

代码世界模型： 把环境动力学写成可执行的程序

从文本智能体的信念程序 到连续控制的规划动力学

白佳欣
香港浸会大学 助理教授

2026 年 9 月

世界模型是什么？



September 1, 2026 Introducing Atlas, our new omni world model for spatial intelligence.

Atlas: A World Model for Spatial Intelligence



Technical Report

Model Cards

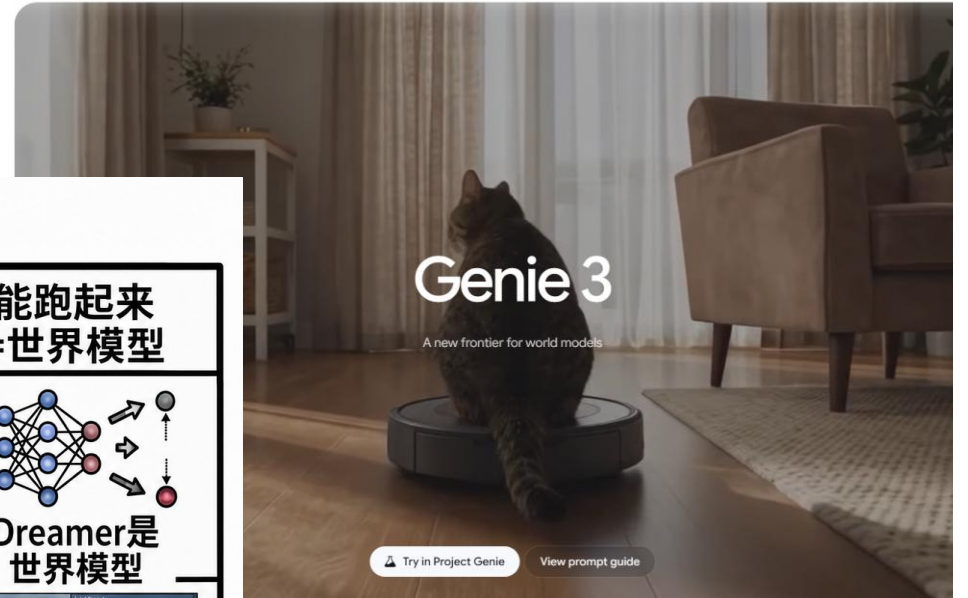
GitHub Code

Product Website

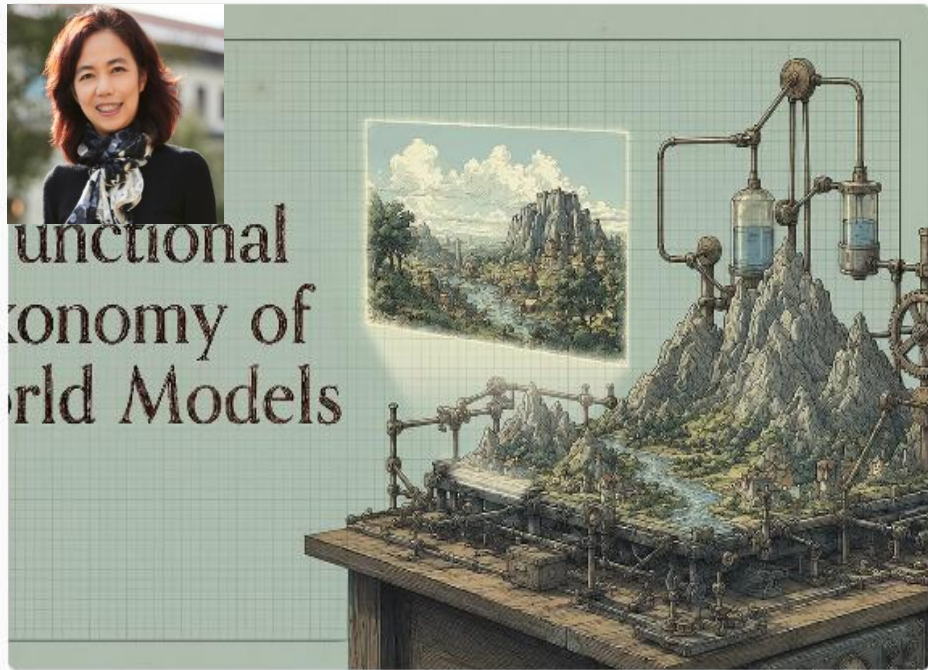
世界模型九宫格

	重建=世界模型	预测下一步=世界模型	能跑起来=世界模型
特征/潜变量	DINO是世界模型	JEPA是世界模型	Dreamer是世界模型
对象/3D	NeRF是世界模型	Scene Flow是世界模型	MuJoCo是世界模型
像素/视频	MAE是世界模型	视频扩散是世界模型	贪吃蛇跑通了，也是世界模型

Credit: Internet



世界模型是什么？渲染器，模拟器，和规划器



A Functional Taxonomy of World Models

Renderers, Simulators, Planners, and the Loop That Connects Them

JUN 4 • FEI-FEI LI

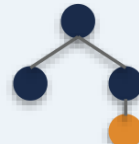
♥ 1,823 💬 123 ↻ 285 ↗



Render observations



Simulate state

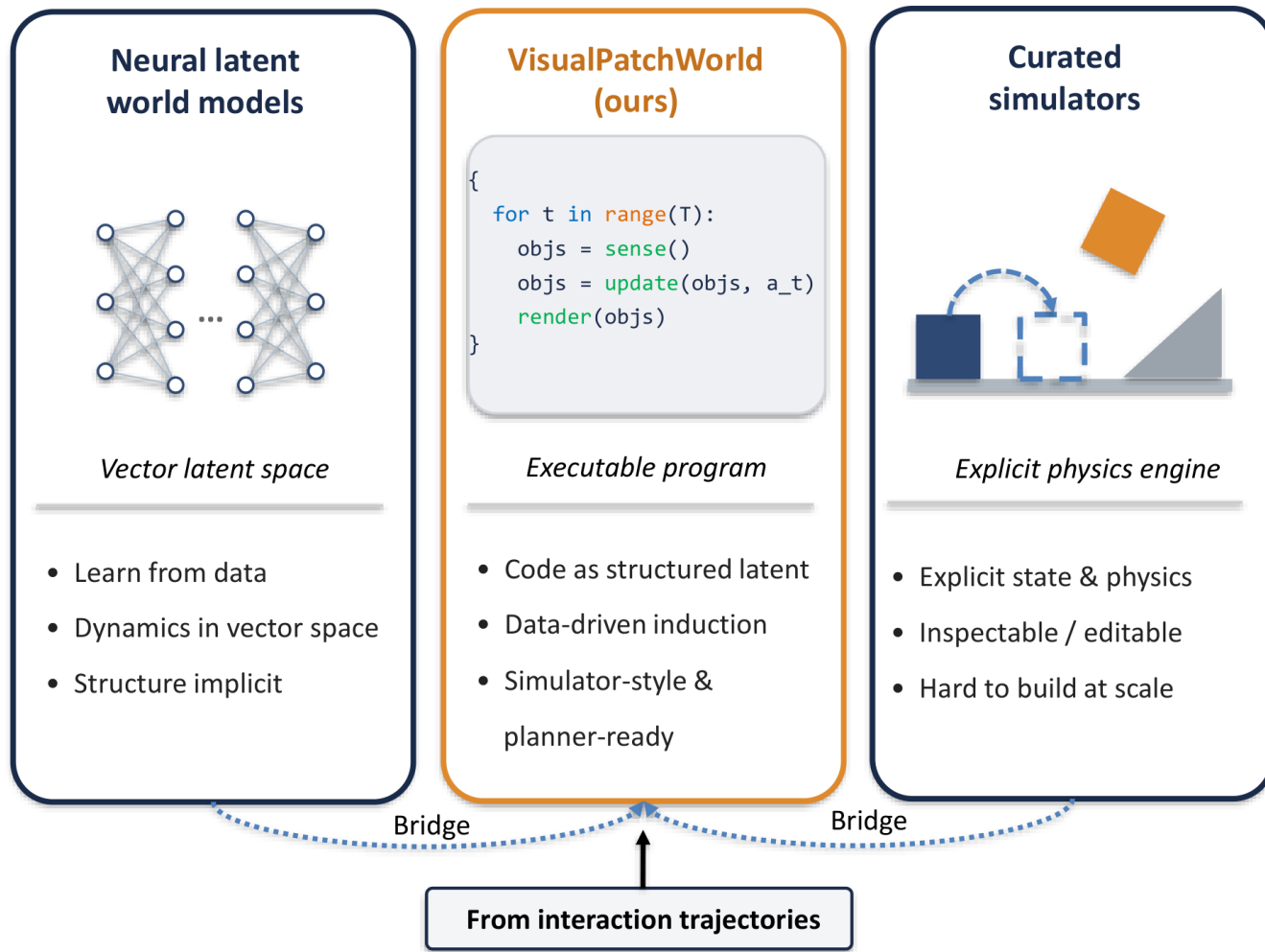


Plan actions

世界模型的功能角色：

- 生成观测
- **模拟仿真**
- 规划动作

- **模拟仿真是关键：** 状态和动力学被显式写出来，观测的样子和动作的后果就都变成可计算的。



- **人工搭建的模拟器。** 状态与物理显式、可编辑、规划能力强；代价是很难扩展到不同环境。
 - Atlas可以从照片复原3D场景
 - 是否接触，惯性？是否可以用来规划？
 - 从经验和交互归纳？
- **神经隐式模型。** 从原始观测序列中归纳动力学，容易扩展；
 - 代价是放弃可检视的结构。
- **代码世界模型。** 从交互数据中归纳出环境背后的执行代码。
 - 既可以通过代码生成扩展到不同场景
 - 又可以维持可检视维护的状态和状态转移公式

- 从黑箱文本环境中 归纳信念程序
- 从视觉环境中 归纳动力学公式

大模型负责提出、修复与筛选世界模型程序，而不是自己去充当状态转移模型。

第一部分

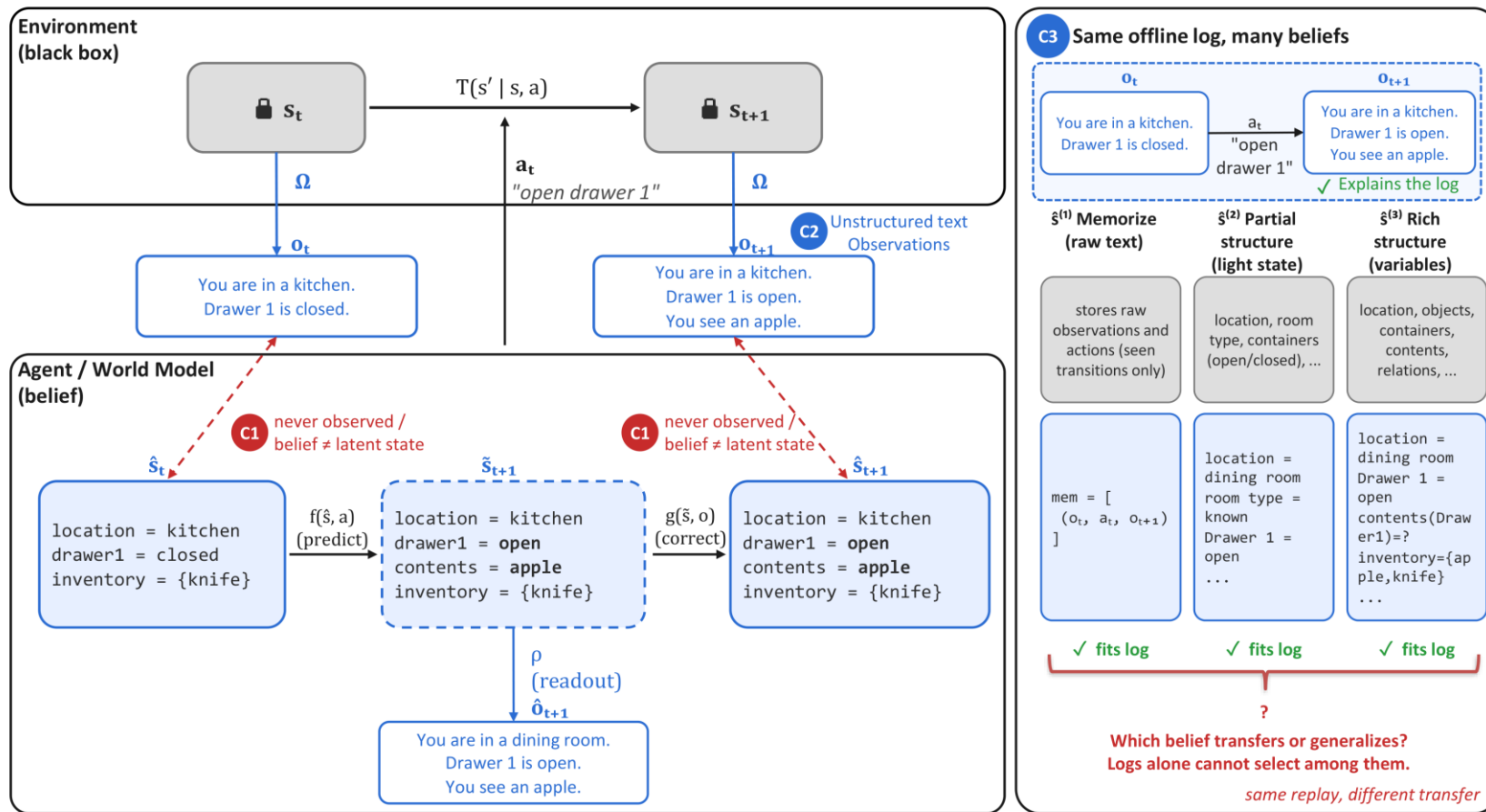
从黑箱文本环境中 归纳信念程序

PatchWorld: Gradient-Free Optimization of Executable World Models for Agent Environments

Jiaxin Bai, Yue Guo, Yifei Dong, Jiaxuan Xiong, Tianshi Zheng, Yixia Li, Tianqing Fang, Zihao Wang, Lihui Liu, Jeff Z. Pan, Yangqiu Song, et al.

arXiv:2605.30880 <https://github.com/HKBU-KnowComp/PatchWorld>

文本智能体环境是黑箱 POMDP



- **挑战1** 潜在事实可能永远无法被智能体直接获得。
- **挑战2** 对同样一个潜在事实，有许多种表面措辞描述。
- **挑战3** 许多程序都能回放同一批日志；单靠日志无法判断哪一个能迁移。

Figure 1, PatchWorld. 环境来自 AgentGym (Xi et al., 2025) : AlfWorld、BabyAI、Maze、SciWorld、TextCraft、WebShop、Wordle。

以数据结构为潜表示，以代码为状态转移方程

状态是数据结构，状态转移律就是源代码。

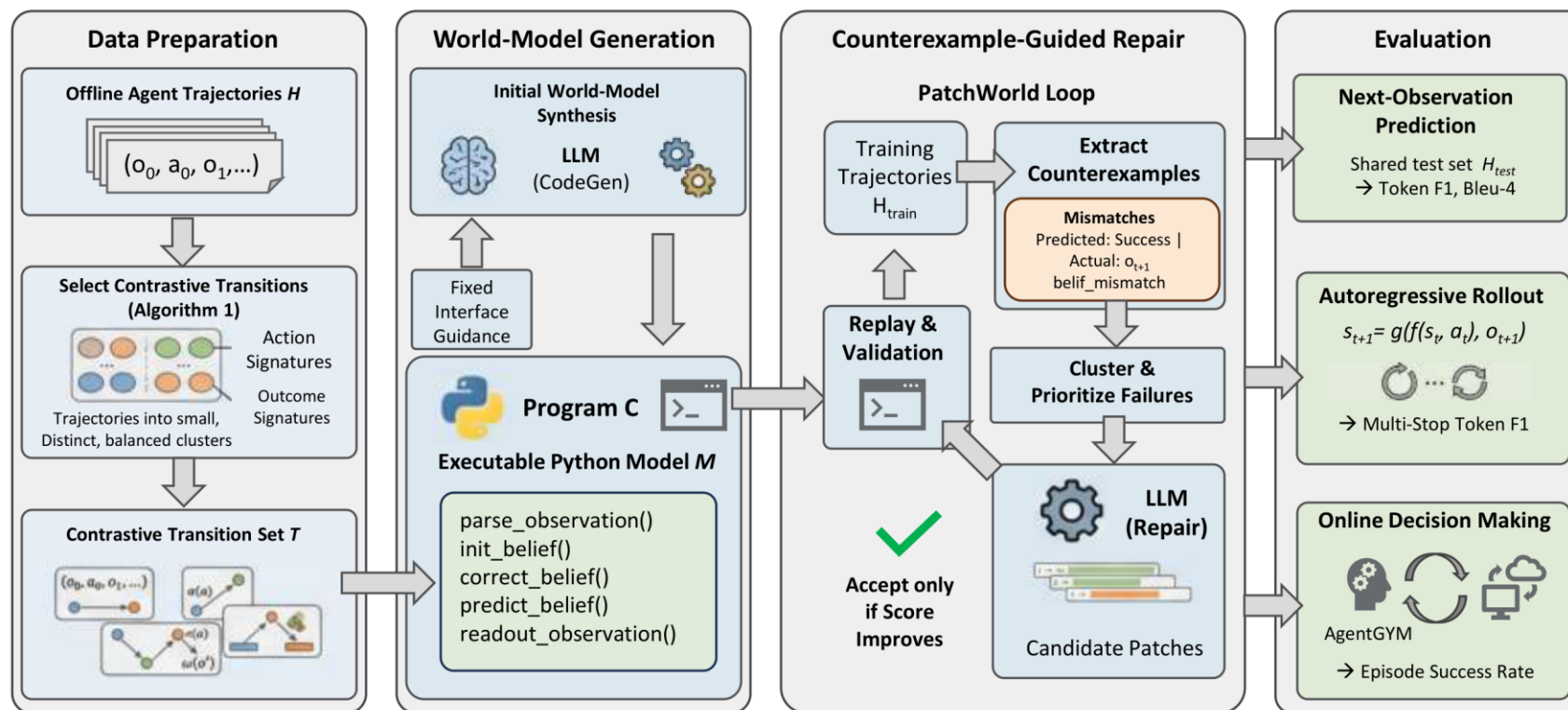
```
class WorldModel:          # 归纳一次，随后冻结
    def parse_observation(o)    -> State
    def init_belief(o)         -> State
    def predict_belief(s, a)    -> State
    def readout_observation(s, a) -> str
    def correct_belief(s, o)    -> State
```

- **回放。** 在记录下来的轨迹上执行，精确定位分歧发生在哪一步。
- **修复。** 直接改写出错的那个分支，而不是重训一个黑箱。
- **复用。** 在任意规划器里向前推演，决策时不再调用任何模型。
- **检视。** 状态是有名字的字段，任何一步的信念都能直接打印出来看。

把信念更新与观测读出分开，是让动力学保持简单的关键：程序不必去复现每一种表面措辞。

前瞻真正需要的只是状态层面的差异；把文本生成写成另一个函数，动力学就不必为措辞买单。

PatchWorld: Generative Optimization of Executable World Models from Agent Trajectories



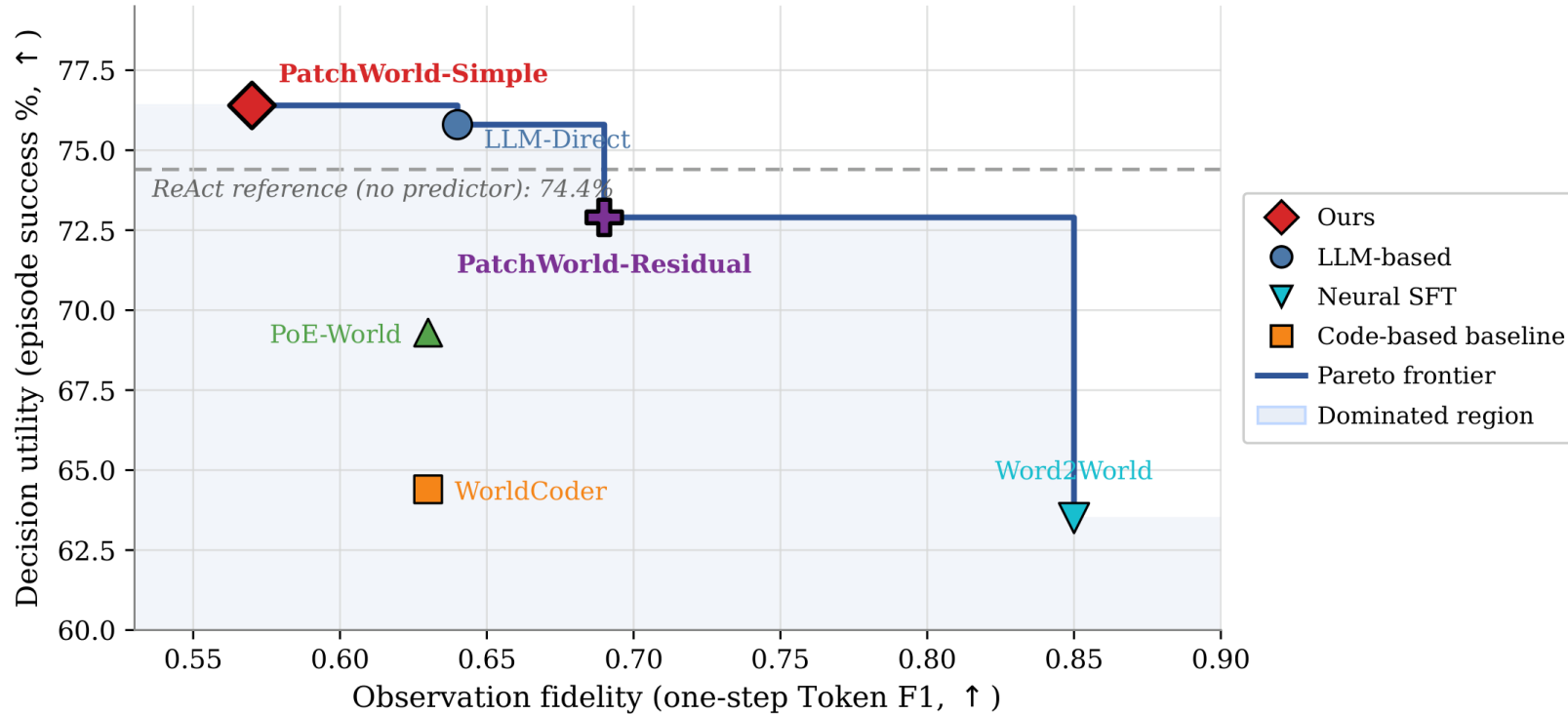
- 每轮只接受一个补丁，且必须在留出验证集的回放得分上严格变好。归纳阶段一次性消耗 17–28 次大模型调用；之后的预测就只是普通的 Python 执行。

Figure 2, PatchWorld. 即把反例引导的归纳式程序合成 (Solar-Lezama et al., 2006) 用在序贯 POMDP 动力学上。

七个 AgentGym 环境，共享同一套单步前瞻流程，骨干模型 Qwen3-Coder-480B。

方法	模型类别	推理时调用大模型	Token F1	成功率
Word2World	神经 (SFT)	是	0.85	63.5%
LLM-Direct	上下文预测	是	0.64	75.7%
ReAct (无世界模型)	仅策略	是	—	74.4%
WorldCoder	程序	否	0.63	64.4%
PoE-World	程序	否	0.63	69.3%
POMDP Coder	程序	否	0.50	74.5%
PatchWorld-Residual	程序 + 残差记忆	否	0.69	72.9%
PatchWorld-Simple	程序	否	0.57	76.4%

- **宏平均成功率 76.4%**。在完全相同的候选集合与选择器下，比 PoE-World 高 7.1 个点，比 WorldCoder 高 12.0 个点。
- **预测开销为 0 token**。LLM-Direct 成功率与我们相当，但每个任务要花约 6.4 万 token 做前瞻。



- **残差记忆。** F1 从 0.57 升到 0.69，成功率却从 76.4% 降到 72.9%。
- **Word2World。** 全研究中保真度最高，0.85 F1；决策却最差，63.5%。
- **消融实验。** 去掉修复循环损失 7.9 个点成功率，去掉对比证据挖掘损失 12.9 个点，去掉结构化信念解析损失 24.5 个点。

前瞻规划需要的是与动作相关的区分度，而不是更接近的表面措辞。

Figure 3, PatchWorld. 留出测试集，七个环境宏平均。

PatchWorld 的经验

- **仅凭文本就能学到可执行的信念。**

不需要仿真器的特权状态；归纳出的程序可检视、可局部修补。

- **大模型作为优化器比作为预测器更好用。**

把它移出推理路径反而提升了决策，并彻底消除了决策时的模型调用。

- **保真度不等于效用。**

两个轴都要报告；它们构成一条前沿，而不是单一最优点。

- **尚未解决。**

信念仍是确定性的；观测读出仍占剩余误差的约 53%；程序仍按环境分别归纳，尚未跨环境共享。

PatchWorld 在自由形式的代码空间里搜索。在连续控制中这还不够 —— 程序可以解析、可以回放，物理却仍然是错的。

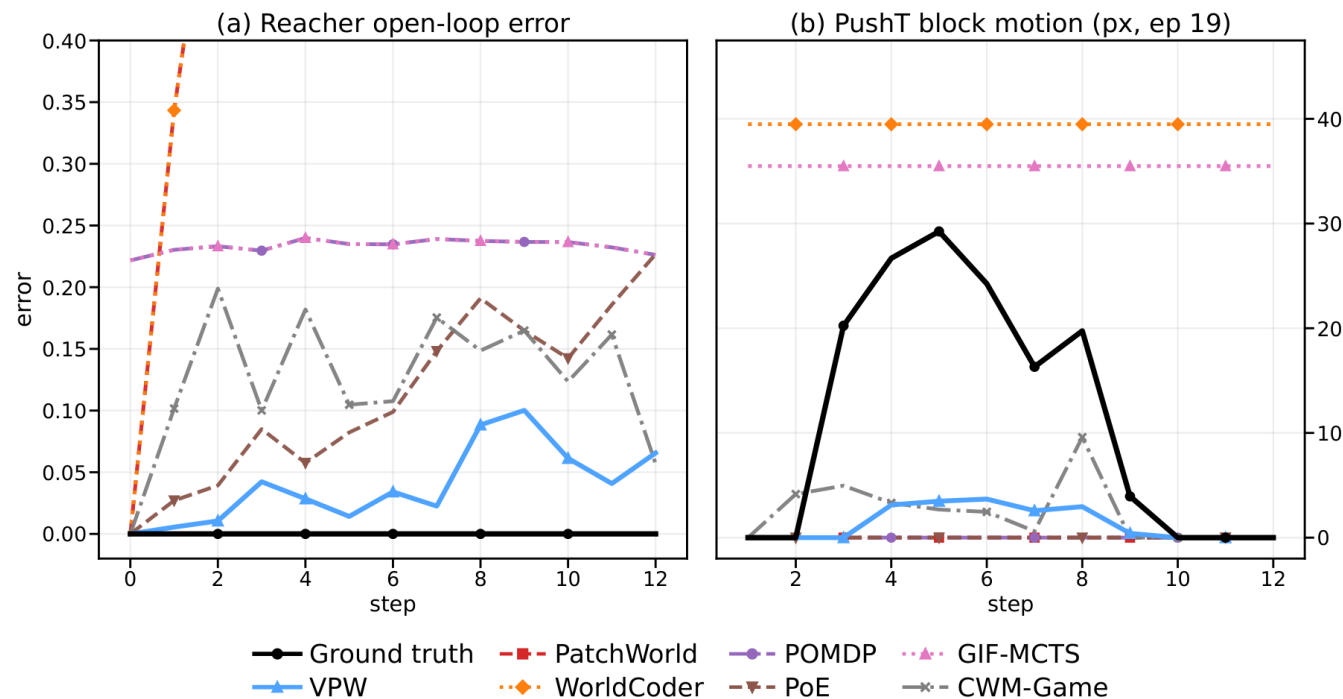
第二部分

从视觉环境 归纳动力学公式

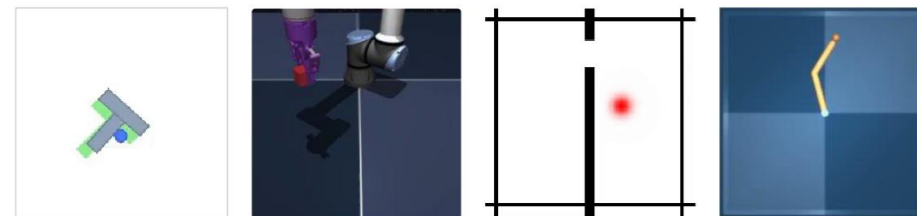
VisualPatchWorld: Code World Models as Latent Structured Representations for Planning

Jiaxin Bai, Jiaxuan Xiong · 香港浸会大学

arXiv:2607.25236 <https://github.com/HKBU-KnowComp/VisualPatchWorld>

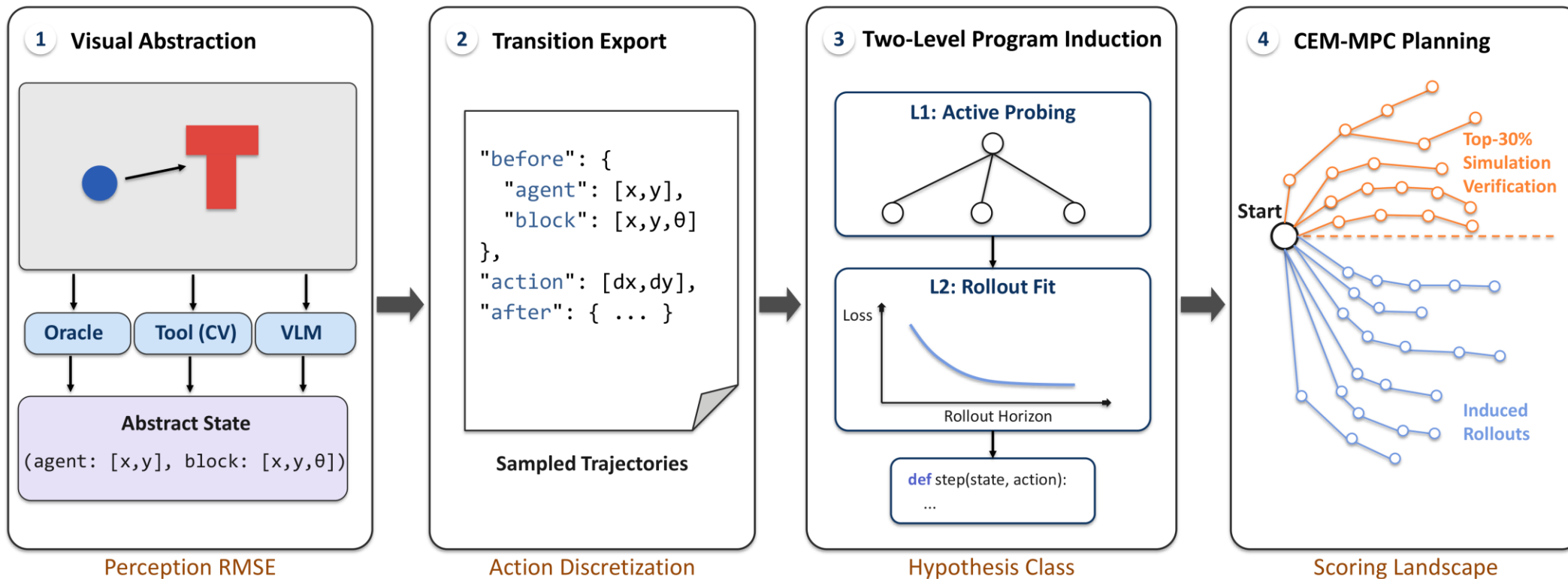


- **Reacher**。基线把指尖当作自由的质点：误差 0.86，成功率 6–8%。改用关节角加拟合的正运动学：误差 0.04，成功率 72%。
- **PushT**。有些实现让物块完全不动，有些则一步跳出几百像素。恢复出的接触位移为 16.2，真值为 140.4。

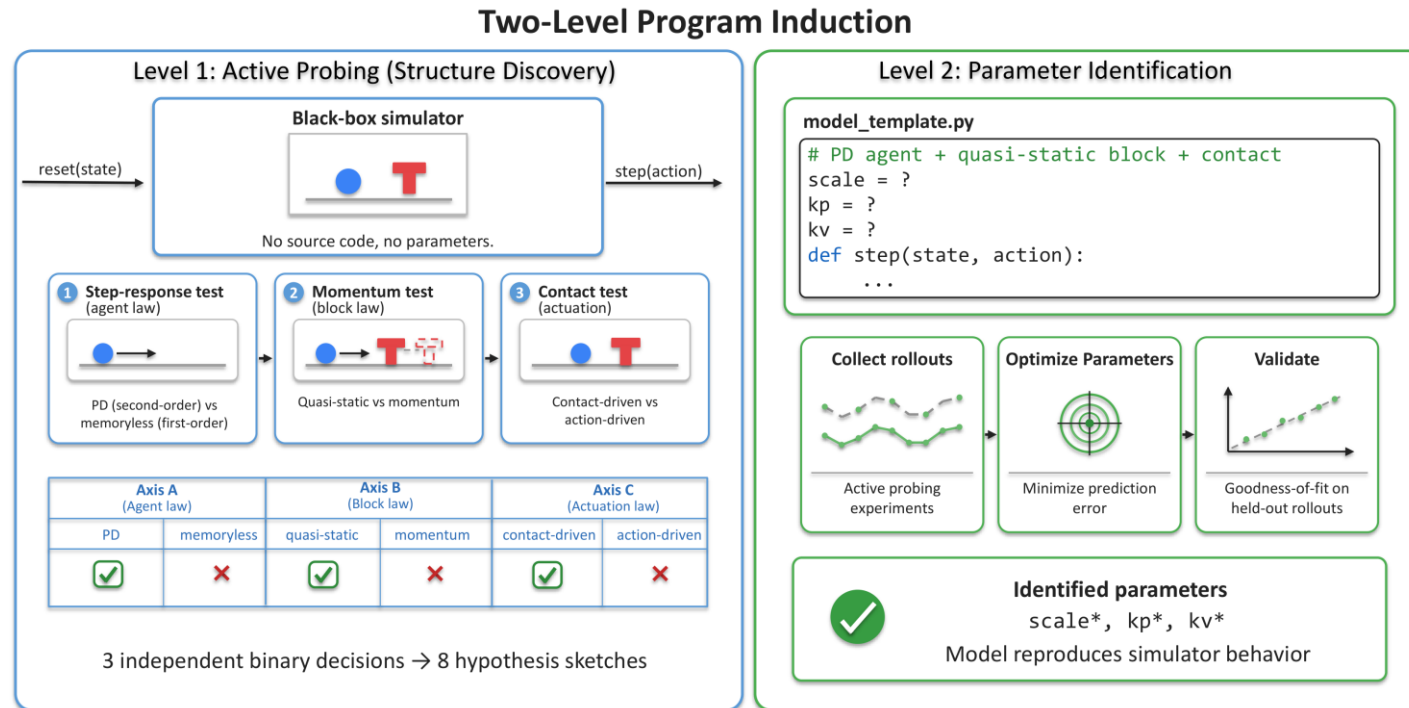


能解析、单步误差低，都不是充分条件。真正的约束是状态转移律的定性形式。

VisualPatchWorld 方法



视觉的抽象化感知、智能体动作轨迹抽取、世界模型结构归纳和用世界模型规划

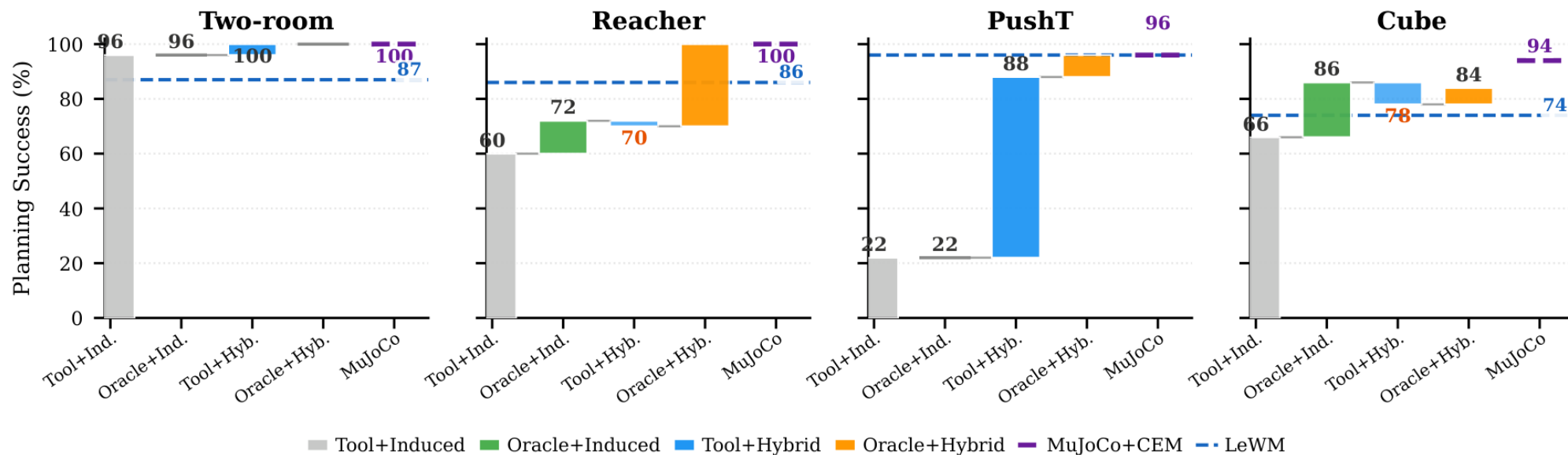


- **第一级选形式。** 通过少量探测实验，从不同的动力学公式库中选择一个最合适的公式。
- **第二级定常数。** 以规划时域上的累积误差为目标，做多次重启最小化，去拟合选定公式之中的。

相同的训练轨迹、相同的 oracle 重规划状态、相同的固定规划器。成功率 %， $n = 50$ ，随机种子 42。

方法	Two-room	Reacher	PushT	Cube	均值
LeWM (神经, RGB 参照)	87	86	96	74	85.8
DINO-WM (神经, RGB 参照)	100	79	74	86	84.8
PatchWorld	98	8	0	66	43.0
WorldCoder	60	6	0	74	35.0
POMDP-Coder	98	18	0	66	45.5
PoE-World	6	18	0	84	27.0
GIF-MCTS	6	20	0	66	23.0
VisualPatchWorld	96	72	22	86	69.0

- **领先幅度。** 比最强的程序类基线 POMDP-Coder 高 23.5 个点，比 PatchWorld 高 26.0 个点。
- **提升来自哪里。** Reacher 从 8 到 72（关节空间动力学）、Cube 从 66 到 86（抓取门控接触）、PushT 从 0 到 22（恢复接触律）。



- **归纳出的结构与环境匹配时，已经接近上限。** 在 Two-room 上达到物理引擎成功率的 0.96，在 Cube 上达到 0.91 ——但在 PushT 上只有 0.23。
- **混合打分把这个差距补上。** 归纳程序仍然搜索完整候选集，只让物理引擎对前 30% 重新打分：PushT 从 22 升到 96，套件均值达到 95.0（上限 97.5），而物理引擎的调用量减少约 70%。

VisualPatchWorld 的经验

- **结构的选择比参数精度更关键。** 在数据、规划器与起点都固定的条件下，选对形式解释了全部 23.5 个点的差距。
- **要优化规划器真正关心的目标。** 让程序在滚动时域控制中可用的，是多步展开损失，而不是单步误差。
- **感知可以与动力学干净地分离。** 用图像得到的状态，相比 oracle 状态只损失 8 个点均值成功率；而一个很强的视觉语言模型语义上正确，度量上却不可用。
- **尚未解决。** 公式草图库仍然小；动力学仍是从结构化轨迹拟合而来，而非端到端从视觉轨迹归纳；也还没有真实机器人上的评测。
 - 我们有相关的从经验发掘物理定律公式的工作: NewtonBench [1]

[1] Tianshi Zheng, Kelvin Kiu Wai Tam, Newt Nguyen Kim Hue Nam, Baixuan Xu, Zhaowei Wang, Cheng Jiayang, Hong Ting Tsang, Weiqi Wang, Jiaxin Bai, Tianqing Fang, Yangqiu Song, Ginny Wong, Simon See: NewtonBench: Benchmarking Generalizable Scientific Law Discovery in LLM Agents [ICLR'26](#)

维度	PatchWorld	VisualPatchWorld
核心问题	仅凭文本能否归纳出信念？	恢复出的动力学形式对不对？
观测	无结构的自然语言文本	图像 → 以物体为中心的场景图
潜状态	从不可见，只能推断	视觉归纳
搜索方式	开放式合成，再做反例修复	在草图族上探测，再拟合参数
用于决策	单步前瞻重排候选动作	CEM 滚动时域控制
主要结果	成功率 76.4%，前瞻 0 token	均值 69.0%，比最强代码基线高 23.5
主要结论	保真度不等于决策效用	形式错了，靠拟合补不回来

贯穿两篇工作的是同一件事：把大模型当作符号优化器，把可执行产物当作世界模型，并以决策而不是重建损失来评价它。

谢谢，欢迎提问。

白佳欣· 香港浸会大学 · baijiaxin@hkbu.edu.hk · <https://bjx.fun>

PatchWorld: arXiv:2605.30880 · VisualPatchWorld: arXiv:2607.25236

<https://github.com/HKBU-KnowComp/PatchWorld>

<https://github.com/HKBU-KnowComp/VisualPatchWorld>