arXiv:2611.30111v1 [cs.RO] 23 Nov 2026

RoboThink：让智能体思考，让策略执行

Yunbei Zhang[1] Zijian Jin[2] Janet Wang[1] Xingchen Zhao[3] Zhenyu Zhang[4] Weirui Ye[5] Yilun Du[6]TulaneUniversity [2]纽约大学 [3]NortheasternUniversity [4]StanfordUniversity [5]麻省理工学院 [6]HarvardUniversity

项目页面与代码： https://Robo-Think.github.io

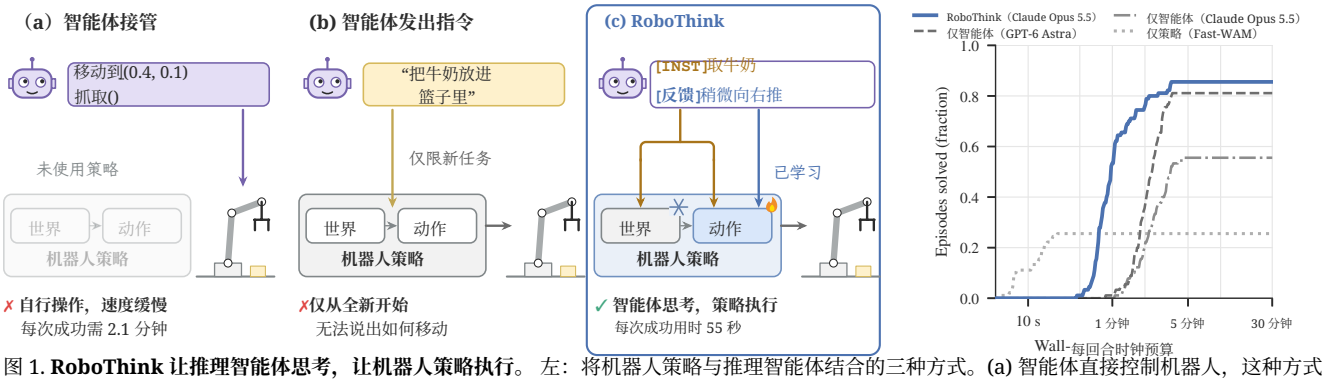


图 1. **RoboThink 让推理智能体思考，让机器人策略执行。** 左：将机器人策略与推理智能体结合的三种方式。(a) 智能体直接控制机器人，这种方式通用但速度慢。(b) 智能体发送指令，由策略执行，但只有从全新开始时才能生效。(c) 在 RoboThink 中，智能体的结论通过两个通道传达给策略：［INST］重写指令，而从智能体处［FB］,已学习的反馈则传达给经过训练的动作模块 ( )，同时世界模块保持冻结 ( )。右：在漂移情况下，实际时间预算内解决的 LIBERO 回合占比。每次成功的中位用时：RoboThink 为 55 秒，单独使用智能体时分别为 2.1 分钟和 2.3 分钟。

摘要

机器人策略可以实时执行动作，但不知道原因就会失败；而观察机器人的推理智能体通常能判断哪里出了问题。现有系统要么将控制权交给智能体，每个回合需要几分钟；要么让智能体发送指令。然而，策略只有从全新开始时才能遵循新指令，因为未完成的任务会使其处于训练数据之外的状态。指令也无法修正机器人的运动方式。如果运动指令上的微小漂移不断将机器人的手拉向左侧，那么“稍微向右推”这样的反馈并不是策略已学习遵循的任务。我们提出 *RoboThink*，让智能体思考，并让策略学会根据智能体的指导执行动作。指令通道告诉策略该做什么。机器人开始执行后，用户说“要牛奶，不要黄油”，智能体便会重写指令，机器人也会撤销自己的动作，无需训练。反馈通道则告诉它如何移动。它

通过智能体标注的 *DAgger*、角色标记和反事实反馈，仅利用智能体进行训练；这些信息仅由策略的动作模块读取。在 *LIBERO* 上，面对未知的恒定漂移和随机漂移，搭配 *Claude Opus 5.5* 的 *RoboThink* 分别将成功率从 *0.09* 提高到 *0.87*，以及从 *0.42* 提高到 *0.84*。它还能以 *0.91* 至 *1.00* 的回合成功完成误听和变更后的请求。我们测试过的最强智能体 *GPT-6 Astra* 单独执行时，每次成功需要 *2.1* 分钟，每回合耗费超过一百万个词元；而 *RoboThink* 仅需 *55* 秒和 *8* 千个词元。同一套配方适用于不同策略、视觉-语言-动作模型和世界动作模型，也适用于不同机器人，从单臂、双臂到移动操作机器人皆可。在 *RoboTwin 2.0* 的全部 *50* 个双臂任务中，面对恒定漂移和随机漂移，成功率分别从 *0.12* 提高到 *0.74*，以及从 *0.31* 提高到 *0.56*。最后，智能体可作为即插即用模块。由一个智能体训练的策略也能与其他智能体配合使用，包括开放权重智能体；随着智能体能力增强，策略也会随之变强，无需重新训练。

1. 引言

视觉-语言-动作模型（视觉-语言-动作模型） [3, 4, 17, 28, 30] 与世界动作模型（世界动作模型） [8, 18, 21, 37, 39, 45] 将摄像头图像和一条指令实时映射为动作片段。然而，一旦超出训练条件，它们就会失败，却不知道原因。在部署时，可能出现两种问题。执行可能偏离正确计划。当被要求拿起橙汁时，策略规划了正确的运动，但每一条运动指令上都存在我们称之为漂移的微小未知漂移，不断将手推离



图2. **教会推理智能体的反馈通道**。(1) 在训练任务中，面对训练漂移，机器人执行经智能体估计修正后的策略计划。(2) 每个状态都会产生三类训练样本：以修正后的计划为目标值的智能体反馈、针对另一种漂移且配有相应目标值的反事实反馈，或以计划本身为目标值的无反馈。只有读取反馈才能拟合这些目标值。(3) 训练的只有动作模块和角色标记，且绝不使用真实漂移。

瓶子（图3a）。策略也可能追求错误目标，因为它所执行的指令并非用户想要的。用户请求的措辞可能不同于任何训练指令，例如用“炉灶现在需要打开”来表示“打开炉灶”，策略常常会误解它。它可能被误听，以至于机器人在用户想要牛奶时去取奶油奶酪（图3b）。或者，用户可能在机器人已经开始后改变主意（“要牛奶，不要黄油”）。在这两种情况下，一个观察机器人的推理智能体，例如视觉语言模型（视觉语言模型），能够检测失败并解释其原因 [9, 14, 34]（图1）。

机器人策略和推理智能体应如何协同工作？现有系统有三种途径。智能体可以接管机器人，通过代码或工具调用来控制它 [6, 15, 22]。这种方法既能处理错误执行，也能处理错误目标，但代价高昂。在我们的实验中，GPT-6 Astra 单独行动时，每次成功需要 2.1 分钟，每回合消耗超过一百万个词元。在 RoboDojo 上，它每回合耗时 8.6 分钟 [15]。智能体也可以改为向策略发出指令 [1, 14, 33]。新指令可以纠正错误目标，但策略只能在熟悉的状态下遵循指令，无法在错误任务进行到一半时照做。给出执行三个动作块后的修正任务，训练过的 WAM 仅能在 0.07 的回合中完成任务。此外，这种方法根本无法纠正错误执行，因为“稍微向右推”并不是一项任务。将这种修正附加到强大的 WAM 的提示词中 [39]，甚至会使它在随机漂移下的成功率从 0.42 降至 0.16。最后，还可以将推理能力训练进策略中 [26, 43]，，但这会将其锁定在权重中，导致策略无法纠正自己从未见过的漂移（第E节），也无法借助更强的智能体提升能力。总之，每种途径要么牺牲策略的速度，要么无法将智能体对错误原因的理解传达给策略。

我们的关键洞见是，智能体与策略相辅相成。策略知道如何行动，却无法看出失败的原因；而智能体能看出原因，却行动得太慢。现有方案未能发挥两者的优势：要么让智能体行动、策略推理，要么让智能体用策略无法执行的语言与策略交流。

难点在于策略，而非智能体。策略只接受过遵循任务指令的训练，因此无法根据关于如何纠正其运动的反馈采取行动，因为它从未学会这类反馈的含义。我们缺少的是一种渠道，让策略学会理解智能体告诉它的内容。为此，我们提出 **RoboThink**：智能体负责思考，策略则学会根据智能体的指导采取行动（图 1）。智能体每隔几个动作块才会被调用一次，因为运动指令或用户请求所带来的偏向通常会从一个动作块延续到下一个动作块；在两次调用之间，策略则根据智能体最新的指导实时行动。指导通过两个渠道传达给策略，分别针对错误目标和错误执行。

通过指令通道 [INST]，可以告知策略应采取的行动，从而纠正错误目标。智能体了解用户请求，并检查策略据此执行的指令是否与用户请求一致。如果请求表述不清，智能体会将其改写为普通指令，例如将“现在需要让炉灶处于打开状态”改写为“打开炉灶”。如果请求被误听或发生了改变，智能体会写出正确的指令，例如在机器人被告知要拿奶油奶酪时，改写为“把牛奶放进篮子里”。一旦机器人已经开始执行错误的任务，仅有纠正后的指令还不够，因为策略只有从熟悉的状态出发才能执行该指令。因此，机器人会先沿着自己的运动轨迹返回到这样的状态，然后再根据新指令行动；我们将这一过程称为先撤销再重做。该通道无需训练，并且适用于任何策略。

这种反馈通道 [反馈] 通过告知策略应如何移动，来修正错误执行。它携带智能体对漂移的估计，例如“稍微向右推”。与指令通道不同，反馈必须经过学习，因为策略从未见过这种反馈；我们教策略仅从智能体提供的信息中读取反馈（图2）。采用智能体标注的 DAgger [31], 时，我们在存在漂移的训练任务上运行策略。机器人执行策略的计划，并由智能体根据其估计的漂移进行修正；这些修正后的计划便成为训练目标，因此无需知道真实漂移。已学习的角色标记使反馈与指令保持分离。反事实反馈将同一状态与不同的

反馈及不同的修正后计划配对，使策略无法根据状态猜出计划，而必须学会读取反馈。最后，只有策略的动作模块读取反馈，而其世界模块仍只读取任务。世界模块根据任务想象场景将如何展开，而在此处加入反馈会扭曲它的想象。动作模块将这一想象的未来转化为运动，因此修正应当作用于此处。

我们在三种模拟器中，使用两个 WAM 和一个视觉-语言-动作模型，以及十个专有和开放权重智能体评估 RoboThink。在 LIBERO [23], 中，我们测试未知漂移、改述、误听和变更的请求，以及受扰场景 [10, 16, 42]。在 RoboTwin 2.0 [5], 中，策略在关节空间中控制两条机械臂；在 RoboCasa [27], 中，移动操作机器人则在未见过的厨房中工作。我们的贡献有三点：

• **RoboThink**，让机器人策略与推理智能体协同工作：智能体负责思考，策略负责执行。智能体的指导通过指令通道传达给策略，该通道无需训练，即可通过先撤销再重做来修正错误目标；指导还通过反馈通道传达，以纠正错误执行。

• **一种仅凭智能体即可教会反馈通道的配方**，以及一项发现：反馈必须传递到动作模块，而非世界模块。该配方既不需要真实漂移，也不需要人工标注，只需约一百个智能体标注的回合即可奏效，并且可迁移至视觉-语言-动作模型、第二个 WAM、两条机械臂和一台移动操作机器人。

• **涵盖三种策略、三种模拟器和十个智能体的证据**。在 LIBERO 上，在恒定漂移下，成功率从 0.09 提升至 0.87；在随机漂移下，则从 0.42 提升至 0.84；对于误听或更改后的请求，回合完成率达到 0.91 至 1.00。在全部 50 个 RoboTwin 2.0 任务上，成功率分别从 0.12 提升至 0.74，以及从 0.31 提升至 0.56。RoboThink 的成功率至少不低于单独执行的最强智能体，同时每次成功所需时间为 55 秒，而非 2.1 分钟。其成功率随着智能体准确率的提高而增长，因此，一次训练的策略也会随着推理智能体的进步而提升。

2. 相关工作

面向机器人的推理智能体。 语言模型通过选择技能来为机器人制定计划 [1], , 通过根据文本反馈重新规划 [14], , 或通过编写控制代码 [22]。如今，视觉语言模型智能体通过工具调用和脚本直接充当策略 [6, 15, 41]。分层策略和双系统策略将输出语言的高层模型与低层策略配对 [28, 30, 33, 40], , 高层只能通过低层受训后能够遵循的指令与低层沟通。相比之下，RoboThink 将智能体排除在控制回路之外，并为其提供一个传达结论而非指令的通道，因此只需每隔几个动作块调用一次智能体，且无需重新训练即可将其替换。

策略内部的纠正与推理。 语言纠正可通过在纠正指令上训练来引导视觉-语言-动作模型 [2, 7, 25,32, 36] ，也可以作为运动偏移量来引导 [19], ；视觉语言模型则可解释机器人故障 [9]。推理也被融入策略训练中，例如在执行动作前生成思维链 [43] ，或在 AW-0 [26], 中结合视频和动作生成代码。与这些方法不同，RoboThink 不会将纠正指令纳入策略的指令中，也不会将推理固化在策略权重里，而是为智能体反馈提供独立通道，仅由动作模块读取，并且只需依靠一个现成的智能体即可学习该通道。

世界动作模型与鲁棒性。世界动作模型基于视频生成器构建策略，从预-测帧中解码动作 [8, 13, 35] ，或联合建模视频与动-作 [18, 20, 21, 37, 39, 45]。测试时自适应会更新其权重或计划 [11, 24, 29, 38], , 而鲁棒性基准则对动作、场景和语-言施加扰动 [10, 12, 16, 42, 44]。我们采用这些扰动，并加入误听和更改后的请求。与测试时自适应不同，RoboThink 在测试时既不改变权重，也不改变计划：策略将智能体的结论作为输入。

3. 方法

RoboThink 将机器人策略与推理智能体相结合（图 1）。策略在每一步执行动作，智能体的结论通过两个渠道传达给它： [INST] 告知要做什么（第3.3节），以及 [反馈] 告知如何移动（第 3.4节）。

3.1. 问题设定

机器人策略 π（视觉-语言-动作模型或 WAM）以动作块为单位执行。在动作块k 处，它接收观测 o_k （图像和本体感知）以及指令 ℓ，输出一个动作块 a_k，并在重新规划前执行其中的前 h个动作。现代策略由两部分组成：一个大型感知或世界模块 P，用于读取图像和指令；以及一个较小的动作模块A，它在关注P的特征以及（在大多数设计中）指令本身的同时，对动作块进行去噪：

$$a_k \sim A\big(\cdot \mid P(o_k, \ell),\, o_k,\, \ell\big). \tag{1}$$

在 Fast-WAM [39] 中， P 是一个视频扩散变换器（Wan2.2-5B），它会像想象未来那样，根据提示词对当前帧进行编码； A 则是一个拥有独立提示词交叉注意力的 1B 动作专家。在 GR00T [28] 中， P是视觉语言骨干网络， A 是一个流匹配动作头。在部署时，执行或任务都可能发生变化。可能会对每条动作指令加入漂移 δ，即策略无法获知的固定偏移；机器人收到的指令 ℓ 也可能不同于用户请求 $\ell^\star$，因为指令可能经过改写、被误听或发生变化。

3.2. 推理智能体

策略无法判断失败原因，因此由一个推理智能体 $\mathcal{C}$——例如冻结的视觉语言模型 Claude Opus 5.5——对其进行观察。在执行完第 1、2、3 个动作块后，以及此后每执行四个动作块，智能体都会查看 $\ell^{\star}$：最后几帧画面，以及按动作块列出的指令末端执行器运动与实际末端执行器运动对照表，并以 JSON 格式作答（第A节）。回答会说明各平移轴和旋转轴上漂移的方向及大致幅度、机器人是否正在执行所请求的任务；如果请求是间接的，或机器人在执行其他操作，还会给出明确指令。$\hat{}$漂移部分会提供数值估计 δ，并通过固定模板生成反馈文本 r （“调整：适度向后推.....”），训练集和测试集使用互不重叠的措辞。一次调用耗时约 8 秒（Opus 的中位数）。在我们的运行中，机器人会在调用期间保持静止，因此报告的所有时间都包括这段时间；两次调用之间，策略则依据当前生效的结论采取动作。智能体可以互换。我们默认使用 Claude Opus 5.5，也使用 GPT-6 Astra、Claude Sonnet 5.5、GPT-6.1 Sol 和开放权重模型 Kimi K3（表5）。

3.3. 指令通道与先撤销再重做

当任务有误时，智能体会写入一条新指令，用它替换 ℓ 两个模块中的原指令。策略会从初始状态开始执行普通指令，但不会从错误计划将其带到的位置开始执行，例如手中拿着错误的物体。因此，RoboThink 会先撤销再重做。当智能体报告机器人正在执行另一项任务时，以及首次修正一条并非用户请求的指令时（可能是误听，也可能是用户后来更改的），机器人会按相反顺序、以相反方向执行自己过去的命令，并保持夹爪张开，然后根据新指令行动。若只是改述用户自己的请求，则会保留原有运动。先撤销再重做只使用机器人的命令历史，因此无需训练，并且适用于任何策略。

3.4. 反馈通道

关于如何移动的结论不应出现在指令中，而在其提示词中读取这些结论的策略，无论这些结论说什么，都会以相同的方式移动（图 7）。RoboThink 为这些内容提供了专属通道。动作模块将指令和反馈作为两个独立编码的片段进行读取，这些片段位于已学习的角色标记之后；而感知模块或世界模块则保留普通指令：

$$\begin{aligned} a_k &\sim A_\theta\big(\cdot \mid P(o_k, \ell),\, o_k,\, c_k\big), \\ c_k &= \big[\texttt{[INST]}, E(\ell), \texttt{[FB]}, E(\text{“Feedback: ”} r_k)\big], \end{aligned} \tag{2}$$

其中 E 是策略冻结的文本编码器，“反馈：无”表示无反馈。避免反馈进入世界模块至关重要：在 WAM 中，该模块会根据普通任务想象场景，而反馈会扭曲它的想象（第 4.3节）。

智能体标注的 DAgger。 智能体就是教师（图 2）。在训练任务中，面对训练分布的漂移，每个执行的动作块都是策略针对普通指令所制定的计划，并由智能体当前的估计进行修正，$\hat{}a'_k = a_k - \delta_k$。我们只微调 A 以及 $(o_k、\ \ell、\ r_k、\ a'_k)$ 中的角色标记，并使用策略自身的损失。

反事实反馈。 在这些轨迹展开中，每个状态都携带智能体碰巧作出的那一个估计，因此策略可以仅根据状态拟合目标值。反事实反馈打破了这种相关性。以概率$p_{\text{cf}} = 0.5$，将一个片段的反馈替换为训练分布中$\tilde{}$某个漂移 δ 的渲染结果，并将其目标值替换为$a - \delta$；以概率 $p_{\varnothing} = 0.1$ 替换为“反馈：无”。并且 a。这样，同一个状态会呈现不同的反馈和不同的目标值，只有读取反馈才能拟合这些目标值。真实漂移从未被使用。通过这种方式，策略学会按照反馈所指示的方向运动，并针对相反的反馈采取另一种方式（图 7）。只有动作模块和角色标记参与训练，而由一个智能体教授的权重也能与其他智能体配合使用（表 5）。

4. 实验

我们设计这些实验来回答五个问题。（Q1）推理智能体应当自行执行动作，还是与策略一起思考（第 4.1节）？（Q2）对于不同的智能体、策略和机器人，策略是否会根据智能体关于运动的反馈采取动作（第 4.2节）？（Q3）是什么使反馈通道发挥作用（第 4.3节）？（Q4）智能体是否也能纠正机器人的行为（第 4.4节）？（Q5）RoboThink 能否处理从未训练过的偏移（第 4.5节）？

设置。 除非另有说明，推理智能体为 Claude Opus 5.5（Opus）和 GPT-6 Astra（Astra）。在 Fast-WAM 上，RoboThink 使用 Opus 作为智能体，通过 840 个 DAgger 回合的反馈训练，在 LIBERO-Spatial、-Object 和 -Goal 的任务 0–6 上进行训练 [23], ，并在每个套件的任务 7–9 上进行测试。这九项任务未参与任何反馈训练；每项任务均从 LIBERO 的 50 个初始状态中前 15 个开始，成本较高的对比实验则使用前 5 个（各表中的回合数见第 B 节）。基础策略使用所有 LIBERO 任务的示范数据训练，因此这些任务测试的是智能体结论的运用，而非技能本身。恒定漂移 会为每个动作维度 $+0.1$ 加上一个偏移量，而随机漂移 则在每个回合中从 $U(-0.1,\ 0.1)$[7]中抽取一个偏移量。两种漂移在整个回合中都保持不变。我们报告成功率，即成功回合所占的比例（越高越好）；例如，0.87 / 0.84 表示恒定漂移和随机漂移下的结果。所有方法运行完全相同的回合。特权规则 会从未经训练的策略所输出的动作中减去智能体的估计值。该规则需要知道动作空间和故障形式，并作为参照。



图 3. **LIBERO 上的定性示例**（前置摄像头）。（a）在恒定漂移的影响下，仅策略会偏离瓶子。使用 RoboThink 时，智能体在第一个动作块之后，根据指令动作与实际运动推断出漂移；策略读取反馈后，抓取果汁并将其放好（共调用智能体五次，耗时 65 秒）。（b）只有智能体知道用户请求。机器人按照指示递送奶油奶酪。使用 RoboThink 时，智能体改写指令，机器人撤销之前的两个动作块并取来牛奶。

表 1. **仅智能体、仅策略，以及两者共同作用时**在相同回合上的表现。时间：成功所需实际耗时的中位数，包含智能体调用时间。词元：每回合的输入词元数，包含缓存上下文。RoboThink 的成功率至少与仅智能体相同，所需时间和词元却只有其一小部分。

	常数	随机	时间	词元
LIBERO				
仅策略（Fast-WAM）	0.09	0.42	11 s	0
仅智能体（GPT-6 Astra）	0.78	0.84	2.1 分钟	1.2 M
仅智能体（Claude Opus 5.5）	0.53	0.58	2.3 分钟	759 k
RoboThink（Claude Opus 5.5）	**0.87**	**0.84**	55 s	8 k
RoboTwin 2.0				
仅策略（Fast-WAM）	0.07	0.30	18 s	0
仅智能体（GPT-6 Astra）	0.45	0.36	4.2 分钟	2.8 M
仅智能体（Claude Opus 5.5）	0.30	0.20	3.7 分钟	1.1 M
RoboThink（Claude Opus 5.5）	**0.73**	**0.48**	33 s	10 k

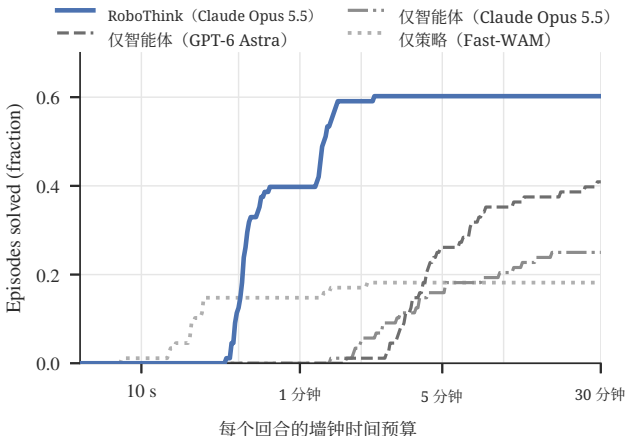


图4。 **双机械臂在时间预算内的成功率。** 在表 1 的回合中，RoboTwin 2.0 在漂移情况下于实际时间预算内完成的回合占比。两分钟内，RoboThink 完成的回合数超过智能体单独在三十分钟内完成的数量。

4.1. 智能体应该执行动作，还是协同思考？

智能体单独行动。 作为策略的智能体 [6,15] 通过代码控制机器人。我们为 GPT-6 Astra 和 Claude Opus 5.5 提供了这样一套控制框架：一个能以闭环方式控制机械手到达三维位置的机器人 API、带有深度信息且经过校准的摄像头，以及与 RoboThink 智能体收到的相同警告，即指令可能发生漂移（第 D 节）。在完全相同的 LIBERO 回合中（表 1），Opus 单独行动时，在存在漂移的情况下成功率为 0.53 / 0.58。与策略协同思考后，同一模型的成功率达到 0.87 / 0.84。我们测试过的最强智能体 Astra 单独行动时成功率为 0.78 / 0.84，与 RoboThink 接近。差异在于每次成功所需的成本。智能体完成一个成功回合的中位用时分别为 2.1 分钟和 2.3 分钟，每回合约消耗一百万词元，其中大部分是每一步都会再次读取的上下文。RoboThink 耗时 55 秒，使用 8 千词元；大部分时间花在智能体调用上（每次 8.2 秒，期间机器人处于等待状态）。使用 Opus 时，智能体单独行动每回合的成本为 0.52 美元，而使用 RoboThink 则为 0.13 美元。

在时间预算内的成功率。 时间有限时，差距最为明显（图 1，右）。每回合限时一分钟时，智能体单独只能完成 0.01 和 0.00 的漂移回合，而 RoboThink 能完成 0.53。限时两分钟时，三者分别能完成 0.37、0.22 和 0.77。只有限时五分钟时，Astra 才接近 RoboThink（0.81 对 0.86）。在 RoboTwin 的双机械臂任务中，差距更大（图 4）。单独执行动作时，Astra 和 Opus 分别在数分钟后完成 0.45 / 0.36 和 0.30 / 0.20 的回合，而 RoboThink 在 33 秒的中位时间内完成 0.73 / 0.48。通过策略执行动作，智能体便能减少思考频率。智能体仅在第 1、2、3 个动作块之后，以及此后每四个动作块之后调用；两次调用之间，策略以自身速率执行动作。因此，智能体应该协同思考，而不是亲自执行动作。

4.2. 根据关于运动的反馈采取行动

主要结果。 将智能体反馈附加到未经训练的策略指令中会适得其反（表 2）：策略对文本的出现作出反应，而非理解其含义（图7a）。之后

表2. **关于运动的反馈**。 在 LIBERO 上使用 Fast-WAM 的成功率，最佳结果以粗体显示。未经训练的策略会受到反馈的负面影响，而 RoboThink 会根据反馈采取行动，表现接近特权规则。

	常数	随机	无
Fast-WAM	0.09	0.42	**0.96**
Fast-WAM + 提示词中的反馈	0.00	0.16	0.84
特权规则	0.84	0.93	0.93
RoboThink（Claude Opus 5.5）	**0.87**	**0.84**	0.89

表3. **其他策略与机器人**，各自在配方前后，配合实时 Claude Opus 5.5。RoboTwin 2.0：双机械臂、关节漂移、反馈训练之外的任务。RoboCasa：未见过的厨房中的移动操作机器人。该配方适用于一个视觉-语言-动作模型、两个世界动作模型和三台机器人。

	常数	随机	无
LIBERO			
GR00T N1.7	0.02	0.24	0.98
RoboThink（GR00T N1.7）	**0.53**	**0.70**	0.89
RoboTwin 2.0			
Fast-WAM	0.07	0.30	**0.95**
RoboThink（Fast-WAM）	**0.73**	**0.48**	**0.95**
RoboCasa			
Cosmos Policy	0.00	0.03	0.50
RoboThink（Cosmos Policy）	**0.25**	**0.30**	0.49

表4。**RoboTwin 2.0 在其官方协议下的**：全部50项任务的成功率，取各任务的平均值。最后一行：RoboThink 优于或逊于 Fast-WAM 的任务数量。AW-0：数据摘自其论文，该论文报告无漂移（漂移情况下的 AW-0：第E节）。几乎所有任务的增益均得以保持。

	干净场景			随机化场景		
	无	常数	随机	无	常数	随机
Fast-WAM	**0.91**	0.12	0.31	**0.91**	0.12	0.30
AW-0	0.81	未报告		0.21	未报告	
RoboThink	**0.91**	**0.74**	**0.56**	**0.91**	**0.75**	**0.57**
任务表现更好 / 更差	0 / 0	46 / 1	44 / 1	11 / 4	47 / 0	43 / 4

经过训练后，同样的反馈将成功率从 0.09 / 0.42 提升至 0.87 / 0.84（图 3a），接近特权规则（0.84 / 0.93）的表现。特权规则可直接应用这些估计值，但必须了解动作空间。没有漂移时，代价为 0.07；此时智能体有时会产生误报。

任何能力出众的智能体。 策略由 Opus 教导，搭配 GPT-6 Astra 同样表现出色。训练和测试时都使用开放权重的 Kimi K3 作为智能体，整个系统便是开放权重的，成绩达到 0.82/0.82。六种智能体每次调用的成本从 $0.001 到 $0.027 不等，带来的增益相近（表 5），其中包括一种只读取运动表的纯文本模型（第 A节）；对于所有智能体，在发生漂移时，成功率都在 0.76 到 0.91 之间。在十种智能体中，发生漂移时的成功率与其估计准确率相关

表 5. **任何有能力的智能体都能胜任**。 RoboThink（由 Claude Opus 5.5 教授）在各智能体实时参与的情况下，面对策略漂移时的成功率；对比对象是仅策略，使用相同的 LIBERO 回合。: 开放权重，其余均为专有模型。仅文本：不具备视觉能力、仅用于反馈通道的语言模型，它读取 com-

指令运动与实际运动。 Δ：相比仅策略的增益。每个智能体都将成功率提升至约 0.8。

智能体	$/调用	常数	Δ	随机	Δ
无（仅 Fast-WAM）	0	0.09		0.42	
Claude Opus 5.5	0.022	0.87	+0.78	0.84	+0.42
Claude Sonnet 5.5	0.011	0.76	+0.67	0.80	+0.38
GPT-6 Astra	0.027	**0.91**	+0.82	0.80	+0.38
GPT-6.1 Sol	0.006	0.82	+0.73	0.84	+0.42
Kimi K3 (	0.009	0.80	+0.71	**0.89**	+0.47
gpt-oss-120b , 仅文本）	0.001	0.78	+0.69	0.76	+0.34

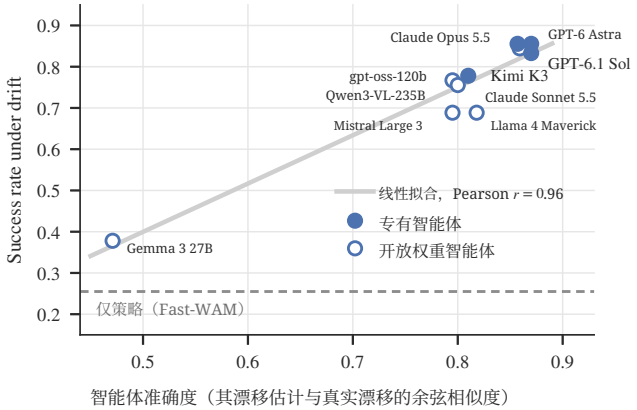


图 5. **智能体越强，机器人就越出色**。 RoboThink（由 Claude Opus 5.5 教授）在 LIBERO 上搭配十个推理智能体：漂移条件下的成功率（两种漂移的平均值）与智能体准确率之间的关系；对每个智能体，都基于相同的已保存调用进行测量。虚线：仅策略。每个智能体都有帮助，而且成功率会随着智能体准确率的提高而上升（$r = 0.96$）。

的智能体（Pearson $r = 0.96$，图 5）。因此，更强大的推理智能体无需重新训练，就能让同一策略变得更强。

其他策略与机器人。该配方还训练了视觉-语言-动作模型 GR00T N1.7，使其成功率从 0.02 / 0.24 提升至 0.53 / 0.70（表 3）。在 RoboTwin 2.0 [5], Fast-WAM 通过绝对关节目标值控制两条六关节机械臂，漂移会重置每个关节的零点，而反馈会点明关节（“左臂：肘部适度下移”）。该策略在 36 项任务上接受训练，并在另外 11 项任务中依据这些反馈执行动作，成功率从 0.07 / 0.30 提升至 0.73 / 0.48，且无漂移时能保留其技能（0.95）。在 AW-0 结果所采用的官方 RoboTwin 协议设置下，这一增益在全部 50 项任务中都能保持，无漂移时也没有代价（0.91，表 4）。RoboThink 在其中 43 至 47 项任务上的表现优于 Fast-WAM，表现较差的任务最多为 4 项；在未纳入反馈训练的 14 项任务中，它在干净场景下的成功率从 0.08 / 0.29 提升至 0.69 / 0.44。AW-0，它

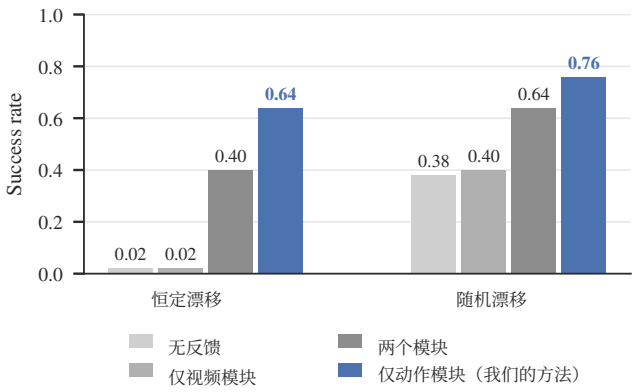


图6。**反馈属于动作模块**。在 LIBERO 上使用完美反馈时的成功率，由同一策略的不同模块读取；该策略通过来自 Claude Opus 5.5 的智能体标注的 DAgger 进行训练。由视频模块读取时，反馈并无帮助。仅由动作模块读取时，增益最大。

表6。**构建反馈通道**。在 LIBERO 上的成功率，由 Claude Opus 5.5 进行教学并运行。最后一行以上的所有行均使用相同的智能体标注的回合进行训练，数量为 RoboThink 的八分之一。DAgger 行：除非另有说明，仅动作模块读取反馈。每一项要素都会增加增益。

	常数	随机
无训练，反馈位于提示词中	0.00	0.16
不含校正后的目标的 DAgger	0.00	0.22
不含反馈文本的 DAgger	0.00	0.22
DAgger 中，两个模块都会读取反馈	0.02	0.42
DAgger 中，只有动作模块会读取反馈	0.22	0.53
+角色标记	0.47	0.56
+ 反事实反馈	0.82	0.80
RoboThink 中，所有由智能体标注的回合	**0.87**	**0.84**

会根据其动作生成代码进行推理，但在漂移情况下会像 Fast-WAM 一样失效，即使提示词中包含智能体反馈也是如此（第E节）。在 RoboCasa [27], 中，面对未见过的厨房里的移动操作机器人，使用该配方并针对自身动作损失进行训练的 Cosmos Policy，成功率从 0.00 / 0.03 提升至 0.25 / 0.30（特权规则为 0.29 / 0.34，见第 F节）。

4.3. 反馈通道为何有效

策略会遵循反馈。 增益来自策略，而不仅仅来自智能体。智能体在一个回合中的初步估计与真实漂移的方向余弦仅为 0.72 到 0.82。在第一个动作块之前提供描述漂移的反馈（图 7 ）时，未经训练的策略即使面对方向相反的反馈也会朝同一方向移动；仅使用 DAgger 的策略只会部分遵循方向（余弦相似度为 0.58），而 RoboThink 会逐轴遵循方向（余弦相似度为 0.95，增益为 0.90），并在反馈方向相反时反向移动。

每个要素都很重要。我们使用相同的智能体标注回合，一次添加一个要素来构建反馈通道（表 6）。DAgger 需要反馈文本和

表 7. **该怎么做**。 成功率。上：LIBERO-Para [16] 的改写后的请求。下，无漂移：告知机器人另一个任务或物体，只有智能体知道请求内容（误听）；或者用户在三个动作块后进行纠正（已更改）。Para.：按 LIBERO-Para 中的方式改写。无智能体：使用不含智能体的相同策略。[INST], 采用先撤销再重做的方式，无需训练，因此也能直接使用 Cosmos Policy。

改写后的请求	无漂移	常数	随机
无智能体	0.51	0.01	0.14
RoboThink（[反馈]）	0.57	0.56	0.49
RoboThink（[反馈]+[INST]）	**0.73**	**0.81**	**0.72**

		误听		已更改	
无漂移	参数。	Task	物体	Task	物体
采用 *RoboThink* 配方的 *Fast-WAM*					
无代理 0.51 0.00			0.53	0.07	0.71
RoboThink([INST]) **0.7**			**0.98**	**0.91**	**1.00**
现成的 *Cosmos Policy*					
无代理	0.76	0.00	0.44	0.20	0.47
RoboThink（[INST]）	**0.90**	**1.00**	**0.98**	**1.00**	**1.00**

校正后的目标值，而且反馈只能传递给动作模块。由两个模块共同读取时，其表现不如单独使用策略（0.02 / 0.42）；在权重相同的情况下，仅由动作模块读取，表现则达到 0.22 / 0.53。使用完美反馈时，这一模式更加明显（图 6）：仅动作模块读取时为 0.64 / 0.76，仅视频模块读取时为 0.02 / 0.40。修正由动作模块生成，而反馈会干扰世界模型，使其键和值发生的变化幅度达到另一项任务指令所致变化的三分之二（第 G节）。角色标记使反馈成为独立片段（0.47 / 0.56 ），而反事实反馈则迫使策略读取反馈：成功率升至 0.82 / 0.80，接近使用八倍数量回合所达到的 0.87 / 0.84；图 7中的方向余弦也从 0.69 升至 0.91。因此，必须在动作模块中教会策略读取反馈。

4.4. 纠正机器人的行为

改写后的请求。 LIBERO-Para 会从词汇、结构和语用层面改写指令（例如，把“打开炉灶”改为“炉灶现在需要打开”）。释义改写会使策略在无漂移的情况下损失一半回合（0.51）；关于运动的反馈可以纠正漂移，却无法解决误解问题（表7，上方）。如果智能体在同一次调用中还将间接请求改写为直接命令，则两种问题都能得到纠正。无漂移、有恒定漂移和有随机漂移时的成功率分别达到 0.73、0.81 和 0.72，而仅有反馈时分别为 0.57、0.56 和 0.49。在没有任何训练的情况下， [INST]也能将 Cosmos Policy 的成功率从 0.76 提升至 0.90。

更改了任务和物体。机器人开始执行另一项任务后，仅有正确的指令并不足够（图7，下方）。经过三个动作块后，训练好的策略在 0.07 的回合中完成了更正后的任务，而 Cosmos Policy

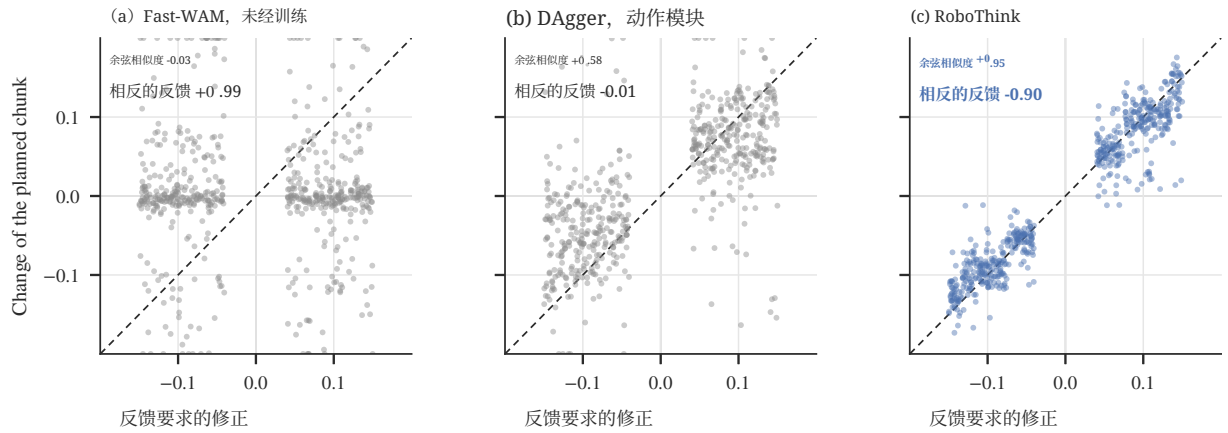


图7. **策略会按反馈所说的去做吗?** 加入描述漂移的反馈后，第一个规划片段的变化（每种情况和手臂轴各一个点）。(a) Fast-WAM 会对文本的出现作出反应，而不是理解其内容；方向相反的反馈也会让它朝同一方向移动。(b) 单独使用 DAgger 只能部分遵循方向。(c) RoboThink 会遵循反馈（余弦相似度 0.95，增益 0.90），并在收到相反反馈时朝另一方向移动。

表8. **变化的偏移、负载和长时任务**。LIBERO 上的成功率。在回合期间，恒定漂移会反向，或者随机漂移会出现或重新抽取。负载：手持物体时施加的向下的力。智能体：漂移发生变化时使用 Claude Opus 5.5，其他情况使用 GPT-6 Astra。智能体会纠正每一次偏移。

	无代理	有智能体
漂移反向	0.09	**0.71**
漂移出现	0.74	**0.84**
漂移重新抽取	0.39	**0.72**
已抓取物体上的负载	0.41	**0.59**
LIBERO-10 ，恒定漂移	0.00	**0.53**
LIBERO-10 ，随机漂移	0.03	**0.54**

这一比例为 0.20。先撤销再重做可将这两个比例分别提高到 0.91 和 1.00，因为回溯其指令会使机器人回到策略已知晓新任务的状态。物体的情况也相同（“要牛奶，不要黄油”）：不撤销时，两个策略分别在 0.71 和 0.47 的回合中取回指定的杂货；撤销后则每个回合都能完成。

误听的请求。 当只有智能体知道用户请求时，它必须察觉到错误。随后，它会改写错误的指令，并让机器人撤销已执行的动作块（图 3b）。这样，请求的任务便能在 0.96 和 1.00 的回合中完成，而不再是一次也无法完成；两种策略都能在 0.98 的回合中选中请求的物体。在 RoboTwin 中，机器人被告知的两个物体顺序颠倒，成功率从 0.52 升至 0.88。等到智能体报告机器人偏离任务后才撤销，为时已晚（0.56 和 0.76，见第 C节）。 [INST] 因此无需训练；先撤销再重做，即使机器人已经开始执行，也能让它奏效。

4.5. 训练中从未出现的变化

类型不同或发生变化的偏移。反馈通道是在回合期间保持恒定的漂移上训练的。它也能处理其他类型的偏移，因为智能体会描述它所看到的情况

表 9. **与训练场景不同或在回合中发生变化的场景**，随机漂移下的成功率。LIBERO-Plus [10]：测试任务的视觉扰动变体。LIBERO-MAX [42] Lite：回合中发生的一次事件。智能体：使用 [反馈] + [INST] 的 Claude Opus 5.5。

	仅策略	RoboThink
LIBERO-Plus	0.22	**0.58**
LIBERO-MAX	0.04	**0.20**

（表8）。当漂移反转、出现或在回合期间被重新设定时，智能体会在下次调用时报告新的漂移，策略随之调整，使成功率从 0.09 提升至 0.71、从 0.74 提升至 0.84，以及从 0.39 提升至 0.72。针对反转，只需在智能体的提示词中加入一句说明漂移可能发生变化的话语；在没有漂移时也不会产生任何代价（第 H节）。仅在持有物体时出现的向下的力也能得到纠正（成功率从 0.41 提升至 0.59，与特权规则的 0.56 相当）。在 LIBERO-10 的十项长时任务中（这些任务均未包含在反馈训练中），漂移条件下的成功率从 0.00 /0.03 提升至 0.53 / 0.54。

与训练场景不同的场景。反馈通道和智能体也能适用于受扰场景（表9）。在 LIBERO-Plus 中，面对新摄像头、灯光和布局等七类扰动，实时 Opus 将随机漂移下的成功率从 0.22 提升至 0.58。LIBERO-MAX 的场景会在回合中发生变化，表现也类似：智能体在事件过程中修正运动，成功率从 0.04 提升至 0.20。由于智能体会描述它所看到的一切，RoboThink 能修正训练中从未遇到过的偏移。

5. 结论

机器人策略能实时执行动作，却无法判断失败的原因；推理智能体能找出问题所在，却行动过慢，无法控制机器人。我们提出了 RoboThink，

在该方法中，智能体负责思考，策略则学会根据其指导采取动作。指令通道用于修正错误目标：智能体重写指令，机器人先撤销之前的运动，全程无需训练。反馈通道用于修正错误执行：它仅通过智能体自身的指导进行学习，且只有动作模块能读取，因此策略可以遵循“向右推，稍微一点”之类无法通过指令表达的纠正指令。在三种策略、三个模拟器和十个智能体上，Robo- Think 将漂移和误解请求导致的失败转化为成功，所需时间和词元仅为智能体独自行动时的一小部分。将智能体作为即插即用组件后，同一策略会随着智能体能力提升而改进，无需重新训练。

大语言模型使用声明

我们使用 ChatGPT 和 Claude 协助润色语言、提升本手稿的可读性，并为手稿提供反馈。作者对本文中的所有研究思路、实验结果和主张负全部责任。

参考文献

[1] Michael Ahn、Anthony Brohan、Noah Brown、Yevgen Chebotar、Omar Cortes、Byron David、Chelsea Finn 等人。《尽我所能，而非照我所说：以机器人可供性为基础的语言》。发表于 机器人学习会议（*CoRL*），2022年。 2，3[2] Suneel Belkhale 等人。《RT-H：基于语言的动作层级》。发表于 机器人学：科学与系统（*RSS*），2024年。arXiv:2403.01823。 3[3] Kevin Black、Noah Brown、Danny Driess、Adnan Esmail、Michael Equi、Chelsea Finn、Niccolo Fusai、Lachy Groom、Karol Hausman、Brian Ichter、Szymon Jakubczak、Tim Jones、Liyiming Ke、Sergey Levine、Adrian Li-Bell、Mohith Mothukuri、Suraj Nair、Karl Pertsch、Lucy Xiaoyang Shi、James Tanner、Quan Vuong、Anna Walling、Haohuan Wang 和 Ury Zhilinsky。 π_0：用于通用机器人控制的视觉-语言-动作流模型。*arXiv* 预印本 *arXiv:2410.24164*，2024年。 1[4] Anthony Brohan、Noah Brown、Justice Carbajal、Yevgen Chebotar、Xi Chen、Krzysztof Choromanski、Tianli Ding、Danny Driess、Avinava Dubey、Chelsea Finn、Pete Florence、Chuyuan Fu、Montse Gonzalez Arenas、Keerthana Gopalakrishnan、Kehang Han、Karol Hausman、Alexander Herzog、Jasmine Hsu、Brian Ichter、Alex Irpan、Nikhil Joshi、Ryan Julian、Dmitry Kalashnikov、Yuheng Kuang、Isabel Leal、Lisa Lee、Tsang-Wei Edward Lee、Sergey Levine、Yao Lu、Henryk Michalewski、Igor Mordatch、Karl Pertsch、Kanishka Rao、Krista Reymann、Michael Ryoo、Grecia Salazar、Pannag Sanketi、Pierre Sermanet、Jaspiar Singh、Anikait Singh、Radu Soricut、Huong Tran、Vincent Vanhoucke、Quan Vuong、Ayzaan Wahid、Stefan Welker、Paul Wohlhart、Jialin Wu、Fei Xia、Ted Xiao、Peng Xu、Sichun Xu、Tianhe Yu 和 Brianna Zitkovich。《RT-2：视觉-语言-动作模型将网络知识迁移至机器人控制》。 *arXiv* 预印本 *arXiv:2307.15818*，2023年。CoRL 2023。 1

[5] Tianxing Chen 等。《RoboTwin 2.0：一种具备强领域随机化能力、用于鲁棒双臂机器人操作的可扩展数据生成器与基准测试》。arXiv 预印本 arXiv:2506.18088，2025年。3，6[6] Yanzhe Chen、Zechen Bai、Zhijun Cao、Wenzheng Zeng、Kevin Qinghong Lin、Yiqi Lin、Guoqiang Liang、Kevin Yuchen Ma、Qiming Huang 和 Mike Zheng Shou。《Show-Harness：仅凭视觉语言模型智能体就能操作机器人》。arXiv 预印本 arXiv:2609.10522，2026年。2，3，5[7] Yinpei Dai 等。《RACER：用于模仿学习的丰富语言引导失败恢复策略》。arXiv 预印本 arXiv:2409.14674，2024年。3[8] Yilun Du、Mengjiao Yang、Bo Dai、Hanjun Dai、Ofir Nachum、Joshua B. Tenenbaum、Dale Schuurmans 和 Pieter Abbeel。《通过文本引导的视频生成学习通用策略》。arXiv 预印本 arXiv:2302.00111，2023年。NeurIPS 2023。1，3[9] Jiafei Duan 等。《AHA：用于检测机器人操作中的失败并进行推理的视觉语言模型》。载于国际学习表征会议（ICLR），2025年。arXiv:2410.00371。2，3[10] Senyu Fei、Siyin Wang、Junhao Shi、Zihao Dai、Jikun Cai、Pengfang Qian、Li Ji、Xinzhe He、Shiduo Zhang、Zhaoye Fei、Jinlan Fu、Jingjing Gong 和 Xipeng Qiu。《LIBERO-Plus：视觉-语言-动作模型的深入鲁棒性分析》。arXiv 预印本 arXiv:2510.13626，2025年。3，8，1[11] Yusen Feng、Bingchen Han、Jiangran Lyu、Kai Liu、Yixin Zheng、Yuxuan Wan、Weiheng Liu、Sun Han、Ruiqin Li、Yulong Zhang、Fangfu Liu、Xuesong Shi、Libin Liu、Yizhou Wang、Zhizheng Zhang 和 He Wang。《WAM-TTT：在测试时通过观察人类操作来引导世界-动作模型》。arXiv 预印本 arXiv:2607.06988，2026年。3[12] Jianing Guo、Zhenhong Wu、Chang Tu、Yiyao Ma、Xiangqi Kong、Zhiqian Liu、Jiaming Ji、Shuning Zhang、Yuanpei Chen、Kai Chen、Qi Dou、Yaodong Yang、Xianglong Liu、Huijie Zhao、Weifeng Lv 和 Simin Li。《RobustVLA：视觉-语言-动作模型对多模态扰动的鲁棒性》。arXiv 预印本 arXiv:2510.00037，2025年。3[13] Yucheng Hu、Yanjiang Guo、Pengchao Wang、Xiaoyu Chen、Yen-Jen Wang、Jianke Zhang、Koushil Sreenath、Chaochao Lu 和 Jianyu Chen。《视频预测策略：一种具有预测性视觉表征的通用机器人策略》。arXiv 预印本 arXiv:2412.14803，2024年。3[14] Wenlong Huang、Fei Xia、Ted Xiao、Harris Chan、Jacky Liang、Pete Florence、Andy Zeng、Jonathan Tompson、Igor Mordatch、Yevgen Chebotar 等。《内心独白：通过语言模型规划实现具身推理》。载于机器人学习会议（CoRL），2022年。2，3[15] Mengzhao Jia、Yang Lin、Xixin Zhang、Zhihan Zhang、Xiaobai Liu 和 Meng Jiang。《智能体即策略：用于机器人操作》。arXiv 预印本 arXiv:2609.12541，2026年。2，3，5[16] Chanyoung Kim、Minwoo Kim、Minseok Kang、Hyunwoo Kim 和 Dahuin Jung。《LIBERO-Para：用于衡量视觉-语言-动作模型释义鲁棒性的诊断基准测试与指标》。arXiv 预印本 arXiv:2603.28301，2026年。3，7，1[17] Moo Jin Kim、Karl Pertsch、Siddharth Karamcheti、Ted Xiao、Ashwin Balakrishna、Suraj Nair、Rafael Rafailov、Ethan Foster、Grace Lam、Pannag Sanketi、Quan Vuong、Thomas Kollar,

Benjamin Burchfiel、Russ Tedrake、Dorsa Sadigh、Sergey Levine、Percy Liang 和 Chelsea Finn。《OpenVLA：一种开源视觉-语言-动作模型》。arXiv 预印本 arXiv:2406.09246，2024年。CoRL 2024。1[18] Moo Jin Kim、Yihuai Gao、Tsung-Yi Lin、Yen-Chen Lin、Yunhao Ge、Grace Lam、Percy Liang、Shuran Song、Ming-Yu Liu、Chelsea Finn 和 Jinwei Gu。《Cosmos Policy：微调视频模型以实现视觉运动控制与规划》。arXiv 预印本 arXiv:2601.16163，2026年。1，3，2 [19] Owen Kwon、Pablo Ortega-Kral、Arthur Bucker 和 Jean Oh。《通过语言反馈对视觉-语言-动作模型失效进行免训练动作纠正》。arXiv 预印本 arXiv:2608.29967，2026年。3[20] Lin Li、Qihang Zhang、Yiming Luo、Shuai Yang、Ruilin Wang、Fei Han、Mingrui Yu、Zelin Gao、Nan Xue、Xing Zhu、Yujun Shen 和 Yinghao Xu。《用于机器人控制的因果世界建模》。arXiv 预印本 arXiv:2601.21998，2026年。3[21] Shuang Li、Yihuai Gao、Dorsa Sadigh 和 Shuran Song。《统一视频动作模型》。arXiv 预印本 arXiv:2503 00200，2025年 1，3 。 .[22] Jacky Liang、Wenlong Huang、Fei Xia、Peng Xu、Karol Hausman、Brian Ichter、Pete Florence 和 Andy Zeng。《代码即策略：用于具身控制的语言模型程序》。载于IEEE国际机器人与自动化会议（ICRA），2023年。2，3[23] Bo Liu、Yifeng Zhu、Chongkai Gao、Yihao Feng、Qiang Liu、Yuke Zhu 和 Peter Stone。《LIBERO：终身机器人学习知识迁移基准测试》。arXiv 预印本 arXiv:2306.03310，2023年。NeurIPS 2023 数据集与基准测试。3，4[24] Changyu Liu、Yiyang Liu、Taowen Wang、Qiao Zhuang、James Chenhao Liang、Wenhao Yang、Renjing Xu、Qifan Wang、Dongfang Liu 和 Cheng Han。《通过测试时强化学习进行即时VLA适应》。arXiv 预印本 arXiv:2601.06748，2026年。3[25] Huihan Liu 等。《通过语言纠正实现交互式机器人学习》。arXiv 预印本 arXiv:2310.17555，2023年。3[26] Zhuoyang Liu、Dantong Niu、Botong Zhang、Boning Shao、Jing Wang、Ruijie Zheng、Yuvan Sharma、Runyu Lu、Jiaming Liu、Renrui Zhang、Roei Herzig、Yutong Bai、Ken Goldberg、Danfei Xu、Yuke Zhu、Guanzhi Wang、Linxi Fan 和 Trevor Darrell。《智能体世界动作模型》。arXiv 预印本 arXiv:2610.30136，2026年。2，3[27] Soroush Nasiriany、Abhiram Maddukuri、Lance Zhang、Adeet Parikh、Aaron Lo、Abhishek Joshi、Ajay Mandlekar 和 Yuke Zhu。《RoboCasa：面向通用型机器人的日常任务大规模仿真》。载于机器人学：科学与系统，2024年。3，7[28] 英伟达。《GR00T N1：面向通用型人形机器人的开放基础模型》。arXiv 预印本 arXiv:2503 14734，2025 1，3 。 .[29] Sangwu Park、Wonjoong Kim、Yeonjun In、Sein Kim、Hongseok Kang 和 Chanyoung Park。《用于视觉预见的视觉-语言-动作模型测试时训练》。arXiv 预印本 arXiv:2605.08215，2026年。3[30] Physical Intelligence、Kevin Black、Noah Brown、James Darpinian、Karan Dhabalia、Danny Driess、Adnan Esmail，

Michael Equi、Chelsea Finn、Niccolo Fusai、Manuel Y. Galliker、Dibya Ghosh、Lachy Groom、Karol Hausman、Brian Ichter、Szymon Jakubczak、Tim Jones、Liyiming Ke、Devin LeBlanc、Sergey Levine、Adrian Li-Bell、Mohith Mothukuri、Suraj Nair、Karl Pertsch、Allen Z. Ren、Lucy Xiaoyang Shi、Laura Smith、Jost Tobias Springenberg、Kyle Stachowicz、James Tanner、Quan Vuong、Homer Walke、Anna Walling、Haohuan Wang、Lili Yu 和 Ury Zhilinsky。 $\pi_{0.5}$：一个具备开放世界泛化能力的视觉-语言-动作模型。*arXiv* 预印本 *arXiv:2504.16054*，2025。1，3[31] St´ephane Ross、Geoffrey J. Gordon 和 J. Andrew Bagnell。《将模仿学习和结构化预测归约为无悔在线学习》。载于 第十四届人工智能与统计国际会议（*AISTATS*）论文集，2011。arXiv:1011.0686。2[32] Lucy Xiaoyang Shi、Zheyuan Hu、Tony Z. Zhao、Archit Sharma、Karl Pertsch、Jianlan Luo、Sergey Levine 和 Chelsea Finn。《对你的机器人喊话：通过语言纠正指令实现即时改进》。载于 机器人：科学与系统（*RSS*），2024。arXiv:2403.12910。3[33] Lucy Xiaoyang Shi、Brian Ichter、Michael Equi、Liyiming Ke、Karl Pertsch、Quan Vuong、James Tanner、Anna Walling、Haohuan Wang、Niccolo Fusai 等。《嗨，机器人：利用分层视觉-语言-动作模型实现开放式指令跟随》。载于 国际机器学习会议（*ICML*），2025。2，3[34]Gemini Robotics Team、Abbas Abdolmaleki、Saminda Abeyruwan、Joshua Ainslie、Jean-Baptiste Alayrac、Montser- rat Gonzalez Arenas、Ashwin Balakrishna、Nathan Batch- elor、Alex Bewley、Jeff Bingham、Michael Bloesch、Kon- stantinos Bousmalis、Philemon Brakel、Anthony Brohan、Thomas Buschmann、Arunkumar Byravan、Serkan Cabi、Ken Caluwaerts、Federico Casarini、Christine Chan、Oscar Chang、London Chappellet-Volpini、Jose Enrique Chen、Xi Chen、Hao-Tien Lewis Chiang、Krzysztof Choromanski、Adrian Col- lister、David B. D'Ambrosio、Sudeep Dasari、Todor Davchev、Meet Kirankumar Dave、Coline Devin、Norman Di Palo、Tianli Ding、Carl Doersch、Adil Dostmohamed、Yilun Du、Debidatta Dwibedi、Sathish Thoppay Egambaram、Michael Elabd、Tom Erez、Xiaolin Fang、Claudio Fantacci、Cody Fong、Erik Frey、Chuyuan Fu、Ruiqi Gao、Marissa Giustina、Keerthana Gopalakrishnan、Laura Graesser、Oliver Groth、Agrim Gupta、Roland Hafner、Steven Hansen、Leonard Hasen- clever、Sam Haves、Nicolas Heess、Brandon Hernaez、Alex Hofer、Jasmine Hsu、Lu Huang、Sandy H. Huang、Atil Is- cen、Mithun George Jacob、Deepali Jain、Sally Jesmonth、Abhishek Jindal、Ryan Julian、Dmitry Kalashnikov、M. Emre Karagozler、Stefani Karp、Matija Kecman、J. Chase Kew、Donnie Kim、Frank Kim、Junkyung Kim、Thomas Kipf、Sean Kirmani、Ksenia Konyushkova、Li Yang Ku、Yuheng Kuang、Thomas Lampe、Antoine Laurens、Tuan Anh Le、Isabel Leal、Alex X. Lee、Tsang-Wei Edward Lee、Guy Lever、Jacky Liang、Li-Heng Lin、Fangchen Liu、Shangbang Long、Caden Lu、Sharath Maddineni、Anirudha Majumdar、Kevis-Kokitsi Maninis、Andrew Marmon、Sergio Martinez、Assaf Hurwitz Michaely、Niko Milonopoulos、Joss Moore、Robert Moreno、Michael Neunert、Francesco Nori、Joy Ortiz、Kenneth Oslund、Carolina Parada、Emilio Parisotto、Amaris Paryag、Acorn

Pooley、Thomas Power、Alessio Quaglino、Haroon Qureshi、Rajkumar Vasudeva Raju、Helen Ran、Dushyant Rao、Kan- ishka Rao、Isaac Reid、David Rendleman、Krista Reymann、Miguel Rivas、Francesco Romano、Yulia Rubanova、Peter Pas- tor Sampedro、Pannag R Sanketi、Dhruv Shah、Mohit Sharma、Kathryn Shea、Mohit Shridhar、Charles Shu、Vikas Sindhwani、Sumeet Singh、Radu Soricut、Rachel Sterneck、Ian Storz、Razvan Surdulescu、Jie Tan、Jonathan Tompson、Saran Tunyasuvunakool、Jake Varley、Grace Vesom、Giulia Vezzani、Maria Bauza Villalonga、Oriol Vinyals、Ren´e Wagner、Ayzaan Wahid、Stefan Welker、Paul Wohlhart、Chengda Wu、Markus Wulfmeier、Fei Xia、Ted Xiao、Annie Xie、Jinyu Xie、Peng Xu、Sichun Xu、Ying Xu、Zhuo Xu、Jimmy Yan、Sherry Yang、Skye Yang、Yuxiang Yang、Hiu Hong Yu、Wenhao Yu、Wentao Yuan、Yuan Yuan、Jingwei Zhang、Tingnan Zhang、Zhiyuan Zhang、Allan Zhou、Guangyao Zhou 和 Yuxiang Zhou。《Gemini robotics 1.5：凭借先进的具身推理、思考和运动迁移，推动通用型机器人的发展前沿》，2025。2[35] Yang Tian、Sizhe Yang、Jia Zeng、Ping Wang、Dahua Lin、Hao Dong 和 Jiangmiao Pang。《预测性逆动力学模型是可扩展的机器人操作学习器》。arXiv 预印本 arXiv:2412.15109，2024。ICLR 2025。3[36]Wenke Xia 等。《Phoenix：一种基于运动的自我反思框架，用于精细化机器人动作纠正》。载于 IEEE/CVF计算机视觉与模式识别会议（CVPR）论文集，2025。arXiv:2504.14588。3[37] Seonghyeon Ye、Yunhao Ge、Kaiyuan Zheng、Shenyuan Gao、Sihyun Yu、George Kurian、Suneel Indupuru、You Liang Tan、Chuning Zhu、Jiannan Xiang、Ayaan Malik、Kyungmin Lee、William Liang、Nadun Ranawaka、Jiasheng Gu、Yinzhen Xu、Guanzhi Wang、Fengyuan Hu、Avnish Narayan、Johan Bjorck、Jing Wang、Gwanghyun Kim、Dantong Niu、Ruijie Zheng、Yuqi Xie、Jimmy Wu、Qi Wang、Ryan Julian、Danfei Xu、Yilun Du、Yevgen Chebotar、Scott Reed、Jan Kautz、Yuke Zhu、Linxi "Jim" Fan 和 Joel Jang。《世界动作模型是零样本策略》。arXiv 预印本 arXiv:2602 15922，2026 1，3。。[38] Ge Yuan、Qiyuan Qiao、Jing Zhang 和 Dong Xu。《AdaWorld-Policy：面向机器人操作的在线自适应学习世界模型驱动扩散策略》。arXiv 预印本 arXiv:2602.20057，2026。3[39] Tianyuan Yuan、Zibin Dong、Yicheng Liu 和 Hang Zhao。《Fast-WAM：世界动作模型需要测试时未来想象吗？》arXiv 预印本 arXiv:2603.16666，2026。1，2，3[40] Jianke Zhang、Yanjiang Guo、Xiaoyu Chen、Yen-Jen Wang、Yucheng Hu、Chengming Shi 和 Jianyu Chen。《HiRT：利用分层机器人 Transformer增强机器人控制》。载于机器人学习会议（CoRL），2024。3[41] Wenbo Zhang、Kaixuan Wang、Yutao Ouyang、Xiaoyu Huang、Liyang Li、Kailun Su、Weiyang Jin、Wenhao Chai、Haotian Liang、Zhiyang Dou、Yue Chen 和 Tianxing Chen。《一项出人意料的机器人策略：GPT-6 Astra 在 RoboDojo 及其他场景中的早期评估》。arXiv 预印本 arXiv:2609 24170，2026 3。。[42] Yunbei Zhang、Zijian Jin 等。《LIBERO-MAX：机器人世界发生变化时，策略会如何适应？*arXiv*预印本 *arXiv:2609.36518*，2026年。3，8，1[43] Qingqing Zhao 等人。CoT-VLA：面向视觉-语言-动作模型的视觉思维链推理。载于 *IEEE/CVF*计算机视觉与模式识别会议论文集（*CVPR*），2025年。arXiv:2503.22020。2，3[44] Xueyang Zhou、Yangming Xu、Guiyao Tie、Yongchao Chen、Guowen Zhang、Duanfeng Chu、Pan Zhou 和 Lichao Sun。LIBERO-PRO：迈向超越记忆化的视觉-语言-动作模型稳健且公平的评估。*arXiv*预印本 *arXiv:2510.03827*，2025年。3[45] Chuning Zhu、Raymond Yu、Siyuan Feng、Benjamin Burchfield、Paarth Shah 和 Abhishek Gupta。统一世界模型：耦合视频与动作扩散，以便在大型机器人数据集上进行预训练。*arXiv*预印本 *arXiv:2504.02792*，2025年。1，3

RoboThink：让智能体思考，让策略执行

补充材料

A. 智能体接口

智能体在第 1、2、3 个动作块之后调用，随后每隔四个动作块调用一次（前三个动作块之后约每 40 个控制步调用一次）。它会接收一张拼接图，展示最近两个动作块的最后八帧（前置摄像头）、用户请求，以及一张表格，其中每个动作块占一行，列出指令末端执行器运动（指令总和乘以在标称回合上拟合的逐轴增益）和实际运动，单位为厘米和度。提示词描述了可能出现的故障（“每条指令都可能附加一个恒定偏差……也可能完全正常”）、尺寸档位（轻微：0。35–0。9 厘米或每个动作块 2–4°，中等：最高 1。6 厘米或 8°，强烈：超过上述范围），并要求以 JSON 格式返回：问题；每个轴（前、左、上、滚转、俯仰、偏航）上手自行移动的方向及幅度；置信度；以及三个句子（评估、原因、调整）。任务检查是同一次调用的第二部分：

> 其次，检查机器人是否正在执行所要求的操作。上面的请求可能采用了间接的措辞……机器人也可能误解请求：伸手去拿错误的物体或放错位置、开始执行另一项任务，或者根本没有开始。如果请求不是清晰明确的命令，或者机器人没有执行所请求的任务，就在“instruction”中写下一个简短明确、供机器人执行的命令……

提示词中的示例来自训练任务。策略不会读取智能体自己写的句子。固定模板会将各轴的方向和尺寸渲染为反馈文本中的三个字段（调整、评估、原因），所用短语取自措辞库；训练和测试使用互不重叠的措辞库。每个尺寸档位对应一个幅度（0.05/ 0.10/ 0.15 个动作单位），由此得到数值估计 δ，作为 DAgger 标签并供特权规则使用。长度为 2–25 个词的指令会替换两个模块中的指令，直到智能体写出另一条指令为止。调用失败时，之前的回答仍然有效。

其他智能体。我们从四种设置中保存了 1,100 次智能体调用，并让每个模型回答相同的输入。估计值与真实漂移之间的方向余弦相似度（x图 5 的轴）在 Opus 和 Kimi K3 上为 0.86，在 Astra 和 Sol 上为 0.87，在 Sonnet 上为 0.81，在仅读取运动表的 gpt-oss-120b 上为 0.80。

示例。图S1 展示了一次智能体调用，图S2 展示了另外两个回合。

B. 基准测试

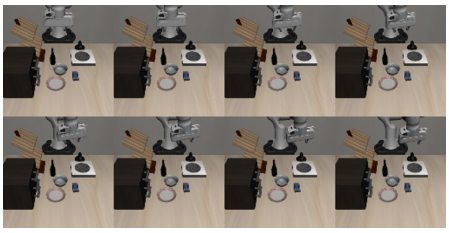
在 LIBERO 中，测试任务是 LIBERO-Spatial、-Object 和 -Goal 的任务 7–9，这些任务未参与任何反馈训练。每项任务均从前 5 个初始状态开始运行（每个单元格 45 个回合）或从前 15 个初始状态开始（135 个回合）。这 45 个回合具有代表性：表6 中的反事实反馈模型在其中 0.82 / 0.80 的回合中成功，在全部 135 个回合中的成功率为 0.81 / 0.75。表1、2、5 和6 以及图1、5 和6中，成功指在 LIBERO 的标准步数预算内完成任务（Spatial、Object 和 Goal 分别为 220、280 和 300 步），这些预算适用于单独行动的智能体。在 LIBERO 的其他实验中，策略的回合运行 400 步（在 LIBERO-10 上为 700 步），并根据撤销操作的时长相应延长。除智能体比较（表5 和图5）、负载和 LIBERO-10（GPT-6 Astra）的实验，以及图6和7（其中向策略提供精确反馈）外，智能体始终为 Claude Opus 5.5。

LIBERO-Para [16]：对每个留出的 LIBERO-Goal 任务提供 30 种释义改写，每个初始状态对应一种。误听的请求：在留出的 LIBERO-Goal 任务中，策略接收同一场景下任务（$t+1$）、（$t+4$）或（$t+7$）的指令（对 10 取模），而智能体接收正确指令（初始状态 0–4）。哪个物体：在留出的 LIBERO-Object 任务中，策略接收“拿起 X 并将其放入篮子”的指令，所指物体为同一场景中三个训练任务的物体 X。任务纠正：使用相同的错误指令，并在第 3 个动作块之后给出正确指令。在*RoboTwin*中，在两个要求将一个物体放到另一个物体旁边的任务中，机器人收到的两个物体顺序被颠倒了。*LIBERO-Plus* [10]：留出任务的 126 种变体。*LIBERO-MAX* [42]：Lite 划分中的 800 个案例（在每种协议下均完成了 795 个），每个案例都会在手首次进入物体（目标物体或容器）0.18 米范围内时触发一个事件（目标物体或容器移动、摄像头偏移、传感器噪声出现、灯光或主题切换、干扰物或障碍物出现）。

C. 实现

Fast-WAM 和 RoboThink。Fast-WAM [39]（Wan2.2-5B 视频模块、1.0B 动作专家、umT5 文本编码器；在 224×224 处使用前置摄像头和腕部摄像头）预测 32 步动作块，并执行前 10 步。角色标记是 4096 维向量，由“Instruction:”和“Feedback:”的编码初始化；两个片段拼接后用零填充至 128 个位置。DAgger 数据：由 Opus 作为智能体，在 21 个训练任务上采集 840 个回合；每个回合分别施加来自训练分布的漂移（幅度为 0.05–0.15，从不将测试值 0.1 作为固定值使用）以及无漂移情况。训练：训练动作专家和角色标记，并使用策略的流匹配损失，学习率为 $2 \cdot 10^{-5}$，以 32 个片段为一批训练 3,000 步；对于在不同于片段反馈文本的反馈文本下执行的动作，将其从损失中屏蔽。两个额外的训练随机种子在 LIBERO 上使用实时 Opus 得到 0.87 / 0.84 / 0.93 和 0.80 / 0.89 / 0.84（固定 / 随机 / 无漂移）

输入



请求：“打开炉灶”

动作块		Fwd.	Left	Up	Roll	俯仰	Yaw
1	cmd.	−1.9	8.5	−1.6	0.1	0.9	−0.8
	act.	−1.6	9.8	−1.0	7.4	6.9	6.0
2	命令	−5.6	5.7	−5.7	−7.0	−2.4	−7.9
	act.	−4.2	7.8	−4.4	−0.3	5.6	−1.1

回答 （节选）

问题：确实存在；置信度：0.75；手部会自行向前移动（轻微）、向左移动（明显）、向上移动，并发生滚转、俯仰和偏航（中等程度）。
评估：“在上一个动作块中，手向左超调了（+2 .1 厘米），并向前超调了（+1 .4 厘米），下落幅度小于指令值 ……并且在滚转、俯仰和偏航方向上比指令值多旋转了约 +7 至 +8 度，因此没有按计划接近炉灶旋钮。”

原因：这表明问题源于地定的执行器偏置残差，并具有相同的正号 . . .

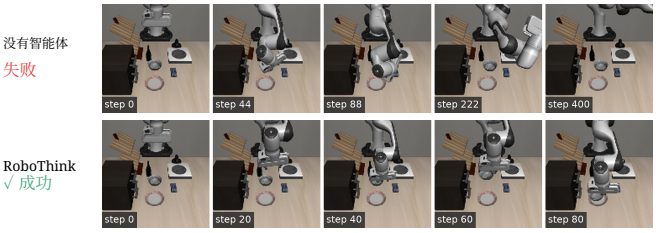
策略读取的内容

[反馈]调整：稍微向后推，大力向机器人的右侧推，并适度向下推。撤销旋转（滚转适度取负值，俯仰适度取负值，偏航适度取负值）。评估：手部正在自行缓慢向前、向机器人的左侧并向上移动。手腕也在转动（滚转为正，俯仰为正，偏航为正）。原因：某种因素持续使每条指令产生偏置。

对六个机械臂轴的估计值为 $\hat{\delta} = (0.05, 0.15, 0.10, 0.10, 0.10, 0.10)$；真实漂移：每个轴均为 0.1。

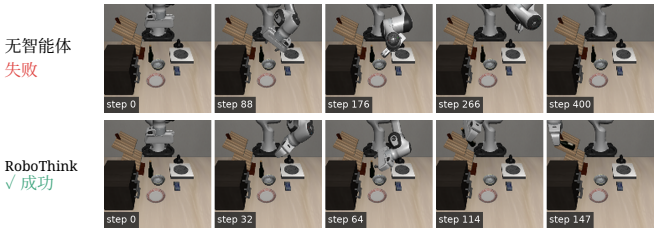
图 S1. **智能体的一次调用** （Claude Opus 5.5），发生在恒定漂移条件下一个留出的 LIBERO-Goal 回合的第 2 个动作块之后。运动单位为厘米和度。模板会将各轴的答案呈现在反馈文本中；智能体自己的句子不会传递出去。

(a) “把碗放在盘子上”



智能体，在第1块之后：“……结果位置偏左约*1.5*厘米，偏低约*0.9*厘米，滚转角多了约*7*度，偏航角多了约*6*度。” [反馈] “向机器人的右侧适度推回，并稍微向上推。将手腕转回（适度向负方向滚转，适度向负方向偏航）……”

(b) “把葡萄酒瓶放在架子上”



智能体，动作块 3 之后：漂移现在出现在全部六个轴上。 [反馈] “适度向后推，向机器人的右侧适度推，并适度向下推。撤销旋转（滚转适度为负，俯仰适度为负，偏航适度为负）…… ”

图 S2。**更多定性示例**：在恒定漂移条件下执行留出的 LIBERO-Goal 任务（前置摄像头）。无智能体：不带智能体的同一策略。(a) 执行第一个动作块后，智能体描述机器人向左、向下，并发生滚转和偏航漂移；策略读取其反馈后将碗放好（共调用智能体四次，耗时 46 秒）。(b) 执行第一个动作块后，智能体报告机器人向左漂移，并发生滚转、俯仰和偏航漂移；执行第三个动作块后，智能体报告机器人在全部六个轴上均发生漂移，与真实漂移一致；策略随后将瓶子放到架子上（共调用智能体五次，耗时 65 秒）。

表 1中的回合，相比之下，第一项分别为 0.87 / 0.84 / 0.89。**先撤销再重做**。撤销块是机器人从开始执行（或上次撤销）以来发出的命令，按逆序排列；机械臂各分量取反，夹爪打开，并用空操作补齐至完整动作块。执行撤销块时不会查询策略，且回合的步数预算会按其长度延长。机器人会在脚本化纠正后撤销；每当智能体写出新指令且同时给出 任务中：否时，也会撤销；此外，在智能体首次改写用户请求以外的指令时也会撤销。等到智能体报告机器人偏离任务后才撤销，对于误听的请求来说已经太迟；而每次改写都撤销，也会把正确的运动一并撤销，因为智能体可能只是在换种说法重述请求（表 S1）。**GR00T N1.7**。发布的 LIBERO 检查点；使用 GR00T 作为策略收集了 315 个 Opus DAgger 回合；使用 GR00T 自带的训练器进行微调，每个 rollout 都制作两个反事实副本和一个无反馈副本。反馈会附加到指令中；视觉语言模型被冻结，因此反馈通过其特征传递至动作头。**Cosmos Policy**。发布的 LIBERO 检查点

表 S1。**何时撤销**。任务偏离：当智能体报告机器人偏离任务时。首次改写：当智能体首次将不属于用户请求的指令进行改写时。LIBERO-Para：向机器人传达用户重新措辞后的请求；误听：向机器人传达另一项任务。

	LIBERO-Para			误听	
Undo	无	常数	随机	RoboThink	Cosmos
任务偏离	**0.73**	**0.81**	**0.72**	0.56	0.76
每次改写	0.68	0.59	0.51	**0.96**	**1.00**
偏离任务，首次重写	**0.73**	**0.81**	**0.72**	**0.96**	**1.00**

要点 [18] （Cosmos-Predict2-2B 在演示数据上微调；T5-11B 文本嵌入通过交叉注意力输入视频-动作转换器），采用公开的推理设置；每个 16 步动作块执行前 10 个动作。在 LIBERO 上不经训练使用（[INST]）。对于 [反馈]，，指令和“反馈：……”的 T5 行分别编码后再拼接，因此没有反馈时，条件输入与指令完全相同；模型在 RoboCasa 上训练（第F节）。

D. 仅智能体

智能体直接在机器人上实时行动，策略不介入其中；它通过一套由这类直接充当策略的智能体所使用的工具来操作 [6, 15]：每次在机器人上运行一段简短的 Python 脚本（应用程序接口可闭环控制手部移动到指定的三维位置和偏航角、打开和关闭夹爪、等待并读取状态），每段脚本运行完后，智能体会查看摄像头图像、深度图和相机标定信息。GPT-6 Astra 在 Codex 中运行，Claude Opus 5.5 在 Claude Code 中运行，均采用默认推理设置，每回合时限为 30 分钟。提示词与 RoboThink 智能体的提示词一样，说明每条命令都可能叠加未知偏差，且手部最终位置可能与命令指定的位置不同。 **LIBERO：** 使用 45 个回合、LIBERO 标准步数预算，以及策略评估中的成功判定；场景稳定后，每条命令都会叠加漂移，且每回合的偏移量与策略所用的相同。策略运行 400 步的回合事后截断至相同的步数预算。策略并不知道预算，因此此处理完全精确；这会使策略的成功次数减少 3 次。 **RoboTwin 2.0：** 对双机械臂使用相同的工具（运动规划至指定姿态），采用 44 个测试回合（第 E节）、RoboTwin 的步数限制和相同的漂移（对于随机漂移，使用与 RoboThink 同种子回合所获得的偏移量）。成功时间从智能体开始运行时算起；对于策略，则从回合开始时算起；终点为首次满足任务成功判定的步数。RoboThink 会同步调用智能体：调用期间机器人会等待。

E. RoboTwin 2.0

协议。 Fast-WAM 发布的 RoboTwin 检查点 [5,39]（三台摄像头，两条六关节机械臂及夹爪的 14 维绝对关节目标值，计划的 32 步中执行了 24 步）。漂移会偏移每个机械臂关节的目标值，就像零位校准误差一样： $+0.05$弧度（恒定）或 $U(-0.06,\ 0.06)$[12] 每回合（随机）。智能体（Opus）会查看头部相机画面以及一张表格，其中列出指令关节角度与测量关节角度，并针对每个关节给出方向和幅度。模板按功能命名关节（腰部、肩部、肘部、手腕俯仰、偏航、滚转）；如果按编号命名关节（“左侧关节 3”），策略根据反馈应对随机偏移的方向余弦为 0.36，而使用功能名称时为 0.88。训练：在 36 个任务上进行 464 个由智能体标注的 DAgger 回合训练，并加入训练漂移（稀疏、密集和共模偏移，范围为 0.025–0.12 弧度）；反事实反馈（$p_{\text{cf}} = 0.8$）；以 32 个片段为一批，训练 3,000 步；仅训练动作专家的文本嵌入、文本交叉注意力和角色标记（3.83 亿个参数）。测试时，反馈生效期间由训练后的权重进行计划，否则使用发布的权重。表1和3中的测试回合：干净场景；在 14 个未参与反馈训练的任务中选取 11 个任务，各取前四个官方随机种子（见下文），但不包括那三个

表 S2. **漂移条件下的 AW-0** 在 RoboTwin 2.0 上：反馈训练之外六项任务的成功率。智能体：Claude Opus 5.5。

	常数	随机	无
Fast-WAM	0.07	0.35	0.82
AW-0	0.11	0.37	0.75
AW-0 + 反馈在指令中	0.13	0.37	0.75
RoboThink	**0.73**	**0.60**	**0.84**

成功判定需要脚本专家状态的任务。**存在漂移时的 AW-0**（表S2）。AW-0 [26]仅报告无漂移时的结果。我们使用其发布的 RoboTwin 检查点及其评估设置，在 11 个任务中的 6 个上进行测试；每个任务 20 个回合，场景均为干净场景。与 Fast-WAM 一样，AW-0 在存在漂移时会失败，即使其指令中加入了 Opus 的反馈也是如此。**官方协议**（表4）。RoboTwin 的评估脚本保留脚本专家使用 cuRobo 成功解决的随机种子（从 100000 起），采用专家回合中的指令，并使用该场景的专家状态评判策略回合；无漂移时每个任务 20 个回合，有漂移时每个任务 10 个回合。在 14 个未参与反馈训练的任务中，恒定漂移／随机漂移下的成功率，在干净场景中从 0.08／0.29 提升至 0.69／0.44，在随机化场景中则从 0.06／0.26 提升至 0.76／0.57。无漂移时不会提供反馈，因此干净场景中的回合完全相同；而在随机化场景中，模拟器并非确定性的。AW-0 在随机化场景中的结果不可比较：它使用 2,500 条干净场景示范数据进行训练，而 Fast-WAM 还使用了 25,000 条随机化场景示范数据。

F. RoboCasa

协议。 Cosmos Policy 发布的 RoboCasa 检查点 [18, 27]（停放的移动底座上的一条机械臂、三台摄像机、计划执行的 32 个动作中执行了 16 个）。留出任务：柜台上的四项抓取与放置任务，在五个包含未见物体的测试厨房中进行；漂移方式与 LIBERO 相同。智能体（Opus）读取机器人基座坐标系中的手部运动。DAgger 数据：在训练分布内的厨房中，针对另外 20 项任务的 600 个智能体标注回合。 **训练。** 仅在动作坐标系上计算损失（这是 Cosmos Policy 自带的选项）时，策略保留了原有技能，并学会遵循反馈（探针余弦相似度 0.93，增益 0.99）。**配方：** 动作损失，仅使用文本交叉注意力（1,956M 个参数中的 411M 个），$p_{\text{cf}} = 0.8$，训练 2,000 步，随后再训练 6,000 步，其间一半反馈文本使用第二个训练短语库改写；根据验证回合（在十个训练厨房中进行的留出任务）选择模型。将特权规则应用于发布的策略，在恒定漂移／随机漂移下的成功率分别为 0.29／0.34，而 RoboThink 为 0.25／0.30（表 3）。

表 S3。**失败后调用智能体**。两次尝试后的成功率，每次尝试 400 步。随机漂移下的 LIBERO-Plus 和 LIBERO-MAX。智能体：Claude Opus 5.5。

	LIBERO			Plus	MAX
	常数	随机	无	随机	随机
直接进行第二次尝试	0.07	0.40	**0.97**	0.29	0.07
使用智能体	**0.84**	**0.84**	**0.97**	**0.62**	**0.21**

G. 为什么反馈属于动作模块

在反馈完美的情况下，图 6 中的 DAgger 策略在两个模块都读取反馈时，回合成功率为 0.40／0.64；仅动作模块读取时为 0.64／0.76；仅视频模块读取时为 0.02／0.40（恒定漂移／随机漂移；无反馈时为 0.02／0.38）。三项探测揭示了原因。（i）修正由动作模块生成：在首次获得反馈的动作块中，较早的 DAgger 策略（由 GPT-6 Astra 教授，与图 7b 中的 DAgger 策略相同）在仅动作模块读取反馈时，执行了所述修正的 0.74；仅视频模块读取时则为 0.02。（ii）反馈会干扰世界模型：相较于普通提示词，它使视频模块的逐层键和值改变 10% 和 22%；这一变化是另一项任务的指令所致变化的三分之二，并使计划动作的变化幅度达到重新采样的七倍。（iii）未经训练的策略会对文本的存在作出反应：在指令后附加一条没有实际内容、共 38 个词的备注，会使 Fast-WAM 在无漂移时的成功率从 0.96 降至 0.51。

H. 更困难的偏移：详情

会变化的漂移（表 8）。恒定漂移会在第 100 步反转；随机漂移会在第 80 步出现，或在第 100 步重新抽取。使用标准提示词时，智能体会晚一次调用才报告反转（成功率为 0.51）。针对反转，提示词中有一句话说明偏差可能发生变化，并要求报告最近一次偏差；之后，智能体会在变化后的首次调用中报告新的漂移。没有漂移时，加入这句话的成功率为 0.89，不加入则为 0.91（45 个回合，400 步）。出现或重新抽取的漂移均使用标准提示词。 **负载和长时任务**。负载是施加在手上的向下的力，仅在持有物体时施加，且训练期间从未出现过。对于负载和 LIBERO-10，智能体均为 GPT-6 Astra，特权规则采用 Astra 的估计值（负载下为 0.56）。LIBERO-10：全部十项任务，初始状态 0–14，恒定漂移和随机漂移。

I. 仅在失败后调用智能体

RoboThink 的策略也可以先单独执行，只有失败后才调用智能体，并从相同的初始状态和偏移重新尝试（表 S3）。如果策略单独执行成功，则不会调用智能体。

J. 局限性

所有实验均在仿真中进行，反馈通道是在回合期间保持恒定的漂移数据上训练的。智能体能够看到并描述其他偏移时，系统会对其进行纠正（表8 和9），但智能体无法赋予策略其本身不具备的技能：在 LIBERO-MAX 的场景事件下，成功率仍为 0.20。每次智能体调用期间，机器人都会等待（中位数为 8.2 秒）；而智能体的误报会导致无漂移的回合中有 0.07 的回合受到影响。先撤销再重做的前提是机器人能够沿原路回退。