

基于场景语义感知与大语言模型推理的行为树生成^①



鄢龙武^{1,2}, 郑王里³, 林云汉^{1,2}

¹(武汉科技大学 计算机科学与技术学院, 武汉 430081)

²(武汉科技大学 智能信息处理与实时工业系统湖北省重点实验室, 武汉 430081)

³(国网电力科学研究院有限公司, 南京 211106)

通信作者: 林云汉, E-mail: yhlin@wust.edu.cn

摘 要: 具身智能 (embodied AI) 需要能够与环境进行互动和感知, 并具备自主规划、决策和行动等能力. 行为树 (BT) 由于其模块化和高效控制的特性, 已经成为机器人技术中广泛使用的方法. 然而, 现有的行为树生成技术在处理复杂任务时仍面临一定的挑战. 这些方法通常依赖于领域专业知识, 生成行为树的能力有限. 此外, 许多现有方法在语言理解方面存在不足, 或者在理论上无法保证行为树的成功, 从而导致在机器人上的实际部署难度较大. 本研究提出一种新的行为树自动生成方法, 该方法基于大语言模型 (LLM) 和场景语义感知, 生成包含任务目标的初始行为树. 本文的方法根据机器人的能力设计机器人动作原语和相关条件节点, 并以此设计提示 (prompt) 使 LLM 输出行为规划 (generated plan), 然后将行为规划转化为初始行为树. 虽然本文以此为示例, 但该方法具有广泛的适用性, 可以根据不同需求应用于其他类型的机器人任务. 同时, 本文将这种方法应用于机器人任务中, 并给出具体实现方法和示例. 在机器人执行任务过程中, 行为树可以根据机器人操作失误和环境变化动态更新, 对外部环境变化具有一定的鲁棒性. 本文进行了初始行为树生成验证实验, 并在仿真机器人环境中进行了验证, 展示了本文方法的有效性.

关键词: 具身智能; 大语言模型; 机器人操作; 行为树; 行为树生成

引用格式: 鄢龙武, 郑王里, 林云汉. 基于场景语义感知与大语言模型推理的行为树生成. 计算机系统应用, 2025, 34(1): 37-46. <http://www.c-s-a.org.cn/1003-3254/9742.html>

Behavior Tree Generation Based on Scene Semantic Perception and Reasoning with Large Language Models

YAN Long-Wu^{1,2}, ZHENG Wang-Li³, LIN Yun-Han^{1,2}

¹(School of Computer Science and Technology, Wuhan University of Science and Technology, Wuhan 430081, China)

²(Hubei Province Key Laboratory of Intelligent Information Processing and Real-time Industrial System, Wuhan University of Science and Technology, Wuhan 430081, China)

³(State Grid Electric Power Research Institute Co. Ltd., Nanjing 211106, China)

Abstract: Embodied AI requires the ability to interact with and perceive the environment, and capabilities such as autonomous planning, decision making, and action taking. Behavior trees (BTs) become a widely used approach in robotics due to their modularity and efficient control. However, existing behavior tree generation techniques still face certain challenges when dealing with complex tasks. These methods typically rely on domain expertise and have a limited capacity to generate behavior trees. In addition, many existing methods have language comprehension deficiencies or are theoretically unable to guarantee the success of the behavior tree, leading to difficulties in practical robotic applications.

① 基金项目: 国家重点研发计划 (2022YFB4700400)

收稿时间: 2024-06-14; 修改时间: 2024-07-18; 采用时间: 2024-08-01; csa 在线出版时间: 2024-11-15

CNKI 网络首发时间: 2024-11-18

In this study, a new method for automatic behavior tree generation is proposed, which generates an initial behavior tree with task goals based on large language models (LLMs) and scene semantic perception. The method in this study designs robot action primitives and related condition nodes based on the robot's capabilities. It then uses these to design prompts to make the LLMs output a behavior plan (generated plan), which is then transformed into an initial behavior tree. Although this paper takes this as an example, the method has wide applicability and can be applied to other types of robotic tasks according to different needs. Meanwhile, this study applies this method to robot tasks and gives specific implementation methods and examples. During the process of the robot performing a task, the behavior tree can be dynamically updated in response to the robot's operation errors and environmental changes and has a certain degree of robustness to changes in the external environment. In this study, the first validation experiments on behavior tree generation are carried out and verified in the simulated robot environment, which demonstrates the effectiveness of the proposed method.

Key words: embodied AI; large language model (LLM); robotic manipulation; behavior tree (BT); behavior tree generation

1 介绍

行为树 (behavior tree, BT) 是一种有向根树, 其中叶节点负责执行任务, 而分支节点则定义了控制流逻辑. 它的一个主要优势在于其在控制代理 (agent) 行为方面的灵活性. BT 提供了模块化和分层的表示, 每个任务或行为都可以被封装为树中的一个独立节点. 这使得在不破坏系统整体架构的前提下, 可以对代理的行为进行有效的修改和扩展. 因此, BT 在机器人领域得到了广泛的应用.

目前, 行为树多数是由人类专家手动设计的, 这需要丰富的经验和领域知识. 随着任务复杂度的增加, 手动设计行为树变得越来越复杂和耗时, 这促使了对自动生成行为树的研究. 然而, 现有的行为树生成技术在任务复杂性、模型适应性和可靠性方面仍然面临挑战, 这些技术在根据复杂任务描述生成行为树的能力上仍然有限, 并且在面临新场景时难以取得良好的性能.

大型语言模型 (LLM) 的出现为这种智能体系的发展提供了更为强大的推理工具, 通过语言理解和生成, 实现了对复杂任务更深层次的理解和响应. 到目前为止, 使用大语言模型来生成行为树的研究处于起步阶段, 一些研究在现有 LLM 基础上微调专用于 BT 生成的语言模型, 此方法需要大量专业的 BT 数据, 而大量专业 BT 数据的获取难度较大. 另一些研究采用设置提示 (prompt) 来生成 BT, 但生成的 BT 对环境的适应性较低, 部署在机器人上的难度较大.

为了解决这些问题, 本文注意到, LLM 擅长推断

语言条件的支持 (affordance) 和约束 (constraint), 因此本文将 LLM 输出的动作序列规定为生成行为树所需的特定格式. 同时, 为了使 LLM 能够理解任务场景并做出正确的推理, 本文将语义地图 (semantic map, 一个包含环境信息的 XML 文件) 和人类指令同时作为 LLM 的输入. 当得到正确的行为规划 (generated plan) 后, 这些行为规划可以转化为行为树, 机器人根据初始行为树执行任务. 在这个过程中, 行为树根据环境信息的变化或机器人自身的操作错误进行动态更新, 这种方式不需要对 LLM 进行单独的训练. 本文的贡献如下.

(1) 本文提出了一种创新且通用的行为树生成方法, 基于大语言模型和场景语义感知, 生成包含任务目标的初始行为树, 并根据机器人的能力设计动作原语和相关条件节点, 具有广泛的通用性.

(2) 本文将生成的行为树应用在机器人操作中并给出了具体实现框架和示例, 展示了其实际应用价值, 并指出该方法具有广泛的适用性, 可以应用于不同类型的机器人任务.

(3) 本文在机器人仿真环境中进行了实验来证明本文的方法的有效性, 并讨论了本文方法的优点和局限性.

本文的其余部分安排如下. 第 2 节是行为树相关背景及研究. 第 3 节展示了本文提出的方法. 第 4 节首先基于本文提出的方法设计了一个机器人总体实验框架, 进行机器人实验, 并基于实验对方法进行分析. 第 5 节进行结论分析和讨论未来的工作.

2 行为树相关背景及研究

行为树 (BT) 由控制流节点、执行节点和根节点组成. 控制流节点通过逻辑触发其子节点的执行. 执行节点则负责检查条件或执行具体操作. 行为树的执行从根节点开始, 根节点会依次激活其子节点. 被激活的节点会向其父节点返回成功、失败或运行中的状态. 本文重点关注以下节点用于生成初始行为树.

- 条件节点 (condition): 评估观察到的状态是否满足指定条件的叶节点, 并根据此评估返回成功或失败.

- 动作节点 (action): 控制机器人执行操作的叶节点, 返回成功、失败、运行, 具体取决于执行结果.

- 序列节点 (sequence): 一个内部节点, 只有当其所有子节点都成功时, 它才会返回成功. 否则, 它会从左到右勾选其子项, 第 1 个返回失败或运行的子项将决定其返回状态.

- 回退节点 (fallback): 逻辑与序列节点相反的内部节点. 仅当其所有子级都失败时, 它才会返回失败. 否则, 在执行期间首次出现成功或运行将成为其返回状态.

除了由专家手动构建 BT, 传统的 BT 生成方法主要分为基于强化学习、进化学习和演示学习的方法. 在 LLM 出现后, 一些研究开始尝试基于 LLM 构建 BT.

基于强化学习 (RL) 的 BT 生成: 强化学习已被证明在生成 BT 中是有效的^[1-3]. 这个过程需要定义状态和动作空间, 创建奖励函数, 建立 RL 模型, 并确定 BT 的适当表示.

基于进化学习的 BT 生成: Nicolau 等人^[4]通过采用选择、交叉和突变过程来模拟生物进化, 以根据适应度评估结果优化 BT. Iovino 等人^[5]遗传编程是用于 BT 生成的一种方法, 它涉及定义编码方法. Neupane 等人^[6]指定语法规则来约束进化算法的搜索空间, 从而简化解决问题.

基于演示学习的 BT 生成: French 等人^[7]从演示中学习可以通过观察专家提供的示例来获取复杂的多步骤任务. Francillette 等人^[8]通过观察人类行为来生成 BT.

Machalewski 等人^[9]使用半自动优化方法调整专家构建的 BT.

基于 LLM 的 BT 生成: 近期一些研究表明 LLM 可作为机器人的智能核心, 赋予其常识, 从而使其能够做出决策. 机器人充当物理执行者, 执行 LLM 生成的计划. 这些计划可以表示为动作序列^[10-12]、控制指令^[13]、代码^[14,15]、值图^[16]等, 然而, 对于 LLM 直接生成 BT 的方法, 需要收集大量的 BT 数据集对语言模型进行微调, 且不能保证 BT 的正确性和鲁棒性, 例如, Li 等人^[17]引入合成数据来训练 BT 生成模型 (BTGen 模型), 并进行数据验证. Lykoy 等人^[18]对 Stanford Alpaca 7B 模型进行微调进行 BT 生成. 通过设置提示来使 LLM 生成 BT 逻辑流可以减少对大量数据的依赖. 例如, Cao 等人^[19]通过阶段步进提示设计, 根据目标任务描述和少量样例实现 BT 生成. Zhou 等人^[12]通过端到端模型将动作序列转化为行为树, Chen 等人^[20]通过设置提示使 LLM 输出解释目标, 随后使用行为树扩展算法 (OBTEA) 构建 BT.

传统的行为树生成方法效率低下, 且严重依赖领域专业知识. 根据复杂任务描述生成 BT 的能力仍然不足, 对新环境的适应性也较低. 因此, 需要进一步研究以增强在动态和不可预测环境中生成 BT 的鲁棒性和通用性. 目前, 使用大型语言模型 (LLM) 来生成 BT 还处于起步阶段. 本文参考 ProgPrompt^[15]的提示结构, 生成满足特定条件的机器人行为规划, 随后将这些行为规划转化为初始行为树.

3 方法

3.1 初始行为树的生成

行为树生成流程如图 1 所示, 行为树 (BT) 生成过程中涵盖 5 个关键组件: 动作原语 (action primitive)、可用对象列表 (available objects list)、外部信息 (external info)、示例任务 (example task) 与任务描述 (task description). 这些组件构成了生成行为树的基础, 也是调用大语言模型的提示语.

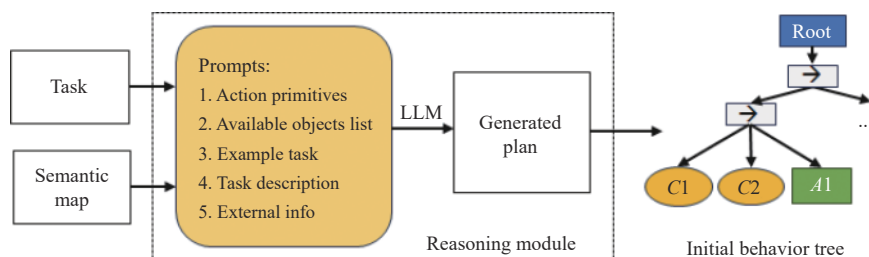


图 1 机器人初始行为树构建流程图

通过递归调用大型语言模型 (LLM) 来综合这些组件, 生成行为树规划 (generated plan), 最终将其转化为初始 BT, 实现任务规划的自动化.

动作原语: 动作原语即动作模板库 (action template library, ATL), 集合定义了机器人可以执行的基本动作, 这些动作直接映射到机器人的物理操作能力, 并为 LLM 提供了完成任务所需的动作框架. 在本方法中, 动作原语被设定如式 (1) 所示, 覆盖了可能的任务执行动作.

$$\text{action} = \{\text{pick}, \text{place}, \text{drop}, \text{rotate}, \text{move}, \text{push}\} \quad (1)$$

为了简化行为树的复杂性, 同时便于其后续更新, 针对预期的任务环境引入了一系列条件节点, 如式 (2) 所示. 条件节点例如 *IsObjectOnGoal* 用于判断任务目标是否完成, 当此条件节点返回正确的结果后将执行下一个节点.

$$\text{ConditionNode} = \left\{ \begin{array}{l} \text{IsObjectOnGoal}(\text{object}_i, \text{goal}_i), \\ \text{IsRobotApproachObject}(\text{object}_i), \\ \text{IsHandEmpty}(), \\ \text{IsObjectOnHand}(\text{object}_i), \\ \text{IsObjectApproachGoal}(\text{object}_i), \\ \text{IsObjectAligned}() \end{array} \right\} \quad (2)$$

在执行每个动作之前, 需先用相应的条件节点判断该动作是否满足执行条件, 本文的设计如下.

- 动作节点 *pick*: 机器人执行抓取动作, 从当前位置拾取一个物体.

条件节点: (1) *IsObjectApproachObject*: 检查机器人是否已接近目标对象. (2) *IsHandEmpty*: 检查机器人手是否空闲, 确保可以进行抓取动作.

- 动作节点 *place*: 机器人将手中的物体放置到目标位置.

条件节点: (1) *IsObjectOnHand*: 检查机器人手中是否有物体, 确保可以进行放置动作. (2) *IsObjectApproachGoal*: 检查机器人是否已接近放置目标位置.

- 动作节点 *drop*: 机器人在当前位置放下手中的物体.

条件节点: *IsObjectOnHand*: 检查机器人手中是否有物体, 确保可以进行放下动作.

- 动作节点 *rotate*: 将物体旋转到特定角度.

条件节点: (1) *IsObjectOnHand*: 检查机器人手中是否有物体. (2) *IsObjectAligned*: 检查物体是否对齐, 以

决定是否需要执行旋转操作.

- 动作节点 *move*: 机器人机械臂从一个位置移动到另一个位置.

条件节点: (1) *IsObjectApproachGoal*: 检查机器人是否已经接近目标位置, 确认是否需要移动. (2) *IsHandEmpty* (可选): 检查机器人手中是否为空, 若移动需要手空闲, 则需要此条件.

- 动作节点 *push*: 机器人执行推动动作, 将物体推到指定位置或清理障碍物.

条件节点: (1) *IsObjectApproachObject*: 检查机器人是否已接近目标对象, 确保可以进行推动动作. (2) *IsObjectOnGoal* (可选): 检查物体是否已经在目标位置, 若推动作是为了调整位置, 则需要此条件.

可用对象列表: 可用对象列表由当前环境的语义地图 (semantic map) 生成, 其中包含了各个对象的详细位置和属性信息, 为行为树的构建提供了必要的环境状态, 其具体实现方法见本文第 3.2 节.

示例任务: 示例任务旨在展示如何综合动作原语、对象列表和外部组件以执行具体的任务. 这一部分是使 LLM 理解行为树生成流程的核心, 由用户根据任务自定义, 它展示了从任务描述到动作执行的转换过程.

任务描述: 任务描述提供了需要执行的任务的详细信息和预期结果, 它定义了行为树生成算法的具体执行目标.

外部组件: 在本方法中, 外部组件 *get_goal* 利用 LLM 的推理能力, 根据任务描述和语义地图, 确定物体的最优放置位置, 如算法 1 所示.

算法 1. *get_goal*

输入: SemanticMap, TaskDescription, OccupiedPositions.

输出: OptimalGoalPosition.

```

1. Function get_goal(SemanticMap, TaskDescription, OccupiedPositions):
2.   OptimalGoalPosition ← None
3.   ParsedSemanticMap ← parse(SemanticMap)
4.   ParsedTask ← parse(TaskDescription)
5.   CandidatePositions ← empty list
6.   for each location in ParsedSemanticMap:
7.     if check(location, ParsedTask)
       and not location in OccupiedPositions:
8.       CandidatePositions.add(location)
9.   best_score ←  $-\infty$ 
10.  for each position in CandidatePositions:
11.    score ← evaluate_position(position, ParsedTask)
12.    if score > best_score:

```

```

13. best_score ← score
14. OptimalGoalPosition ← position
15. return OptimalGoalPosition

```

该算法的目标是确定任务中物体的最优放置位置。该函数首先解析语义地图和任务描述,然后筛选出未被占用且符合任务要求的所有可能位置,对每个候选位置进行评分,最终选取得分最高的位置作为最优放置点。此外,辅助函数 *is_occupied* 用于检查给定位置是否已在之前的计划中被占用,以确保所选位置的有效性和适用性。

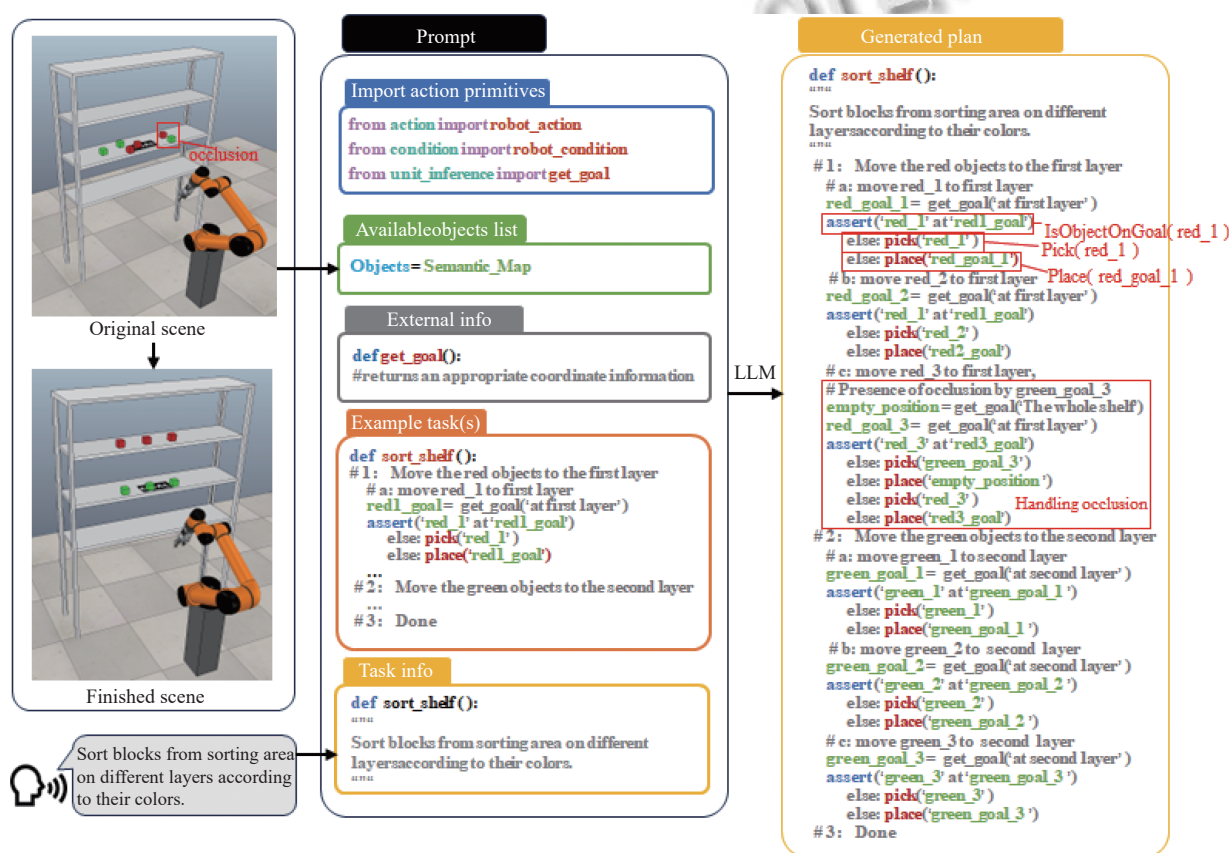


图2 推理模块 (reasoning module) 示意图

3.2 语义地图转化和变化检测

实时场景中的数据由深度相机捕获,视觉识别如图3所示。在本文的方法中,将场景信息转化为语义地图,一是使 LLM 更好地理解环境信息从而构建正确的初始行为树,二是为行为树更新拓展提供场景信息。本文选择 XML 作为存储语义映射的格式,它具有出色的可扩展性、清晰的结构以及有效表示不同属性之间关系的能力,且 LLM 在此类文件处理速

由 5 个组件构成的 prompt 的表示如图 2 所示,在构建货架整理任务的初始行为树时,本文选择了条件节点而非直接动作节点。生成的初始行为树如图 1 所示,这种方式允许行为树根据环境的实时状态并动态做出决策,优于直接使用动作节点,因为它赋予行为树响应环境变化的能力。尽管在条件明确时本文可以直接应用动作节点如 *pick* 和 *place*,但将动作节点预设为条件节点的子节点,使得本文能在满足特定条件时激活它们。因此,本文将动作节点作为潜在的执行候选,它们在条件节点验证成功时被调用,保证了任务的适应性和鲁棒性。

度上表现突出。为了更新语义地图,本文使用 3D 识别技术,这样做的好处是能够更准确地捕捉和表示复杂的三维环境信息,从而提高行为树的构建和更新的精度。变化监测 (change-detection) 是将两幅匹配图像的像素加以比较从而检测出其中物体差别的技术,本文使用 2D 识别算法监督机器人运行过程,因为 2D 识别在变化检测上速度更快,能够及时发现环境中的变化。

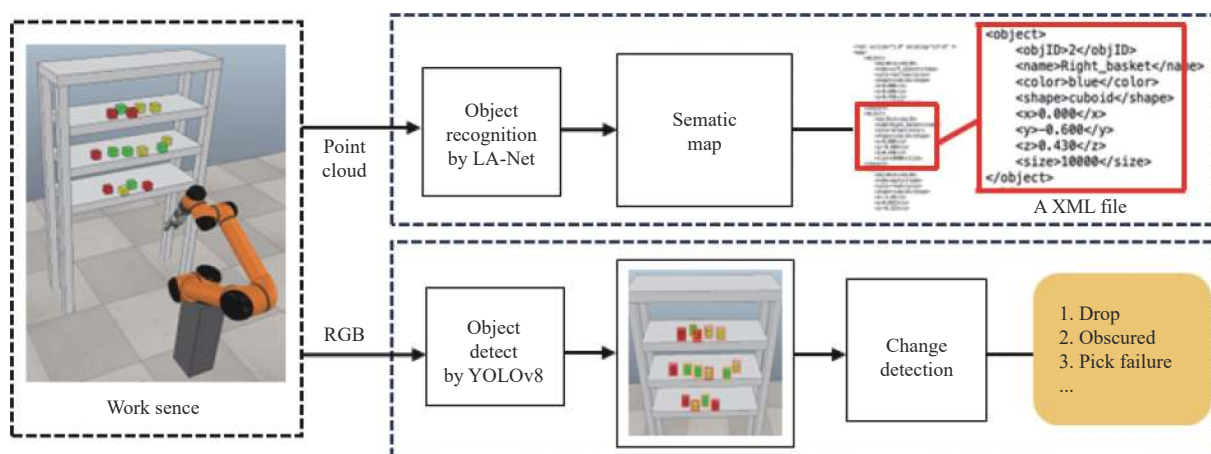


图3 视觉识别模块

语义地图的实现: 通过 3D 识别网络 LA-Net^[21]处理环境点云信息, 以识别并标记物体. 识别结果被保存至一个 XML 文件, 该文件详细记录了物体的类别、三维坐标、颜色、体积及形状等信息.

变化检测的实现: 利用 YOLOv8^[22]技术获取场景中物体的边界框信息, 并根据语义地图对识别出的物体进行标记. 随后采用 XMem^[23]来追踪这些标记. 通过变化检测机制, 如式 (3) 所示, 当发现物体滑落、抓取失败或遮挡等问题时, 相应的行为树节点会被更新, 确保系统能够根据初始行为树 (BT) 正常运作.

$$\Delta D = |C_t - G_t| - |C_{t-1} - G_{t-1}| \quad (3)$$

其中, C_t 和 G_t 表示当前帧物体和夹爪的中心坐标, C_{t-1} 和 G_{t-1} 表示上一帧的中心坐标, 则 ΔD 为相对位置变化的大小, 在抓取过程中, 当 ΔD 超过了设定的阈值, 则判定物体在抓取过程中抓取失败或物体滑落; 同理, 当抓取过程中两物体的中心坐标高度重合则判定为存在遮挡.

4 实验

4.1 实验设计

本文在第 3 节讲述了行为树的生成方法及总体运行框架, 本节中, 本文首先进行初始行为树生成实验, 随后基于初始行为树, 进行仿真机器人实验验证, 整体运行框架如图 4 所示.

本文将其分为两个阶段, 在第 1 阶段, 本文的目标是得到正确的初始行为树, 将场景信息转化为语义地图 (semantic map) 与用户指令一同作为大语言模型 (LLM) 的输入, 生成可与 LLM 交互的代码, LLM 根据

人类指令和相关提示 (prompt), 生成特定格式的约束和动作, 即行为规划, 随后将其转化为初始行为树, 在这个过程中, 用户可与 LLM 多次交互以保证大语言模型对任务的理解; 在第 2 阶段, 本文的目标是保证机器人按照正确初始行为树执行任务, 首先由用户确认初始行为树的正确性, 当判定为正确的初始行为树后机器人根据初始行为树的节点信息开始执行任务, 并根据环境信息动态更新子树, 为了保证对环境变化和机器人操作失误的适应性, 本文使用变化检测算法来监督机器人作业过程, 引入 LLM-BT^[12]的 BTs-Update 方法, 当机器人操作失误时使用 BTs-Update 进行行为树的更新.

4.2 初始行为树生成实验

在机器人系统中, 初始行为树 (BT) 的生成扮演了至关重要的角色, 它不仅满足了规定要求, 同时保障了功能的正确执行. 本实验的目的验证生成的初始 BT 是否按照动作原语构成, 同时能有效地执行在不同场景下设定的任务. 为了验证本文所提出的动作原语的适用性和可行性, 本文选取了货架整理和轴孔装配两种代表性场景进行深入分析.

货架整理: 机器人需要对货架上的物品进行整理, 确保每件物品按照人类指令放至正确的位置.

轴孔装配: 机器人需要精确地将轴插入孔中, 确保装配过程中的对齐和固定, 保证组件的正确组装和功能性.

实验参数设定: 货架整理采用总量为 6 个 (红色积木和蓝色积木各一半) 积木随机在 3 层货架上摆放; 轴孔装配为一个孔一个轴在桌子上随机摆放. 本研究选择 GPT-4 作为核心推理器, 使用第 3.1 节讲述的 5 个

相关组件构成 prompt, 同时为了得到更加准确的结果, 本文在调用模型时设置了相关参数, temperature 设置为 0.2, 较低的温度会使模型生成的文本更确定和保守, 从而提高准确性. Top-*p* 设置为 0.9, 可以在保证准确性

的同时, 允许一定程度的多样性. Frequency penalty 设置为 0.5, 减少重复生成的频率, 确保生成的行为树内容丰富且不重复. Presence penalty: 设置为 0, 使模型按照提示要求生成行为树.

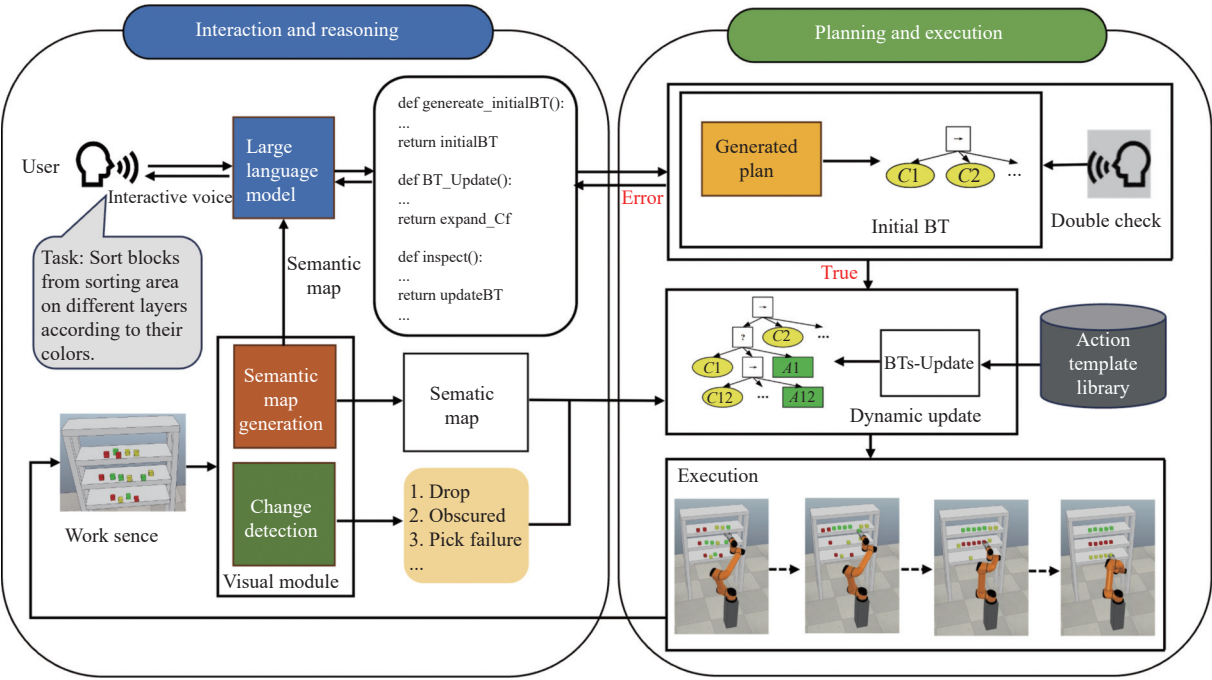


图 4 结合大语言模型和行为树的机器人任务执行框架图

评价指标: 鉴于本文的方法涵盖了多轮生成过程, 本文特别关注首轮生成结果, 将生成的行为树的正确性和结构作为评价标准. 正确性是指行为树在按照其逻辑执行任务时, 是否能够满足任务需求, 由人工进行判断; 结构是指生成的行为树是否由动作原语构成, 包括 6 个机器人基本动作和执行动作前的条件节点. 本文使用行为树的分支因子作为行为树结构的评价指标, 其计算如式 (4) 所示:

$$B = \frac{\sum_{v \in \text{non-leaves}} (\text{conditions}(v) + \text{actions}(v))}{N_{\text{non-leaves}} - 1} \quad (4)$$

其中, *B* 是分支因子, *condition*(*v*) 表示条件节点的数量, *action*(*v*) 表示动作节点的数量, *N_{non-leaves}* 表示非叶子节点的数量, 计算时不考虑 root 节点. 为了提高生成的行为树的容错性并便于后续扩展, 本研究设计的行为树在执行操作前至少需要一个条件节点作为判断, 因此, 分支因子 *B* 的最佳范围在 2~3 之间.

在进行实验时, 本研究将本文的方法与 Cao 等人^[19]提出的方法和 LLM-BT^[12]提出的方法进行了比较.

Cao 等人通过设置阶段步进提示 (phase-step prompt), 促使大型语言模型输出行为树, 本文使用同类任务的行为树作为其源任务, 同时将语义地图和动作原语作为大语言模型的输入. LLM-BT 使用 LLM 根据任务目标生成动作步骤, 随后采用端到端模型将任务步骤转化为行为树. 对每种场景进行了 15 次实验, 实验结果如表 1 所示.

表 1 4 种任务下初始行为树的生成实验结果

方法	Scene1		Scene2	
	Accuracy	Avg. <i>B</i>	Accuracy	Avg. <i>B</i>
文献[19]+Semantic map	6/15	1.6	10/15	1.7
LLM-BT ^[12]	10/15	1	13/15	1
Ours	11/15	2.8	13/15	2.8

表 1 中, Accuracy 表示在该场景下 15 实验生成的行为树正确的次数, Avg. *B* 表示在该场景 15 次实验中分支因子的平均值. 本文的方法在生成的行为树的正确性和结构上均表现最优. 需要注意的是, 在分支因子的计算中, LLM-BT^[12]通过端到端模型直接提取动作序列为行为树, 其结构固定, 因此 *B* 的计算结果始终为 1.

本文的方法在实验中的总体正确率为 80%, 在货架整理实验中的正确率为 73%, 在轴孔装配实验中的成功率为 86.6%。货架整理实验中可能存在遮挡, 导致推理步骤增加, 从而降低了 GPT-4 推理的正确率。本文的方法

生成的行为树的分支因子平均为 2.8, 通常在动作节点之前由 2 个条件节点进行判断是否满足执行标准, 但在执行 *push*、*move* 或 *drop* 动作时, 可能只需一个条件节点, 例如图 5 所示, 在执行 *drop* 时只有一个条件节点。

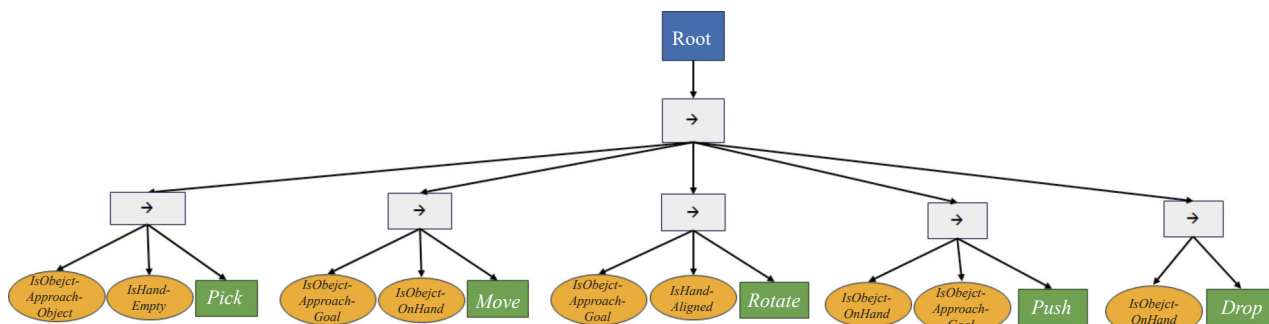
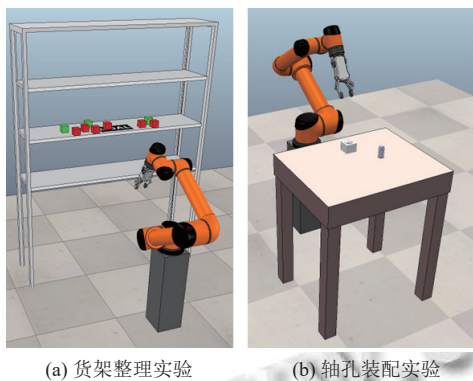


图 5 轴孔装配行为树示例

4.3 仿真实验

为了验证生成的初始行为树在机器人系统中的可行性, 本文进行了机器人仿真实验。

实验参数设定: 本文的仿真实验环境搭建在 Ubuntu 22.04 系统, 使用 Vrep 3.3.2 构建仿真实验环境如图 6 所示。



(a) 货架整理实验

(b) 轴孔装配实验

图 6 仿真实验场景

评价指标如下。

(1) 仿真实验行为树生成的正确性: 衡量生成的行为树是否能够准确反映任务需求和操作步骤。具体指标包括行为树的结构是否合理、节点逻辑是否正确, 以及在仿真实验中是否能够正确指导机器人完成任务。

(2) 鲁棒性: 评估行为树在面对环境变化和操作失误时的稳定性和适应能力。通过引入 BTs-Update 方法, 观察机器人在执行任务过程中遇到意外情况时, 行为树的自我调整和恢复能力。

(3) 任务成功率: 衡量机器人在不同任务场景下的

任务完成情况。具体包括在货架整理和轴孔装配等任务中的成功执行率, 反映行为树在实际应用中的有效性。

在每种场景下都进行了 20 次实验, “Scene1”表示货物分拣, “Scene2”表示轴孔装配, 为了控制变量, 在实验时不考虑三维识别的成功率。“S/All”表示在 20 次实验中成功的次数, 其中“Overall”表示该场景下实验的总体成功率, “Initial BT”表示成功构建初始行为树, “BTs-Update”表示机器人运行过程中行为树的更新, 只有当初始行为树构建成功时, “BTs-Update”才会继续进行。BTs-Update 更新如图 7 所示。

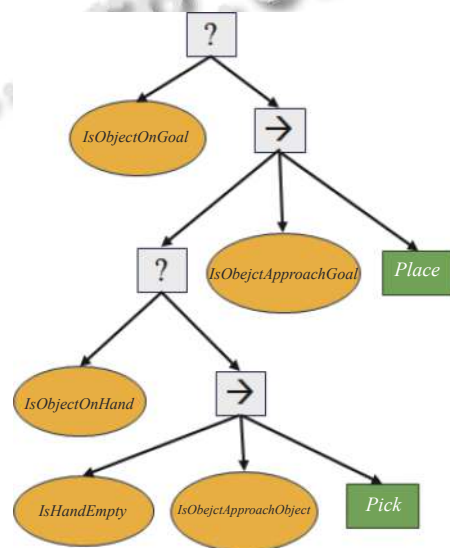


图 7 行为树更新 (BTs-Update) 示例

实验结果如表 2 所示, 货物分拣的成功率为 70%。在 20 次实验中, 有 5 次失败是由于生成错误的初始行

为树,有1次失败是由于行为树动态更新失败.轴孔装配的成功率为80%,失败的原因均为构建错误的初始行为树.

表2 仿真实验结果(S/All)

Scene	Overall	Initial BT	BTs-Update
Scene1	14/20	15/20	14/15
Scene2	16/20	16/20	16/16

实验结果表明,本文提出的方法在仿真实验中表现出较高的成功率,并且具有良好的鲁棒性.

5 结论

在这项工作中,本文提出一种新的行为树生成方法,使用场景语义感知和LLM生成初始行为树,并通过动态更新来控制机器人执行任务,这种方法对动态环境具有一定的适应性,对大语言模型不需要额外的训练,同时本文将方法应用在机器人操作中,并给出了应用框架和具体示例.本文的方法具有几个局限性,首先是推理依赖于LLM,当环境复杂化或推理难度较大时,LLM无法给出正确行为规划导致生成错误的初始行为树;其次是需要根据机器人设计动作原语和对应条件节点,这些动作原语和条件节点是构建初始树和更新行为树的关键组件,当系统需要添加一种新的动作节点时,需要专家或工程师手动添加相应的动作模板.在未来的工作中,本文将收集BT数据,微调BT生成专用的大语言模型,使其能够生成完整的BT,并在任务中自动更新.

参考文献

- Ahmad F, Mayr M, Krueger V. Learning to adapt the parameters of behavior trees and motion generators (BTMGs) to task variations. Proceedings of the 2023 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS). Detroit: IEEE, 2023. 10133–10140.
- Zhu XW. Behavior tree design of intelligent behavior of non-player character (NPC) based on Unity3D. Journal of Intelligent & Fuzzy Systems, 2019, 37(5): 6071–6079.
- Chen YX, Karkus P, Ivanovic B, *et al.* Tree-structured policy planning with learned behavior models. Proceedings of the 2023 IEEE International Conference on Robotics and Automation (ICRA). London: IEEE, 2023. 7902–7908.
- Nicolau M, Perez-Liebana D, O'Neill M, *et al.* Evolutionary behavior tree approaches for navigating platform games. IEEE Transactions on Computational Intelligence and AI in Games, 2017, 9(3): 227–238. [doi: [10.1109/TCIAIG.2016.2543661](https://doi.org/10.1109/TCIAIG.2016.2543661)]
- Iovino M, Styurd J, Falco P, *et al.* A framework for learning behavior trees in collaborative robotic applications. Proceedings of the 19th IEEE International Conference on Automation Science and Engineering (CASE). Auckland: IEEE, 2023. 1–8.
- Neupane A, Goodrich MA. Efficiently evolving swarm behaviors using grammatical evolution with PPA-style behavior trees. arXiv:2203.15776, 2022.
- French K, Wu SY, Pan TY, *et al.* Learning behavior trees from demonstration. Proceedings of the 2019 International Conference on Robotics and Automation (ICRA). Montreal: IEEE, 2019. 7791–7797.
- Francillette Y, Bouchard B, Bouchard K, *et al.* Modeling, learning, and simulating human activities of daily living with behavior trees. Knowledge and Information Systems, 2020, 62(10): 3881–3910. [doi: [10.1007/s10115-020-01476-x](https://doi.org/10.1007/s10115-020-01476-x)]
- Machalewski T, Marek M, Ochmann A. Optimization of parameterized behavior trees in RTS games. Proceedings of the 21st International Conference on Artificial Intelligence and Soft Computing. Zakopane: Springer, 2022. 387–398.
- Driess D, Xia F, Sajjadi MSM, *et al.* PaLM-E: An embodied multimodal language model. Proceedings of the 40th International Conference on Machine Learning. Honolulu: PMLR, 2023. 8469–8488.
- Zitkovich B, Yu TH, Xu SC, *et al.* RT-2: Vision-language-action models transfer web knowledge to robotic control. Proceedings of the 7th Conference on Robot Learning. Atlanta: PMLR, 2023. 2165–2183.
- Zhou HT, Lin YH, Yan LW, *et al.* LLM-BT: Performing robotic adaptive tasks based on large language models and behavior trees. Proceedings of the 2024 IEEE International Conference on Robotics and Automation (ICRA). Yokohama: IEEE, 2024. 16655–16661.
- Brohan A, Brown N, Carbajal J, *et al.* RT-1: Robotics Transformer for real-world control at scale. arXiv:2212.06817, 2022.
- Vemprala S, Bonatti R, Bucker A, *et al.* ChatGPT for robotics: Design principles and model abilities. https://www.microsoft.com/en-us/research/uploads/prod/2023/02/ChatGPT_Robotics.pdf. (2023-02-20).
- Singh I, Blukis V, Mousavian A, *et al.* ProgPrompt: Generating situated robot task plans using large language models. Proceedings of the 2023 IEEE International

- Conference on Robotics and Automation (ICRA). London: IEEE, 2023. 11523–11530.
- 16 Huang WL, Wang C, Zhang RH, *et al.* VoxPoser: Composable 3D value maps for robotic manipulation with language models. Proceedings of the 7th Conference on Robot Learning. Atlanta: PMLR, 2023. 540–562.
- 17 Li F, Wang XY, Li B, *et al.* A study on training and developing large language models for behavior tree generation. arXiv:2401.08089, 2024.
- 18 Lykov A, Tsetserukou D. LLM-BRAIn: AI-driven fast generation of robot behaviour tree based on large language model. arXiv:2305.19352, 2023.
- 19 Cao Y, Lee CS. Robot behavior-tree-based task generation with large language models. Proceedings of the 2023 AAAI Spring Symposium on Challenges Requiring the Combination of Machine Learning and Knowledge Engineering. San Francisco: CEUR Workshop Proceedings, 2023.
- 20 Chen XL, Cai YS, Mao YX, *et al.* Integrating intent understanding and optimal behavior planning for behavior tree generation from human instructions. arXiv:2405.07474, 2024.
- 21 Lin YH, Liu T, Zhang Y, *et al.* LA-Net: LSTM and attention based point cloud down-sampling and its application. Measurement and Control, 2023, 56(7-8): 1261–1277. [doi: 10.1177/00202940221149074]
- 22 Jocher G, Chaurasia A, Qiu J. Ultralytics YOLO (version 8.0.0) [Computer software]. <https://github.com/ultralytics/ultralytics>. [2023-06-21].
- 23 Cheng HK, Schwing AG. XMem: Long-term video object segmentation with an Atkinson-Shiffrin memory model. Proceedings of the 17th European Conference on Computer Vision. Tel Aviv: Springer, 2022. 640–658.

(校对责编: 孙君艳)