RoboCodeX

Introduction

Input

RGBD data

Human Instructions

Structure

tree-of-thought

Procedure

将高级人类指令分解为通过文本代码注释定义的若干以对象为中心的操作单元

Units

Target pose proposals generation

通过对环境中的物体进行详细的三维空间分析生成目标姿势建议

Object physical characteristic prediction

利用视觉灵感 (visual inspiration) 和三维几何信息（如铰接物体上的平移或旋转轴约束）,确保生成现实可行的运动计划

Preference prediction

利用对环境和任务目标的多模态理解进行偏好预测，选择最佳抓取位置和接近方向

Trajectory generation

利用规划算法生成轨迹，以生成安全、优化且符合碰撞和物理约束条件的运动计划

Method

Problem setup

1. high level free-form human instruction

$$L_{global}$$

2. RGBD data

$$I$$

从左、右、上三个不同的深度摄像机获得的 view

3. decompose

$$U_{task} \rightarrow (u_1, u_2, \cdots, u_n)$$

4. motion trajectory set

$$\{\tau_i\}_{i=0}^{n}$$

我们将 τ_i 表示为由 Operational Space Controller (Khatib，1987)（另一个框架，用于分析和控制机械手系统） 执行的 dense end-effector waypoints 序列，其中每个waypoint由所需的 6-DoF end-effector pose, end-effector velocity 和 gripper action 组成。

5. 给定 ground-truth 指令 l_i^* 所描述的第 i 个子任务，提出一个优化问题，其定义如下：

$$\min_{\tau_i} \sum_{i=0}^{N} \{S_{task}(\tau_i, l_i^*) + S_{control}(\tau_i)\} \quad s.t. \quad C(\tau_i)$$

其中，S_{task} 表示 τ_i 完成指令 l_i^* 的程度，$S_{control}$ 表示控制成本

$C_(\tau_i)$ 表示动力学或运动学约束

6. 通过对每个子任务 l_i 的优化求解，我们得到了一系列机器人轨迹，这些轨迹共同完成了指令 L_{global} 指定的总体任务。

Multi-modal Tree-of-thought Code Generation

1. Language description of the sub-task

$$L_i$$

2. Manipulation preference

$$pf_i$$

表示考虑到控制稳定性的接近方向或首选触摸位置

3. decompose

$$L_{units}, pf_{units} = f(I, L_{global})$$
$$L_{units} = \{L_0, L_1, \cdots, L_N\}$$
$$pf_{units} = \{pf_0, pf_1, \cdots, pf_N\}$$

4. accurate 3D point cloud

$$O_i$$

对于每个以对象为中心的操作单元，将从RGB流中获得的接地二维位置（*grounded 2D positions*）与点云中的三维盒子（*3D boxes from the point cloud*）进行匹配，匹配依据是它们在空间上的重叠程度和方向的一致性（*based on overlap and orientation consistency*）。这个过程使得可以从点云中提取出精确的三维点云数据，记作 O_i，它代表了与任务相关的对象。

这里的“RGB流”指的是彩色图像序列，而“接地二维位置”意味着从这些彩色图像中提取的、与地面平面对齐的二维坐标。“点云”是一个三维数据集，它包含了场景中每个点的三维坐标，而“三维盒子”是对场景中对象的包围框，它定义了对象在三维空间中的位置和大小。

匹配过程考虑了两个因素：一是二维位置和三维盒子在空间上的重叠程度，二是它们在方向上的连贯性或一致性。

因此，人话版：对于场景中的每个对象，系统会从彩色图像序列中提取该对象的二维位置，并将其与从点云数据中获得的对应的三维盒子进行匹配。这种匹配是基于二维位置和三维盒子在空间上的重叠程度以及它们的方向是否一致来进行的。这种方法有助于将二维图像中的对象信息与三维空间中的实际对象位置相对应，从而实现更精确的对象识别和场景理解。

5. the object-centric trajectory generation problem

$$\tau_i = g(O_i, L_i, pf_i)$$

在 sub-task 的 language instruction L_i 条件下

τ_i 是第 i 个 sub-task 的运动轨迹，用函数 $g(\cdot)$ 根据第 i 个 sub-task 的 preference pf_i 和 language instruction L_i 生成。

τ_i 包含了一系列 dense end-effector waypoints，这些 waypoint 需要被执行，以及用于控制 gripper 开合的离散控制指令。

6. 然后，聚合的对象特定感知推断、物理洞察和操作参数被系统地编译成结构化的可执行动作代码。每个单元被视为一个父节点，并进一步扩展到以下部分：

    a. ***part-level affordance*** $A_i = h(O_i, L_i)$ ***prediction***
    这是与任务相关的部件点云分割，例如抽屉的把手区域。

    b. ***grasp pose proposals*** pc_i ***prediction***

$$pc_i = \{pc_{i0}, pc_{i1}, \cdots, pc_{ik}\} = \mathrm{cand}(A_i)$$

    函数 $\mathrm{cand}(\cdot)$ 是一个从 $\mathbb{N}^3$ 到 $\mathrm{SE}(3)$ 的映射，用于生成 $\mathrm{SE}(3)$ 姿态。其中 $\mathbb{N}^3$ 表示三维欧几里得空间，而 $\mathrm{SE}(3)$ 应该就是 ***Homogeneous Transformation Matrix***，即*齐次变换矩阵*。

    论文使用了预训练的 **Anygrasp** (Fang et al., 2023) 来生成 grasp pose proposals，但具体来说，实际上使用了两种模型，允许代理根据由高级算法框架RoboCodeX推断出的偏好进行选择：

i. Anygrasp
Anygrasp是一个复杂的预训练模型，擅长生成抓取姿态。它使用单个RGB图像和相应的点云来提出无碰撞的抓取，特别适合平行颚式夹具。该模型的有效性在于其能够解释复杂场景并提出可行的抓取解决方案。

ii. "central lift" method
通过生成自上而下的抓取姿态采用不同的方法。这种方法简化了抓取生成过程，专注于物体的中心提升点，这些点往往与其质心对齐。

为了进一步细化抓取选择过程，三个具体偏好被整合到“parse_adaptive_shape_grasp_pose()”函数中。这些偏好如下：

- **Preferred Position**
  Optional，表示夹具尖端点的首选位置。
- **Preferred Approach Direction**
  Optional，表示夹具的首选接近方向。
- **Preferred Plane Normal**
  Optional，表示夹具平面首选手选法线方向。

一旦确定了最优的抓取姿态，接下来的阶段涉及其执行。利用 Dasari et al. (2023) 的见解，采用了一种简化的预抓取策略：假设 $\vec{p}$ 表示识别的抓取点，$\vec{a}$ 示抓取模型建议的接近向量。机器人夹具遵循的轨迹数学表达为：

$$< \vec{p} - 0.1\vec{a}, ; \vec{p} - 0.08\vec{a}, ; \vec{p} >$$

这表明从预定义位置向目标物体逐渐减少运动增量，以确保精度和稳定性。这种渐进减速是过程中的一个关键方面，尤其是对于轻质或易碎的物体，它们容易因突然运动而移位或损坏。

c. ***object physical property prediction***

$$\phi_i = \{\phi_{i0}, \phi_{i1}, \cdots, \phi_{ik}\} = z(O_i, L_i)$$

例如铰接物体的关节信息。

文章使用了 GAMMA (Yu et al., 2023b) 模型来预测关节物体的物理属性，它将关节物体点云分割成刚体部分并估计关节参数。具体来说，是使用多视角的改进点云进行微调的 GAMMA 模型，GAMMA 使用 PointNet++ (Qi et al., 2017) 提取点特征，然后进行分割、部件偏移和关节轴回归，GAMMA 的具体训练过程如下：

> *首先使用多视角改进是因为GAMMA 只使用单个深度摄像机生成的点云，会使实验设置中出现域差距 (domain gaps)，除此之外一切都与 GAMMA 相同。*

> *GAMMA 框架首先分析铰接物体，将其分割成不同的刚性部分，并从部分点云观测值 $P = \{p_i \in \mathbb{R}^3\}_{i=1}^N$ 估算其铰接参数。如刚才所述，GAMMA 利用 PointNet++ 骨干进行特征提取，包括三个关键部分：**Segmentation**、**Offset** 和 **Joint Parameter**。*
>
> i. ***Segmentation Head***
> *将每个点分类为静态 static，旋转 revolute，棱柱 prismatic ($\hat{c}_i \in \{0, 1, 2\}$)*
>
> ii. ***Offset Head***
> *计算偏移向量 ($\hat{o}_i \in \mathbb{R}^3$)，引导个点朝向其部分中心点*
>
> iii. ***Joint Parameter Head***
> *预测关节参数，将各点投影到关节轴（$\hat{v}_i \in \mathbb{R}^3$）上并估计轴方向（$\hat{d}_i \in \mathbb{R}^3$）*
>
> *损失函数结合了 segmentation, offset, and joint parameter losses：*
>
> $$L = \frac{1}{N}\sum_i^N [L_c(\hat{c}_i, c_i) + L_o(\hat{o}_i, o_i) + L_d(\hat{v}_i, v_i) + L_d(\hat{d}_i, d_i)]$$
>
> *其中 L_o 是 offset loss*
>
> $$L_o(\hat{o}_i, o_i) = ||\hat{o}_i - o_i|| - (\frac{o_i}{||o_i||_2} \cdot \frac{\hat{o}_i}{||\hat{o}_i||_2})$$

文章还在代码生成过程中调用 Open3D (Zhou et al., 2018) 中的平面检测功能，以对象的点云作为输入，输出检测到的平面以及法向量信息。

d. ***trajectory planning***

通过整合运动规划算法和机器人操作系统（ROS）操作模块，模型最终输出动态可行的机器人轨迹，确保碰撞避免和奇点排除。

$$\tau_i = Z(pc_i, \phi_i, pf_i, L_i)$$

具体来说，文章在轨迹规划中使用零阶优化，通过随机采样轨迹并使用 $S_{control}$ 和占用图约束进行评估。

Dataset Preparation

Pre-training Dataset

文章借助 GPT-4 开发了一个多样化的机器人多模态代码生成数据集。为确保数据集的多样性，我们设计了一个程序化数据生成框架：

首先从 HM3D 数据集 (Ramakrishnan et al., 2021) 中随机抽样家庭场景，

随后在语义合适的地方插入额外的物体，例如，将物体放在桌子和柜台的顶部。

然后在场景中插入额外的物体，这些物体分为独立物体（如球、玩具、水果）和容器物体（如碗、盘、杯）。这些物体采样自 Google Scan Dataset (Downs et al., 2022), YCB Dataset (Calli et al., 2015), OmniObject3D Dataset (Wu et al., 2023b), and articulated object dataset AKB-48(Liu et al., 2022).

通过随机选择物体类别和数量，构建复杂场景配置。

基于生成的环境，研究人员形成自然语言描述，这些描述被输入到GPT-4语言模型中，以产生适合给定场景配置的自由形式任务描述，例如对象操纵和重新排列。

对于每个生成的任务，GPT-4 （不用 GPT4V 是为了降低成本）从任务描述和额外的参数化输入中生成完成任务的编程代码，并从每个任务中采样10个高质量的代码样本，由GPT-3.5评估，并过滤掉语法错误的样本，以获得机器人可以在模拟中执行的可执行程序。

通过这种将环境生成、任务说明和程序合成与大型语言模型相结合的流水线，文章建立了一个多样化的数据集，用于预训练机器人视觉语言推理模型，以完成各种日常家务劳动。

为了避免过拟合，同时采用了通用视觉模型数据集和生成的数据集，通用视觉模型数据集来自 ShareGPT4V dataset(Chen et al., 2023c), SViT dataset(Zhao et al., 2023) 和 LLaVA Visual Instruct 150K(Liu et al., 2023d).

SFT Dataset

有监督微调（supervised fine-tuning, SFT）过程经过了精心设计，通过对在评估中表现出较高成功率的高质量代码序列进行训练，以提高模型生成机器人代码的能力。

数据集基于 RT-1(Brohan et al., 2022) 和 LIBERO(Liu et al. 2023b) 数据集，从中随机组合为不同种类任务提供多样化的任务集合。

对每种任务，都提供了人工的经过模拟和现实测试的高质量示例，将高质量示例和 API 说明输入给 GPT-4 以让其生成相关代码。

对于语法正确却执行失败的代码，想办法找到最佳执行，使用 GPT-4V 分析为什么这样好，将其总结并编码注释，作为思维链 (chain-of-thought) 的标签。

对于如何调整可选设置都无法正确执行的代码，进行手动标记并放入人工标记数据池。

同样需要防止过拟合，因此也使用了通用视觉语言数据，使用与 ShareGPT4V-7B 模型 SFT 过程中使用的 ShareGPT4V 数据集相同的部分。

Vision Language Model Design

此论文的模型采用了和 BLIP2(Li et al., 2023b) 相同的范式，由 vision transformer, Q-Former 和 language model 组成，其中 Q-Former 的作用是连接和压缩视觉模态的 token 量以节省 GPU memory。

为了从图片中获取层次化特征，论文特地设计了一种 vision adapter ，它能将不同阶段的特征汇总到 vision transformer 中，具体来说：

首先将视觉模型的隐藏层按比例分为四部分， 然后从每部分的的最后一层提取 class token（因其通过该层的 self-attention 和所有 token 互动），并通过 vision adapter 使它们相交互。此 vision adapter 是一个 channel-wise attention network(Yan et al., 2021)，首先通过线性层减少通道维度，然后使用 SILU(Swish-Relu) (Paul et al., 2022) 激活函数选择特征，最后，通过另一个线性层恢复原始的通道维度。

聚合的特征令牌与 vision transformer 的其他令牌结合，结合后的令牌集作为后续模块的输入。

通过这种设计，研究人员能够从视觉Transformer模型中提取层次化的特征，并确保这些特征之间能够相互交互和融合。这种层次化的特征提取有助于模型更好地理解图像中的不同层次信息，从而在多模态任务中提供更准确和细粒度的预测。

```
class Adapter(nn.Module):
    def __init__(self, config):
        super().__init__()
        self.config = config
        self.activation_fn = ACT2FN("silu")
        hidden_size = config.vision_config.hidden_size
        intermediate_size = hidden_size // 4
        output_size = config.qformer_config.hidden_size
        self.fc1 = nn.Linear(hidden_size, intermediate_size)
```

```
        self.fc2 = nn.Linear(intermediate_size, output_size)
        self.layernorm = nn.LayerNorm(output_size,
eps=config.vision_config.layer_norm_eps)

    def forward(self, hidden_states: torch.Tensor) ->
torch.Tensor:
        hidden_states = self.fc1(hidden_states)
        hidden_states = self.activation_fn(hidden_states)
        hidden_states = self.fc2(hidden_states)
        hidden_states =  self.layernorm(hidden_states)
        return hidden_states

```