arXiv:2602.02411v1 [cs.RO] 2 Feb 2026

面向杂乱空间中工期高效的物体重排的多智能体蒙特卡洛树搜索

Hanwen Ren+、Junyong Kim+、Aathman Tharmasanthiran 和 Ahmed H. Qureshi

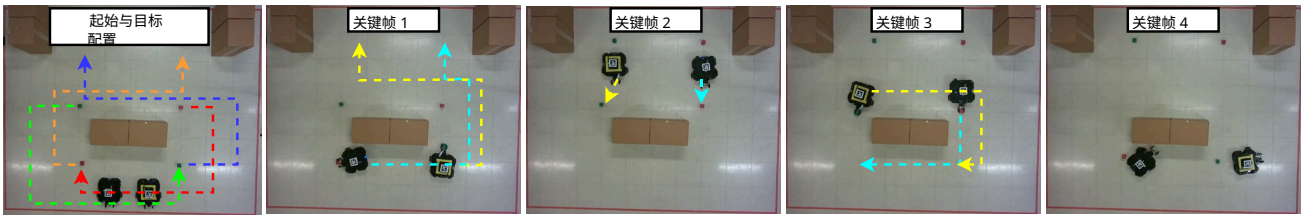


图 1：该图展示了一个非单调四物体搬迁任务。左图显示物体的起始状态和目标状态，其他图则描绘了关键的机器人抓取与放置动作，最终实现所需的物体重排配置on.

摘要——**在复杂、拥挤的环境中进行物体重排规划，是仓库、家庭和救援现场面临的一项常见挑战。以往研究大多针对单调实例，而现实任务往往是非单调的：物体彼此阻挡，必须先暂时移至中间位置，才能到达最终目标。在此类场景中，有效的多智能体协作能够显著缩短任务完成所需的时间。本文提出集中式异步多智能体蒙特卡洛树搜索（CAM-MCTS），这是一种面向复杂环境、用于通用高效完工时间物体重排规划的新框架。CAM-MCTS 将集中式任务分配与异步任务执行策略相结合：在集中式任务分配中，智能体能够了解彼此计划执行的动作，从而促进全局优化规划；异步任务执行策略则在适当的时间步为智能体分配新任务，而无需等待其他智能体，并以一步前瞻成本估计作为指导。这种设计最大限度地减少了空闲时间，避免了不必要的同步延迟，并提升了系统整体效率。我们在拥挤环境中的多种单调和非单调任务上评估了CAM-MCTS，结果表明，与强基线方法相比，该方法始终能够缩短完工时间。最后，我们在不同配置下的真实多智能体系统中验证了该方法，进一步证实了其有效性和鲁棒性。**

一、引言

物体重排规划是日常生活中的常见任务——例如，整理货架物品以高效利用空间，或在物流中心对新送达的包裹进行排序。它们使仓库能够实现库存管理自动化并降低运营成本；在灾害现场，则使救援队能够部署机器人，通过移动残骸来清理受阻的路径。多智能体系统尤其适合这些任务，因为机器人之间的协作能够缩短完工时间，并相较于单个机器人提高效率。图1展示了一个示例，其中两个机器人协同解决一项非单调重新定位任务。

尽管人类觉得物体重排很自然，但对多智能体系统而言，这项任务仍面临重大挑战。首先，即使在单机器人场景中，该问题通常也是 NP难 [1], [2] 问题，因为规划必须同时考虑高层任务分配和底层运动生成。其次，其他因素也会进一步增加复杂性：在非单调实例中，如果物体的目标区域最初被占据，就必须多次移动某些物体；碰撞避免等环境约束也会增加难度。最后，多智能体场景使这些挑战更加复杂——任务规划器必须评估大量任务分配组合，才能获得高质量的解；而运动规划器则必须为受限工作空间中的所有智能体生成无碰撞轨迹。

现有大多数工作研究单机器人物体重排规划 [3]，通常使用固定机械臂操作器进行桌面重排，或使用移动机器人在受限环境中作业 [4], [5]。这些方法通常依赖树搜索，其中节点表示由单机器人动作连接的环境状态；搜索从初始配置开始扩展，直至所有物体都被重新定位。近期研究已将问题扩展至多智能体系统，但大多数去中心化方法 [6], [7]仅限于单调的抓取与放置任务。处理非单调重排的方法 [8] 通常基于学习，需要大量示范数据，因此训练成本高昂。此外，这些方法强制采用同步执行：所有智能体必须完成当前任务后，才能分配新任务。因此，即使机器人提前完成任务，也不得不等待，导致推理期间的完工时间不必要地延长。

本文提出了集中式异步多智能体蒙特卡洛树搜索（CAM-MCTS）方法，用于在复杂杂乱环境中进行单调和非单调物体重排规划。我们的

+ 表示贡献相同。Hanwen Ren、Junyong Kim、Aathman Tharmasanthiran 和 Ahmed H. Qureshi 均就职于美国印第安纳州西拉斐特的普渡大学计算机科学系，邮编 47907。

方法为整个多智能体系统进行集中式规划，同时允许智能体异步行动——这一特性通常与去中心化方法相关联。这种异步行为使智能体无需等待其他智能体完成任务即可接手新任务，从而充分发挥多智能体协作的优势，并得出完工时间高效的解决方案。总之，本文的主要贡献如下：

- 一种集中式任务分配模块，根据各智能体的当前状态——正在执行先前分配的任务或处于空闲状态——来分配任务，从而显著缩小搜索空间。
- 一种异步任务执行策略，利用一步前瞻成本估计，使智能体能够在适当的时间步接取新任务，而不必等待其他智能体。
- 一种集中式、异步的多智能体规划器，利用蒙特卡洛树搜索范式，高效生成完工时间较短的物体重排方案。

我们通过在仿真和真实环境中进行全面测试，评估了所提出的方法在各种单调和非单调重排规划任务上的表现。结果表明，与表现强劲的基线方法相比，该方法持续提升了任务完成成功率，并缩短了完工时间。

II. 相关工作

与 CAM-MCTS 最相关的研究包括多智能体取送、物体重排规划和蒙特卡洛树搜索方面的方法。下文将讨论这些研究，并明确阐述我们的方法与既有工作的不同之处，以及在此基础上的推进。

多智能体取送（MAPD）问题 [9]旨在为多智能体系统制定任务和运动规划，以完成一系列取送操作。例如， [7] 提出基于边际成本和后悔值的任务选择策略，为智能体分配最合适的任务，并尽量减少总行程延误。同样，[10] 将任务分配表述为旅行商问题（TSP） [11]，通过成本估算为每个智能体生成任务序列，从而缩短完工时间。针对终身在线场景，也开发了多种解耦式 MAPD技术，例如令牌传递与任务交换方法 [6]，，在该方法中，智能体依次规划路径，甚至可以重新分配尚未执行的任务。

然而，这些MAPD方法本质上假设重排场景是单调的，即起始区域和目标区域在执行过程中始终保持可行 [12], [13]。因此，这些方法不适用于现实世界中的非单调任务；在这类任务中，物体往往相互阻挡，必须被移至中间位置。相比之下，CAM-MCTS明确处理单调和非单调物体重排，使其能够在复杂的迷宫式环境中有效规划。

物体重排规划 [14]是机器人学中一个研究充分的问题，尤其是在任务与运动规划（TAMP） [15], [16]中。该问题通常是 NP 难题 [1]，[2]，，因为它需要对高层任务规划和低层运动规划进行联合推理。大多数现有研究都在各种环境中采用单智能体场景 [17]，[18]来解决该问题。与单个智能体相比，多智能体系统能够更高效地解决复杂任务 [19]。然而，应用多智能体系统也带来了重大挑战：任务分配空间呈组合式增长 [20]，，运动规划问题也变得更加复杂。

为应对这些挑战， [8]，提出了一种基于多智能体学习的框架，通过迭代选择物体、确定重定位区域并将其分配给机器人。然而，该框架依赖大量示范数据，导致训练成本高昂；同时，它假设任务同步执行，迫使所有机器人等待其他机器人完成后才能继续。相比之下，我们的方法不基于学习，并支持异步执行，使智能体能够在新任务可用时立即接手，从而减少空闲时间并提高整体效率。

蒙特卡洛树搜索（MCTS）是一种通用算法，结合了树搜索的精确性与随机采样的普适性 [21]。借助蒙特卡洛模拟过程，MCTS 几乎不需要或完全不需要领域知识；因此，它在人工智能领域取得了巨大成功 [22]。在围棋中，AlphaGo 和 AlphaZero [23], [24] 将 MCTS 与经过训练的神经网络相结合，以预测下一步的搜索概率以及当前阶段的胜者，并击败了当时排名最高的人类围棋棋手。受 MCTS 在游戏人工智能领域取得成功的推动，它也被应用于调度、规划和组合优化等其他领域 [25]。例如， [4]的作者通过解耦的多阶段蒙特卡洛树搜索，解决了狭窄受限环境中的非单调物体重排规划问题。 [26]的研究将 MCTS 与生成式循环神经网络（RNN）相结合，用于动态环境中的路径规划，并通过提高运动预测准确率展现出良好的性能。尽管 MCTS 主要被用作生成同步解的集中式方法，我们的方法引入了异步特性，从而缩短了完工时间。

III. 问题表述

将工作空间表示为 $\mathcal{W} \subseteq \mathbb{R}^2$。在该空间中，存在一组不可移动障碍物 $\mathcal{B}$、一组物体 $\mathcal{O} = \{o_1, \ldots, o_n\}$ 和一个由机器人智能体组成的团队$\mathcal{Q} = \{q_1, \ldots, q_m\}$。障碍物 $\mathcal{B}$构成环境边界，阻止智能体穿行其中。在任意时间步 t，各智能体的位置 p_q^t 构成智能体位置集合 $\mathcal{P}_Q^t \subseteq \mathcal{W} \backslash \mathcal{B}$，而各物体的位置 p_o^t 构成物体位置集合 $\mathcal{P}_O^t \subseteq \mathcal{W} \backslash \mathcal{B}$。每个智能体都能够拾取物体，并将其放置在可行位置。我们将智能体 q拾取物体 o_i 的动作表示为 $r^{pick}{}_{(q,o_i)}$，将放置物体 o_i 的动作表示为 $r^{place}{}_{(q,o_i)}$。与每次拾取或

放置动作相关联的智能体路径，表示为智能体位置的有序序列 $v^{pick/place}(_{q,o_i}) = [p_q^t, p_q^{t+1}, \ldots, p_q^{t+k}]$。请注意，在上述智能体路径中，智能体位置可以在若干个时间步内保持不变，这意味着它可以选择空闲。随后，单个智能体 q_i 的任务分配可表示为一系列拾取和放置动作 $\pi_q = \{(r^{pick}(_{q,o_i})\, r^{place}(_{q,o_i})) \ldots, (r^{pick}(_{q,o_j})\, r^{place}(_{q,o_j}))\}$，以及相应的路径 $v_q = \{(v^{pick}(_{q,o_i})\, v^{place}(_{q,o_i})) \ldots, (v^{pick}(_{q,o_j})\, v^{place}(_{q,o_j}))\}$。最后，所有智能体任务分配的集合构成多智能体系统的全局任务分配 $\pi = \{\pi_{q_i}, \ldots, \pi_{q_m}\}$，并对应全局路径 $V = \{v_{q_i}, \ldots, v_{q_m}\}$。最后，对于某个全局任务分配$\pi$，假设智能体移动到相邻位置需要一个时间步，则完工时间 T定义为所有智能体所需时间步数中的最大值，这也等于最大路径长度，因此 $T = \max_{k=1}^{m} |v_{q_k}|$。

在物体排列规划任务中，输入由物体起始位置配置 $a^s = \{p_{o_1}^s, \ldots, p_{o_n}^s\}$、物体目标位置配置 $a^g = \{p_{o_1}^g, \ldots, p_{o_n}^g\}$ 以及智能体起始位置配置 $\{p_{q_1}^s, \ldots, p_{q_m}^s\}$ 组成。根据是否需要先将某些物体移至缓冲区域，再将其送至实际目标位置，任务可进一步分为单调实例和非单调实例。给定这些输入，我们的目标是为多智能体系统 $\pi = \{\pi_{q_1}, \ldots, \pi_{q_m}\}$ 找到无碰撞全局任务分配及其对应路径$V = \{v_{q_1}, \ldots, v_{q_m}\}$，以便将物体从起始配置 a^s 移至目标配置 a^g，并使单调实例和非单调实例的完工时间 T 最短。该问题可形式化表述为：寻找全局策略 π 及其对应路径 V，使得

$$\text{minimize} \quad \max_{k=1}^{m} |v_{q_k}| \text{ subject to}$$
$$\pi(a^s) = a^g \text{ and } p_{q_k}^t \neq p_{q_u}^t \quad \forall\, k \neq u,\ \forall\, 0 \leq t \leq T \quad (1)$$

IV. 方法

本节介绍我们提出的集中式、异步多智能体蒙特卡洛树搜索（CAM-MCTS）方法，用于在复杂且拥挤的环境中进行物体重排规划。现有文献 [25]声称，蒙特卡洛树搜索非常适合解决可建模为马尔可夫决策过程（MDP） [27]的问题。因此，在我们的集中式搜索树中，每个节点都会存储当前问题求解过程的所有信息，包括当前物体配置$a^t = \{p_{o_1}^t, \ldots, p_{o_n}^t\}$、目标物体配置 $a^g = \{p_{o_1}^g, \ldots, p_{o_n}^g\}$以及每个智能体的当前状态 q_i。智能体的状态包括其当前位置 $p_{q_i}^t$ 和手持物体 o_j。此外，父树节点和子树节点通过异步任务执行相连，并记录相应智能体的路径 $v = \{v_{q_1}, \ldots, v_{q_m}\}$。接下来的段落将按照由选择、扩展、仿真和反向传播组成的标准蒙特卡洛树搜索范式，详细介绍我们的 CAM-MCTS。

A. 选择

在每次迭代中，CAM-MCTS 通过依次遍历上置信界（UCB） [28]值最大的子节点来选择一个叶节点。在我们的设计中，UCB 的构造如下：

$$UCB = -\frac{\sum_{i=1}^{h}(\sum_{k=1}^{m} |v_{q_k}| + \alpha \max_{k=1}^{m} |v_{q_k}|)}{B_n} + 2\sqrt{\frac{\log B_p}{B_n}} \quad (2)$$

第一项的分子是累积成本的相反数，该成本定义为多智能体系统的总行驶距离$\sum_{k=1}^{m} |v_{q_k}|$ 与截至目前已执行的所有 h次仿真中的最大完工时间 $\max_{k=1}^{m} |v_{q_k}|$ 的加权组合。可调超参数α 用于平衡移动距离与完工时间之间的成本。通过在选择过程中最大化 UCB 值，我们选择最有希望的叶节点，这些节点具有较短的移动距离和较短的完工时间。方程 2 中的 B_p和 B_n 分别是父节点和所选节点的访问次数。

B. 扩展

根据所选节点的状态，扩展模块会提出各种后续任务和运动规划，使其向解决问题迈进一步。我们设计的两个主要组成部分是集中式任务分配方法和异步任务执行策略。

1) 集中式任务分配：集中式任务分配方法会生成多个候选规划，然后为每个规划中的每个智能体分配最合适的任务。在物体重排规划过程中的任意时间步，所有手中持有物体的智能体都会被归入活动智能体集合，其余智能体则被归入空闲集合。根据智能体所属的集合，将采用不同的策略。

活跃智能体 Q_{active} 被迫不断重新定位手中的物体。然而，重新定位区域取决于每个活跃智能体所属的类别。第一类智能体刚抵达物体 o_j的拾取区域，需要确认物体 $a^g[o_j]$ 的目标位置是否被占用。如果有其他物体尚未被任何智能体拾取，则将该目标位置标记为已占用。如果目标区域空闲，则直接将目标设为期望的重新定位区域；否则，将一个缓冲位置 $p_{o_j}^b$ 作为临时目的地。我们的方法通过从以目标为中心的标准正态分布中采样一个位置来设置缓冲位置。缓冲区域必须可行，且不能是其他活跃智能体当前的目标。将缓冲位置采样在目标附近，旨在推动任务尽可能乐观地取得进展，因为从缓冲区域到实际目标的最后一步可以尽量缩短。当同时生成多个缓冲位置时，必须确保它们彼此可区分。第二类活跃智能体是先前所述、正在前往缓冲位置以重新定位物体的智能体。我们的方法会重新验证手中物体的目标位置是否可行。

如果目标区域空闲，智能体会将目的地从缓冲区改为实际目标。否则，智能体会沿当前路径继续前往缓冲区。最后，第三类正在前往目标区域的智能体，其任务保持不变。

另一方面，空闲智能体 Q_{idle} 存在多个潜在的任务分配方案。我们将剩余物体按空闲智能体的数量进行所有可能的组合。然后，对于每种组合，根据欧几里得距离为智能体指定最近的物体进行拾取；欧几里得距离是一种可采纳启发式，常用于各种基于搜索的方法进行成本估算 [29]。通过将每个空闲智能体方案与活跃智能体方案相结合，生成多智能体系统的最终任务分配。

2）异步任务执行：根据任务分配，任务执行模块会计算相应的路径，并提出最合适的时间步来终止当前迭代，使得较早完成任务的智能体可以开始执行新任务，从而实现异步任务执行。与同步设计相比，异步设计的主要优势在于提高了完工时间效率，因为智能体不必等待最后一个智能体完成任务后才能执行下一个可用任务。因此，我们提出了一种新颖的异步任务执行设计，利用一步前瞻进行成本估算。其基本思路如图 2 所示。在图中所示的示例中，我们的方法会提前结束当前迭代，因为等待智能体3会妨碍智能体1和智能体2完成下一个可用任务。我们的方法兼顾了解的质量与可扩展性，从而为长时域物体重排规划任务提供完工时间高效的解决方案。

算法1：异步任务执行

数据： $\pi = \{p_{q_1}, \ldots, p_{q_m}\}$

1 v =ICBS(π) ▷ 同步任务执行路径
2 $\{(t_i, q_i)\}_{\{m\}}$ = FindFinishTime(v).sort()
3 $t_h = \infty$ ▷ 容差时域
4 t' =无 ▷ 异步执行结束时间
5 **对于** $(t_i,\ q_i) \in \{(t_i,\ q_i)\}_{\{m\}}$ **执行**
6 **如果** $t_i \le t_h$ **则**
7 $t' = t_i$
8 **如果** $q_i \in Q_{idle}$ **则**
9 $t_h = \min(t_h,\ t_i + \min_{j=1}^{k} \text{C2G}(p_{q_i},\ p_{o_j}))$
10 **否则，如果** $q_i \in Q_{active}$ **则**
11 $t_h = \min(t_h, t_i + \text{C2G}(p_{q_i}, p_{o_j}))$
12 **else**
13 跳出
14 **对于** $v_{q_i} = [p_{q_i}^{t} \ldots, p_{q_i}^{t+k}] \in v$ **执行**
15 $v_{q_i} = v_{q_i}[0 : t']$
16 **返回** v ▷ 异步任务执行路径

算法1展示了该算法的工作流程。首先，



图 2：该图展示了智能体 A1 至 A4 的异步任务执行设计。从同步任务执行（a）出发，我们的方法利用一步前瞻成本估算（b），找到最合适的时间步 t'_a（c）来终止当前迭代。这样，智能体 A1 和 A2 无需等待 A3 和 A4，即可开始执行新任务。

给定任务分配 π，我们首先采用改进的基于冲突的搜索（ICBS）方法 [30] 生成初始同步任务执行路径 v （第 1 行）。ICBS 以分布式方式为每个智能体生成一条路径，并扩展二叉约束树，逐步解决冲突。随后，给定初始路径 v，我们按任务完成时间从短到长对智能体排序（第 2 行）。接下来，我们遍历排序后的智能体列表，通过一步前瞻机制检查其等待时间是否足够长，以便接取下一个可用任务。若是，我们希望提前结束执行，让等待中的智能体能够接取新任务。为此，我们设定一个容差时域 t_h，表示所有等待中的智能体都不可能完成任何后续任务的最长时间步数。然后，对于排序列表中的每个智能体，如果其任务完成时间 t_i 早于容差时域 t_h，我们就将异步执行结束时间 t' 设为 t_i ，并根据以下两种情况更新容差时域 t_h 。首先，假设该智能体在完成任务时 t_i变为空闲。在这种情况下，与其对应的容差时域是前往最近物体起始位置所需的时间步数，该时间由到达目标代价（C2G）函数估算（第 8–9 行）。但是，如果该智能体在时间 t_i手持物体，则将容差时域设为预计将手持物体重新布置到目标位置所需的时间（第 10–11 行）。到达目标代价函数通过将起始区域与目标区域之间的曼哈顿距离乘以成本比率，来预测任务完成时间。该成本比率根据搜索树中生成的所有路径，记录智能体在工作空间中移动一个单位距离所需的平均时间步数。另一方面，任务完成时间为

晚于时间 t_i t_h 的智能体不会在当前迭代中完成任务，因为等待它们会阻止其他智能体完成下一个潜在任务（第 12–13 行）。最后，根据 t'对智能体的路径进行分段，从而实现异步任务执行（第 14–15 行）。

利用问题的当前状态和前面介绍的两个子模块，MCTS 扩展方法会根据可能的任务分配创建固定数量的新树节点，并将每个节点与相应的异步任务执行路径关联起来。

C. 并行模拟与顺序反向传播

模拟过程会从一个未访问的树节点开始，逐步扩展一棵病态树，直至问题得到解决。在我们的 CAM-MCTS 中，为了提高时间效率，会并行执行未访问树节点的模拟步骤。回顾公式 2 中所示的 UCB 计算。对于未访问的树节点，由于访问次数 B_n 为 0，其 UCB 值为无穷大，这意味着必须对所有已扩展的树节点执行模拟。由于搜索具有 MDP 特性，对所有新创建的树节点并行执行模拟，所得结果与按顺序执行相同。然而，并行模拟所有节点时，奖励会被缓存，而不是反向传播；只有在之后选中相应节点时，才会使用这些奖励。

D. 结果提取

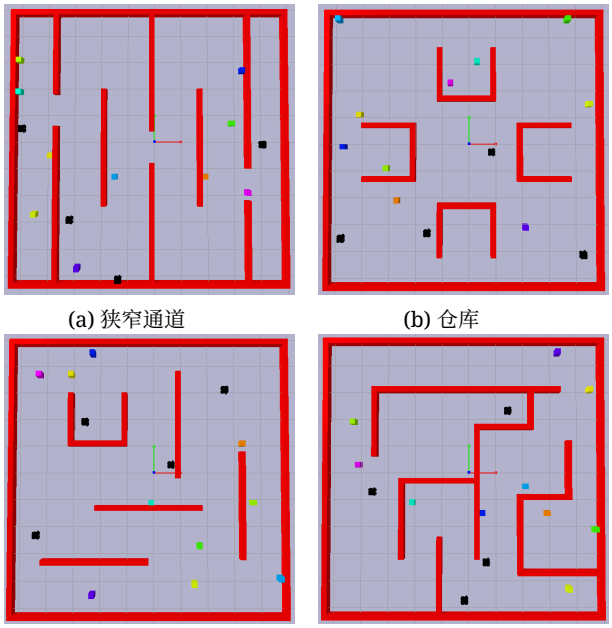
当选择过程中得到的树节点表明所有任务均已完成时，我们的 CAM-MCTS 即宣告成功，即所有物体均已从起始配置重新放置到目标配置。最终的异步执行路径 $V = \{v_{q_1}, \ldots, v_{q_m}\}$和全局策略 π 可通过反向树遍历直至根节点，再经一次逆序操作恢复。

V. 实验

在本节中，我们介绍模拟设置、基线比较、消融研究以及真实世界实验。

A. 模拟设置

我们创建了四个复杂的模拟环境，分别代表狭窄通道、仓库、随机障碍物和迷宫场景，每个环境的尺寸为（50×50）。对于每个场景，我们为由2至4个智能体组成的团队放置4至10个物体以进行重新定位，从而形成20种物体-智能体组合，范围从（2个智能体，4个物体）到（4个智能体，10个物体）。在物体数量超过七个的场景中，我们纳入四个智能体，否则任务分配将过于简单。然后，对于每种组合，物体的起始配置和目标配置按照三种方案设置：（1）随机——起始和目标均随机生成。（2）排序——起始随机生成，而目标遵循结构化模式。（3）打乱——起始和目标共享同一组位置，但以不同配置排列。随机和排序场景涵盖单调与非单调物体



(a) 狭窄通道　(b) 仓库　(c) 随机障碍物　(d) 迷宫

图3：本图展示了狭窄通道、仓库、随机障碍物和迷宫环境。彩色立方体是物体，而方块状的是机器人。

重排规划任务，而洗牌案例仅包含非单调任务。我们总共创建了480个测试实例，以进行全面评估。不同环境中的测试实例示例见图3。为定量评估性能，我们使用以下指标

•**成功率（SR）**：成功率是已求解案例所占的百分比，未在300秒内完成的案例被定义为失败。

• **规划时间（PT）**：规划时间用于记录规划器求解该实例所消耗的时间。

• **行进距离（TD）**：该指标记录了物体重排规划过程中所有智能体的总行进距离。

• **总时长（MS）**：如问题表述部分中的公式所述，完工时间量化了重排规划任务所需的总时间步数。

B. 仿真结果

1）基线比较：我们考虑以下两种基线方法，以与我们的方法进行比较：

• **随机同步规划器（RSP）**：RSP 是一种同步树搜索方法，它根据问题的当前状态持续随机生成任务分配，直到问题得到解决。

• **集中式异步多智能体一致代价搜索（CAM-UCS）：**在这种方法中，我们用一致代价搜索方法替代蒙特卡洛树搜索框架[31]。CAM-UCS 使用与 CAM-MCTS 相同的代价函数，通过扩展最低代价节点来保证最短完工时间。

（#物体，#智能体）	CAM-MCTS（本文方法）				RSP				CAM-UCS			
	SR ↑	PT ↓	TD ↓	MS ↓	SR ↑	PT ↓	TD ↓	MS ↓	SR ↑	PT ↓	TD ↓	MS ↓
(6,2)	**100.0**	2.7	430	239	83.3	**0.4**	479	329	58.3	160.8	**385**	**205**
(6,3)	**100.0**	5.8	405	163	100.0	**0.4**	435	247	62.5	115.7	**372**	**145**
(7,2)	**100.0**	5.6	**512**	**281**	100.0	**0.7**	574	426	0.0	-	-	-
(7,3)	**100.0**	6.5	502	202	100.0	**0.5**	555	323	8.3	220.2	**397**	**150**
(8,2)	**100.0**	10.2	**562**	**306**	83.3	**0.7**	650	448	0.0	-	-	-
(8,3)	**100.0**	12.7	575	227	100.0	**0.5**	613	349	4.2	1.8	**393**	**172**
(8,4)	**100.0**	12.6	532	168	95.8	**0.6**	547	244	12.5	115.7	**437**	**133**
(9,2)	**100.0**	13.4	**616**	**337**	95.8	**0.8**	669	467	0.0	-	-	-
(9,3)	**100.0**	31.2	**608**	**235**	100.0	**0.6**	655	362	0.0	-	-	-
(9,4)	**100.0**	23.5	**600**	**185**	100.0	**0.6**	614	293	0.0	-	-	-
(10,2)	**100.0**	21.6	**733**	**400**	91.7	**1.0**	799	553	0.0	-	-	-
(10,3)	**100.0**	39.7	**720**	**275**	100.0	**1.0**	774	445	0.0	-	-	-
(10,4)	**100.0**	40.9	**720**	**219**	100.0	**0.9**	771	362	0.0	-	-	-

表 I：本表比较了我们的 CAM-MCTS 方法与基线方法。总体而言，CAM-MCTS 在成功率、解的完工时间和可扩展性方面表现最佳。CAM-UCS 在物体较少的场景中表现良好，但规划时间（PT）非常长，且随着任务难度增加，其成功率降至零，表明其可扩展性较差。RSP 的规划时间较短，成功率与我们的方法相当，但完工时间极长，限制了其实用性。总之，CAM-MCTS 的综合表现最佳：它能够扩展至复杂场景，保持最低的完工时间，同时规划时间也在可接受范围内，因此是多智能体重排任务的理想规划器。

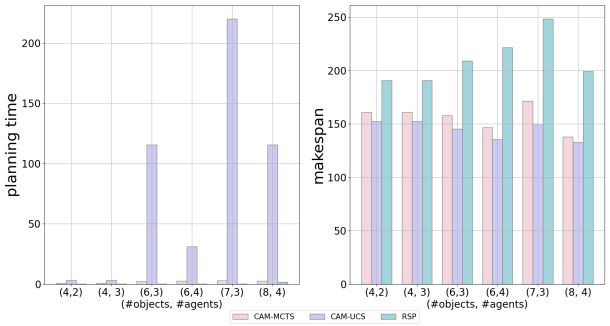


图 4：这些结果仅比较 CAM-MCTS、CAM-UCS 和 RSP 在所有方法均成功的场景中的表现，应结合表 1 一并解读。如图所示，在物体较少的任务中，CAM-UCS 的完工时间略短，但仅以微弱优势胜过 CAM-MCTS，代价则是规划时间显著增加。另一方面，RSP 的规划时间较短，但完工时间最长。综合来看，这些结果表明，CAM-MCTS 在规划时间、完工时间以及应对更具挑战性场景的可扩展性之间实现了最佳权衡。

请注意，先前的物体重排规划方法主要关注单调任务，因此不适用于我们的场景。对于非单调情形，近期唯一一种基于学习的多智能体方法是 Maner [8]，但它需要大量训练数据。相比之下，我们的方法不基于学习，因此避免了这一局限。

表 I 报告了难度级别逐步提高时的结果；除成功率外，所有指标均对成功实例取平均。图 4 仅在所有方法均成功的场景中比较完工时间与规划时间。

RSP 的运行速度快于 CAM-MCTS，但完工时间表现较差。随着任务复杂度增加，CAM-MCTS 一直在成功率、行驶距离和完工时间方面优于 RSP。这一优势源于 CAM-MCTS 的集中式任务分配模块：该模块会生成多个高质量候选规划，并通过 MCTS 模拟对其进行评估。相比之下，RSP 会随机选择有效分配，往往舍弃最优选择。随着任务层级加深，这些随机决策不断累积，导致解的质量不稳定且方差较大。

在物体与智能体较少的简单情形中，CAM-UCS 的完工时间优于 CAM-MCTS，但物体数量超过七个后，其性能急剧下降，成功率因时间限制而降至接近于零。UCS 会扩展当前代价最小的节点，而忽略长期潜力。该策略在小规模问题中有效，但随着复杂度上升，会迫使 UCS 探索多得多的状态，从而严重降低效率。相比之下，蒙特卡洛树搜索扩展最有前景的节点，从而改善整体可扩展性。

最后，图 4 在所有方法均成功的情形下比较了 CAM-MCTS、CAM-UCS 与 RSP（需结合表 1 解读）。在小型任务中，CAM-UCS 的完工时间略好，但仅以微弱优势实现，且规划代价高得多。RSP 虽然更快，却具有最高的完工时间。总体而言，CAM-MCTS 在规划时间、完工时间与可扩展性之间取得了最佳平衡，因而成为具有挑战性的多智能体重排任务中最有效的规划器。

2) 消融研究：我们还通过以下两种方法开展消融研究，以验证异步任务执行模块中单步前瞻设计的有效性，以及异步框架的使用效果：

- **不含前瞻的 CAM-MCTS：** 该方法以智能体完成任务后立即终止迭代的思路，取代了我们方法中的单步前瞻设计，从而允许立即重新分配任务。

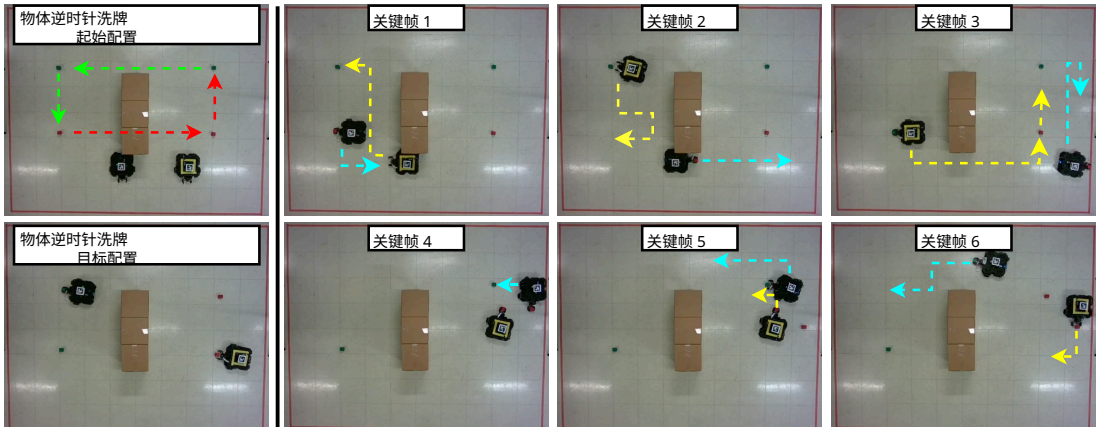


图 5：该图展示了一个非单调四物体逆时针洗牌案例。左侧展示起始配置和目标配置，红色和绿色箭头表示每个物体的目标位置。其余图像表示涉及物体抓取与放置的关键帧。我们的 CAM-MCTS 在 0.83 秒内找到异步解，而实际执行耗时 394 秒。

(#物体，#智能体)	CAM-MCTS（我们的方法）				不含前瞻的 CAM-MCTS				CSM-MCTS			
	SR ↑	PT ↓	TD ↓	MS ↓	SR ↑	PT ↓	TD ↓	MS ↓	SR ↑	PT ↓	TD ↓	MS ↓
(6,2)	**100.0**	**2.7**	430	239	100.0	7.4	438	**233**	100.0	2.9	**420**	270
(6,3)	**100.0**	5.8	405	163	100.0	15.8	422	**156**	100.0	**2.9**	**383**	192
(7,2)	**100.0**	**5.6**	512	281	100.0	17.8	508	**268**	91.7	7.5	**484**	317
(7,3)	**100.0**	**6.5**	502	202	100.0	31.9	518	**196**	100.0	9.8	**477**	245
(8,2)	**100.0**	**10.2**	562	306	100.0	21.6	580	**301**	95.8	11.6	**540**	341
(8,3)	**100.0**	12.7	575	227	100.0	61.5	590	**217**	95.8	**11.2**	**538**	270
(8,4)	**100.0**	**12.6**	532	168	95.8	41.6	550	**161**	91.7	21.7	**508**	196
(9,2)	**100.0**	**13.4**	616	337	100.0	26.7	610	**317**	95.8	15.5	**568**	371
(9,3)	**100.0**	31.2	608	235	100.0	114.8	617	**228**	91.7	**20.3**	**565**	265
(9,4)	**100.0**	**23.5**	600	185	100.0	81.7	613	**178**	100.0	24.1	**563**	225
(10,2)	**100.0**	**21.6**	733	400	100.0	56.7	742	**389**	70.8	36.1	**686**	438
(10,3)	**100.0**	**39.7**	720	275	91.6	129.6	740	**260**	91.7	48.0	**669**	321
(10,4)	**100.0**	**40.9**	720	219	79.2	139.0	754	**218**	91.7	55.0	**691**	278

表 I：消融研究凸显了单步前瞻和异步任务执行策略对于解决长时域多智能体物体重排问题的重要性。CSM-MCTS 采用同步执行，即所有智能体都要等待彼此完成后才能继续。虽然这略微缩短了行驶距离，却显著增加了总完工时间。单步前瞻策略展现出明显优势。虽然不采用该策略时，该方法实现了较短的完工时间，但代价是规划时间大幅增加、成功率降低。综合来看，这些结果表明，CAM-MCTS 在规划时间、行驶距离和完工时间之间实现了最佳权衡，是一种用途广泛且有效的物体重排规划器。

•集中式同步多智能体蒙特卡洛树搜索 (CSM-MCTS)：CSM-MCTS 是我们方法的同步版本，所有智能体同时拾取并重新定位物体。

结果汇总于表 II。无论在成功率还是完工时间方面，我们的方法始终优于同步基线 CSM-MCTS。这一改进主要得益于我们的异步任务执行策略：智能体不会在等待其他智能体完成任务时闲置，而是立即承担新任务，从而提高整体效率并缩短完工时间。此外，将每个树节点的子节点数限制为固定值，可以防止搜索树不受控制地增长，从而提高具有挑战性的实例上的成功率。与不具备前瞻能力的 CAM-MCTS 相比，尽管完工时间略长，我们的方法在复杂场景中仍能取得更高的成功率和更好的时间效率。

总体而言，将蒙特卡洛树搜索与异步执行相结合，显著增强了长时域多智能体重排任务的可扩展性，并在规划时间、行驶距离和完工时间之间实现了均衡权衡。

C. 真实机器人实验

我们在一个尺寸为（300 cm，300 cm）的竞技场中，部署由两台 TurtleBot3 机器人组成的多智能体系统，并在其中运行 CAM-MCTS。我们设计了五个涉及四个物体的非单调重排任务，以验证所提方法在真实环境中的性能，其中一个示例见图 1。实验中，我们的方法成功解决了所有任务，平均规划时间为 0.45 秒，平均执行时间为 383 秒。图 5 展示了一个成功解决的非单调物体逆时针洗牌任务。CAM-MCTS 在 0.83 秒内找到了有效规划。在生成的规划中，

机器人执行异步任务（关键帧 1-3），并相互协作（关键帧 4-6），以较短的完工时间完成物体重排任务。其他场景可在我们的补充视频中查看。

VI. 结论与未来工作

本文提出了一种新颖的集中式、异步、多智能体完工时间优化规划器，用于在复杂杂乱环境中进行物体重排，并处理单调和非单调任务。我们的方法将集中式任务分配与异步任务执行相结合，为多智能体系统生成多样且高质量的任务分配方案。借助蒙特卡洛树搜索框架，该规划器能够生成高效的物体重排方案，缩短完工时间并显著提升系统整体性能。实验结果表明，在不同难度级别下，无论是单调还是非单调场景，我们的方法始终优于基线方法。未来，我们计划进一步提升可扩展性，以处理更复杂的情况，并通过引入学习型奖励预测函数来加速节点评估，从而增强 MCTS 模拟。

参考文献

[1] John Reif 和 Micha Sharir。存在移动障碍物时的运动规划。美国计算机协会学报（*JACM*），41(4):764–790，1994。[2] Gordon Wilfong。存在可移动障碍物时的运动规划。载于第四届计算几何年度研讨会论文集，第279–288页，1988。[3] Rui Wang、Yinglong Miao 和 Kostas E Bekris。杂乱狭窄空间中高效且高质量的抓取式重排。载于*2022*年国际机器人与自动化会议（*ICRA*），第1968–1975页。IEEE，2022。[4] Hanwen Ren 和 Ahmed H. Qureshi。用于狭窄受限环境中非单调物体重排规划的多阶段蒙特卡洛树搜索。载于*2024*年*IEEE/RSJ*智能机器人与系统国际会议（*IROS*），第12078–12085页，2024。[5] Junyong Kim、Hanwen Ren 和 Ahmed H Qureshi。集成主动感知与重排规划，以便从未知的受限杂乱环境中高效取回物体。载于*2025*年*IEEE*机器人与自动化国际会议（*ICRA*），第12572–12578页。IEEE，2025。[6] Hang Ma、Jiaoyang Li、TK Kumar 和 Sven Koenig。面向在线取送任务的终身多智能体路径规划。*arXiv*预印本 *arXiv:1705.10868*，2017。[7] Zhe Chen、Javier Alonso-Mora、Xiaoshan Bai、Daniel D. Harabor 和 Peter J. Stuckey。面向容量约束多智能体取送的集成任务分配与路径规划。*IEEE*机器人与自动化快报，6(3):5816–5823，2021。[8] Vivek Gupta、Praphpreet Dhir、Jeegn Dani 和 Ahmed H. Qureshi。Maner：拥挤环境中的多智能体神经网络物体重排规划。*IEEE*机器人与自动化快报，8(12):8295–8302，2023。[9] Zefang Zong、Meng Zheng、Yong Li 和 Depeng Jin。Mapdp：通过协作式多智能体强化学习解决取送问题。载于*AAAI*人工智能会议论文集，第36卷，第9980–9988页，2022。[10] Minghua Liu、Hang Ma、Jiaoyang Li 和 Sven Koenig。多智能体取送的任务与路径规划。载于自主智能体与多智能体系统国际联合会议（*AAMAS*）论文集，2019。[11] Tolga Bektas。多旅行商问题：问题表述与求解方法概述。*omega*，34 219，2006 3 209( ): – 。[12] Oren Salzman 和 Roni Stern。多智能体路径规划和多智能体取送问题中的研究挑战与机遇。载于第*19*届自主智能体与多智能体系统国际会议论文集，第1711–1715页，2020。[13] Qinghong Xu、Jiaoyang Li、Sven Koenig 和 Hang Ma。多目标多智能体取货与配送。载于2022年IEEE/RSJ智能机器人与系统国际会议，第9964–9971页。IEEE，2022。[14] Huihui Guo、Fan Wu、Yunchuan Qin、Ruihui Li、Keqin Li 和 Kenli Li。机器人任务与运动规划的最新趋势：综述。《ACM计算调查》，55(13s):1–36，2023。[15] Caelan Reed Garrett、Rohan Chitnis、Rachel Holladay、Beomjoon Kim、Tom Silver、Leslie Pack Kaelbling 和 Tom´as Lozano-P´erez。集成任务与运动规划。《控制、机器人与自主系统年度评论》，4:265–293，2021。[16] Siddharth Srivastava、Eugene Fang、Lorenzo Riano、Rohan Chitnis、Stuart Russell 和 Pieter Abbeel。通过可扩展的规划器无关接口层实现任务与运动规划的结合。载于2014年IEEE机器人与自动化国际会议（ICRA），第639–646页。IEEE，2014。[17] Yann Labb´e、Sergey Zagoruyko、Igor Kalevatykh、Ivan Laptev、Justin Carpentier、Mathieu Aubry 和 Josef Sivic。用于高效视觉引导重排规划的蒙特卡洛树搜索。《IEEE机器人与自动化快报》，5(2):3715–3722，2020。[18] Hanwen Ren 和 Ahmed H. Qureshi。利用机器人手持RGB-D传感器感知受限空间中的物体取回神经重排规划。载于2024年IEEE国际机器人与自动化会议（ICRA），第15388–15394页，2024。[19] Ali Dorri、Salil S. Kanhere 和 Raja Jurdak。多智能体系统：综述。《IEEE Access》，6:28573–28593，2018。[20] Martin Levihn、Takeo Igarashi 和 Mike Stilman。分配空间中的多机器人多物体重排。载于2012年IEEE/RSJ智能机器人与系统国际会议，第5255–5261页，2012。[21] Guillaume Chaslot、Sander Bakkes、Istvan Szita 和 Pieter Spronck。蒙特卡洛树搜索：游戏人工智能的新框架。载于AAAI人工智能与互动数字娱乐会议论文集，第4卷，第216–217页，2008。[22] Cameron B Browne、Edward Powley、Daniel Whitehouse、Simon M Lucas、Peter I Cowling、Philipp Rohlfshagen、Stephen Tavener、Diego Perez、Spyridon Samothrakis 和 Simon Colton。蒙特卡洛树搜索方法综述。《IEEE计算智能与游戏人工智能汇刊》，4(1):1–43，2012。[23]David Silver、Thomas Hubert、Julian Schrittwieser、Ioannis Antonoglou、Matthew Lai、Arthur Guez、Marc Lanctot、Laurent Sifre、Dharshan Kumaran、Thore Graepel 等。一种通过自博弈精通国际象棋、日本将棋和围棋的通用强化学习算法。《科学》，362(6419):1140–1144，2018。[24] Thomas M Moerland、Joost Broekens、Aske Plaat 和 Catholijn M Jonker。A0C：连续动作空间中的 Alpha Zero。arXiv预印本 arXiv:1805.09613，2018。[25] Maciej Swiechowski、Konrad Godlewski、Bartosz Sawicki 和 Jacek Ma´ndziuk。蒙特卡洛树搜索：近期改进与应用综述。《人工智能评论》，56 2562，2023 3 2497( ): – 。[26] Stuart Eiffert、He Kong、Navid Pirmarzdashti 和 Salah Sukkarieh。利用生成式循环神经网络和蒙特卡洛树搜索在动态环境中进行路径规划。载于2020年IEEE国际机器人与自动化会议（ICRA），第10263–10269页。IEEE，2020。[27] Martin L Puterman。马尔可夫决策过程。《运筹学与管理科学手册》，2:331–434，1990。[28] Aur´elien Garivier 和 Eric Moulines。关于切换多臂老虎机问题的上置信界策略。载于算法学习理论国际会议，第174–188页。Springer，2011。[29] Xiang Liu 和 Daoxiong Gong。完美迷宫中用于搜救的A*算法比较研究。载于2011年国际电气信息与控制工程会议，第24–27页。IEEE，2011。[30] Eli Boyarski、Ariel Felner、Roni Stern、Guni Sharon、Oded Betzalel、David Tolpin 和 Eyal Shimony。ICBS：改进的多智能体寻路基于冲突的搜索算法。载于组合搜索国际研讨会论文集，第6卷，第223–225页，2015。[31] Ariel Felner。立场论文：迪杰斯特拉算法与一致代价搜索，或反对迪杰斯特拉算法的理由。载于组合搜索国际研讨会论文集，第2卷，第47–51页，2011。