅依赖于状态的基准所能获得的方差减少优势。基于该理论，为了权衡方差减少和计算复杂度，我们将任务环境参数空间划分为子空间，并提出了方差减少的域随机化方法（VRDR），其具有理论上方差减少和更快收敛的保证。我们在实验上验证了它在训练过程中具有更小的策略梯度方差以及能够加速策略训练过程中的收敛。

7) 深度元强化学习中的双重鲁棒保证的重标记算法。元强化学习通过利用不同任务之间共享的潜在结构，使强化学习智能体能够快速适应新任务，通常仅需从新任务上获得少量采样就能到新的控制策略。然而，当前元强化学习方法的训练阶段仍然需要大量的样本来获得良好的自适应性能，其中来自不同任务的样本被简单集成用来训练是主要原因之一。为了通过提高元强化学习训练过程中的采样效率，我们在本项目中提出了一种双重鲁棒增强的重标记方法（DRR），该方法可以结合任何基于价值函数的元强化学习算法中，在元训练阶段通过 DRR 可以在任务之间正确地迁移样本。分析表明，所提出的方法在双重鲁棒保证下是无偏的，并且与传统的基于重要性抽样的估计器相比，其方差更低。考虑到目标值的方差会加剧强化学习中的高估误差（或值近似误差），我们还从理论上推导了所提出的 DRR 方法中值估计器的方差和达到最小方差的最优动力学重要性权重。然后，我们受到启发，将重要性权重缩小到其最优值附近，以进一步减少方差。在与几个元强化学习的基准测试算法的对比实验中，我们证明 DRR 可以加速训练过程并实现更好的渐近性能。

8) 基于 D*算法的四足机器人路径规划与避障。D*是一种可以面对未知环境或环境存在动态变化的情况下进行启发式路径搜索的算法。然而 D*算法规划路径的主要依据是从终点向起点传播，带有一定的方向性，且规划过程中将机器人视为一点，因此在四足机器人的路径规划中极易出现碰撞。本项目在传统 D*的基础上开发了适用于四足机器人的路径规划与避障算法。

4.1 实物与仿真结合的训练环境构建

为了训练出能在甲板及舱室环境下行走的四足机器人，需要先在仿真器中训练，如果直接在真实环境中训练容易损坏四足机器人，因此需要搭建能模拟甲板和舱室环境的仿真环境。

利用两种广泛使用的名为 mujoco 和 pybullet 的仿真物理引擎分别搭建了四足机器人的仿真训练平台。其中，基于名为 mujoco 物理引擎的仿真平台是通过继承 python 的 gym.envs.mujoco 工具包中名为 mujoco_env 的基类，并利用该基类的 load_model_from_path 函数导入宇树科技提供的四足机器人模型文件来实现的。而基于名为 pybullet 物理引擎的仿真平台是通过直接调用 python 中名为 pybullet 的工具包中的 loadURDF 函数导入四足机器人模型文件来实现。

综合考虑各个物理引擎的性能，pybullet 在如图 4.1.1 所示的各项指标下综合评分较高，库函数齐全，且开源更早。因此后续仿真平台工作基于名为 pybullet 的物理引擎。

Simulation Environment	Reproducible	Parallel	Photorealistic Rendering	Accelerated	Modular	GPU-Accelerated Simulation	Open Source	Total Score
ROS & Gazebo				✓	✓		✓	1.55
CoppeliaSim	✓			✓				2.0
Pybullet	✓	✓		✓			✓	3.4
MuJoCo	✓	✓		✓				3.0
RaiSim	✓	✓	✓	✓				3.8
IsaacGym	✓	✓	✓	✓		✓		**4.8**
Surreal	✓	✓		✓				3.0
Unity ML	✓	✓	✓	✓				3.8

图 4.1.1 物理引擎性能对比

仿真平台具体的实现框架如图 4.1.2 所示。

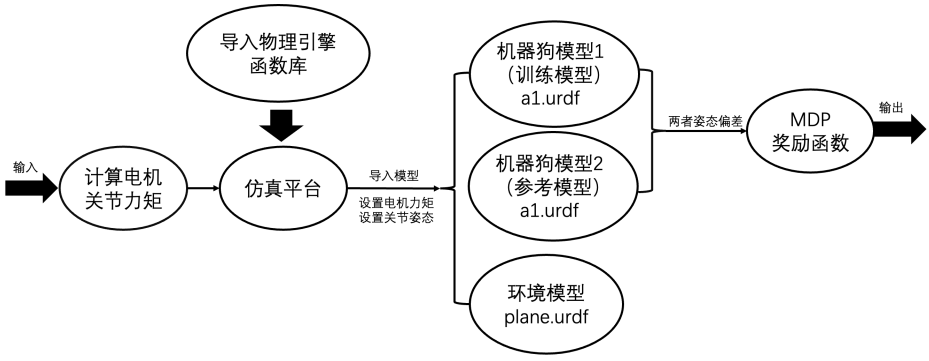


图 4.1.2 仿真平台实现框架图

四足机器人的导入和关节电机控制都通过物理引擎的库函数实现。直接建模

甲板和舱室环境比较困难，需要用较简单的模型来模拟。考虑到船舱往往存在晃动，因此使用晃动的平面来模拟。甲板高低不平，因此使用坡度较小的斜面来模拟。晃动的平面和斜面的仿真都可以通过修改环境模型文件来实现。

为了使用深度强化学习训练仿生四足机器人的运动控制策略，仿真平台还需能计算强化学习奖励函数。强化学习将问题建模成马尔可夫决策过程。

$$M = \{S, A, P, R, \gamma\}$$

这里 S 是状态空间，A 是行为空间，P 是环境状态转移概率，R 是奖励函数，γ 是计算累计奖励函数的参数）。

强化学习的目标是在马尔可夫决策过程下，得到最大化累计奖励函数的策略。

$$\pi^*(s) = \arg\max_{\pi} E\left[\sum_{h=0}^{\infty} \gamma^h r(s_h, a_h) \mid s_0 = s, a_h \sim \pi(s_h), s_{h+1} = P(s_h, a_h)\right]$$

其中奖励函数 R 对于强化学习的训练起到重要的效果，决定了训练出来的策略能完成的任务。本课题为了模仿参考策略中狗的行为完成向前行走的任务，将参考策略表现与真实策略表现的差（狗关节的位置、速度，脚的位置之差）作为马尔可夫决策过程中奖励函数的正则项。

$$r_t = w^p r_t^p + w^v r_t^v + w^e r_t^e + w^{rp} r_t^{rp} + w^{rv} r_t^{rv}$$

$$r_t^p = \exp\left[-5 \sum_j \left\| \hat{q}_t^j - q_t^j \right\|^2\right]$$

$$r_t^v = \exp\left[-0.1 \sum_j \left\| \hat{\dot{q}}_t^j - \dot{q}_t^j \right\|^2\right]$$

$$r_t^e = \exp\left[-40 \sum_j \left\| \hat{x}_t^e - x_t^e \right\|^2\right]$$

$$r_t^{rp} = \exp\left[-20 \left\| \hat{x}_t^{root} - x_t^{root} \right\|^2 - 10 \left\| \hat{q}_t^{root} - q_t^{root} \right\|^2\right]$$

$$r_t^{rv} = \exp\left[-2 \left\| \hat{\dot{x}}_t^{root} - \dot{x}_t^{root} \right\|^2 - 0.2 \left\| \hat{\dot{q}}_t^{root} - \dot{q}_t^{root} \right\|^2\right]$$

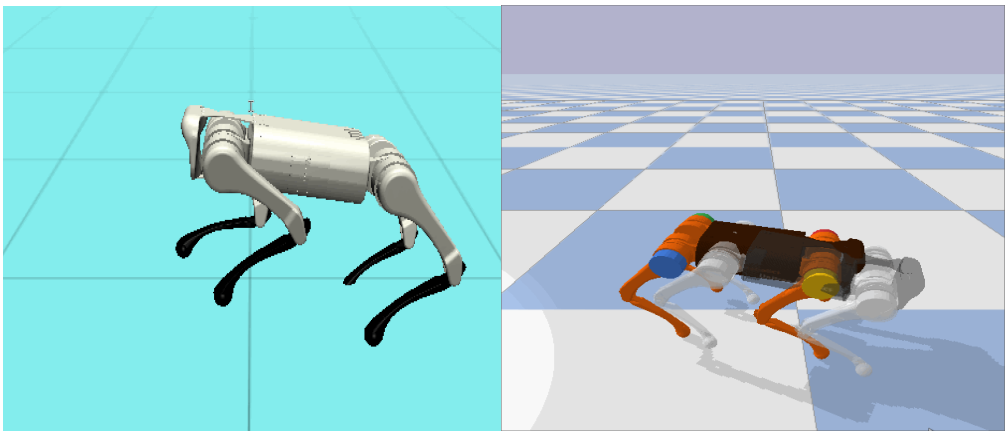
这里 r_t^p、r_t^v、r_t^e、r_t^{rp} 和 r_t^{rv} 分别表示机器人关节位置、机器人关节速度、机器人足端位置、机器人躯干位置和机器人躯干速度对应的奖励函数。而对应上标的 w 表示这些奖励函数加权求和时的权重。在这种奖励函数的设定下，若当前策略控制下四足机器人的步态与参考策略的步态差距过大，则奖励函数值接近 0，表示不鼓励当前的行为。

在搭建的仿真平台上，使用深度强化学习的 Proximal Policy Optimization 算法（PPO 算法），完成了基于模仿学习、深度强化学习的四足机器人控制策略的初步训练。PPO 算法更新策略参数的时候不仅仅要最大化累计奖励函数，还增加了对策略的约束，使得前后策略下采样得到的(s,a)分布差距不会太大，保证了策略更新的单调性。

$$\theta_{k+1} = \arg\max_{\theta} \mathop{\mathrm{E}}_{s,a\sim\pi_{\theta_k}} [L(s,a,\theta_k,\theta)]$$

$$L(s,a,\theta_k,\theta) = \min\left(\frac{\pi_\theta(a|s)}{\pi_{\theta_k}(a|s)} A^{\pi_{\theta_k}}(s,a),\ \ \mathrm{clip}\left(\frac{\pi_\theta(a|s)}{\pi_{\theta_k}(a|s)}, 1-\epsilon, 1+\epsilon\right) A^{\pi_{\theta_k}}(s,a)\right)$$

如图 4.1.3 所示是基于 mujoco 和 pybullet 搭建的 A1 型号机器人仿真平台。



(a) (b)

图 4.1.3 基于 mujoco 和 pybullet 搭建的 A1 型号机器人仿真平台(a)：基于 mujoco 搭建的仿真平台(b)：基于 pybullet 搭建的仿真平台

如图 4.1.4 所示是基于 Pybullet 的，倾角为 2° 斜面仿真平台。

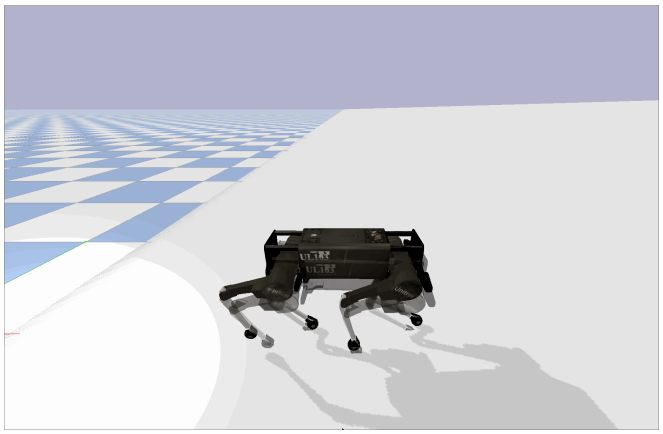


图 4.1.4 基于 Pybullet 的，倾角为 2° 斜面仿真平台

如图 4.1.5 所示是基于 Pybullet 的晃动平面仿真平台。

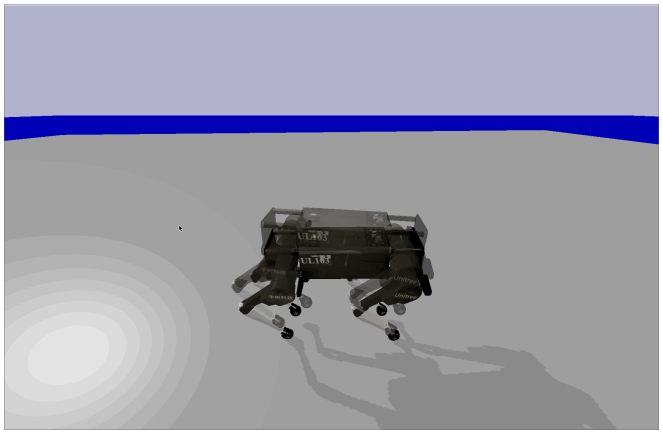


图 4.1.5 基于 Pybullet 的，晃动平面仿真平台

4.2 四足机器人的运动控制

四足机器人的运动学可以认为是通过改变四条腿足端的位置，来改变机器人机身的位置和姿态。而四足机器人的动力学则是利用地面对机器人足端的作用力来改变机器人的运动状态。再通过单腿的运动学和静力学分析就能将机器人整体的运动与每个关节的命令联系在一起。

首先要分析的就是如何移动四足机器人的足端。机器人足端只有触地（T_{stance}）和腾空（T_{swing}）两种状态，触地的腿称为支撑腿，腾空的腿称为摆动腿。当四足机器人行走时，每个足端都会再这两种状态之间周期性切换。这四个足端切换状态的不同模式成为步态。用条状图表示足端在一个周期的状态。其中黑色部分代表触地状态，白色部分代表腾空状态。如图 4.2.1 所示是四足机器人的对角步态。

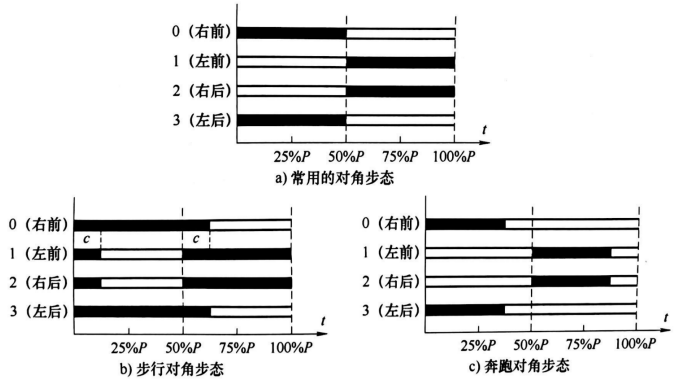


图 4.2.1 四足机器人对角步态

如图 4.2.2 所示是四足机器人的其他常用步态。

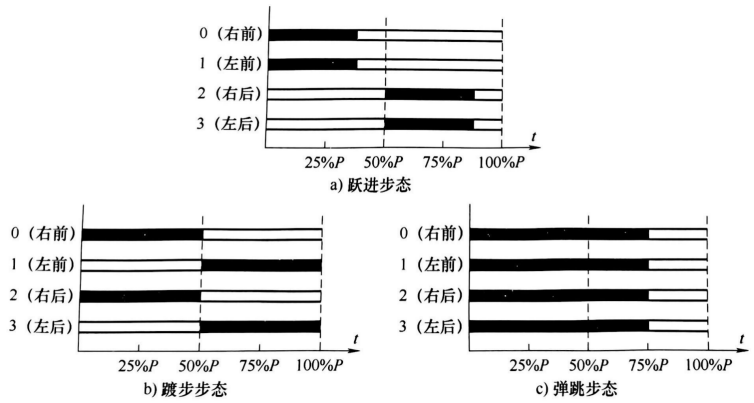


图 4.2.2 四足机器人其他常用步态

步态确定了触地和腾空在时间上的分布，在触地状态时，机器人的足端位置在世界坐标系下保持静止，但是在腾空状态时，机器人的足端在世界坐标系下是运动的。摆动腿的腾空足端总是从初始触地位置运动到落脚点，因此为了确定足端的运动轨迹，需要确定落脚点的位置。

如图 4.2.3 所示是机器人以匀速 v_x 平移时的落脚点选择，取机器人的一条腿来研究。

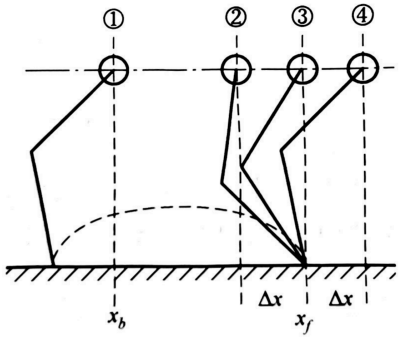


图 4.2.3 四足机器人平移落脚点

其中 1 时刻，足端刚刚开始腾空，沿途中的虚线运动。2 时刻触地。3 时刻足端位于大腿关节正下方。4 时刻足端又开始腾空。由 Raibert 启发落脚点理论，可以计算落脚点的坐标。

$$\begin{cases} x_f = x_p(p) + v_x(1-p)T_{\text{swing}} + \dfrac{1}{2}v_x T_{\text{stance}} + k_x(v_x - v_{xd}) \\ y_f = y_p(p) + v_y(1-p)T_{\text{swing}} + \dfrac{1}{2}v_y T_{\text{stance}} + k_y(v_y - v_{yd}) \end{cases}$$

其中 T_{stance} 和 T_{swing} 分别表示一个运动周期中触地和腾空的时间，p 是步态的相位，v_x 是机器人匀速运动的速度。

运动学中控制的变量都是机器人关节的角度、足端的位置。为了实现更好的控制效果，需要加入对足端力的控制，建立足端力与机器人运动的力学关系。四足机器人模型较为复杂，需要进行简化方便进行力学分析。四足机器人简化模型如图 4.2.4 所示。

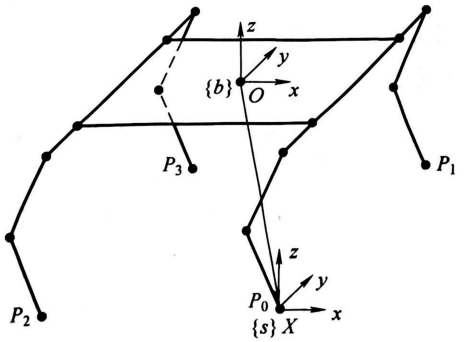


图 4.2.4 四足机器人简化模型图

忽略机器人各个部件在外力作用下形变，将其视为多个刚体连接组成的系统。考虑到四足机器人运动时很少快速旋转，通过力学公式的推导能得到如下所示的简化动力学方程。

$$\begin{bmatrix} \mathbf{I} & \mathbf{I} & \mathbf{I} & \mathbf{I} \\ [\mathbf{p}_{g0}]_x & [\mathbf{p}_{g1}]_x & [\mathbf{p}_{g2}]_x & [\mathbf{p}_{g3}]_x \end{bmatrix} \begin{bmatrix} \mathbf{f}_{0s} \\ \mathbf{f}_{1s} \\ \mathbf{f}_{2s} \\ \mathbf{f}_{3s} \end{bmatrix} = \begin{bmatrix} m(\dot{\mathbf{v}}_s - \mathbf{g}) \\ \mathbf{R}_{sb}\mathbf{I}_b\mathbf{R}_{sb}^T\dot{\boldsymbol{\omega}}_s \end{bmatrix} \Rightarrow \mathbf{Af} = \mathbf{b}_d$$

其中 I 是单位矩阵，$\mathbf{p}_{gi}$是每个足端力的力矩，$\mathbf{f}_{is}$是每条腿的足端力，$\mathbf{R}_{sb}$是机器人姿态，v_s和ω_s都是机器人在世界坐标系下的运动状态，$\mathbf{I}_b$是四足机器人在机身坐标系下的转动惯量。当某条腿处于腾空状态，没有与地面接触时，只需将这条腿对应的足端力赋值为零向量即可。

利用二次规划的方法可以求得每条腿的足端力$\mathbf{f}_{is}$。当机器人腿触地时，腿速度变化比较小，且机器人腿质量小，可以认为单腿的总功率近似为 0。所以在足端力控制时，可以直接使用单腿静力学来求各个关节的力矩。

$$\tau_i = -J_i^T f_{ib} = -J_i^T R_{bs} f_{is} = -J_i^T R_{sb}^T f_{is}$$

其中τ_i是机器人第 i 条腿上三个关节的力矩向量；J_i是该腿的雅可比矩阵；f_{ib}是机身坐标系下地面对足端的作用力，即求解机器人动力学方程得到的足端力。

4.3 实时视频传输

为了能实时监测甲板情况，需要用四足机器人在甲板上巡逻检视。机器人检视时，通过携带的摄像头实时采集视频数据，远程传输给终端，实现远程的实时检视。

该问题的难点主要在于摄像头的选型和远程传输的实现。

其中摄像头型号需要满足 4K 分辨率、20 倍变焦和 H264/AVC 编码。经市场调研，选择生华视通型号名为 SH-HK590A 的网络摄像头，如图 4.3.1 所示。

图 4.3.1 SH-HK590A 的网络摄像头

该摄像头的具体参数如图 4.3.2 所示。

摄像机、镜头参数				
光学变焦	12倍	12倍	20倍	30倍
焦距范围	4.4~52.8mm	3.9~46.8mm	5.5~110mm	4.3~129mm
视场角	8.2°(窄角) 72° (广角)	6.3°(窄角) 72.5° (广角)	3.3°(窄角) 54.7° (广角)	2.34°(窄角) 65.1° (广角)
光圈系数	F1.8 – F2.6	F1.8– F2.4	F1.6 – F3.5	F1.6 – F4.7
图像传感器	1/1.8英寸 CMOS	1/2.7英寸 CMOS图像传感器		
有效像素	851万像素	207万像素		
4K版本 视频信号	HDMI输出支持的视频格式包括 4KP30、4KP25、1080P60、1080P50、1080P30、1080P25、720P60、720P50、1080P59.94、1080P29.97、720P59.94、720P29.97; "P" 代表逐行（Progressive）扫描格式的图像。 "I" 代表隔行（interlaced）扫描格式的图像。 USB3.0输出支持视频格式包括 YUY2/NV12 支持 1920×1080P30、1280×720P30、1024×576P30、800×600P30、800×448P30、640×360P30、480×270P30、320×180P30 MJPEG/H.26/H.265 支持 3840×2160P30、2560×1440P、1920×1080P30、1600×896P30、1280×720P30、1024×576P30、960×540P30、800×600P30、800×448P30、720×576P30、720×480P30、640×360P30、640×480P30、480×270、320×240P30、352×288P30、320×176P30 USB3.0兼容USB2.0输出支持视频格式包括 YUY2/NV12 支持 1920×1080P10、1280×720P25、1024×576P30、800×600P30、800×448P30、640×360P30、480×270P30、320×180P30 MJPEG/H.26/H.265 支持 3840×2160P30、2560×1440P、1920×1080P30、1600×896P30、1280×720P30、1024×576P30、960×540P30、800×600P30、800×448P30、720×576P30、720×480P30、640×360P30、640×480P30、480×270、320×240P30、352×288P30、320×176P30			

图 4.3.2 SH-HK590A 的网络摄像头参数表

由参数表可知该型号摄像头分辨率达 4K，且能实现 20 倍变焦，主码流使用

H.264 编码标准。其中变焦功能可通过手机下载摄像头厂商提供的 UI 界面来控制。

该摄像头使用 DC 12V 供电，其最大功耗为 24W。而宇树科技 Aliengo 型号四足机器人提供的接口如图 4.3.3 所示。可以利用其中的 12V 输出接口给摄像头供电。

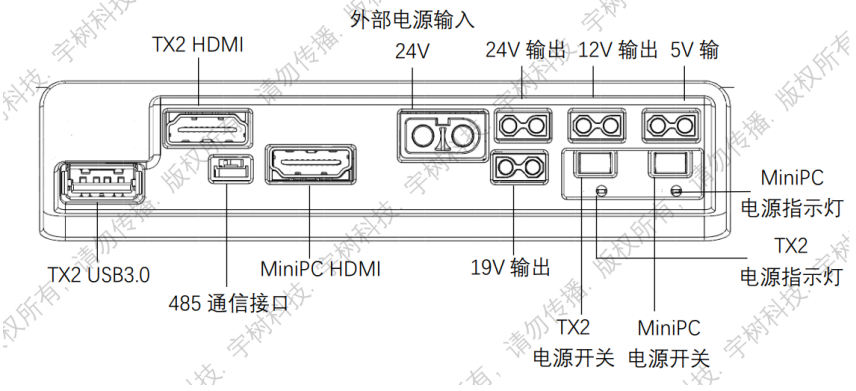


图 4.3.3 宇树科技 Aliengo 型号机器人接口

远距离传输视频数据需要依赖 RTSP 协议。RTSP 协议全名 Real Time Streaming Protocol，是 TCP/IP 协议体系中的一个应用层协议，该协议定义了一对多应用程序如何有效地通过 IP 网络传送多媒体数据。如图 4.3.4 所示是实时视频传输系统的框架图。

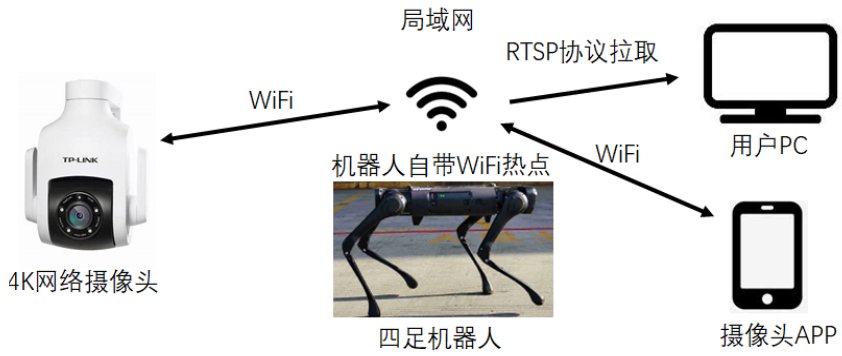


图 4.3.4 实时视频传输系统框架图

在使用时，将网络摄像头、手机和电脑均连接到四足机器人发射的 WiFi 热点上，形成局域网，每个设备都能在该局域网下分配到一个 IP 地址。由图 2 可知，SH-HK590A 型号网络摄像头支持 RTSP 协议，因此可以在连入该局域网的电脑上使用多媒体播放器（如 VLC 等）通过 RTSP 取流地址拉取视频流。如图

4.3.5 所示。

图 4.3.5 RTSP 取流地址示意图

4.4 单调的鲁棒策略优化算法

4.4.1 基于域随机化的强化学习训练仿真环境介绍

如图 4.4.1，基于强化学习理论我们将机器人训练环境与机器人中强化学习智能体的交互建模成马尔可夫决策过程$< \mathcal{O}, \mathcal{A}, \mathcal{T}, R >$。机器人如图 4.4.2 所示，为一双足机器人。影响训练环境状态转移模型$\mathcal{T}$的关键物理参数记为$\rho \in \mathcal{P}$。在每个时间步长t，模拟器中的环境处于状态$\mathbf{s_t} \in \mathcal{S}$，智能体接收来自机器人传感器对于环境状态的感知观察$\mathbf{o_t} \in \mathcal{O}$，根据当前的控制策略$\pi$，智能体输出控制动作$\mathbf{a_t} \in \mathcal{A}$作用于环境，并计算回报$r_t = R(\boldsymbol{s}_t, \boldsymbol{a}_t)$，此后环境发生状态转移，在确定的环境参数设置$\rho$下，环境的状态转移概率$\mathcal{T}$，产生下一个时间步长$t+1$的状态$\boldsymbol{s}_{t+1} \sim \mathcal{T}(\cdot|\boldsymbol{s}_t, \boldsymbol{a}_t)$。强化学习算法的目标是找到一个策略$\pi$能够极大化在环境上交互的期望累计折扣奖励$\eta(\pi|p) = \mathbb{E}_\tau\left[G(\tau|\rho)\right]$，其中$G(\boldsymbol{\tau}|\rho) = \sum_{t=0}^{\infty}\gamma^t r_t$为累计折扣奖励，$\boldsymbol{\tau} = \{s_t, a_t, r_t, s_{t+1}\}_{t=0}^{\infty}$为策略$\pi$在环境上交互得到的经验轨迹，$\gamma \in [0,1]$为折扣因子。策略在环境上的性能记为$\eta(\pi|\rho)$。基于域随机化技术，调整的物理参数$\rho$记为随机变量，其参数空间为$\mathcal{P}$服从的随机分布为$P$。所述调整的物理参数为机器人关节处的关节阻尼和机器人身体的质量。对于另一方面，强化学习智能体可以根据交互得到的经验计算处于某个状态和采取某个动作对于完成任务的价值：某个状态下使用策略π进行交互的期望累计折扣奖励记为状态价值函数$V_\pi(s) = \mathbb{E}[\sum_{k=0}^{\infty}\gamma^k r_{t+k}|s_t = s]$，某个状态下采用了某个控制动作后使用策略$\pi$进行交互的期望累计折扣奖励记为动作价值函数

$Q_\pi(s)=\mathbb{E}[\sum_{k=0}^{\infty}\gamma^k r_{t+k}|s_t=s,a_t=a]$，以及优势函数记为$A_\pi(s,a)=Q_\pi(s,a)-V_\pi(s)$。

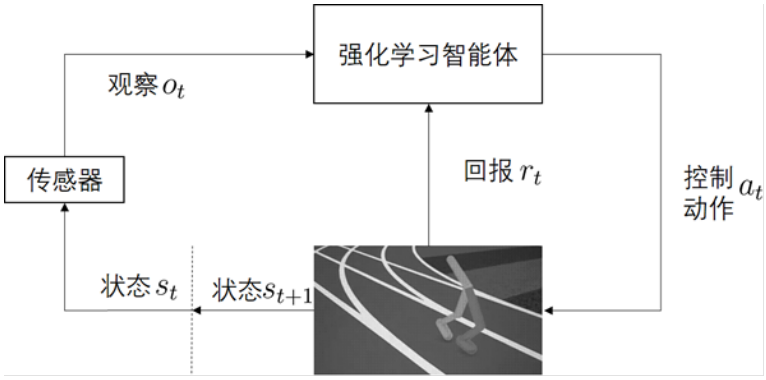


图 4.4.1 基于域随机化的强化学习训练仿真环境

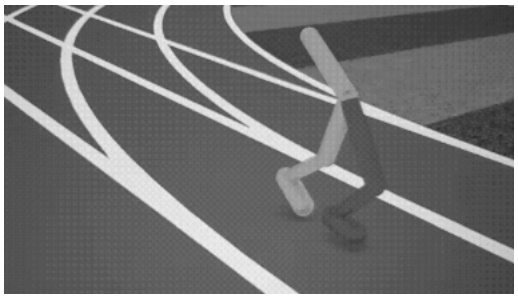


图 4.4.2 基于域随机化的机器人示意图

4.4.2 对策略与环境交互过程中最差性能和平均性能的分析

在第k次策略迭代下，更新前的策略π_k在对应的最差环境ρ_w^k上的性能为$\eta(\pi_k\|\rho_w^k)$。本次更新后的策略在环境ρ_w^k上表现$\eta(\pi_{k+1}\|\rho_w^k)$和在环境参数空间上的期望表现$\mathbb{E}_{\rho\sim P}[\eta(\pi_{k+1}\|\rho)]$满足如下不等式关系：

$$\eta(\pi_{k+1}|\rho_w^k)\geq\mathbb{E}_{\rho\sim P}\left[\eta(\pi_{k+1}|\rho)\right]-2|r|_{\max}\frac{\gamma\mathbb{E}_{\rho\sim P}\left[\epsilon(\rho_w^k\|\rho)\right]}{(1-\gamma)^2}-\frac{4|r|_{\max}d(\pi_k,\pi_{k+1})}{(1-\gamma)^2}。$$

其中$|r|_{\max}$为奖励的最大值，$\epsilon(\rho_w\|\rho)$为环境ρ_w和ρ状态转移分布之间的距离，即

$$\epsilon(p_w\|p)\triangleq\max_t\mathbb{E}_{s',a}D_{TV}\left(\mathcal{T}(s|s',a,p_w)\|\mathcal{T}(s|s',,p)\right),$$

$d(\pi_k,\pi_{k+1})$为更新前后策略的分布距离，即

$$d(\pi,\tilde{\pi})\triangleq\max_t\mathbb{E}_{s'}D_{TV}\left(\pi(a|s')\|\tilde{\pi}(a|s')\right)。$$

在上述不等式关系中，第一项为更新后策略在所有环境上的期望性能，第二

项可以看作对于分布P赋予环境ρ_w以及与之状态转移分布相近的环境低概率的惩罚，第三项可以看作对于单次迭代策略更新过大的惩罚。

基于上式不等式关系，建立单调的鲁棒策略优化问题如下（其中每个参数的含义可在上下文中对应获取）：

目标优化问题：$\max_{\pi_{k+1},P} \mathbb{E}_{\rho\sim P}\left[\eta(\pi^{k+1}|\rho)\right]$

约束条件：$d(\pi_k,\pi_{k+1}) \leq \delta_1, \quad \mathbb{E}_{\rho\sim P}\left[\epsilon(\rho_w\|\rho)\right] \leq \delta_2$。

其中，优化变量为更新后的控制策略π_{k+1}和物理参数服从的随机分布P。

优化目标为：极大化所有可能环境上的期望性能$\mathbb{E}_{\rho\sim P}\left[\eta(\pi^{k+1}|\rho)\right]$。

约束条件为：第一个条件引入了策略更新的信赖域限制，第二个条件引入了对于分布P需要给环境ρ_w及类似环境以更高概率的限制。

给出两步优化过程来近似解决所述单调的鲁棒策略优化问题。所述的两步优化过程执行过程如下（其中每个参数的含义可在上下文中对应获取）：

优化随机分布P：不考虑策略π_{k+1}的优化，即令$\pi_{k+1}=\pi_k$，仅考虑优化变量为随机分布P。在此步骤下，第一个约束条件恒成立，我们将第二个关于随机分布的限制条件转移到优化目标中，得到如下的无限制的分布优化问题：

目标优化问题：$\max_P \mathbb{E}_{\rho\sim P}\left[\mathcal{E}(\rho,\pi_k)\right]$,

其中$\mathcal{E}(\rho,\pi_k) \triangleq \eta(\pi_k|\rho) - \frac{2|r|_{max}\gamma\epsilon(\rho\|\rho_w)}{(1-\gamma)^2}$为更新前策略$\pi_k$在环境$\rho$上的一个衡量，其中的第一项表示策略$\pi_k$在环境$\rho$上的性能，第二项表示环境$\rho$和$\rho_w$动态模型之间的差距。因为所述分布优化问题目标函数关于P是线性的，因此分布P的更新方向是赋予具有更大 $\mathcal{E}(\rho,\pi_k)$值的环境更高的概率。为了兼顾考虑平均性能和最差性能，根据条件$\mathcal{E}(p,\pi_k) \geq \mathcal{E}(p_w,\pi_k)$选取环境赋予均等的采样概率，不满足条件的环境赋予为零的采样概率。

(b) 优化策略π_{k+1}：采用在步骤(a)中得到的分布P进行测量的优化，得到如下的策略优化问题：

目标优化问题：$\max_{\pi_{k+1}} \mathbb{E}_{\sim P}\left[\eta(\pi_{k+1}|\rho)\right]$

约束条件：$d(\pi_k,\pi_{k+1}) \leq \delta_1$

所述策略优化问题可以转换为信赖域鲁棒策略优化问题使用近端策略优化算法来解决。

4.4.3 单调的鲁棒策略优化

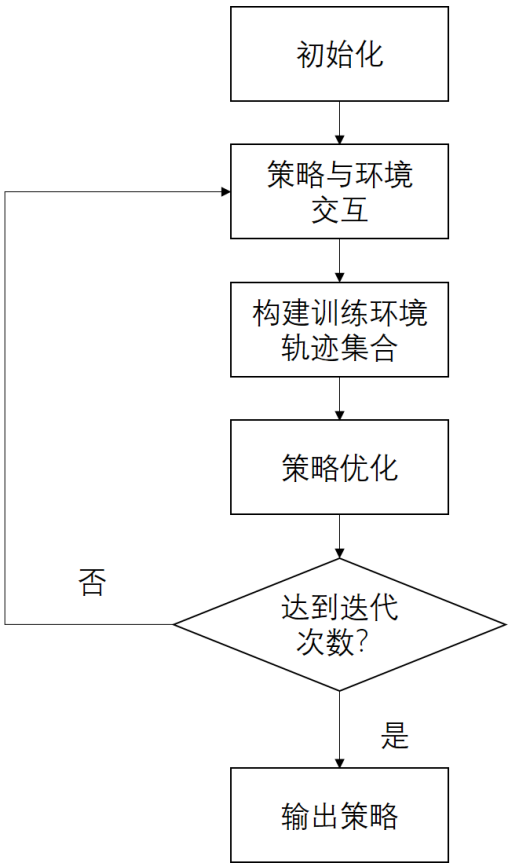


图 4.4.3 单调的鲁棒策略优化算法流程图

如图 4.4.3 所示，基于如上所述两步优化过程，给出单调的鲁棒策略优化算法，最终得到兼顾平均性能和最差性能的鲁棒策略。所述的单调的鲁棒策略优化算法执行过程如下（其中每个参数的含义可在上下文中对应获取）：

(a) 初始化：设置初始神经网络策略为π_0，环境参数分布为均匀分布U，每次迭代采样的环境数量为M，每个环境上采样的经验轨迹数量为L，最大迭代次数N，采样的环境参数集合$\mathbb{S}=\emptyset$和训练环境轨迹集合$\mathbb{T}=\emptyset$。

(b) 当前策略π_k与环境的交互步骤（$i=0,1,2,\ldots,N-1$）：

根据均匀分布U采样环境参数ρ_i并加入到环境参数集合$\mathbb{S}$中。在环境ρ_i上使用策略π_k采样L条经验轨迹$\{\boldsymbol{\tau}_{i,j}\}_{j=0}^{L-1}$。计算策略$\pi_k$在环境$\rho_i$上的经验性能估计$\hat{\eta}(\pi_k|\rho_i)=\sum_{j=0}^{L-1}G(\boldsymbol{\tau}_{i,j}|\rho_i)/L$。

(c) 构建训练环境轨迹集合$\mathbb{T}$步骤（$i=0,1,2,\ldots,M-1$）：

根据$\hat{\eta}(\pi_k|\rho_i)$计算当前策略表现最差的环境$\rho_w^k=\arg\min_{\rho_i\in\mathbb{S}}\hat{\eta}(\pi_k|\rho_i)$。计算当

前策略π_k在环境ρ_i上的经验衡量$\hat{\mathcal{E}}(\rho_i,\pi_k)=\hat{\eta}(\pi_k|\rho_i)-\kappa\|\rho_i-\rho_w^k\|$，如果$\hat{\mathcal{E}}(\rho_i,\pi_k)\geq\hat{\mathcal{E}}(\rho_w^k,\pi_k)$，则将环境$\rho_i$的$L$条经验轨迹$\{\boldsymbol{\tau}_{i,j}\}_{j=0}^{L-1}$加入到$\mathbb{T}$中。

(f)策略优化：在$\mathbb{T}$上使用近端策略优化算法得到更新后的策略π_{k+1}。

(g)判定步骤：设置$\mathbb{T}=\emptyset$，$\mathbb{S}=\emptyset$，如果迭代次数达到最大迭代次数N，则停止迭代；否则，转到(b)。

4.5 域随机化中控制策略优化的方差减少研究

4.5.1 强化学习策略梯度算法中的基于基准的方差减少方法

环境参数直接决定任务环境的动力学模型。在域随机化中，环境参数p是一个服从采样分布$\mathcal{P}$的随机变量。令$|\mathcal{P}|$表示任务集合上环境的总个数。在域随机化中，策略优化的目的是要极大化所有可能环境参数上策略的期望累计奖励：

$$\max_{\pi}\mathbb{E}_{p\sim P}[\eta(\pi,p)]。$$

使用不依赖于动作的基准，策略梯度方法解析式可以给出如下：

$$\nabla_\theta\mathbb{E}_{p\sim P}[\eta(\pi,p)]=\mathbb{E}_P[\mathbb{E}_{\mu_\pi^p,\pi}[\nabla_\theta\log\pi_\theta(a|s)[Q_\pi(s,a,p)-b]]],$$

其中$\mu_\pi^p(s)=\sum_{t=0}^{\infty}\gamma^t P_\pi(s_t=s|p)$定义为折扣的状态访问频率。此外上式中基准用$b$表示，代表策略梯度中减去的一项。我们定义给定状态动作对在环境p下的策略梯度为$g(\theta,s,a,p)\triangleq\nabla_\theta\log\pi_\theta(a|s)\,[Q_\pi(s,a,p)-b]$。当基准$b$与动作无关时，能获得如下结论

$$\mathbb{E}_a[\nabla_\theta\log\pi_\theta(a|s)b]=\nabla_\theta\mathbb{E}_a[b]=0,$$

$$\mathbb{E}_{P,\mu_\pi^p,\pi}[g]=\mathbb{E}_{P,\mu_\pi^p,\pi}[\nabla_\theta\log\pi_\theta(a|s)Q_\pi(s,a,p)]\triangleq\mathbb{E}[g]。$$

因此这时的策略梯度估计是无偏的，同时减去基准后的梯度期望值变小，因此能够降低策略梯度估计的方差。

4.5.2 域随机化中最优的依赖于环境和状态的基准

本小节介绍如何推导最优的依赖于环境和状态的基准。我们定义任意依赖于环境和状态的基准为$b(s,\mathcal{P})=\{b(s,p_i)\}_{i=1}^{|\mathcal{P}|}$，建立如下以该基准为优化变量的优化问题：

$$\min_{b(s,\mathcal{P})} F(b(s,\mathcal{P}))=\mathbb{E}_{p\sim P,\mu_\pi^p}[\mathbb{E}_\pi[G(a,s)[Q_\pi(s,a,p)-b(s,p)]^2]],$$

解该优化问题可得最优的依赖于环境和状态的基准：

$$b^*(s,\mathcal{P})=\{b^*(s,p_i)\}_{i=1}^{|\mathcal{P}|}，\text{其中}b^*(s,p_i)=\frac{\mathbb{E}_\pi[G(a,s)Q_\pi(s,a,p_i)]}{\mathbb{E}_\pi[G(a,s)]}。$$

该基准相较于任意仅依赖于状态的基准，减少的方差可以理论上给出为：

$$Var^{b(s)}(g)-Var^{b^*(s,\mathcal{P})}(g)=\mathbb{E}_{P,\mu_\pi^p}[\sqrt{\mathbb{E}_\pi[G(a,s)]}b(s)-\frac{\mathbb{E}_\pi[G(a,s)Q_\pi(s,a,p)]}{\sqrt{\mathbb{E}_\pi[G(a,s)]}}]^2,$$

其中$Var^{b(s)}(g)$为使用任意仅依赖于状态基准的策略梯度方差，$Var^{b^*(s,\mathcal{P})}(g)$为最优的依赖于环境和状态的基准。

4.5.3 基于聚类的方差减少算法框架

本小节介绍基于聚类的方差减少算法框架，算法框架图如图 4.5.1 所示。图中双足机器人相同部位的不同颜色表明该部位的环境参数不一样，会导致最终机器人的动力学方程上的差异，如果颜色相近则表明机器人该部位的动力学方程接近。例如在p_0和p_3中的两个机器人左腿都是绿色，右腿都是红色表明这两个机器人具有相同的动力学方程，因此将其归为$\mathcal{C}_1$类，同理将p_3和p_{H-1}归为$\mathcal{C}_2$类，即将环境参数划分为不同的子空间。之后为不同类上的任务环境都分别学习一个依赖于环境和子空间的的基准。之后我们利用环境即策略在环境上的性能表现进行聚类，得到聚类原型以进行对后续迭代中采样到的环境参数的聚类操作。

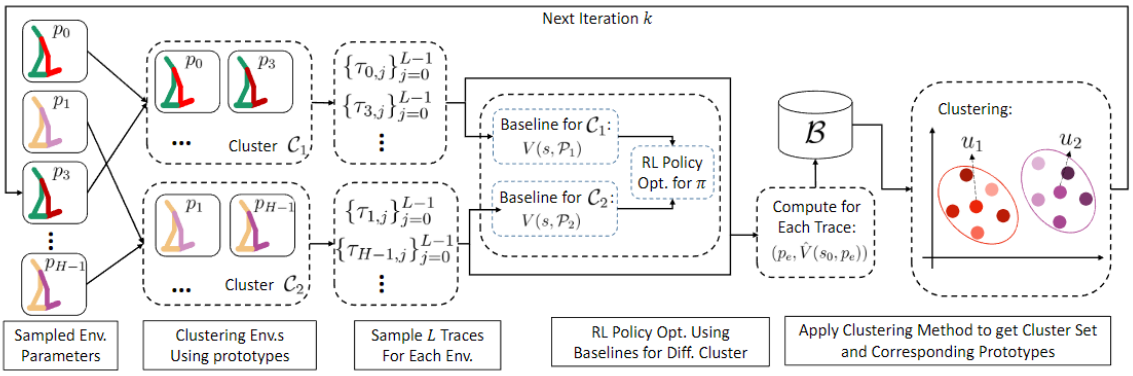


图 4.5.1 基于聚类的方差减少算法框架图

4.5.4 算法部分实验展示

我们在强化学习标准测试环境 MUJOCO 上测试了我们所提出的算法和其他对比方案，算法训练曲线如图 4.5.2 所示。仿真实验表明我们的方法能够加速收敛并且获得更好的收敛性能。

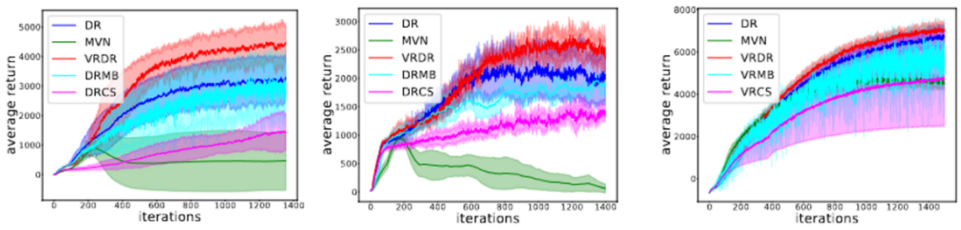


图 4.5.2 算法训练曲线

此外，我们在仿真环境中计算了所有算法在训练过程中策略梯度的方差估计值，其曲线如图 4.5.3 所示。结果表明我们的方法能够获得最小的策略梯度方差。

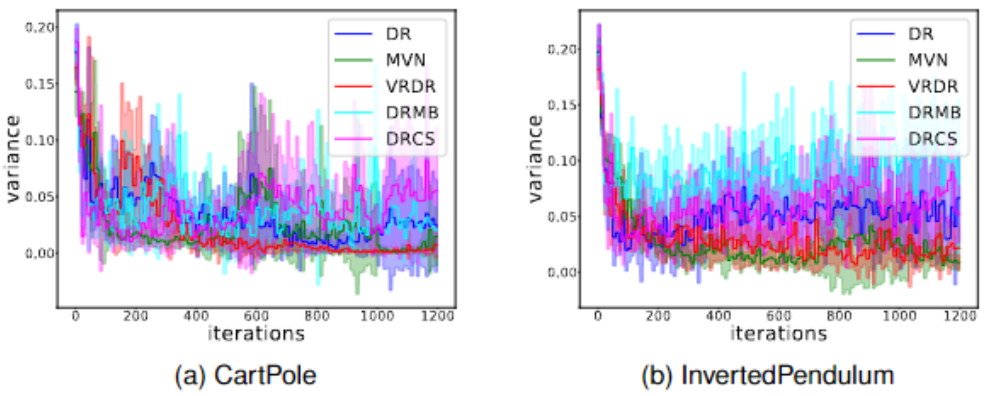


图 4.5.3 策略梯度的方差估计值

4.6 深度元强化学习中的双重鲁棒增强的重标记算法

4.6.1 基准元强化学习算法介绍

传统的深度强化学习为每个任务分别学习一个策略，每个任务上都需要与环境交互产生大量的采样来学习，因此用这种方法来学习大量不同的行为令人望而却步。幸运的是，很多我们想要解决的问题都有共享的结构，例如在瓶子上拧上瓶盖和转动门把手都需要抓住手中的物体并转动手腕。元学习方法通过从一个任务分布集合上收集到的大量经验来学习这种共享的结构，一旦成功习得，给定新任务上的少量经验就能快速地适应到新任务上去。深度元学习，顾名思义，就是将元学习技术应用到深度强化学习领域，深度元强化学习技术可以使得强化学习智能体通过少量在新任务上的采样就能够学习到新技能。

给定一个任务集合，其中每个任务都被建模为一个马尔可夫决策过程（MDP）$M_i \in \mathcal{M}$，每个 MDP 都由一个四元组决定，即$M_e =< \mathcal{S}, \mathcal{A}, \mathcal{T}_e, R_e >$。注意到这些任务拥有相同的状态空间$\mathcal{S}$和动作空间$\mathcal{A}$但是拥有不同的动态转移$\mathcal{T}_e$和奖励函数$R_e$。元强化学习的优化目标可以概括为

$$\max_{\phi,\varphi} \mathop{\mathbb{E}}_{p(\mathcal{T})} \mathop{\mathbb{E}}_{\substack{s_t\sim p_{\mathcal{T}} \\ a_t\sim\pi_{\phi'_{\mathcal{T}}}}} \left[\sum_{t=0}^{T}\gamma^t r_{\mathcal{T}}(s_t,a_t)\right], \text{s.t.}\ \ \phi'_{\mathcal{T}} = f_{\varphi}(\phi,\mathcal{T}),$$

其中f_{φ}是元强化学习要训练的适应过程，将元策略参数ϕ更新到不同的任务上得到该任务上的策略参数$\phi'_{\mathcal{T}}$。其中算法的Q值和适应过程参数φ由以下损失函数训练：

$$\mathcal{L}_c \quad = \frac{1}{2}\mathbb{E}_{(s,a,r,s')\sim\mathcal{B},z\sim q_{\varphi}(z|c)}[Q_{\theta}(s,a,z) - \tilde{Q}(s,a,\overline{z})]^2。$$

其中算法的元策略参数由以下损失函数训练：

$$\mathcal{L}_a = \mathbb{E}[D_{KL}(\pi_{\phi}(a|s,\overline{z}) \parallel \frac{\exp\left(Q_{\theta}(s,a,\overline{z})\right)}{Z_{\theta}(s)})]。$$

本项目所使用的基准元强化学习算法如图 4.6.1 所示。

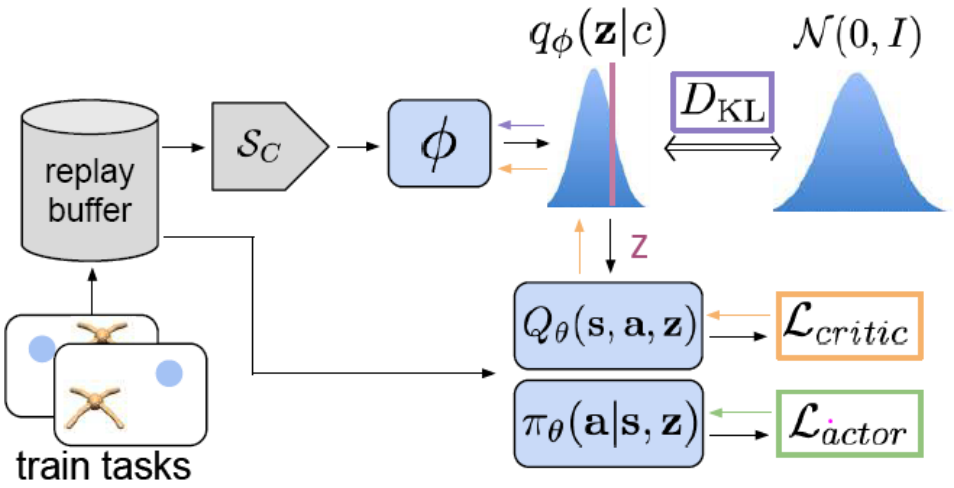


图 4.6.1 基准元强化学习算法

4.6.2 双重鲁棒保证的值估计器

双重鲁棒保证的估计器将低方差值函数近似方法与无偏的重要性加权值估计方法相结合，从而为控制策略的离策略值估计提供了无偏估计的双重鲁棒保证。对于离策略估计，双重鲁棒保证的值估计器为：

$$V^{DR}(s_t = s) = \hat{V}(s_t = s) + \rho_\pi(t)(r_t + \gamma V^{DR}(s_{t+1}) - \hat{Q}(s, a_t))。$$

其中$\hat{V}(s) = \mathbb{E}_{a\sim\pi_e}[\hat{Q}(s,a)]$，策略重要性权重$\rho_\pi = \pi_e(a|s)/\pi_b(a|s)$。双重鲁棒的含义为当 1）重要性权重是准确的，那么对值估计器求期望得到$V^{DR}(s_t = s) = \mathbb{E}_{a\sim\pi_e}[r_t + \gamma V^{DR}(s_{t+1})]$，因此此时的$V^{DR}$是无偏估计器；2）$\hat{Q}$对于$Q$值的拟合是准确的，那么$r_t + \gamma V^{DR}(s_{t+1}) - \hat{Q}(s, a_t) = 0$，而$V^{DR}(s_t = s) = \mathbb{E}_{a\sim\pi_e}[\hat{Q}(s,a)]$，此时的$V^{DR}$依旧是无偏估计。

4.6.3 双重鲁棒增强的重标记算法

基于上述提到的双重鲁棒保证的值估计器，我们提出双重鲁棒增强的重标记方法（DRR），结合所使用的基准元强化学习算法，整体算法的流程框图如图 4.6.2 所示。所提出 DRR 在图中由蓝色背景着重标出。在每个元训练迭代中，由待重标记样本选择策略$\mathcal{S}_I$选择样本数据集$\mathcal{D}$，在该据集$\mathcal{D}$上，我们拟合环境动态并用

于估计动力学和策略的重要性权重，最后利用双重鲁棒保证的值估计器计算重标记之后的状态价值并用来更新基准元强化学习算法中的网络参数。

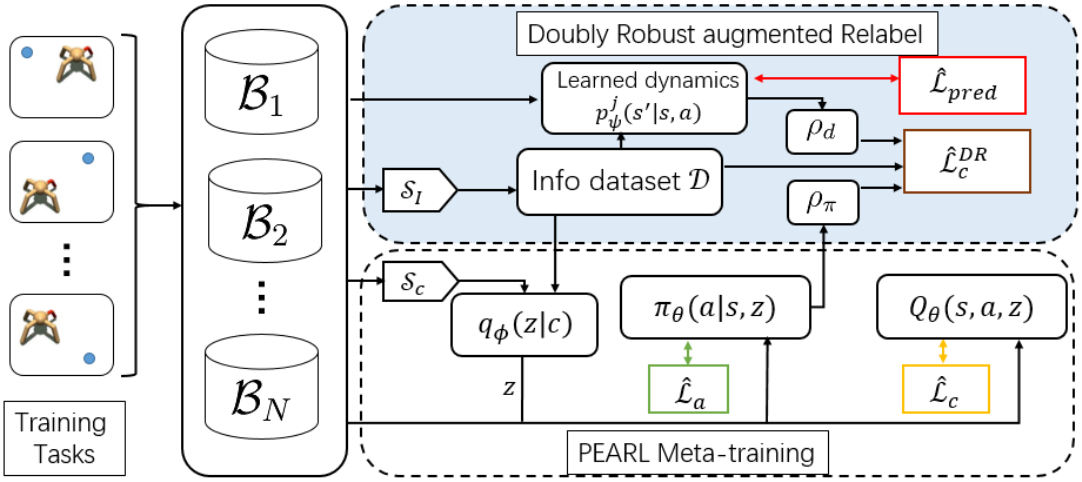


图 4.6.2 基于双重鲁棒增强的重标记算法的元强化学习框架图

4.7 基于 D*算法的四足机器人路径规划与避障

D*算法是一种启发式的路径搜索算法，适合面对周围环境未知或者周围环境存在动态变化的场景。D*算法的主要原理是：通过维护一个优先队列（OpenList）来对场景中的路径节点进行搜索，以目标点为起始，通过将目标点置于 OpenList 来开始搜索，直到机器人当前位置节点从队列中出队为止。如果中间某节点的状态有动态改变，且被算法检测出来的时候，就需要重新寻路。但并不需要重新从头到尾进行寻路，只需要检测动态改变的节点周围节点的状态即可。

地图中的路径点用 state 类来表示，每一个 state 类包含如下信息：

i. **Back-pointer：** 指向前一个 state 的指针，指向的 state 被称为当前状态的父辈，目标（即终点）的 state 无 Back-pointer。路径搜索完毕后，通过机器人当前所在的 state（即起点）与 Back-pointer 即可一步步地移动到路径规划的目标 Goal。$b(X) = Y$表示节点 X 的父辈为 Y。

ii. **Tag：** 表示当前 state 的状态，有 New、Open、Closed 三种状态。其中，New 表示该 state 从未被置于 OpenList 中，Open 表示该 state 正位于 OpenList 中，Closed 表示该 state 已经不再位于 OpenList 中。

iii. **H(X)：**代价函数的估计，表示从当前 state 到目标 Goal 的开销估计。

iv. **K(X)：**即 Key function，该值是优先队列 OpenList 中的排序依据，K 值最小的 state 位于优先队列的队列头。对于处于 OpenList 上的 state X，K(X)表示但 X 被置于 OpenList 后，X 到目标 Goal 的最小代价 H(X)。可以简单理解为：K(X)将位于 OpenList 中的 state X 划分为两种不同的状态，一种状态为“上升”状态，即$K(X) > H(X)$，用于传递这条路径开销的增加（比如某两点之间的开销增加，会导致与之相关的节点到目标点路径的开销随之增加)；另一种状态为“下降”状态，即$K(X) < H(X)$，用于传递这条路径开销的减少（比如某两点之间的开销减少，或者某一新的节点被加入到 OpenList 中，就可能导致与之相关的节点到目标路径的开销随之减少)。

v. **K_{min}：**表示所有位于 OpenList 中的 state 的最小 K 值。

vi. **C(X,Y)：**表示 X 与 Y 之间的路径开销。

vii. **OpenList：**依据 K 值由小到大进行排序的优先队列。

但是，由于 D*算法规划路径的主要依据是从终点向起点传播，带有一定的方向性。因此，当起点和终点间存在墙体时，D*算法很容易规划出一条“贴着墙”的路径，在较为宽敞的地方，这种路径显然并不是最好的。同时，由于 D*算法只考虑了质点的路径规划，在转弯处就很容易出现“急转弯”的情况。因此，对于有一定初速度和体积实际物体，这样的路径具有很大的危险性。因此，我们在保留 D*算法的基本思路的情形下，就路径安全性方面进行优化。

主要思路为：在传统 D*的基础上，新增一个实时距离检测，用于测试当前状态 state 距离地图两边界的距离。

如果当前 state 距离某一边界过近，且距离两边界的距离差过大，则此时的位置就被认为是可优化的，需要向距离较远的边界移动。值得注意的是，如果直接沿着垂直平分线移动，很可能会出现“往回走“的情况，这种情况也不是我们希望看到的。因此，我们选择当前行进方向与距离较远侧垂直平分线的夹角的角平分线作为新的行进方向，向前行进一段距离。同时，我们并不是直接让机械狗沿着这个方向行进相应距离，而是让机械狗“虚拟”前进相应距离，将这个虚拟位置作为一个“关键点”，随后在关键点和机械狗当前位置、关键点和终点间分别再进行两次修改后的 D*算法，这样就还能保证 D*算法的实时避障功能。否则，

如果新的路线中间突然出现障碍物，机器狗就会撞上去。

计算与两边界距离的方法：由于直接计算最近距离可能会涉及较大的计算复杂度，并且这样计算出的最近距离可能是距离四足机器人身后最近点位的距离，因此我们选择采用垂直平分线法来计算当前state与两边界的距离。具体操作是，我们先通过当前 state 与其“父辈”估计出机械狗的大致行进方向。然后通过计算这个方向所在直线的垂直平分线，再计算这条垂直评分线与两边界的交点，最后计算当前 state 与两个交点的距离，作为当前 state 与两边界的距离。

表 1　调整后方向的真值表

k	dif	c	nk	nc
1	+	+1	0	1
1	+	-1	2	-1
1	-	+1	2	1
1	-	-1	0	-1
2	+	+1	1	1
2	+	-1	-1	1
2	-	+1	-1	-1
2	-	-1	1	-1
0	+	+1	1	1
0	-	+1	-1	1
0	+	-1	-1	-1
0	-	-1	1	-1
-1	+	+1	0	1
-1	+	-1	2	1
-1	-	+1	2	-1
-1	-	-1	0	-1

确定新的行进方向的方法：新的行进方向可以由“当前方向所在直线”、“当前方向”、“距离哪一侧较远”三者共同确定。由于在像素坐标中，相邻两点之间连线的斜率只有 0、1、-1、∞、-∞这五个选项，因此我们可以用 2 来代替∞的情况，然后用 x 坐标增加或减小作为直线的延伸方向（如果斜率为∞则将 y 轴增加或减小作为直线的延伸方向）。这样我们就能得到如表 1 所示的真值表。其中，k 代表当前行进方向，nk 代表调整后的行进方向；dif 代表离两边界距离的大小，

dif 为正代表离 x 坐标（k 为 0 时为 y 坐标）增加一侧较远，dif 为负则代表离 x 坐标（k 为 0 时为 y 坐标）减小一侧较远；c 与 nc 代表 x 坐标增加或减小（斜率为∞则为 y 坐标增加或减小），为+1 时代表 x 坐标增加（斜率为∞则为 y 坐标增加），为-1 时代表 x 坐标减小（斜率为∞则为 y 坐标减小）。

在新方向上的虚拟行进距离的确定方法：取以下三个距离的最小值：距离行进方向上障碍物的距离，1m 和两边界距离之差的一半。这样就能保证机器狗既能调整方向，又能保证安全性。

5 知识产权情况

通过本项目实施，获得了一系列知识产权，主要包括学术论文、授权专利、软件著作权等，具体详情如下。

5.1 学术论文列表

[1] Yuankun Jiang, Chenglin Li, Wenrui Dai, Junni Zou, Hongkai Xiong, "Variance Reduced Domain Randomization for Reinforcement Learning with Policy Gradient", ***IEEE Transactions on Pattern Analysis and Machine Intelligence***, vol.46, no. 2, pp. 1031-1048, Feb. 2024. (CCF-A 类国际期刊，SCI 影响因子：23.6)

[2] Qin Yang, Yuqi Li, Chenglin Li, Hao Wang, Sa Yan, Li Wei, Wenrui Dai, Junni Zou, Hongkai Xiong, Pascal Frossard, "SVGC-AVA: 360-Degree Video Saliency Prediction with Spherical Vector-Based Graph Convolution and Audio-Visual Attention", ***IEEE Transactions on Multimedia***, vol. 26, pp. 3061-3076, Feb. 2024. (SCI 影响因子：7.3)

[3] Hao Zhang, Chenglin Li, Wenrui Dai, Junni Zou, Hongkai Xiong, "FedCR: Personalized Federated Learning Based on Across-Client Common Representation with Conditional Mutual Information Regularization", ***International Conference on Machine Learning (ICML'2023)***, July 2023. (CCF-A 类国际会议)

[4] Wenxuan Gao, Chenglin Li, Haoran Lv, Wenrui Dai, Junni Zou, Hongkai Xiong, Xinlong Pan, Haipeng Wang, "Improving Generalization for Neural Adaptive Video Streaming via Meta Reinforcement Learning", ***Picture Coding Symposium (PCS'2022)***, Dec 2022.

[5] Jiayi Xu, Qin Yang, Chenglin Li, Junni Zou, Hongkai Xiong, Xinlong Pan, Haipeng Wang, "Rotation-Equivariant Graph Convolutional Networks For Spherical Data via Global-Local Attention", ***IEEE International Conference on Image Processing (ICIP'2022)***, Oct. 2022.

5.2 授权发明专利列表

[1] 李成林，刘春苗，戴文睿，邹君妮，熊红凯，基于神经网络的目标算法拟合方法、终端以及应用，中国发明专利，授权公告日：2023 年 4 月 28 日，专利号：

CN201911153108.3。

[2] 李成林，杨琴，戴文睿，邹君妮，熊红凯，旋转等变的图卷积神经网络的球形图像分类方法及系统，中国发明专利，授权通知日：2023 年 3 月 9 日，专利号：CN201911330871.9。

[3] 李成林，阚诺文，戴文睿，李劭辉，邹君妮，熊红凯，码率自适应分配方法，中国发明专利，授权公告日：2022 年 5 月 17 日，专利号：CN202110796984.9。

[4] 李成林，吕浩然，杨琴，邹君妮，戴文睿，熊红凯，360 度图像的显著性预测方法及系统，中国发明专利，授权公告日：2022 年 10 月 25 日，专利号：CN202010932741.9。

5.3 申请发明专利列表

[1] 李成林，潘新龙，徐佳怡，杨琴，戴文睿，邹君妮，熊红凯，王海鹏，刘瑜，基于旋转等变性的图卷积层的球形图像分类与分割方法，中国发明专利，申请日：2022 年 9 月 29 日，专利号：CN202211196141.6。

[2] 李成林，高文轩，潘新龙，吕浩然，戴文睿，邹君妮，熊红凯，王海鹏，刘瑜，基于分片的 360 度视频流编码方法及其优化方法和系统中国发明专利，申请日：2022 年 10 月 31 日，专利号：CN202211365606.6。

[3] 李成林，刘卓群，蒋远堃，戴文睿，邹君妮，熊红凯，潘新龙，王海鹏，刘瑜，基于元学习采样分布调整的图像小样本分类方法及系统，中国发明专利，申请日：2022 年 11 月 18 日，专利号：CN202211450947.3。

5.4 申请软件著作权列表

[1] 熊红凯，李成林，蒋远堃，朱首行，陈致远，戴文睿，邹君妮，可降低碰撞发生概率的路径规划系统 V1.0，软件著作权，申请日：2024 年 3 月 15 日，受理号：2024R11S0464520。

参考文献

[1] LEE J, HWANGBO J, WELLHAUSEN L, et al. Learning quadrupedal locomotion over challenging terrain[J/OL]. Science Robotics, 2020, 5(47): eabc5986. DOI:10.1126/scirobotics.abc5986.

[2] BLEDT G, POWELL M J, KATZ B, et al. MIT Cheetah 3: Design and Control of a Robust, Dynamic Quadruped Robot[C/OL]//2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS). 2018: 2245-2252. DOI:10.1109/IROS.2018.8593885.

[3] HUTTER M, GEHRING C, JUD D, et al. ANYmal - a highly mobile and dynamic quadrupedal robot[C/OL]//2016 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS). 2016: 38-44. DOI:10.1109/IROS.2016.7758092.

[4] 卞泽坤, 王兴兴. 四足机器人控制算法——建模、控制与实践[M]. 北京: 机械工业出版社, 2022: 65-128.

[5] MIKI T, LEE J, HWANGBO J, et al. Learning robust perceptive locomotion for quadrupedal robots in the wild[J/OL]. Science Robotics, 2022, 7(62): eabk2822. DOI:10.1126/scirobotics.abk2822.

[6] LI Z, CHENG X, PENG X B, et al. Reinforcement Learning for Robust Parameterized Locomotion Control of Bipedal Robots[EB/OL]. arXiv, 2021[2023-06-13]. http://arxiv.org/abs/2103.14295.

[7] HAUSKNECHT M, STONE P. Deep Recurrent Q-Learning for Partially Observable MDPs[EB/OL]. arXiv, 2017[2023-06-13]. http://arxiv.org/abs/1507.06527. DOI:10.48550/arXiv.1507.06527.

[8] FUJIMOTO S, HOOF H, MEGER D. Addressing Function Approximation Error in Actor-Critic Methods[C/OL]//Proceedings of the 35th International Conference on Machine Learning. PMLR, 2018: 1587-1596[2023-06-13]. https://proceedings.mlr.press/v80/fujimoto18a.html.

[9] VAN HASSELT H, GUEZ A, SILVER D. Deep Reinforcement Learning with Double Q-Learning[J/OL]. Proceedings of the AAAI Conference on Artificial Intelligence, 2016, 30(1)[2023-06-13]. https://ojs.aaai.org/index.php/AAAI/article/view/10295. DOI:10.1609/aaai.v30i1.10295.

[10] SCHULMAN J, WOLSKI F, DHARIWAL P, et al. Proximal policy optimization

algorithms[EB/OL]. arXiv, 2017[2023-06-13]. http://arxiv.org/abs/1707.06347.

[11] HAARNOJA T, ZHOU A, ABBEEL P, et al. Soft Actor-Critic: Off-Policy Maximum Entropy Deep Reinforcement Learning with a Stochastic Actor[C/OL]//Proceedings of the 35th International Conference on Machine Learning. PMLR, 2018: 1861-1870[2023-06-13]. https://proceedings.mlr.press/v80/haarnoja18b.html.

[12] SILVER D, WIERSTRA A G I A D, RIEDMILLER M. Playing atari with deep reinforcement learning[EB/OL]. arXiv, 2013[2023-03-31]. http://arxiv.org/abs/1312.5602.

[13] HAARNOJA T, TANG H, ABBEEL P, et al. Reinforcement Learning with Deep Energy-Based Policies[C/OL]//Proceedings of the 34th International Conference on Machine Learning. PMLR, 2017: 1352-1361[2023-03-29]. https://proceedings.mlr.press/v70/haarnoja17a.html.

[14] WANG Z, SCHAUL T, HESSEL M, et al. Dueling Network Architectures for Deep Reinforcement Learning[C/OL]//Proceedings of The 33rd International Conference on Machine Learning. PMLR, 2016: 1995-2003[2023-06-13]. https://proceedings.mlr.press/v48/wangf16.html.

[15] SCHULMAN J, LEVINE S, ABBEEL P, et al. Trust Region Policy Optimization[C/OL]//Proceedings of the 32nd International Conference on Machine Learning. PMLR, 2015: 1889-1897[2023-06-13]. https://proceedings.mlr.press/v37/schulman15.html.

[16] SILVER D, LEVER G, HEESS N, et al. Deterministic Policy Gradient Algorithms[C/OL]//Proceedings of the 31st International Conference on Machine Learning. PMLR, 2014: 387-395[2023-06-13]. https://proceedings.mlr.press/v32/silver14.html.

[17] HEESS N, TB D, SRIRAM S, et al. Emergence of locomotion behaviours in rich environments[EB/OL]. arXiv, 2017[2023-06-13]. http://arxiv.org/abs/1707.02286.

[18] LILLICRAP T P, HUNT J J, PRITZEL A, et al. Continuous control with deep reinforcement learning[EB/OL]. arXiv, 2019[2023-06-13]. http://arxiv.org/abs/1509.02971.

[19] PENG X B, COUMANS E, ZHANG T, et al. Learning agile robotic locomotion skills by imitating animals[EB/OL]. arXiv, 2020[2023-04-06]. http://arxiv.org/abs/2004.00784.

[20] ARNDT K, HAZARA M, GHADIRZADEH A, et al. Meta Reinforcement Learning for Sim-to-real Domain Adaptation[EB/OL]. arXiv, 2019[2023-06-13]. http://arxiv.org/abs/1909.12906.

[21] DUAN Y, SCHULMAN J, CHEN X, et al. Rl $^ 2$: Fast reinforcement learning via slow reinforcement learning[EB/OL]. arXiv, 2016[2023-06-13]. http://arxiv.org/abs/1611.02779.

[22] SCHOETTLER G, NAIR A, OJEA J A, et al. Meta-reinforcement learning for robotic industrial insertion tasks[EB/OL]. arXiv, 2020[2023-06-13]. http://arxiv.org/abs/2004.14404.

[23] WANG J X, KURTH-NELSON Z, TIRUMALA D, et al. Learning to reinforcement learn[EB/OL]. arXiv, 2017[2023-06-13]. http://arxiv.org/abs/1611.05763.

[24] MISHRA N, ROHANINEJAD M, CHEN X, et al. A simple neural attentive meta-learner[EB/OL]. arXiv, 2018[2023-06-13]. http://arxiv.org/abs/1707.03141.

[25] YU W, TAN J, BAI Y, et al. Learning fast adaptation with meta strategy optimization[EB/OL]. arXiv, 2020[2023-06-13]. http://arxiv.org/abs/1909.12995.

[26] FAKOOR R, CHAUDHARI P, SOATTO S, et al. Meta-q-learning[EB/OL]. arXiv, 2020[2023-06-13]. http://arxiv.org/abs/1910.00125.

[27] FINN C, ABBEEL P, LEVINE S. Model-Agnostic Meta-Learning for Fast Adaptation of Deep Networks[C/OL]//Proceedings of the 34th International Conference on Machine Learning. PMLR, 2017: 1126-1135[2023-06-13]. https://proceedings.mlr.press/v70/finn17a.html.

[28] KAUSHIK R, ANNE T, MOURET J B. Fast online adaptation in robotics through meta-learning embeddings of simulated priors[EB/OL]. arXiv, 2021[2023-06-13]. http://arxiv.org/abs/2003.04663.

[29] Dijkstra E W. A note on two problems in connexion with graphs[M]//Edsger Wybe Dijkstra: His Life, Work, and Legacy. 2022: 287-290.

[30] 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.

[31] Stentz A. Optimal and efficient path planning for partially-known environments[C]//Proceedings of the 1994 IEEE international conference on robotics and automation. IEEE, 1994: 3310-3317.