SIMPACT:

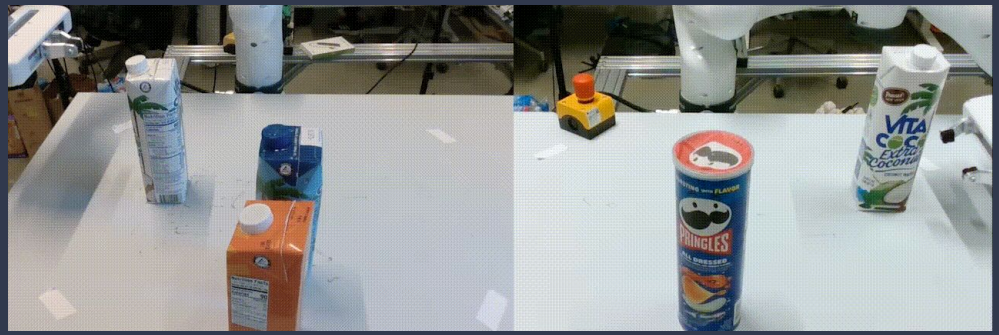
Simulation-Enabled Action Planning using Vision-Language Models

Paper by:

Haowen Liu, Shaoxiong Yao, Haonan Chen, Jiawei Gao, Jiayuan Mao, Jia-Bin Huang, Yilun Du

Presentation by:

Kieran Paranjpe



2 Parts

World Model



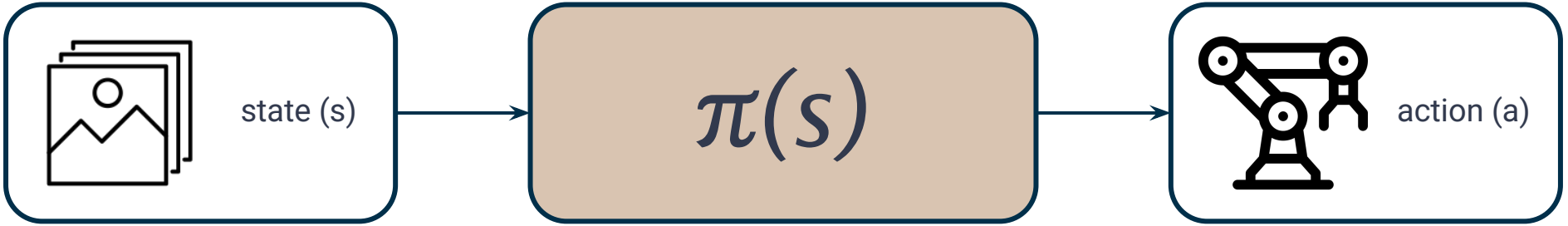
VLM planning framework



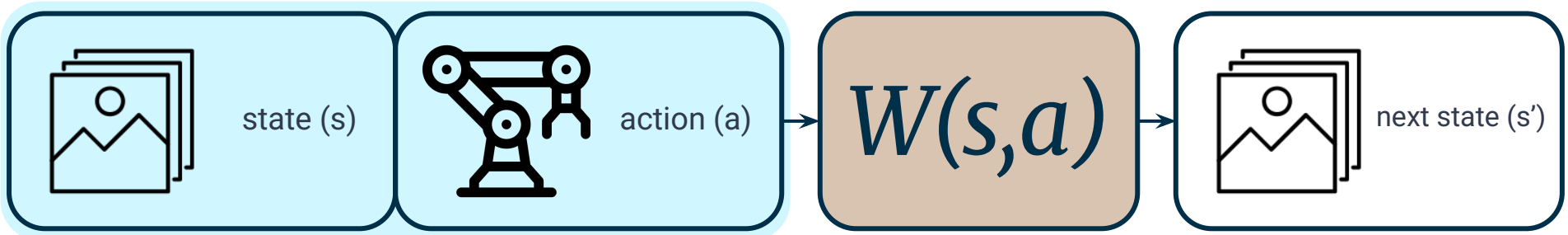
Part 1: World Model

What is a world model (abstract level)?

So far we've seen a policy:



Now we look at world models:



Why world models?

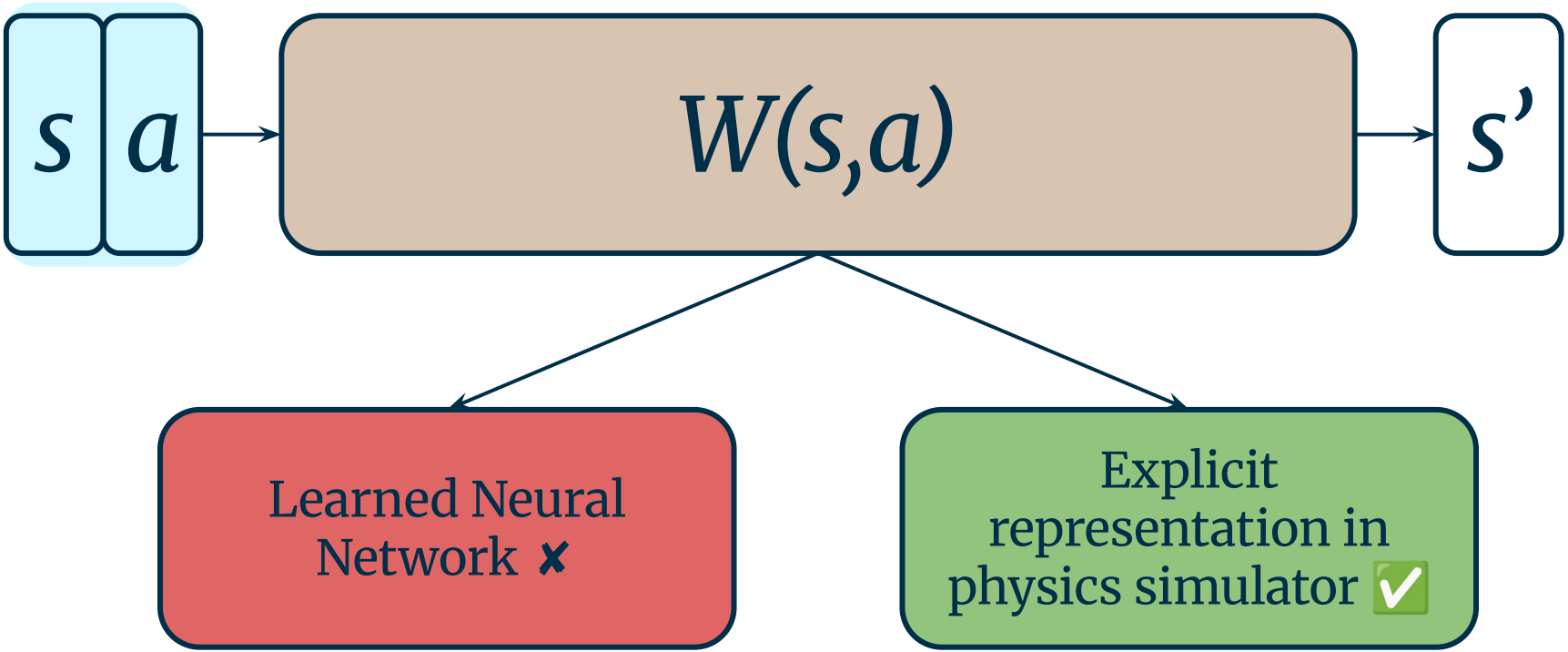
Policy

1. Tells us what to do next
2. **Does not predict** the effect of their actions before taking them
3. Run blind

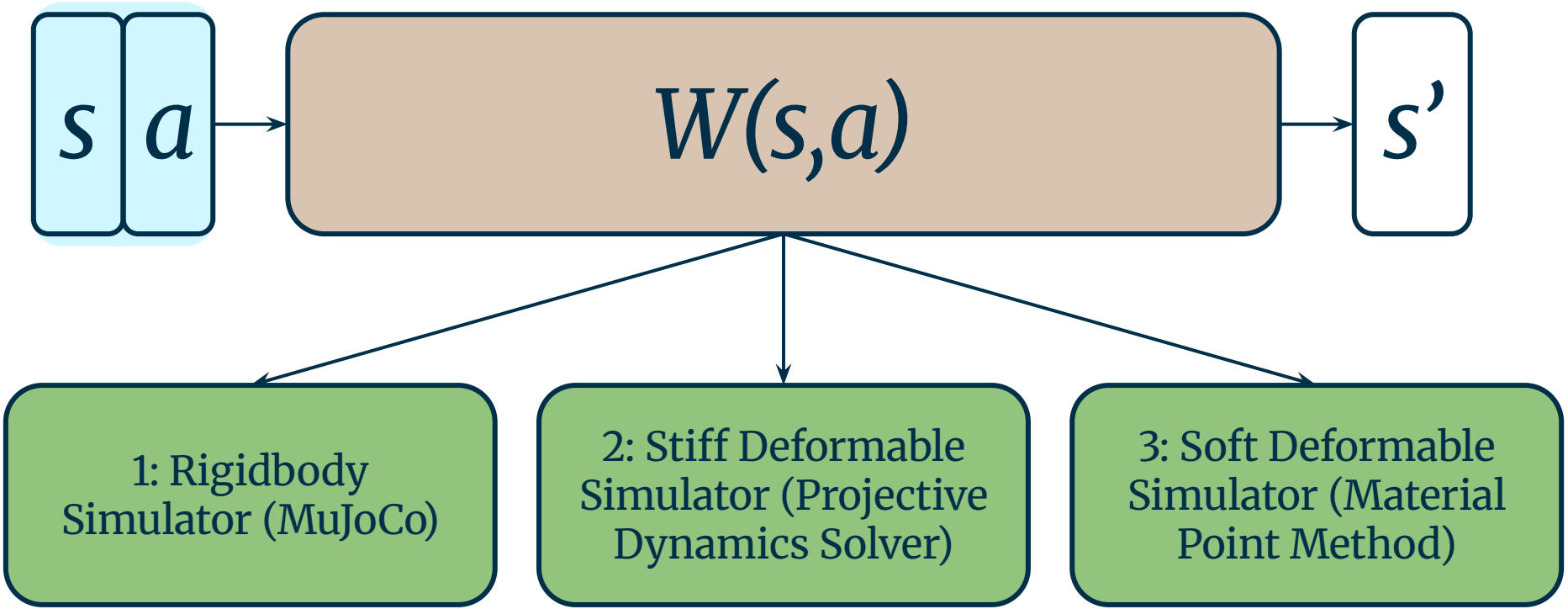
World Model

1. **Predict what might happen** if they take some action
2. Can be used **autoregressively** to predict many states into the future (feed output back into itself)
3. Do not tell us which actions to select

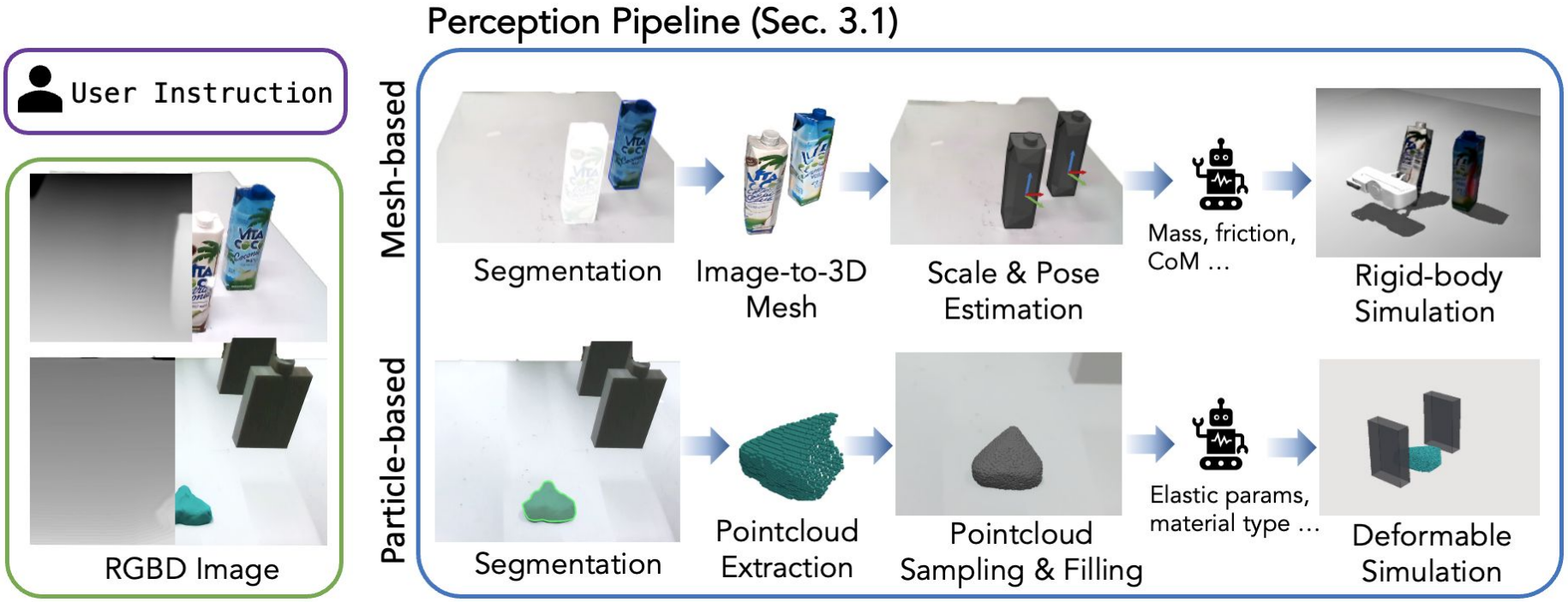
How to implement W(s,a)?



SIMPACT architecture



The main work is how to setup the simulator



Some more details

Rigid Body

1. Requires **Image-to-3D-mesh** model
2. Uses VLM to estimate physical properties of rigidbodies (mass, friction, center of mass). Based on “common sense”.
3. Each mesh is scaled/centered and then placed in estimated position according to **FoundationPose**

Deformable

1. Fills in 3d points it can’t directly see with **backprojection**
    a. Assume the hidden side of the object extends down towards the ground. Uniformly sample points in this region.

Some benefits to this method

1. No need to teach model about laws of physics
2. Exploit pre-existing simulators
3. Can explicitly understand the simulation
4. Has potential in environments we know a lot about (warehouse, factory, etc)

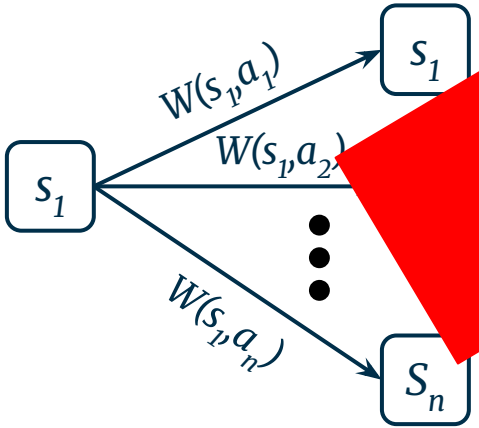
Potential problems to think about

1. Is **Image-to-3D-mesh** model good?
2. How to model scenes with both deformable and rigidbody?
3. Can we trust VLM to estimate physical properties?
4. Requires explicit and certain representation of everything (VLM estimates mesh and physical properties)
5. Computation time to build sim
6. Assumes static environment

Part 2: VLM planning framework

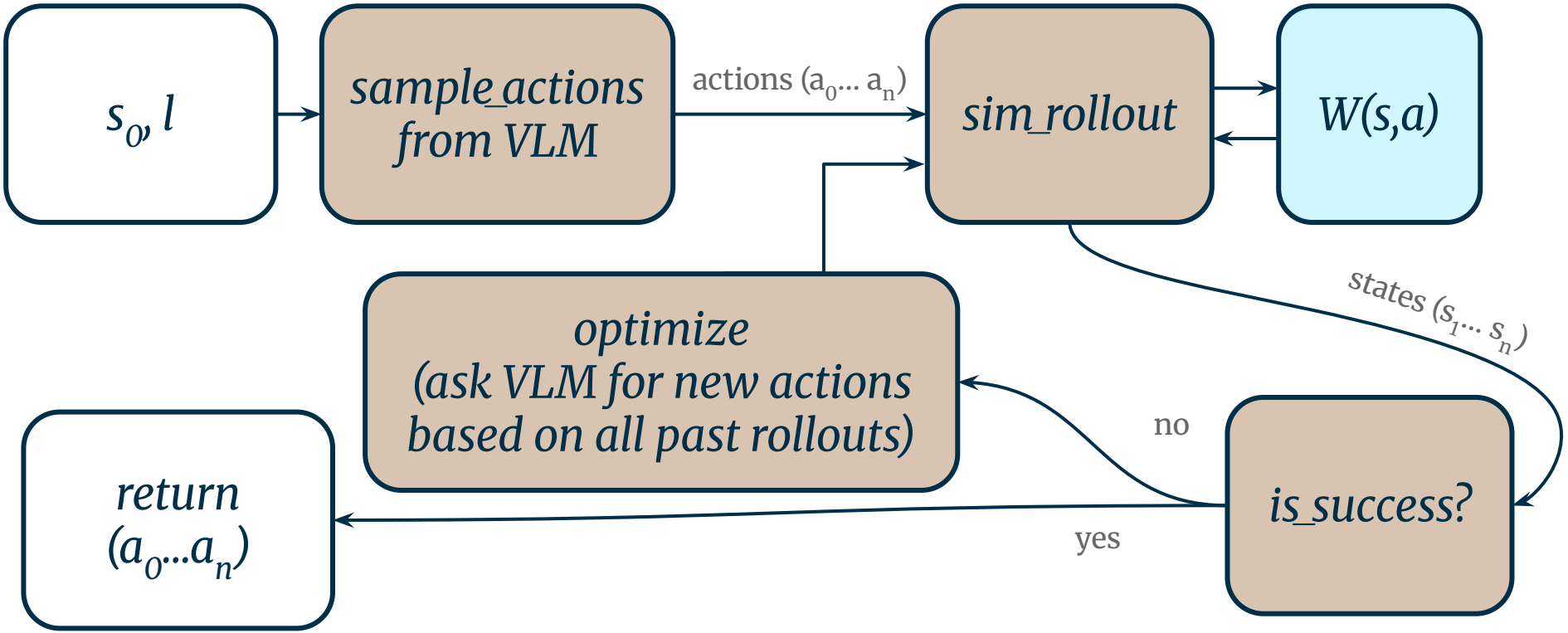
The alternative

Once we have a w… ight be tem… ssical AI planning algorithm to plan … as a search problem:



… model, we could use some … etc) or some more … lgorithm to select actions.

Use the VLM to plan



Analogy to a coding agent

Coding Agent

1: Prompt LLM to write some code. Maybe include requirements / completion criteria.

2: LLM writes some code. Probably buggy.

3: LLM writes and runs some tests, based on own criteria, or user provided criteria.

4: LLM uses test output to modify code and repeats until it works, or is cancelled.

SIMPACT planner

1: Prompt VLM/robot to complete a task. Maybe include requirements / completion criteria

2: VLM comes up with a set of possible action sequences. Probably don’t work perfectly.

3: Construct world model (based on VLM + more), rollout action sequences to test what happens.

4: VLM uses state sequence rollout information to improve on new action sequences. Repeat until works or timeout.

Some benefits to this method

1. Can be used regardless of how we choose to implement W(s,a)
2. Use “common sense” reasoning for improving on action sequences
3. Can learn/reason about why certain actions led to failure by examining past rollouts

Potential problems to think about

1. Planning takes time
2. Plan whole trajectory at once, do not resample during run
    a. Can’t recover from failure mid-run
    b. Would need to remake simulation to do this

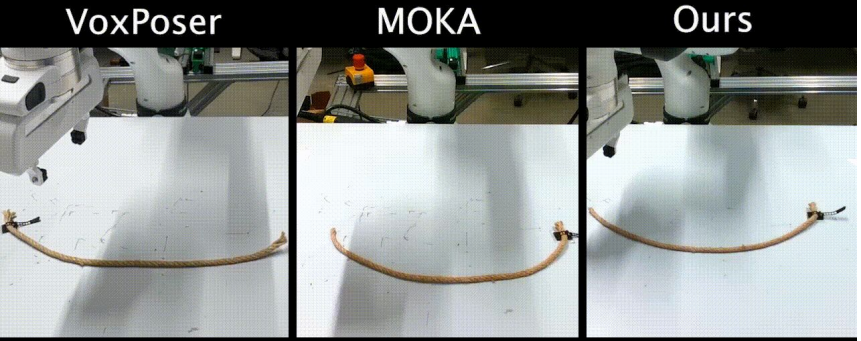
Results & Experiments



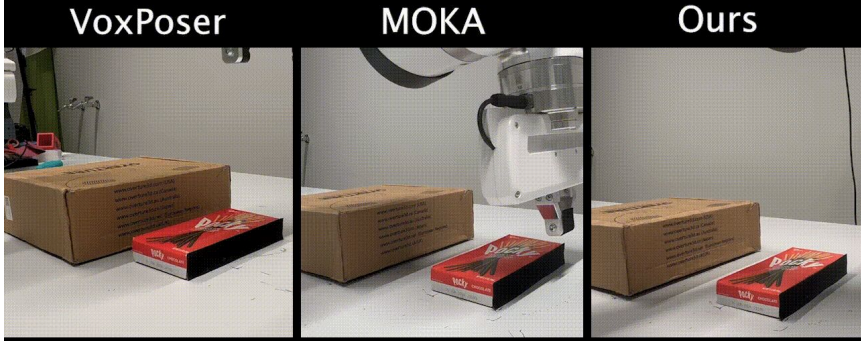
Task: Push white carton forward to align with blue carton



Task: Stack pink bowl with blue bowl



Task: Shape rope into a ‘U’



Task: Lean the Pocky box against the brown box

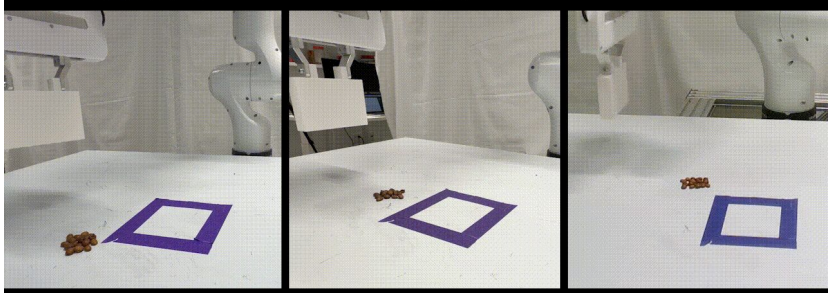
Experiments (1)



Task: Shape playdough into square



Task: Move bottle around obstacle



Task: Sweep coffee beans into box

Experiments (2)

Results Table

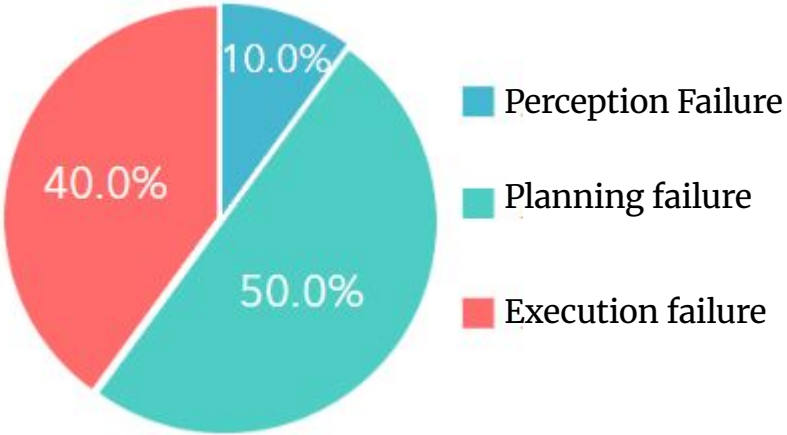
Method	Non-toppling push	Bowl stacking	Lean on box	Shape rope	Shape dough	Avoid obstacle	Sweeping
$\pi_{0.5}$	0%	0%	0%	0%	0%	0%	0%
VoxPoser	0%	20%	0%	0%	0%	0%	20%
MOKA	0%	10%	0%	20%	0%	0%	0%
SIMPACT	80%	60%	40%	90%	80%	80%	70%

*** $\pi_{0.5}$, VoxPoser and MOKA are all VLA / VLM flavours. No comparison to other world models.**

Looks good, right?

Avg. Computation Time: 10 samples / task	
Component	Time (mins)
Simulation Construction	1.9
Action Sampling	2.8
Simulation rollout	0.8
Action Optimization	0.9
Total	6.4

Failure Case Breakdown:



They also compare to classical planning and said it has near 0% success rate.

Problems

1. They don’t compare to any other world model solutions
    a. Does it make sense to use this method over learned world models?
2. Experiments are not cluttered or realistic
3. Long computation time
4. Unclear if they fine tuned any comparison models or planning methods

My Takeaway

1. World models are effective for improving success rate
2. No clear argument that explicit world models are better than learned world models
3. Not clear that this method will scale to more complex / real situations

Thank you

Questions?