

# 关于基于大模型的智能代理 在具身智能的应用研究

刘帅

2020212267

2024 年 1 月 13 日

## 摘要

随着以 GPT-4[3] 为代表的大语言模型 (LLMs) 和以 Flamingo 为代表的视觉语言模型 (VLMs)[9] 的发展，提升了人工智能对于世界信息获取、知识泛化和推理的能力，进一步为发展具身智能代理 (Embodied Agent) 带来了新的方向和研究思路，并取得了显著的进展 [45][61][3]。一般而言，Embodied Agent 通常包括三个关键部分：**大脑、感知和行动**。针对大脑，它主要由一个大语言模型组成，负责存储了关键的记忆、信息和知识，并承担着信息处理、决策、推理和规划等重要任务，是决定代理能否展现智能行为的关键因素。对于感知模块，它类似于人类的感觉器官，主要功能是将代理的感知空间从仅限文本扩展到包括文本、声音、视觉、触觉、嗅觉的多模态感知空间，这种扩展使代理能够更好地感知来自外部环境的信息，对于一个“看不见”的 LLM 来说，这部分需要通过游戏环境的反馈信号并用合适的数据表征将知识传递给“大脑”，而对于 VLM 而言，则需要为文本和图像特征对齐。对于行动模块，用于扩展代理的行动空间，从而使代理能够具备文本输出、采取具体行动并使用工具，以便更好地对环境做出响应、提供反馈，甚至改变和塑造环境。本文将以 Minecraft 的游戏环境作为例子，结合目前常见的 LLM-driven 和 VLM-driven 的智能代理，对其主要构成、技术特点、目前局限及发展趋势等方面进行分析。

## 1 Embodied Agent 的介绍

### 1.1 什么是 Embodied Agent

Embodied Agent 是一个被设计用于与物理世界交互的人工智能 (AI) 系统。这可以包括机器人、虚拟助手和其他类型的智能系统。Embodied Agent 的一个关键特征是它位于环境中：这意味着它可以以自然的方式与其周围的环境进行交互。例如，一个机器人可以拾取物体或在房间中移动。Embodied Agent 还拥有一个身体：这可以是一个物理身体，如机器人，也可以是虚拟身体，如虚拟现实系统中的虚拟人。随着我们朝着将人工智能系统更加融入我们日常生活的未来前进，Embodied Agent 变得越来越重要。它们提供了与世界更自然互动的方式，并可以用于医疗保健、教育和娱乐等任务。

### 1.2 Embodied Agent 的任务描述及其挑战

Embodied agent 的目标是创造一个可以学会创造性地解决需要与环境互动的复杂任务的智能体，例如一个机器人。随着 Chatgpt[3] 等大模型的不断涌现，以及 Alpaca[56],M3IT[36] 等大型数据集的不断增加，使得人工智能的知识存储、理解、推理能力超越了人类表现，解决了以前被认为难以解决的各种人工智能任务。计算机视觉、语音识别和自然语言处理在输入输出任务（如语言理解和图像处理）方面经历了革命性的变革，强化学习在互动任务（如游戏）中也取得了十分出色的表现。这些进展已经极大地推动了具体体现人工智能的发展，使越来越多的研究人员能够迅速取得进展。具体而言，创造智能体的目标包括：

- 看：通过视觉或其他感知方式感知他们的环境。

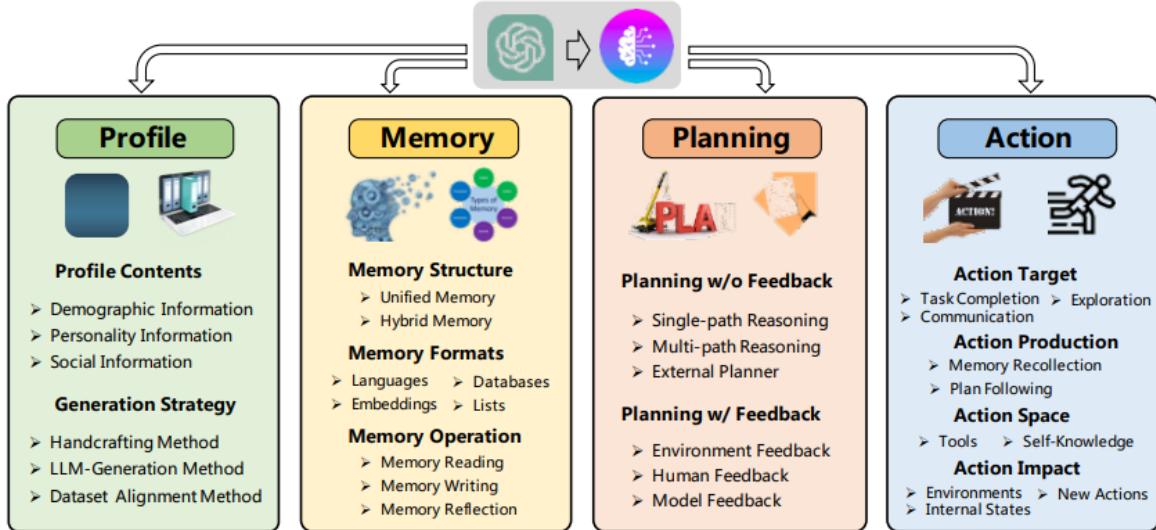


图 1: 基于 LLM 的代理的示意结构

- 说: 进行基于环境的自然语言对话。
- 听: 理解并对场景中的音频输入作出反应。
- 行动: 在环境中导航并与之互动以实现目标。
- 思考: 考虑并规划其行动的长期后果。

### 1.3 Embodied Agent 的组成结构

在这个章节, 我们将针对构成 Embodied Agent 的结构进行简要介绍。如图所示 3, Embodied Agent 主要分为信息、记忆、规划和执行四部分内容。

其中, 信息的部分是 Embodied Agent 执行的先验, 针对 LLM 的情况, 由于 LLM 经由大量数据训练, 因此囊括绝大多数泛化的世界信息, 从而能够作为合适的提供信息的大脑。

针对记忆部分, 通常包含两方面的因素: 1. 关于游戏环境目前状态的短期存储: 通常将一些重要信息作为 prompt 输入, 从而进行下一轮的任务规划、执行。2. 关于一些较大规模信息的长期存储: 由于 LLM 已经具备了关于泛化世界信息的存储, 我们这一步的目的是存储更多针对游戏环境的特定信息, 例如在 Minecraft 游戏中, STEVE-EYE[8] 为了训练一个多模态大模型, 除了利用 GPT 生成指令-回答对以外, 特定的针对 Minecraft 信息, 例如关于游戏内物品的文本介绍、当前动作的简要描述等, 利用 Minedojo[20] 对 minecraft 的数据进行了存储和训练。Voyager[59] 为了完成挖钻石的任务, 利用 GPT4 构建了执行任务的技能数据库。

针对任务规划方面, 在一篇综述 [60] 中提到, 任务规划的结构如图 2 所示。对于单路径的规划而言, 为 LLM 输入对应的 prompt 和最终的任务要求, 则会在一次推理以内将任务分解成子任务。对于 re-prompting 的范式, 则会在每一步子任务的生成和执行后重新进行一次 query, 也就是说, 它可以及时的改变来自环境的反馈并确保任务能够正确执行, 目前大部分基于具身智能的智能代理都采用了该范式进行任务的规划和执行。针对多种路径的规划范式, 则是通过多条路径的树状结构, 一次性给出多种解决方案, 从而并行的进行任务规划。由于这种范式很容易造成空间和时间的开销, 以及, 以游戏为例的环境存在的占用显存或者进程冲突的情况, 在游戏的代理应用方面并不十分常见。同时, 为了将游戏环境和任务情况进行更为形式化的描述, PDDL[25][57] 中提到了一种更为形式化的描述, 从而为 LLM 的任务规划提供了更结构化的先验, 并为场景图生成 [69] (Scene Graph Generation) 等任务提供了充分的数据来源。

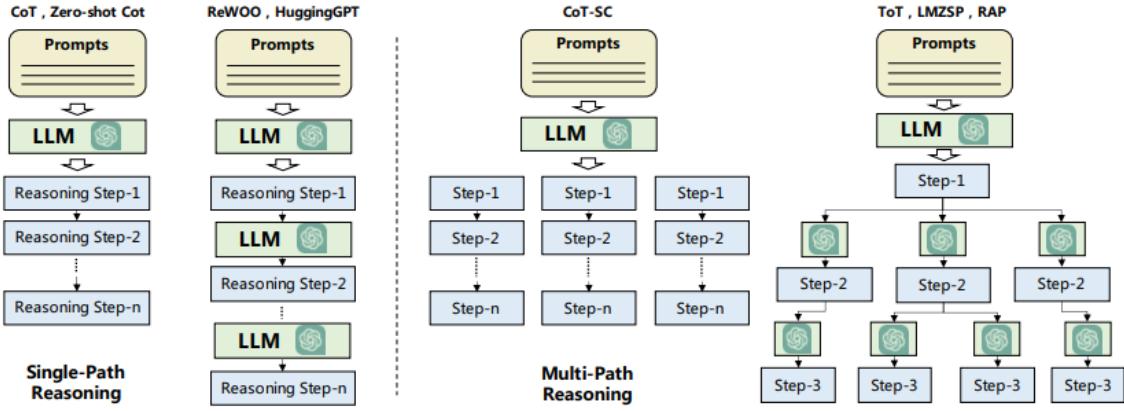


图 2: 基于 LLM 进行任务规划的结构示意

## 2 Embodied AI 数据集介绍

本章节将针对 Embodied AI 任务的一些代表性数据集和收集数据流程进行介绍。由于 Embodied AI 任务的性质, 需要 Agent 在特定的仿真环境执行, 因此, 如何建立高效的 pipeline 并针对需求收集特定数据是构建 Embodied AI 数据集的核心, 以下列出了部分代表性工作的数据收集和训练流程:

**Mindoj** 关于 Minecraft 数据库的存储和训练在 Minedoj[20] 有所提出, 他的数据由 740k 条 YouTube 视频, 7k 个维基页面及 350k 条 Reddit 帖子所构成的图像-文本对构成。在细节层面, Minedoj 利用类似于 CLIP[49] 的方法训练了 MINECLIP 模型, 同时, MINECLIP 的 logits 由于直接输出了图像文本的匹配相似度, 可以直接的作为 reward-model, 从而参与 RLHF 和一些传统 RL 方法的 PPO 训练过程。具体来讲, MINECLIP 的构成如下。

- 分帧的图像编码器  $\phi_I$ : MINECLIP 利用 ViT-B/16 用来将 rgb 帧编码成 512 维的向量, 仅仅对于最后两层进行微调。
- 时序融合模块  $\phi_a$ : MINECLIP 采用了 average pooling 和自注意力两种方法来实现相邻帧之间的融合, 在融合之后, 利用了 CLIP Adapter[23] 进行 clip 的训练。
- Text encoder  $\phi_G$ : 利用 GPT model 进行文本信息编码, 同样只对最后两层的参数进行微调。

**STEVE-EYE** 同样地, 在 STEVE-EYE[8] 中, 如图所示 8, LLM 的训练数据来源于多种数据类型, 例如指令对、QA 对、关于图片的简要概述、以及基于上下文的任务规划。Steve-Eye 在获取世界的基础知识、理解周围环境的细微差别以及生成可执行计划以完成各种开放性任务方面表现出色。此外, Steve-Eye 通过视觉或基于文本的提示来响应用户的指令, 提高了人机交互的便捷性和灵活性。但是, STEVE-EYE 由于数据集的构成是利用的非连续的图像数据, 对于视频或者连续帧的行为推理能力表现并不理想。

**Otter** Otter[33] 是经由多模态数据集 MIMIC-IT[32] 进行训练的多模态大模型, 如图 7, 并如图 6 流程进行数据收集。MIMIC-IT 的数据构建流程将系统消息、视觉注释和上下文示例作为 GPT 的 prompt。系统消息定义生成的指令-响应对的期望语调和风格, 而视觉注释提供关键的图像信息, 如边界框和图像描述。上下文示例帮助 ChatGPT 在上下文中学习。并使用冷启动策略来增强上下文示例, 然后进行大规模查询。在冷启动阶段, 仅通过系统消息和视觉注释提示 ChatGPT 来收集上下文示例, 采用启发式方法。这个阶段仅在满意的上下文示例被识别出时结束, 一旦获取了指令-响应对, pipeline 将它们扩展到中文 (zh), 日语 (ja), 西班牙语 (es), 德语 (de), 法语 (fr), 韩语 (ko) 和阿拉伯语 (ar)。

Butcher Economic Trade					 Butcher	 <b>Copper Ingot</b> <b>Iron Ingot</b> <b>Gold Ingot</b> <b>Diamond</b>	<b>Ingredient</b> <b>Exp</b> <b>Description</b>
Level	Item wanted	Default quantity	Item given	Quantity			
<b>Novice</b>	Raw Chicken	14	Emerald	1			
	Raw Porkchop	7	Emerald	1			
	Raw Rabbit	4	Emerald	1			
<b>Apprentice</b>	Emerald	1	Rabbit Stew	1			
	Coal	15	Emerald	1			
	Emerald	1	Cooked Porkchop	5			
<b>Journeyman</b>	Emerald	1	Cooked Chicken	8			
	Raw Mutton	7	Emerald	1			
	Raw Beef	10	Emerald	1			
<b>Expert</b>	Dried Kelp Block	10	Emerald	1			
<b>Master</b>	Sweet Berries	10	Emerald	1			

Hostile mobs		 Blaze Chicken Jockey Creeper Drowned Elder Guardian Endermite Evoker Ghast Guardian Hoglin Husk Magma Cube Phantom Piglin Brute Pillager Ravager Shulker Silverfish Skeleton Skeleton Horseman Slime Spider Jockey Stray Warden Witch Wither Skeleton Zoglin Zombie Zombie Villager	<b>Biome name</b> <b>Features</b> <b>Description</b> <b>Screenshot</b> <a href="#">[hide]</a>
Biome name	Features		
<b>Nether Wastes</b>	Netherrack, Glowstone, Soul Sand, Nether Quartz Ore, Ghasts, Blazes, Zombified Piglins, Nether Fortresses, Wither Skeletons, Lava, Magma cubes, Gravel, Magma Blocks, Bastion Remnants, Ruined Portals, Piglins, Nether Gold Ore	<b>Temperature: 2.0. Rainfall: 0.0.</b> This is one of the biomes used to generate the Nether. Within this biome mobs such as <b>ghasts</b> , packs of <b>piglins</b> , <b>zombified piglins</b> and the occasional <b>magma cubes</b> and <b>endermen</b> spawn. Certain structures, such as <b>Nether quartz ore</b> and <b>glowstone blobs</b> , and <b>Nether fortresses</b> generate only in the Nether.	 Nether Wastes

图 3: Mindojo 数据库信息汇总

**Octopus** Octopus 是基于多模态大模型 Otter[33] 的针对 Embodied AI 任务进行微调的大模型。如图 5 和图 4 (a) 所示，我们为每个状态收集一个环境消息，其中包括诸如观察到的对象、观察到的关系、库存等属性。具体来说，模拟器可以在每个状态提供我们一个精确的场景图，用于构建前两个部分的内容。库存信息可以在模拟器中轻松获取。任务，例如图 5 中的“Cook a Bacon”，由任务目标表示。

### 3 Embodied AI 评测标准

关于 Embodied AI 的评测标准，通常考虑以下四个维度：效率、社交性、价值观以及持续演化的能力。

**效率** 目前，LLM 和 VLM Agent 作为人类助手，要么独立完成任务，要么协助人类完成任务 [16][62][18]。因此，在任务执行期间效果和效用是关键的评估标准。具体来说，任务完成的成功率是评估效用的主要指标 [67][40]。该指标主要包括代理是否实现了规定的目标或达到了预期的得分 [48][43]。例如，AgentBench[42] 汇集了来自不同现实场景的挑战，并引入了一个系统性的基准来评估 LLM 的任务完成能力，包括环境理解、推理、规划、决策、工具利用和具体行动能力，研究人员可以对这些具体能力进行更详细的评估，例如，SayCan[6] 将 LLM 做出计划和执行计划的成功率作为两个指标，并引入了部分人为评估，为了了解提出的方法的性能，SayCan 提出了两个主要的指标：第一个是计划成功率，它通过人工对 planning 的合理性进行评估的方法，衡量了模型选择的技能是否正确，无论它们是否实际执行成功。需要注意的是，对于许多指令，可能存在多个有效的解决方案。例如，如果指令是“拿来一个海绵并扔掉罐头”，计划可以选择先拿来海绵，或者先扔掉罐头。第二个指标是执行成功率，它衡量了完整的

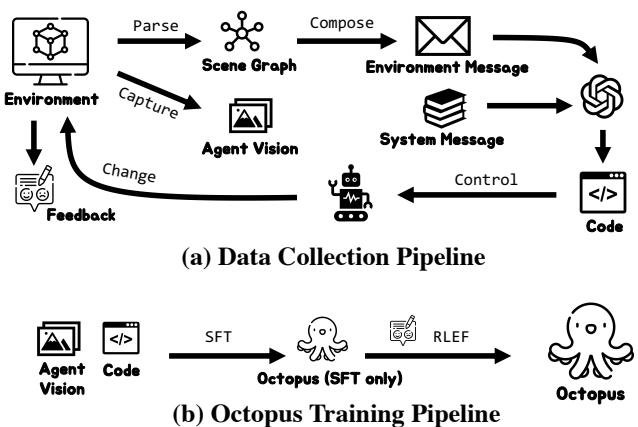


图 4: Octopus Data Collection

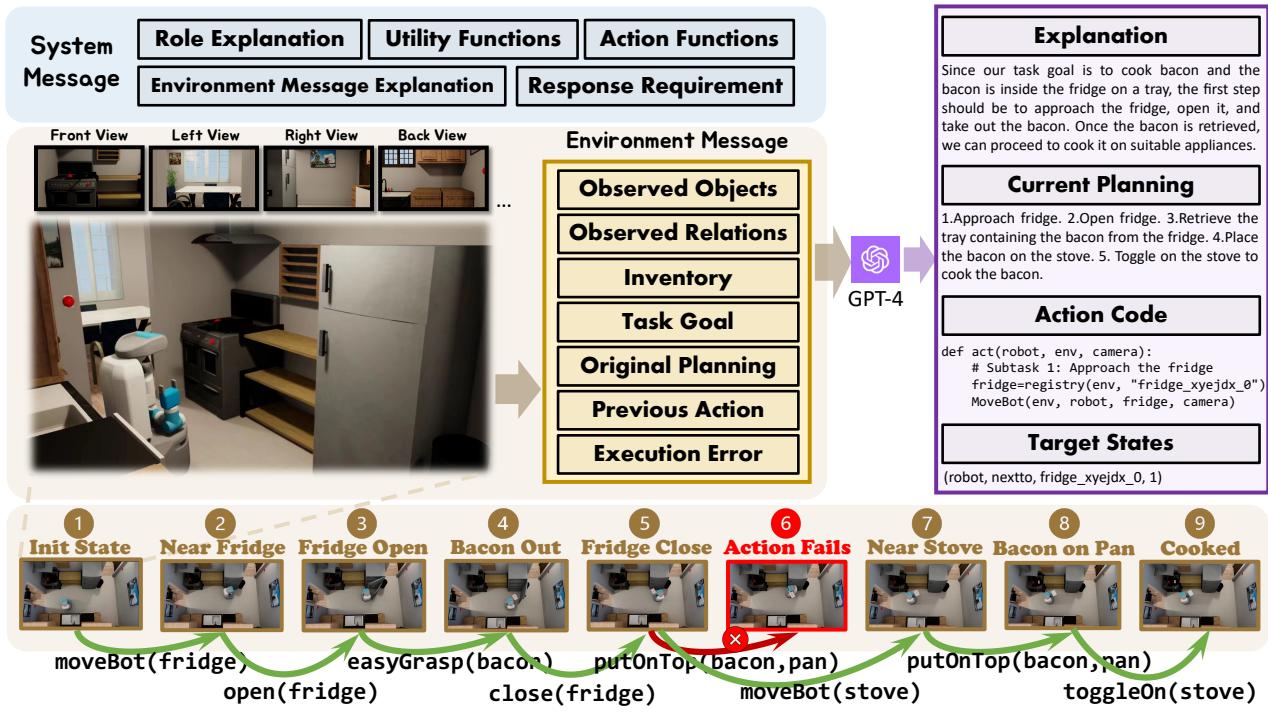


图 5: Cook a Bacon 任务的数据收集示例 GPT-4 通过“环境消息”感知环境，并根据详细的“系统消息”生成预期的计划和代码。随后，这段代码在模拟器中执行，指导代理前往下一个状态。对于每个状态，我们收集“环境消息”，其中“观察到的对象”和“关系”被以自我中心的图像替代，用作训练输入。来自 GPT-4 的响应充当训练输出。具体来说，记录了环境反馈，特别是确定是否达到了每个目标状态，用于 RLEF 训练。

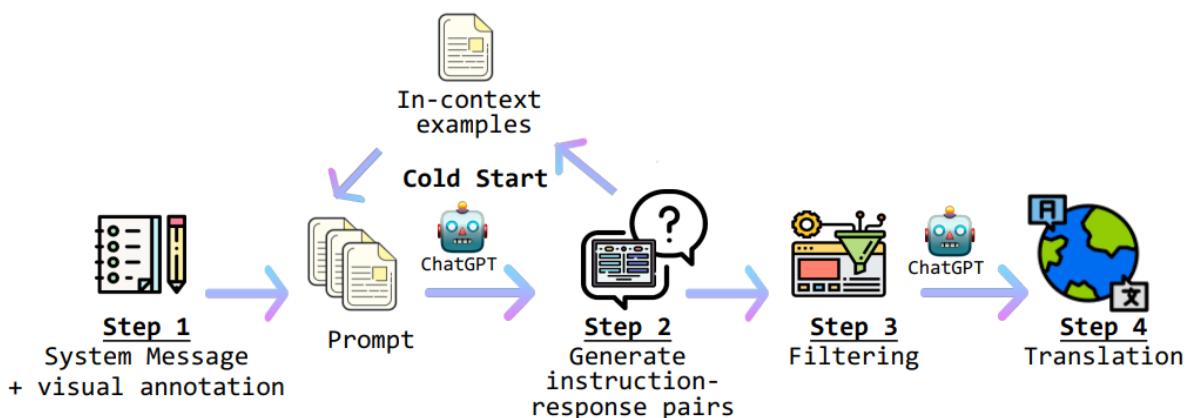


图 6: MIMIC-IT 的数据构建流程 MIMIC-IT 采用了冷启动阶段，以确定在给定数据集中查询指令-响应对的最佳系统消息和上下文示例。随后在步骤 1 到 4 之间生成了八种语言的高质量指令-响应对。

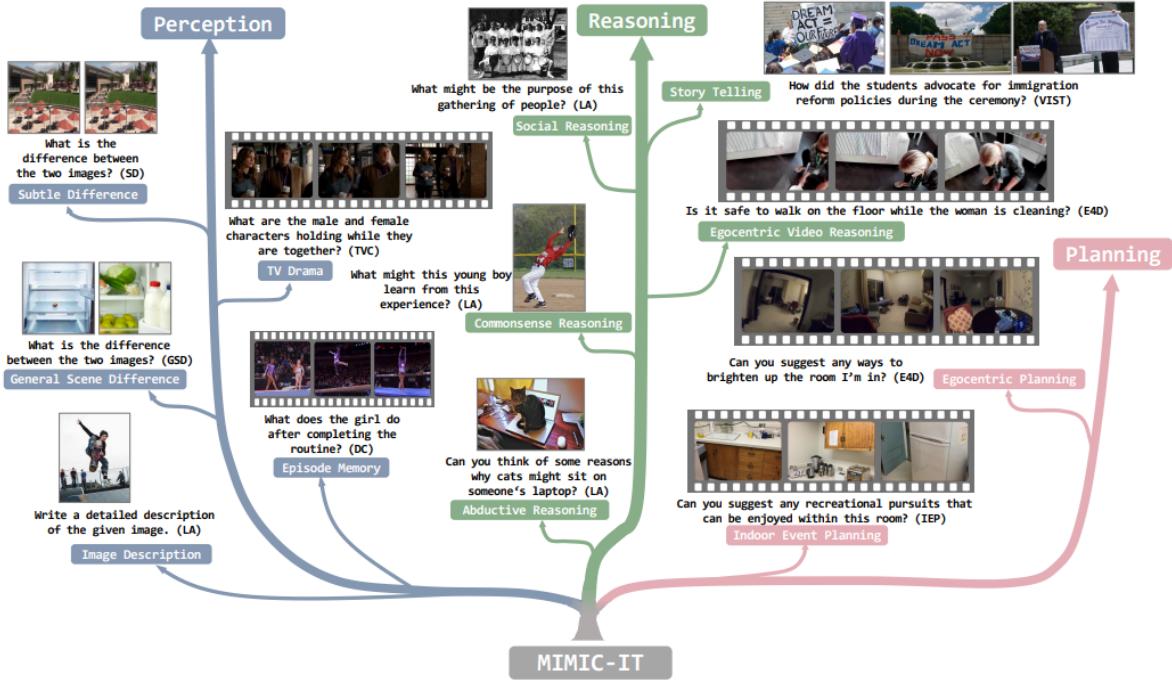


图 7: MIMIC-IT 数据集包括 280 万多个模式指令-响应对，涵盖了基本能力：感知、推理和规划。每个指令都伴随着多模式的对话上下文，使在 MIMIC-IT 上训练的语言模型能够展现出在 zero-shot 通用化方面的强大能力，即在没有预先训练的情况下，能够有效地遵循交互式指令。

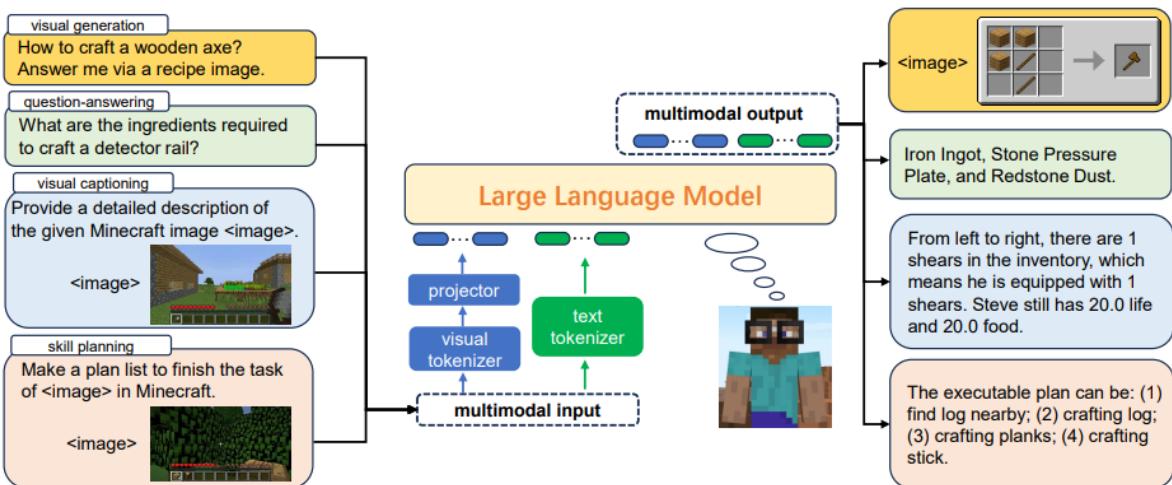


图 8: STEVE 的数据库构成

SayCan 系统是否成功执行了所需的指令。通过人工观看机器人的执行，来评估机器人是否完成了任务。除了衡量任务规划和执行的成功率以外，还应考虑它们的效率，这是用户满意度的关键决定因素 [60]。代理不仅应具备足够的能力，还应能够在适当的时间内和适当的资源投入下完成预定的任务。

**社交性** 除了基于 LLM 的代理在任务完成和满足人类需求方面的效用之外，它们的社交性也至关重要 [48]。它影响用户的沟通体验，显著影响沟通效率，包括它们是否可以与人类和其他代理无缝互动 [5][30]。具体而言，社交性的评估可以从以下几个角度来考虑：

**语言沟通能力** 语言沟通能力是一个基本的能力，包括自然语言理解和生成。自然语言理解要求代理不仅理解字面意义，还要理解隐含的意义和相关社交知识，如幽默、讽刺、攻击和情感 [29][14]。另一方面，自然语言生成要求代理在适应情境的情况下产生流畅、语法正确和可信的内容，同时调整适当的语调和情感 [39]。

**合作和协商能力** 合作和协商能力要求代理在有序和无序的情境中有效地执行其分配的任务 [35]。他们应该与其他代理合作或竞争，以实现更好的性能。测试环境可能涉及代理协作的复杂任务或代理自由互动的开放平台 [46]。评估指标不仅关注任务完成的顺利程度和代理协调合作的可信度 [22][26]

**角色扮演能力** 角色扮演能力要求代理忠实地扮演其分配的角色，表达与其指定身份相符的陈述和行为 [53]。这确保在与其他代理或人类进行长期任务的互动时，代理能够清晰地区分角色。此外，代理应该保持其身份，避免不必要的混淆。

**价值观** 随着基于 LLM 的代理在能力上不断进步，确保它们成为对世界和人类无害的实体至关重要。具体来说，代理需要遵守与人类社会价值观一致的道德和伦理准则 [68]。我们的首要期望是代理能够保持诚实，提供准确、真实的信息和内容。它们应该具备识别自己在完成任务时的能力，并在无法提供答案或帮助时表达不确定性的意识。此外，代理必须保持无害的立场，避免直接或间接的偏见、歧视、攻击或类似行为。他们还应该避免执行人类请求的危险行为，如创建破坏性工具或破坏地球。此外，代理应该能够适应特定的人口统计数据、文化和背景，并在特定情况下表现出与情境相关的社会价值观。与价值观相关的评估方法主要涉及在构建诚实、无害或特定情境基准上进行性能评估。

**持续演化的能力** 从静态的角度看，一个具有高度效用、社交性和适当价值观的代理可以满足大多数人类需求，并有可能提高生产率。然而，从动态的角度看，一个不断演化和适应不断变化的社会需求的代理可能更符合当前的趋势 [15]。由于代理可以自主演化，因此可以显著减少人类干预和所需的资源（如数据收集工作和培训的计算成本）。在这个领域已经进行了一些探索性工作，例如，Voyager 允许 Agent 从头开始在虚拟世界中进行生存任务，并将成功实现的技能存储到 Skill Library<sup>9</sup>。Octopus[65] 的工作在 Omnidigibson 的环境中定义了一系列的子动作，例如“走到”、“拿起”等，通过这些动作的组合，完成技能的学习。

## 4 关于 LLM Agent 模型及训练架构分析

本章节将针对 LLM 以及视觉语言模型 (Vision Language Model, VLMs) 的主流模型及训练架构进行分析。关于以 LLM 作为大脑所驱动的智能代理最早出现在 Ghost[70] 的工作中，具体而言，图 10 包括 LLM 分解器、LLM 规划器和 LLM 与 minecraft 的仿真环境交互的接口，它们分别负责将子目标、结构化操作和键盘操作进行分解。在 Minecraft 中给定一个目标，LLM 分解器首先根据从互联网收集的基于文本的知识将其分解成一系列明确定义的 subtasks。然后，LLM 规划器为每个子目标规划一系列结构化操作。这些结构化操作具有清晰的语义和相应的反馈，使 LLM 能够理解周围环境并在认知层面做出决策。LLM 规划器还将成功的行动列表记录和总结到基于文本的存储器中，以增强未来的规划能力。最后，LLM 界面执行这些结构化操作，通过处理原始键盘输入并接收原始观察结果与环境互动。

在 Voyager[59] 的工作所构建的 agent 中同样包含了任务规划、子任务分解、memory 构建部分，他们的研究重点集中在技能库 (skill library) 的构建上，利用该工作的先前工作 CLIP 作为外部的知识模型。可以在 Minecraft

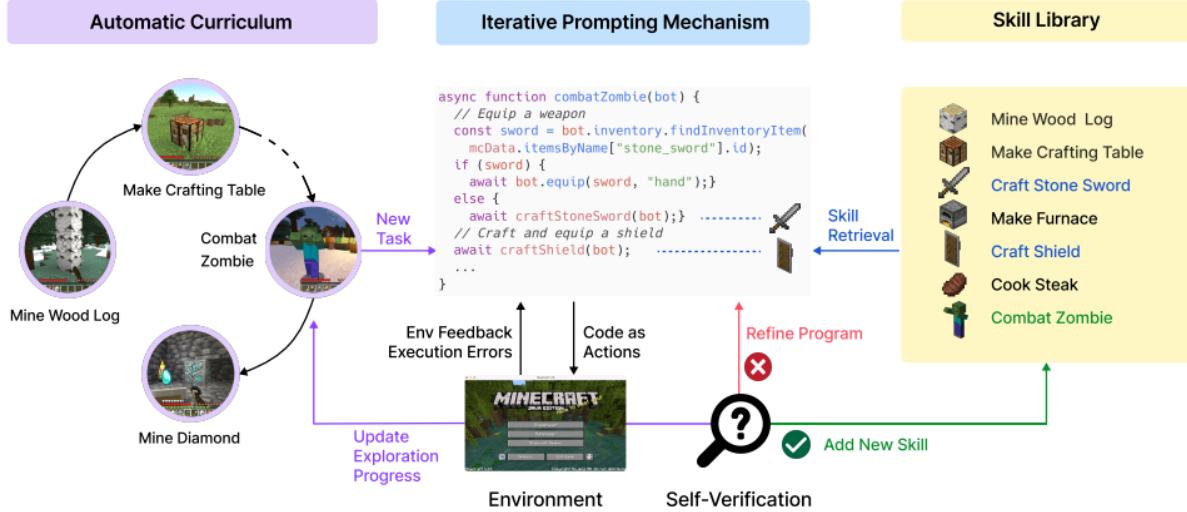


图 9: Voyager 的模型架构

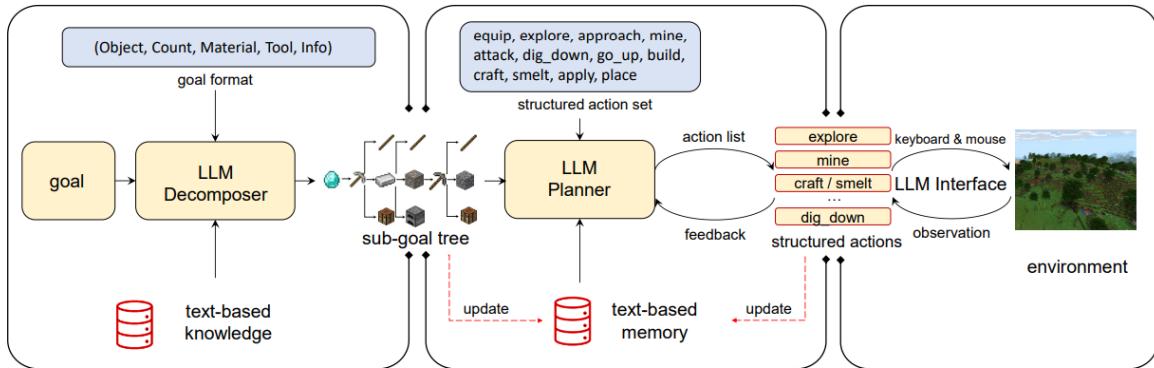
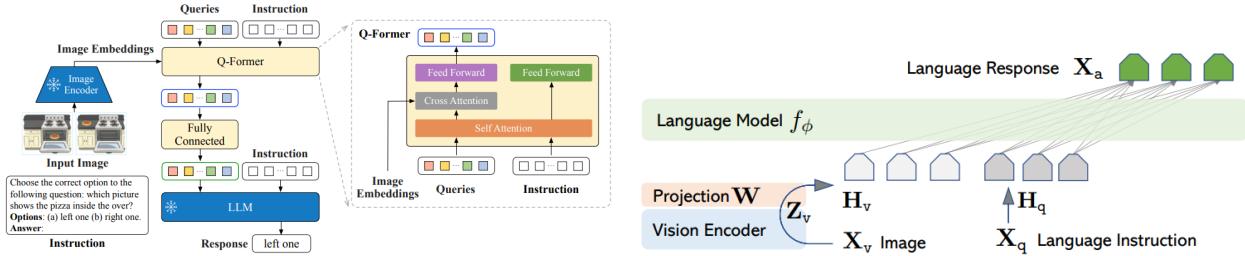


图 10: Ghost 的架构

中进行探索、掌握各种技能，并不断进行新的发现，无需人类干预。该工作，如图 9 通过三个关键模块实现：1) 最大化探索的自动课程；2) 用于存储和检索复杂行为的技能库；以及 3) 生成具体控制代码的新的迭代提示机制。我们选择使用代码作为行动空间，而不是低级运动命令，因为程序可以自然地表示时间上延伸和组合的动作 [37][55]，这对于 Minecraft 中的许多长期任务至关重要。VOYAGER 通过 prompt 和上下文学习 [61][51] 与一个 GPT-4 进行交互。该方法绕过了需要访问模型参数和显式基于梯度的训练或微调的需求。

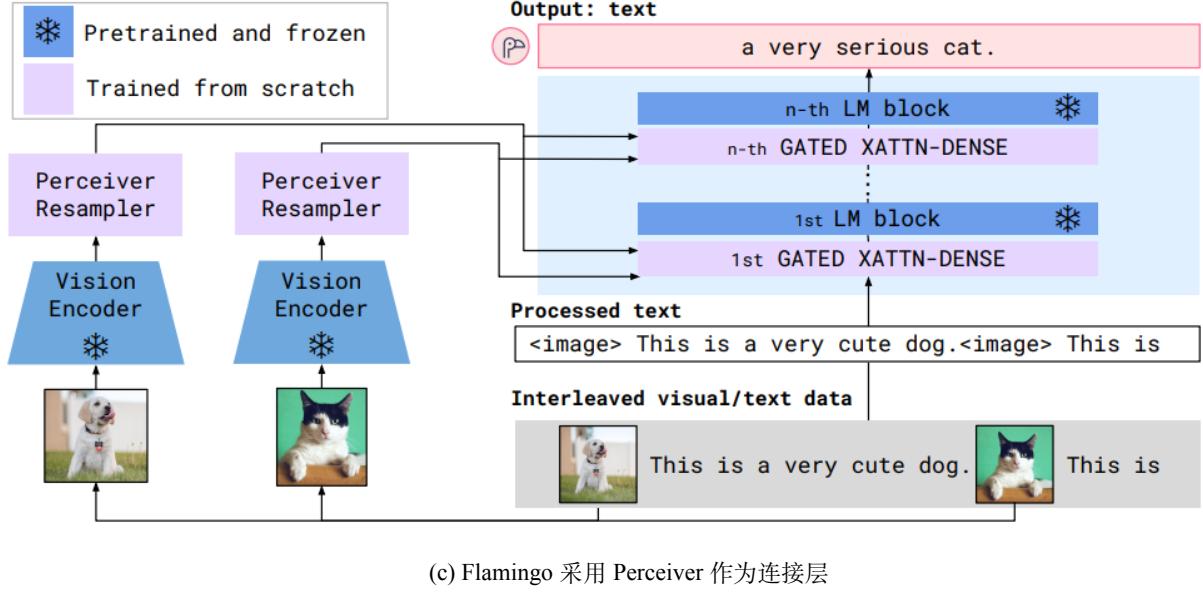
**VLMs 相关模型架构** 由于 LLM 所驱动的智能代理并不能真正的“看到”环境，比如如何利用他眼前的石块去做一个石镐，这个过程通常需要利用游戏的相关结构将环境信息转化为形式化的文本信息输入给 LLM，而由于将图像信息转化为文本的过程由于语义稠密性的差异，可能造成一定的语义缺失。因此，一个自然而常见的思路则是，如何直接利用视觉、语言模型来进行具身智能任务的理解、规划和执行。本章节将介绍主流的 VLMs 及连接层架构，如图 11，分别以 LLaVA[41] 为代表的 projection 架构，InstructBLIP[17] 为代表的 q-former 架构以及 Flamingo[2] 为代表的 Perceiver 架构。

**以 LLaVA[41] 为代表的 projection 架构** 图 11b 展示了 llava 的训练架构，对于一个输入图片  $X_v$ ，我们采用了一个预训练的 CLIP 模型的 encoder，ViT-L/14[49]，提取出了视觉特征  $Z_v = g(X_v)$ 。llava 在训练的过程中只考虑了最后一个 Transformer 层之前和之后的网格特征，使用一个简单的线性层将图像特征连接到词嵌入空间。具体来说，llava 使用一个可训练的投影矩阵  $W$  将  $Z_v$  转换为 embeddings  $H_v$ ，它们的维度与语言模型中的 embeddings 空间相同。多模态性能：Projection 结构允许将图像信息与文本信息进行交互和整合。这使得系统能



(a) InstructBLIP 采用 Q-Former 作为连接层

(b) LLAVA 采用 Projection 作为连接层



(c) Flamingo 采用 Perceiver 作为连接层

图 11: VLMs 的三种主流架构

够同时处理视觉和文本输入，从而更好地理解和生成多模态内容。该架构的优点如下：

优点：

- 高效的特征提取：利用预训练的 CLIP 模型的 ViT-L/14 编码器，有效提取视觉特征。
- 多模态性能：能够处理视觉和文本输入，提高对多模态内容的理解和生成能力。

缺点：

- 模型复杂性：可能需要大量的数据和计算资源来训练和微调。
- 特定于任务的调整：可能需要针对特定任务进行调整，以优化性能。
- 语义信息融合能力：由于 projection 结构将图像 embedding 先投影再和语义 embedding 进行对齐，这种方法可能不能很好地融合图像和语义的信息。

**InstructBLIP[17] 为代表的 q-former 架构** InstructBLIP 利用了 Q-Former11a，从一个冻结的图像编码器中提取视觉特征。Q-Former 的输入包括一组 K 个可学习的 embedding，这些 embeddings 通过交叉注意力与图像编码器的输出进行交互。Q-Former 的输出包括 K 个编码的视觉向量，每个对应一个查询嵌入，然后经过线性投影并馈送到冻结的 LLM。Q-Former 在进行指令微调之前分两个阶段使用图像-标题数据进行预训练。第一阶段使用冻结的图像编码器对 Q-Former 进行预训练，用于视觉-语言表示学习。第二阶段将 Q-Former 的输出调整为用于文本生成的指令信息，与冻结的 LLM一起使用。在预训练之后，使用指令微调对 Q-Former 进行微调，其中 LLM 的输入包括来自 Q-Former 的视觉编码和任务指令。

优点：

- 灵活性：通过交叉注意力机制和可学习的 embedding，提供了对视觉特征的灵活处理。
- 分阶段训练：先进行视觉-语言表示学习，后进行指令微调，使得模型更加精确。
- 对齐效果：具有更好的语义、图像对齐效果。

缺点：

- 训练阶段的复杂性：需要多阶段的预训练和微调，同时注意力结构可能导致更大的训练开销，使得训练过程更加复杂和耗时。
- 对预训练数据的依赖：预训练阶段对大量的图像-标题数据有较高依赖。

**Flamingo[2] 为代表的 Perceiver 架构** 为了更有效地整合视觉信号，Flamingo 采用了基于 Perceiver 的架构，从大量的视觉输入特征中生成了数百个标记，然后使用交叉注意力层与 LM 层交替，将视觉信息融合到语言解码过程中。训练目标是自回归的，采用 NLL (Negative Log-Likelihood) 作为 loss。

优点：

- 高效的信息融合：通过交叉注意力层有效地整合视觉信号到语言解码过程。
- 自回归训练目标：使用 NLL 作为 loss，提高了训练的效率和模型的预测能力。
- 即插即用：由于 perceiver 框架的灵活性，对于各种最先进的语言模型具有一定的灵活度。

缺点：

- 处理大量特征的挑战：从大量的视觉输入生成数百个标记可能导致计算负担增加。
- perceiver 架构的具体设计较为复杂，可解释性较弱，还需要进一步探索。

## 5 关于目前学术界常见的仿真环境和 Embodied AI Model 调研

**仿真环境** 在游戏及其相关的具身智能领域，仿真环境的性质往往决定了哪些环境状态可以作为 Embodied Agent 的输入，目前学术界常见的一些仿真环境如下：

表 1: Overview of Embodied AI simulator 数据集特征与用于构建这些数据集的仿真环境密切相关，在该摘要中，概述了在创建数据集的过程中常用的 Embodied AI Simulator。

Simulation Environment	Kinematics	Continuous Extended States	Flexible Materials	Deformable Bodies	Realistic Fluid	Realistic Action Execution	TaskPlanning and/or Control	Game-Based or World-Based	Well-Formulated Tasks	Code Execution
OpenAI Gym [10]	✓	✗	✗	✗	✗	✓	C	G	✗	✓
Matterport3D [13]	✗	✗	✗	✗	✗	✗	C	W	✗	✗
AI2THOR [31]	✓	✗	✗	✗	✗	✗	TP	G	✗	✓
VirtualHome [47]	✗	✗	✗	✗	✗	✗	TP	G	✗	✗
House3D [63]	✗	✗	✗	✗	✗	✗	TP	W	✗	✗
Habitat 1.0 [54]	✓	✗	✗	✗	✗	✓	C	W	✗	✓
Robosuite [71]	✓	✗	✗	✗	✗	✓	C	W	✗	✓
RFUniverse [21]	✓	✗	✓	✓	✓	✓	TP+C	W	✗	✓
Minecraft [4]	✓	✗	✓	✗	✗	✓	TP+C	G	✓	✓
GTA [1]	✓	✓	✓	✓	✓	✓	TP+C	G	✗	✗
Omnigibson [34]	✓	✓	✓	✓	✓	✓	TP+C	W	✗	✗
OctoGTA [65]	✓	✓	✓	✓	✓	✓	TP+C	G	✓	✓
Octogibson [65]	✓	✓	✓	✓	✓	✓	TP+C	W	✓	✓

表 2: 相关 Embodied AI 模型的介绍

Models	Release	Supported	Vision	Code	Action	LLM Training
	Date	Environment	Model	Generator	w/ Feedback	Enabled
Text2Motion[38]	Mar. 2023	Sim	✗	✓	✓	✗
Instruct2Act[27]	May 2023	Sim	✗	✓	✗	✗
Lang2Rewards[66]	Jun. 2023	Sim	✗	✓	✓	✗
VoxPoser[28]	Jul. 2023	Sim	✓	✗	✗	✗
SayCan[7]	Apr. 2022	Real	✓	✗	✓	✗
PALM-E[19]	Mar. 2023	Sim, Real	✓	✗	✓	✓
RT-2[11]	Jul. 2023	Real	✓	✗	✓	✓
SayPlan[52]	Jun. 2023	Real	✗	✗	✓	✗
EmbodiedGPT[44]	May 2023	Sim	✓	✗	✓	✓
TaPA[64]	Jul. 2023	Sim	✗	✗	✗	✓
Voyager[59]	May 2023	Game	✗	✓	✓	✗
Octopus	Oct. 2023	Sim, Game	✓	✓	✓	✓

**Embodied AI 相关模型** 最近的研究浪潮集中在将大型语言模型 (LLMs) 与体验 AI 任务相结合 [50, 12, 45, 58]。例如, VoxPoser 通过无监督方法解决了机器人操控问题 [28]。一组项目, 如 SayCan[7], Palm-E[19], RT-2[11] 和 EmbodiedGPT[44], 有效地将视觉或语言线索与机器人操控数据集成在一起。在机器人操控领域之外, 像 Voyager[59] 和 Smallville[46] 这样的倡议利用了 GPT 的能力, 与游戏功能进行交互, 依靠预设的功能来管理复杂的操作。与此平行的是, VisProg[24] 利用 GPT-3 语言提示来编写 Python 程序, 为多种引人入胜的应用打开了大门。虽然 Octopus[65] 也制定了计划和代码, 但其特点在于在程序和代码生成中无缝集成了视觉输入。这也与其他体验规划器 (如 TAPA[64] 和 SayPlan[52]) 形成对比, 它们使用单独的视觉模块将视觉数据转化为 LLMs 的语言输入。Octopus[65] 模型作为一个统一的视觉-语言模型表现出色, 不仅提供计划, 还提供可执行的代码。

## 6 未来发展趋势分析

在和 minecraft 相关的 Embodied Agent 的目前发展当中, 只能生成简洁的代码。当面临复杂任务时, 它经常会出现错误尝试, 并且在校正方面严重依赖环境反馈, 通常无法最终成功。未来的努力可以通过使 Embodied Agent 能够在更具挑战性的环境和任务中导航, 或将其与擅长创建复杂、结构良好的程序的最新 LLM 相结合, 以解决这些缺点。

此外, 现有的 Game Agent 仅在模拟环境中运行。转向现实世界可能会引入许多复杂性。例如, 现实世界的情境可能不提供像 MinEcraft 中那样容易获得的 groundtruth 信息, 这会使分辨环境细微差别变得更加复杂。目前对静态图像输入的依赖也引发了关于视频输入在提高任务性能方面效果的问题。

**场景图生成** 对于 LLM 驱动的智能代理来说, 如何以更高级别的形式来表示场景信息成为一个共同的问题, 其中一个解决方案是利用场景图生成 [69]。如图 12 所示提供了一种利用场景图信息从而使现实机器人进行任务规划和决策的示例。同时, 由于场景图信息富含 high-level 的语义信息, 因此可以替代相对稠密的文字信息作为 prompt。或者将 scene graph 的 token 集成至多模态模型的输入

**LLM 和 VLM 是否是正确的 AGI 方向** : 鉴于 GPT-4 功能的广度和深度, 一些研究人员认为, GPT-4 所代表的大型语言模型可以作为早期版本的 AGI 系统。根据这一思路, 基于 LLMs 和 VLMs 构建 Agent 有可能带来更先进的 AGI 系统。这一论点的主要支撑点在于, 只要在足够大且多样化的数据集 (这些数据集是真实世界的投

影，包含丰富的任务）上对它们进行训练，Embodied Agent 就能具有 AGI 的能力。

然而，另一部分人（被称为反对者）认为，LLM,VLM-based Embodied Agent 并不能发展出真正的强人工智能。他们的主要论点是，依赖于自回归下一个 token 预测的自回归模型无法产生真正的智能，因为它们没有模拟真正的人类思维过程，而只是提供被动反应。

**虚拟仿真环境与真实物理世界之间存在很大差距**：虚拟环境受场景限制，针对特定任务，以模拟的方式进行交互，而真实世界的环境是无限的，可容纳各种任务，以物理的方式进行交互。因此，要弥合这一差距，Agent 必须应对来自外部因素和自身能力的各种挑战，使其能够在复杂的物理世界中有效导航和操作。首先，最关键的问题是在物理环境中部署 Agent 时需要合适的硬件支持。这对硬件的适应性提出了很高的要求。在模拟环境中，Agent 的感知空间和行动空间都是虚拟的。这意味着，在大多数情况下，无论是感知输入还是生成输出，都能保证 Agent 操作的结果。当 Agent 过渡到真实物理环境时，其指令可能无法被传感器或机械臂等硬件设备很好地执行，从而严重影响 Agent 的任务效率。

此外，在面对一个无限开放的世界时，Agent 的有限环境也会带来巨大挑战。这决定了 Agent 能否有效处理来自世界的大量信息并顺利运行。最后，在模拟环境中，Agent 的输入和输出都是虚拟的，可以进行无数次的试错尝试。在这种情况下，对错误的容忍度很高，不会造成实际伤害。然而，在物理环境中，Agent 的不当行为或错误可能会对环境造成真正的伤害，有时甚至是不可逆转的伤害。因此，非常有必要制定适当的法规和标准。我们需要关注 Agent 在做出决定和产生行动时的安全性，确保它们不会对现实世界造成威胁或伤害。

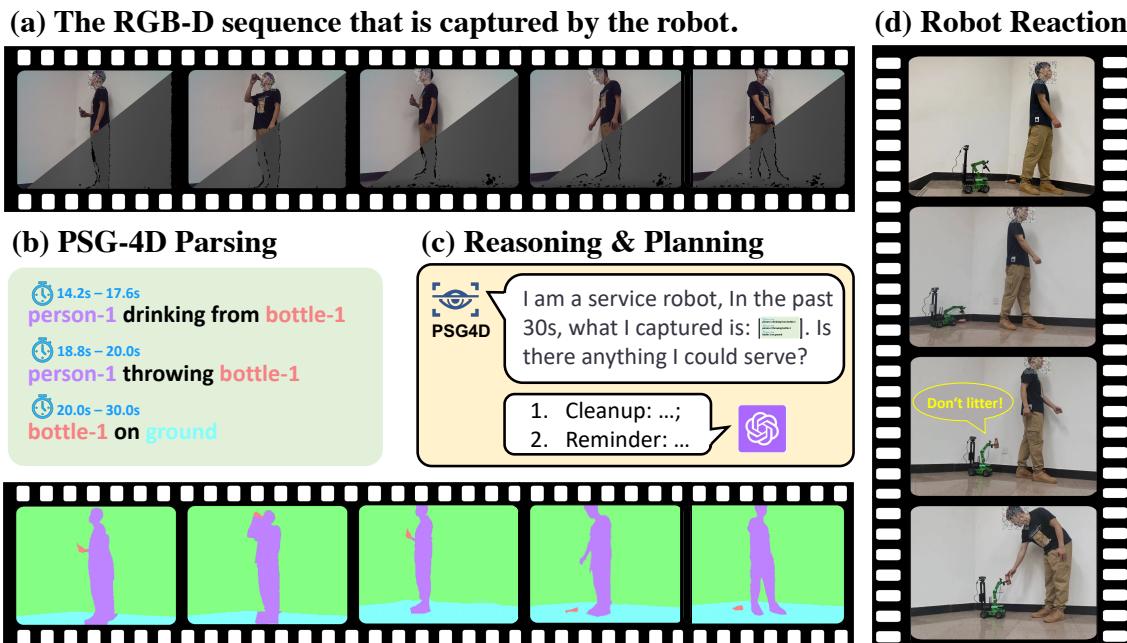


图 12: PSG4D 模型部署的机器人演示。服务机器人解释了 (a) 中显示的 RGB-D 序列，其中一个人在喝咖啡后将空瓶子扔在地上。机器人处理这个序列，将其转化为 (b) 中所示的 4D 场景图。该图包括一组具有时间戳的三元组，每个对象与一个全景掩模相关联，准确地定位在 3D 空间中。机器人定期更新其 PSG4D 到 GPT-4，等待反馈和指令。在这种情景下，GPT-4 建议机器人清理弃置的瓶子并提醒那个人关于他的行为。这一指令被转化为机器人的行动，如 (d) 中所示。

## 参考文献

- [1] Grand theft auto v, 2014.
- [2] Flamingo: a visual language model for few-shot learning, 2022.
- [3] Gpt-4 technical report, 2023.
- [4] Minecraft, 2023.
- [5] D. Adiwardana, M.-T. Luong, D. R. So, J. Hall, N. Fiedel, R. Thoppilan, Z. Yang, A. Kulshreshtha, G. Nemade, Y. Lu, and Q. V. Le. Towards a human-like open-domain chatbot, 2020.
- [6] M. Ahn, A. Brohan, N. Brown, Y. Chebotar, O. Cortes, B. David, C. Finn, C. Fu, K. Gopalakrishnan, K. Hausman, A. Herzog, D. Ho, J. Hsu, J. Ibarz, B. Ichter, A. Irpan, E. Jang, R. J. Ruano, K. Jeffrey, S. Jesmonth, N. Joshi, R. Julian, D. Kalashnikov, Y. Kuang, K.-H. Lee, S. Levine, Y. Lu, L. Luu, C. Parada, P. Pastor, J. Quiambao, K. Rao, J. Rettinghouse, D. Reyes, P. Sermanet, N. Sievers, C. Tan, A. Toshev, V. Vanhoucke, F. Xia, T. Xiao, P. Xu, S. Xu, M. Yan, and A. Zeng. Do as i can and not as i say: Grounding language in robotic affordances. In [arXiv preprint arXiv:2204.01691](#), 2022.
- [7] M. Ahn, A. Brohan, N. Brown, Y. Chebotar, O. Cortes, B. David, C. Finn, C. Fu, K. Gopalakrishnan, K. Hausman, A. Herzog, D. Ho, J. Hsu, J. Ibarz, B. Ichter, A. Irpan, E. Jang, R. J. Ruano, K. Jeffrey, S. Jesmonth, N. Joshi, R. Julian, D. Kalashnikov, Y. Kuang, K.-H. Lee, S. Levine, Y. Lu, L. Luu, C. Parada, P. Pastor, J. Quiambao, K. Rao, J. Rettinghouse, D. Reyes, P. Sermanet, N. Sievers, C. Tan, A. Toshev, V. Vanhoucke, F. Xia, T. Xiao, P. Xu, S. Xu, M. Yan, and A. Zeng. Do as i can and not as i say: Grounding language in robotic affordances. In [arXiv preprint arXiv:2204.01691](#), 2022.
- [8] Anonymous. Steve-eye: Equipping LLM-based embodied agents with visual perception in open worlds. In [Submitted to The Twelfth International Conference on Learning Representations](#), 2023. under review.
- [9] A. Awadalla, I. Gao, J. Gardner, J. Hessel, Y. Hanafy, W. Zhu, K. Marathe, Y. Bitton, S. Gadre, S. Sagawa, J. Jitsev, S. Kornblith, P. W. Koh, G. Ilharco, M. Wortsman, and L. Schmidt. Openflamingo: An open-source framework for training large autoregressive vision-language models. [arXiv preprint arXiv:2308.01390](#), 2023.
- [10] G. Brockman, V. Cheung, L. Pettersson, J. Schneider, J. Schulman, J. Tang, and W. Zaremba. Openai gym. [arXiv preprint arXiv:1606.01540](#), 2016.
- [11] A. Brohan, N. Brown, J. Carbajal, Y. Chebotar, X. Chen, K. Choromanski, T. Ding, D. Driess, A. Dubey, C. Finn, et al. Rt-2: Vision-language-action models transfer web knowledge to robotic control. [arXiv preprint arXiv:2307.15818](#), 2023.
- [12] T. Brown, B. Mann, N. Ryder, M. Subbiah, J. D. Kaplan, P. Dhariwal, A. Neelakantan, P. Shyam, G. Sastry, A. Askell, et al. Language models are few-shot learners. [Advances in neural information processing systems](#), 33:1877–1901, 2020.
- [13] A. Chang, A. Dai, T. Funkhouser, M. Halber, M. Niessner, M. Savva, S. Song, A. Zeng, and Y. Zhang. Matterport3d: Learning from rgb-d data in indoor environments. [arXiv preprint arXiv:1709.06158](#), 2017.
- [14] M. Choi, J. Pei, S. Kumar, C. Shu, and D. Jurgens. Do llms understand social knowledge? evaluating the sociability of large language models with socket benchmark, 2023.
- [15] C. Colas, L. Teodorescu, P.-Y. Oudeyer, X. Yuan, and M.-A. Côté. Augmenting autotelic agents with large language models, 2023.
- [16] H. Dai, Y. Li, Z. Liu, L. Zhao, Z. Wu, S. Song, Y. Shen, D. Zhu, X. Li, S. Li, X. Yao, L. Shi, Q. Li, Z. Chen, D. Zhang, G. Mai, and T. Liu. Ad-autogpt: An autonomous gpt for alzheimer’s disease infodemiology, 2023.
- [17] W. Dai, J. Li, D. Li, A. M. H. Tiong, J. Zhao, W. Wang, B. Li, P. Fung, and S. Hoi. Instructblip: Towards general-purpose vision-language models with instruction tuning, 2023.
- [18] X. Deng, Y. Gu, B. Zheng, S. Chen, S. Stevens, B. Wang, H. Sun, and Y. Su. Mind2web: Towards a generalist agent for the web, 2023.
- [19] D. Driess, F. Xia, M. S. M. Sajjadi, C. Lynch, A. Chowdhery, B. Ichter, A. Wahid, J. Tompson, Q. Vuong, T. Yu, W. Huang, Y. Chebotar, P. Sermanet, D. Duckworth, S. Levine, V. Vanhoucke, K. Hausman, M. Toussaint, K. Greff, A. Zeng, I. Mordatch, and P. Florence. Palm-e: An embodied multimodal language model. In [arXiv preprint arXiv:2303.03378](#), 2023.
- [20] L. Fan, G. Wang, Y. Jiang, A. Mandlekar, Y. Yang, H. Zhu, A. Tang, D.-A. Huang, Y. Zhu, and A. Anandkumar. Minedoj: Building open-ended embodied agents with internet-scale knowledge, 2022.
- [21] H. Fu, W. Xu, R. Ye, H. Xue, Z. Yu, T. Tang, Y. Li, W. Du, J. Zhang, and C. Lu. Rfuniverse: A multiphysics simulation platform for embodied ai, 2023.

- [22] Y. Fu, H. Peng, T. Khot, and M. Lapata. Improving language model negotiation with self-play and in-context learning from ai feedback, 2023.
- [23] P. Gao, S. Geng, R. Zhang, T. Ma, R. Fang, Y. Zhang, H. Li, and Y. Qiao. Clip-adapter: Better vision-language models with feature adapters, 2021.
- [24] T. Gupta and A. Kembhavi. Visual programming: Compositional visual reasoning without training. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 14953–14962, 2023.
- [25] W. Hill, I. Liu, A. D. M. Koch, D. Harvey, G. Konidaris, and S. James. Mineplanner: A benchmark for long-horizon planning in large minecraft worlds, 2023.
- [26] S. Hong, M. Zhuge, J. Chen, X. Zheng, Y. Cheng, C. Zhang, J. Wang, Z. Wang, S. K. S. Yau, Z. Lin, L. Zhou, C. Ran, L. Xiao, C. Wu, and J. Schmidhuber. Metagpt: Meta programming for a multi-agent collaborative framework, 2023.
- [27] S. Huang, Z. Jiang, H. Dong, Y. Qiao, P. Gao, and H. Li. Instruct2act: Mapping multi-modality instructions to robotic actions with large language model. *arXiv preprint arXiv:2305.11176*, 2023.
- [28] W. Huang, C. Wang, R. Zhang, Y. Li, J. Wu, and L. Fei-Fei. Voxposer: Composable 3d value maps for robotic manipulation with language models. *arXiv preprint arXiv:2307.05973*, 2023.
- [29] J.-H. Jhan, C.-P. Liu, S.-K. Jeng, and H.-Y. Lee. Cheerbots: Chatbots toward empathy and emotionusing reinforcement learning, 2021.
- [30] S. S. Y. Kim, E. A. Watkins, O. Russakovsky, R. Fong, and A. Monroy-Hernández. “help me help the ai” : Understanding how explainability can support human-ai interaction. In *Proceedings of the 2023 CHI Conference on Human Factors in Computing Systems, CHI ’ 23*. ACM, Apr. 2023.
- [31] E. Kolve, R. Mottaghi, W. Han, E. VanderBilt, L. Weihs, A. Herrasti, M. Deitke, K. Ehsani, D. Gordon, Y. Zhu, et al. Ai2-thor: An interactive 3d environment for visual ai. *arXiv preprint arXiv:1712.05474*, 2017.
- [32] B. Li, Y. Zhang, L. Chen, J. Wang, F. Pu, J. Yang, C. Li, and Z. Liu. Mimic-it: Multi-modal in-context instruction tuning. 2023.
- [33] B. Li, Y. Zhang, L. Chen, J. Wang, J. Yang, and Z. Liu. Otter: A multi-modal model with in-context instruction tuning. *arXiv preprint arXiv:2305.03726*, 2023.
- [34] C. Li, R. Zhang, J. Wong, C. Gokmen, S. Srivastava, R. Martín-Martín, C. Wang, G. Levine, M. Lingelbach, J. Sun, et al. Behavior-1k: A benchmark for embodied ai with 1,000 everyday activities and realistic simulation. In *Conference on Robot Learning*, pages 80–93. PMLR, 2023.
- [35] G. Li, H. A. A. K. Hammoud, H. Itani, D. Khizbullin, and B. Ghanem. Camel: Communicative agents for ”mind” exploration of large language model society, 2023.
- [36] L. Li, Y. Yin, S. Li, L. Chen, P. Wang, S. Ren, M. Li, Y. Yang, J. Xu, X. Sun, L. Kong, and Q. Liu. M<sup>3</sup>it: A large-scale dataset towards multi-modal multilingual instruction tuning. *arXiv preprint arXiv:2306.04387*, 2023.
- [37] J. Liang, W. Huang, F. Xia, P. Xu, K. Hausman, B. Ichter, P. Florence, and A. Zeng. Code as policies: Language model programs for embodied control, 2023.
- [38] K. Lin, C. Agia, T. Migimatsu, M. Pavone, and J. Bohg. Text2motion: From natural language instructions to feasible plans. *arXiv preprint arXiv:2303.12153*, 2023.
- [39] Y.-T. Lin and Y.-N. Chen. Llm-eval: Unified multi-dimensional automatic evaluation for open-domain conversations with large language models, 2023.
- [40] B. Liu, Y. Jiang, X. Zhang, Q. Liu, S. Zhang, J. Biswas, and P. Stone. Llm+p: Empowering large language models with optimal planning proficiency, 2023.
- [41] H. Liu, C. Li, Q. Wu, and Y. J. Lee. Visual instruction tuning, 2023.
- [42] X. Liu, H. Yu, H. Zhang, Y. Xu, X. Lei, H. Lai, Y. Gu, H. Ding, K. Men, K. Yang, S. Zhang, X. Deng, A. Zeng, Z. Du, C. Zhang, S. Shen, T. Zhang, Y. Su, H. Sun, M. Huang, Y. Dong, and J. Tang. Agentbench: Evaluating llms as agents, 2023.
- [43] N. Mehta, M. Teruel, P. F. Sanz, X. Deng, A. H. Awadallah, and J. Kiseleva. Improving grounded language understanding in a collaborative environment by interacting with agents through help feedback, 2023.
- [44] Y. Mu, Q. Zhang, M. Hu, W. Wang, M. Ding, J. Jin, B. Wang, J. Dai, Y. Qiao, and P. Luo. Embodiedgpt: Vision-language pre-training via embodied chain of thought. *arXiv preprint arXiv:2305.15021*, 2023.

- [45] L. Ouyang, J. Wu, X. Jiang, D. Almeida, C. L. Wainwright, P. Mishkin, C. Zhang, S. Agarwal, K. Slama, A. Ray, J. Schulman, J. Hilton, F. Kelton, L. Miller, M. Simens, A. Askell, P. Welinder, P. Christiano, J. Leike, and R. Lowe. Training language models to follow instructions with human feedback, 2022.
- [46] J. S. Park, J. C. O’Brien, C. J. Cai, M. R. Morris, P. Liang, and M. S. Bernstein. Generative agents: Interactive simulacra of human behavior, 2023.
- [47] X. Puig, K. Ra, M. Boben, J. Li, T. Wang, S. Fidler, and A. Torralba. Virtualhome: Simulating household activities via programs. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 8494–8502, 2018.
- [48] C. Qian, X. Cong, W. Liu, C. Yang, W. Chen, Y. Su, Y. Dang, J. Li, J. Xu, D. Li, Z. Liu, and M. Sun. Communicative agents for software development, 2023.
- [49] A. Radford, J. W. Kim, C. Hallacy, A. Ramesh, G. Goh, S. Agarwal, G. Sastry, A. Askell, P. Mishkin, J. Clark, G. Krueger, and I. Sutskever. Learning transferable visual models from natural language supervision, 2021.
- [50] A. Radford, J. Wu, R. Child, D. Luan, D. Amodei, I. Sutskever, et al. Language models are unsupervised multitask learners. *OpenAI blog*, 1(8):9, 2019.
- [51] C. Raffel, N. Shazeer, A. Roberts, K. Lee, S. Narang, M. Matena, Y. Zhou, W. Li, and P. J. Liu. Exploring the limits of transfer learning with a unified text-to-text transformer, 2023.
- [52] K. Rana, J. Haviland, S. Garg, J. Abou-Chakra, I. Reid, and N. Suenderhauf. Sayplan: Grounding large language models using 3d scene graphs for scalable task planning. *arXiv preprint arXiv:2307.06135*, 2023.
- [53] L. Salewski, S. Alaniz, I. Rio-Torto, E. Schulz, and Z. Akata. In-context impersonation reveals large language models’ strengths and biases, 2023.
- [54] M. Savva, A. Kadian, O. Maksymets, Y. Zhao, E. Wijmans, B. Jain, J. Straub, J. Liu, V. Koltun, J. Malik, et al. Habitat: A platform for embodied ai research. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 9339–9347, 2019.
- [55] I. Singh, V. Blukis, A. Mousavian, A. Goyal, D. Xu, J. Tremblay, D. Fox, J. Thomason, and A. Garg. Progprompt: Generating situated robot task plans using large language models, 2022.
- [56] R. Taori, I. Gulrajani, T. Zhang, Y. Dubois, X. Li, C. Guestrin, P. Liang, and T. B. Hashimoto. Stanford alpaca: An instruction-following llama model. [https://github.com/tatsu-lab/stanford\\_alpaca](https://github.com/tatsu-lab/stanford_alpaca), 2023.
- [57] M. Tatsubori, A. Munawar, and T. Moriyama. Design and implementation of linked planning domain definition language, 2019.
- [58] H. Touvron, T. Lavril, G. Izacard, X. Martinet, M.-A. Lachaux, T. Lacroix, B. Rozière, N. Goyal, E. Hambro, F. Azhar, et al. Llama: Open and efficient foundation language models. *arXiv preprint arXiv:2302.13971*, 2023.
- [59] G. Wang, Y. Xie, Y. Jiang, A. Mandlekar, C. Xiao, Y. Zhu, L. Fan, and A. Anandkumar. Voyager: An open-ended embodied agent with large language models, 2023.
- [60] L. Wang, C. Ma, X. Feng, Z. Zhang, H. Yang, J. Zhang, Z. Chen, J. Tang, X. Chen, Y. Lin, W. X. Zhao, Z. Wei, and J.-R. Wen. A survey on large language model based autonomous agents, 2023.
- [61] J. Wei, Y. Tay, R. Bommasani, C. Raffel, B. Zoph, S. Borgeaud, D. Yogatama, M. Bosma, D. Zhou, D. Metzler, E. H. Chi, T. Hashimoto, O. Vinyals, P. Liang, J. Dean, and W. Fedus. Emergent abilities of large language models, 2022.
- [62] Y. Wu, S. Y. Min, Y. Bisk, R. Salakhutdinov, A. Azaria, Y. Li, T. Mitchell, and S. Prabhumoye. Plan, eliminate, and track – language models are good teachers for embodied agents, 2023.
- [63] Y. Wu, Y. Wu, G. Gkioxari, and Y. Tian. Building generalizable agents with a realistic and rich 3d environment. *arXiv preprint arXiv:1801.02209*, 2018.
- [64] Z. Wu, Z. Wang, X. Xu, J. Lu, and H. Yan. Embodied task planning with large language models. *arXiv preprint arXiv:2307.01848*, 2023.
- [65] J. Yang, Y. Dong, S. Liu, B. Li, Z. Wang, C. Jiang, H. Tan, J. Kang, Y. Zhang, K. Zhou, and Z. Liu. Octopus: Embodied vision-language programmer from environmental feedback, 2023.
- [66] W. Yu, N. Gileadi, C. Fu, S. Kirmani, K.-H. Lee, M. G. Arenas, H.-T. L. Chiang, T. Erez, L. Hasenclever, J. Humplik, et al. Language to rewards for robotic skill synthesis. *arXiv preprint arXiv:2306.08647*, 2023.
- [67] H. Zhang, W. Du, J. Shan, Q. Zhou, Y. Du, J. B. Tenenbaum, T. Shu, and C. Gan. Building cooperative embodied agents modularly with large language models, 2023.

- [68] R. Zheng, S. Dou, S. Gao, Y. Hua, W. Shen, B. Wang, Y. Liu, S. Jin, Q. Liu, Y. Zhou, L. Xiong, L. Chen, Z. Xi, N. Xu, W. Lai, M. Zhu, C. Chang, Z. Yin, R. Weng, W. Cheng, H. Huang, T. Sun, H. Yan, T. Gui, Q. Zhang, X. Qiu, and X. Huang. Secrets of rlhf in large language models part i: Ppo, 2023.
- [69] G. Zhu, L. Zhang, Y. Jiang, Y. Dang, H. Hou, P. Shen, M. Feng, X. Zhao, Q. Miao, S. A. A. Shah, and M. Bennamoun. Scene graph generation: A comprehensive survey, 2022.
- [70] X. Zhu, Y. Chen, H. Tian, C. Tao, W. Su, C. Yang, G. Huang, B. Li, L. Lu, X. Wang, Y. Qiao, Z. Zhang, and J. Dai. Ghost in the minecraft: Generally capable agents for open-world environments via large language models with text-based knowledge and memory, 2023.
- [71] Y. Zhu, J. Wong, A. Mandlekar, R. Martín-Martín, A. Joshi, S. Nasiriany, and Y. Zhu. robosuite: A modular simulation framework and benchmark for robot learning. *arXiv preprint arXiv:2009.12293*, 2020.