# Towards Efficient Assembly Motion Planning via Vision-Language Models

Team 5
Jung Geumyoung, Kim Jaemin

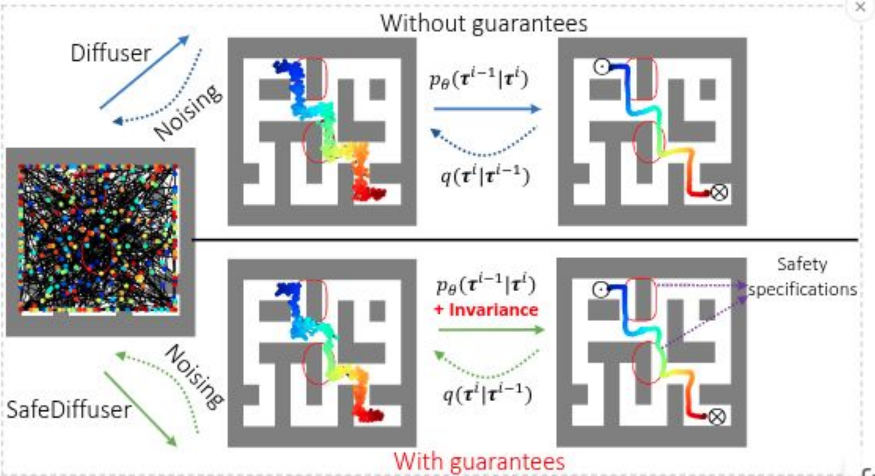
KAIST

# **SafeDiffuser** : Safe Planning with Diffusion Probabilistic Models

**Motivation** : Diffusion model-based approach do not guarantee safety

**Key Idea :**

- Diffuser + Control Barrier Functions = SafeDiffuser
- embed finite-time diffusion invariance into denoising procedure using a class of control barrier functions to ensure safety




KAIST

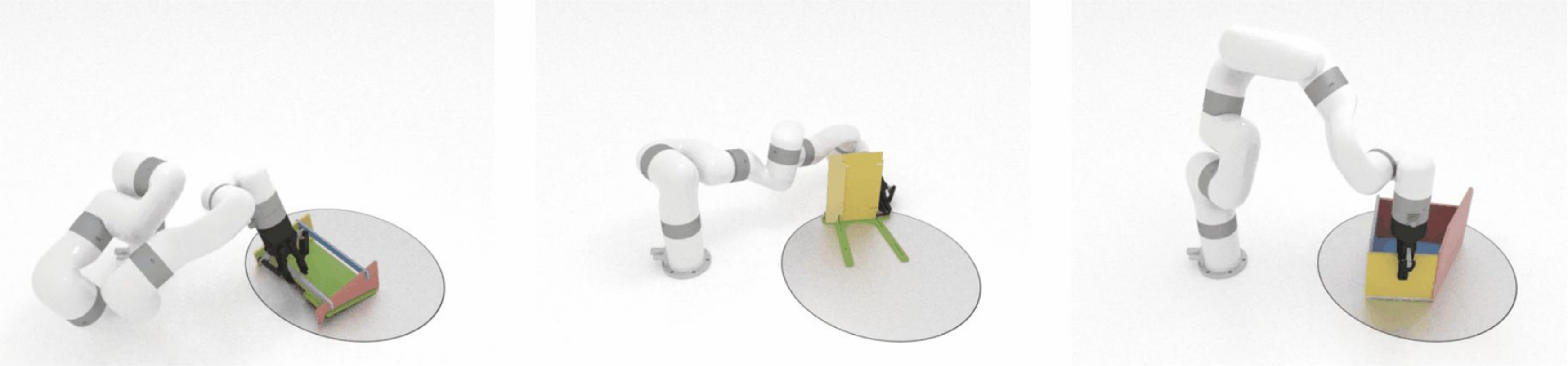
# Contents


- Introduction
- Challenges
- Method
- Experiments
- Conclusion

# Introduction

**Assembly part motion planning :**

process of planning collision-free path of a part to move and join individual components into final assembled product

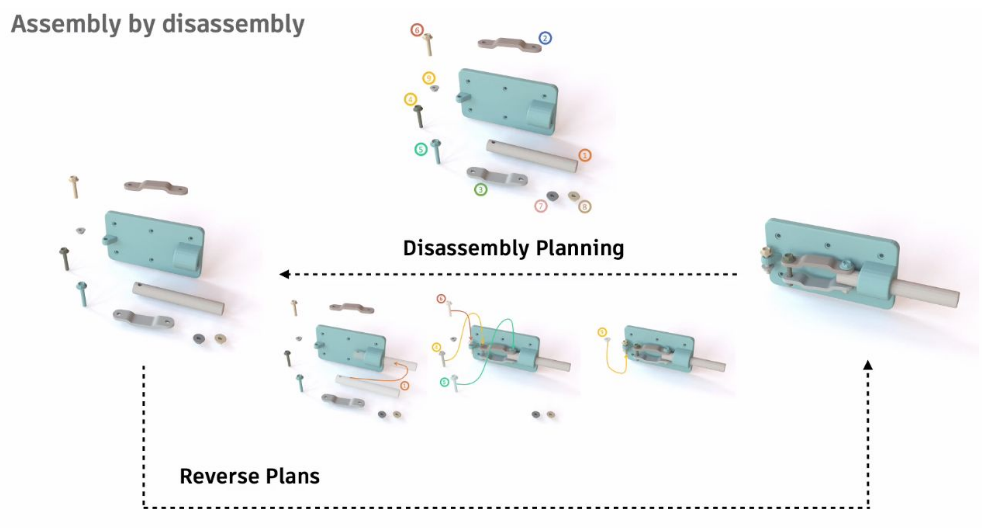


# **Introduction** | Assembly by Disassembly

Assembly of n parts → **O(n!)** search space

Assume all parts are **rigid bodies**, then

- Assembly = reversed disassembly
- Disassembly of n parts → **O($n^2$)** search space (efficient!)

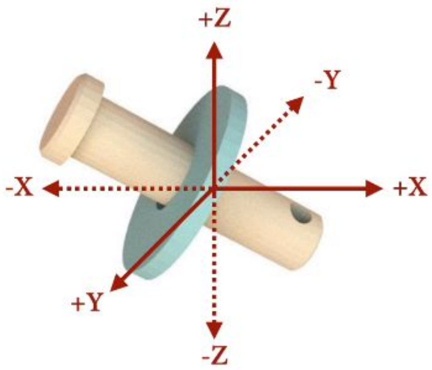


# **Introduction** | Baseline

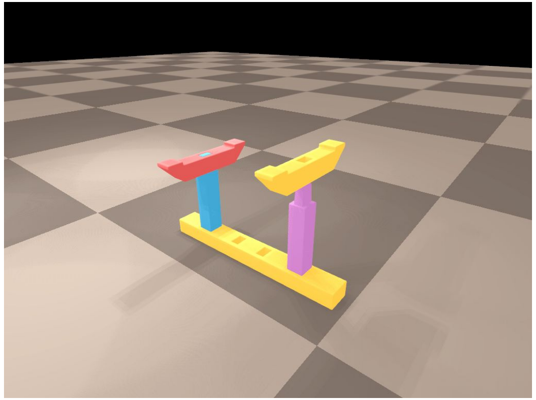
Automated Sequence Planning for Complex Robotic Assembly with Physical Feasibility

ASAP's method for Part motion planning :

- Samples discretized actions (forces) in 6 directions
- Apply the actions in **physics-based simulation**, check disassembly success



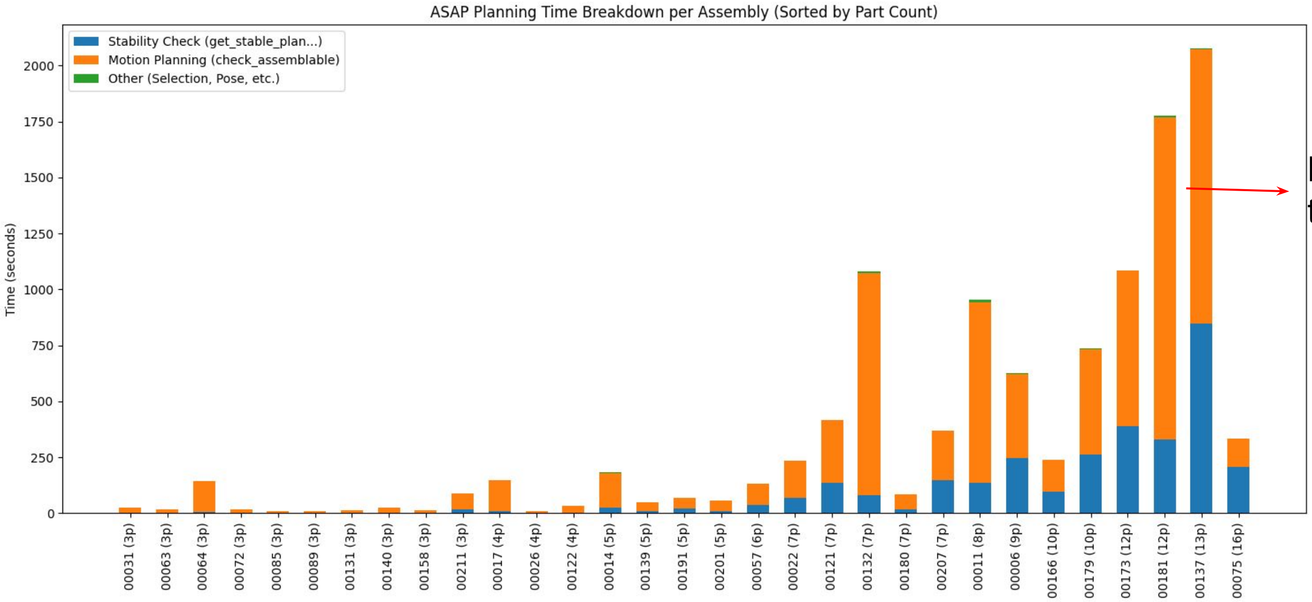
6 actions


+Z applied on assembly

# **Challenge** | Running Time

Physics-based simulation comes with **high computation costs**

Main functions: check_assemblable (**motion planning**), get_stable_plan (**stability check)**




Part motion planning (in orange) is the most time-consuming task

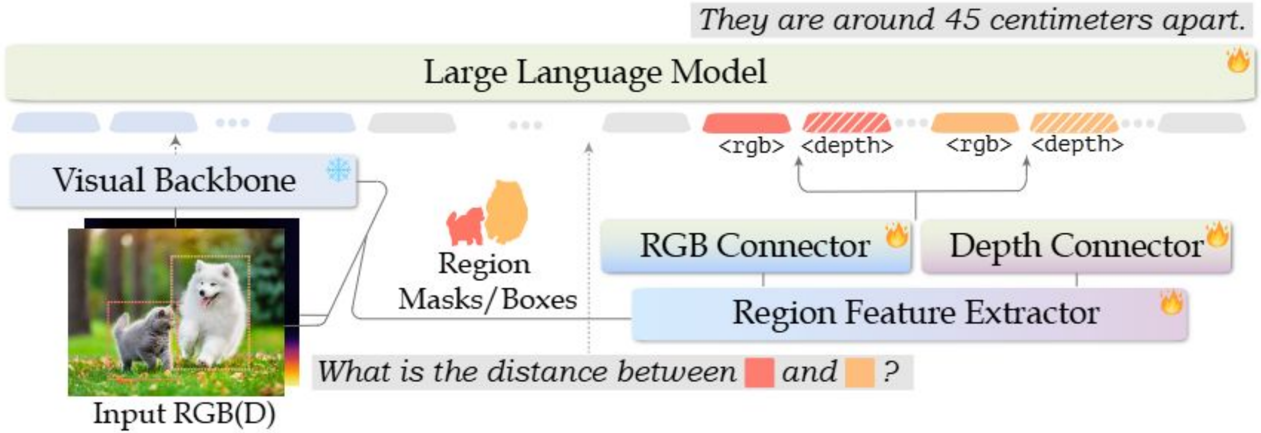
thought : can’t we do better than randomly sampling actions?

# **Method** | Approach

**less action trials(accurate action prediction), Reduce computation time**

Utilize **Spatial reasoning capable FM** to guess initial disassembly force direction

Reduce the number of physics-based action queries to **speed up the motion planning**

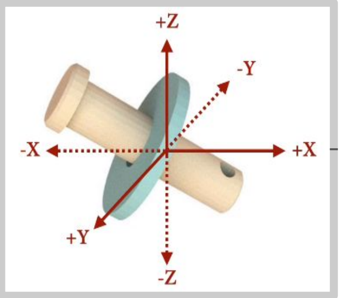



Requires memory > 24GB for inference

ChatGPT API

GPT-4o

# **Method** | System Overview




**[ Input ]**
Picture of an assembly

**[ Input ]**
text prompt

In 3D space,
give me **action vector** to
pull **magenta part**
out of **cyan part**

**Spatial Reasoning Foundation Model**

**[ Output]**
predicted action vector

[x, y, z] = [1, 0, 0], [0, 0, -1]

Prioritize actions
closer to prediction

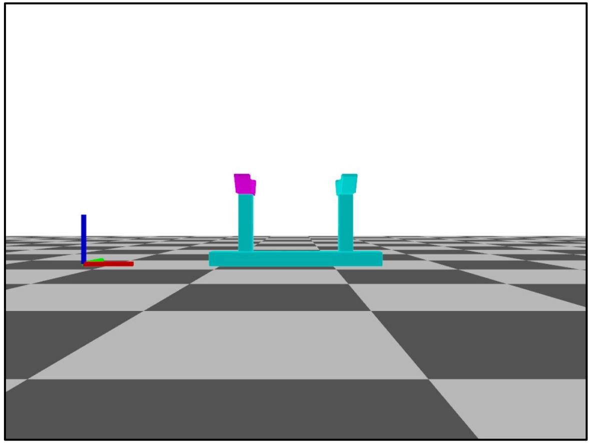
physics-based simulator

Try [1, 0, 0], [0, 0, -1]

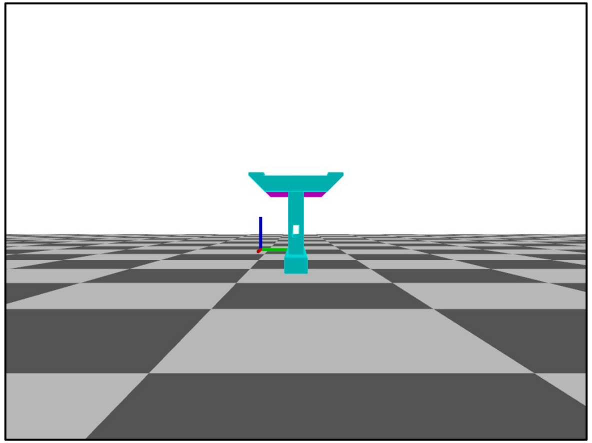
# **Method** | Action Prediction via VLM

Given an **assembly G** and a **target part P** to remove from G:

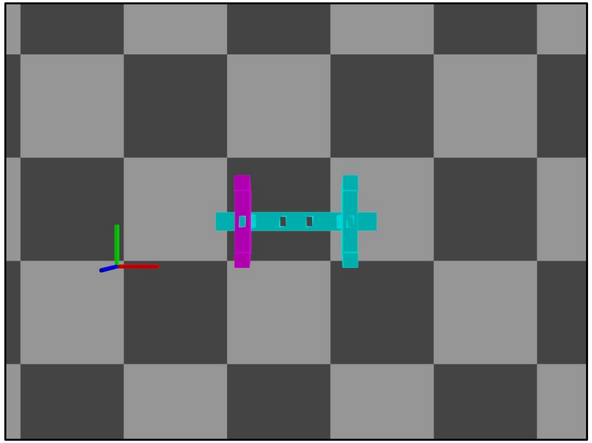
1. Render G, with **P distinctly colored** (e.g. magenta) in **3 principal views**
2. Pass the 3 images + text prompt to the VLM
3. VLM will rank actions based on predicted (disassembly) success rates



Front view



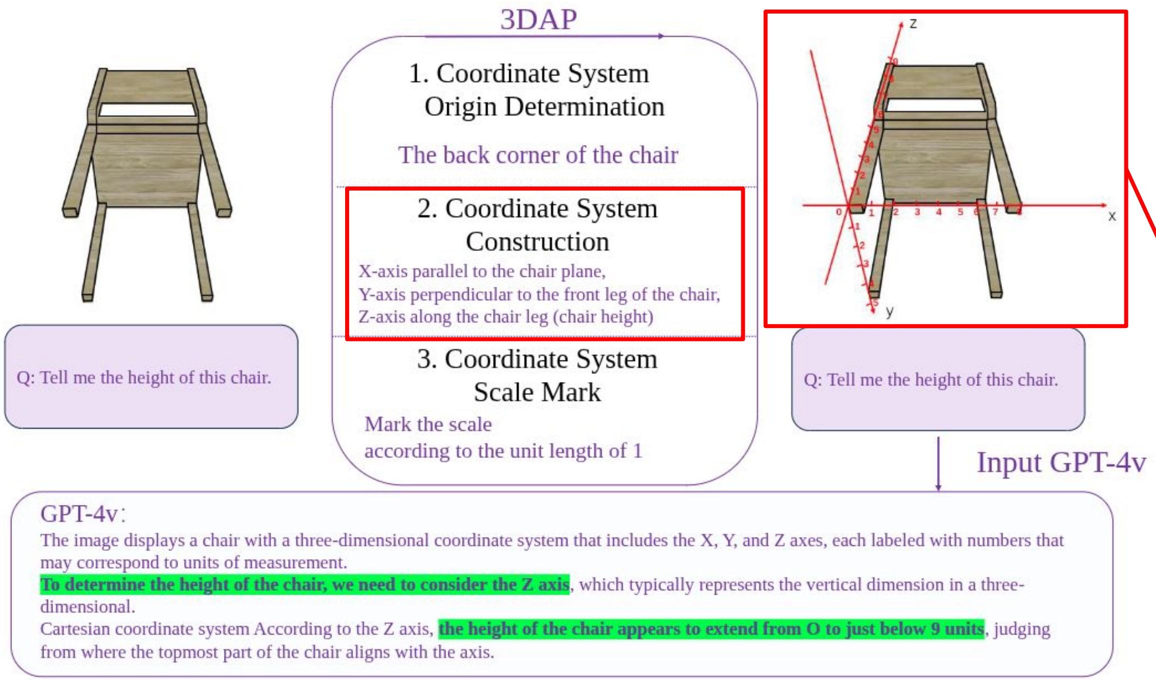
Side view

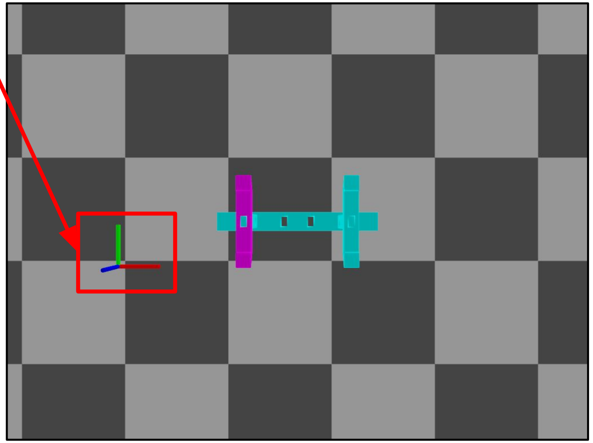


Top view

# **Method** | Action Prediction via VLM

Constructing **coordinate axis** can improve VLM's spatial reasoning capability



Top view

Can LLM be a Good Path Planner based on Prompt Engineering? Mitigating the Hallucination for Path Planning (2024.08. arXiv)

# **Method** | Action Prediction via VLM

## Text instruction for spatial reasoning

```
"You are an expert in spatial reasoning and mechanical assemblies."


"You are given three orthographic views (front, side, and top) of a 3D mechanical assembly."
"One part is distinctly colored magenta and is designated as the target part to disassemble."
"All other parts are rigidly fixed in place."
"A 3D coordinate system is overlaid consistently across all views for spatial reference:"
"Red arrow: +X direction (right)"
"Green arrow: +Y direction (forward, into the page)"
"Blue arrow: +Z direction (up)"

"Based on these views and axes:"
"Analyze the spatial configuration of the magenta part relative to its neighboring fixed parts."
"Identify potential occlusions, constraints, and free space in all six axis-aligned directions."
"Rank the six directions by the likelihood of successful disassembly, from highest to lowest."

"Allowed Directions:"
"+X (right), -X (left), +Y (forward), -Y (backward), +Z (up), -Z (down)"
"Output Format:"
"Respond with a single line of six signed directions, comma-separated,
“with no explanations, no extra text, and no newlines."
"Example: +Y, -Z, +X, +Z, -X, -Y"
```

**#1 set role**

**#2 explains input images :**
color, physical environment,
reference axis

**#3 Define a goal action :**
rank the six directions

**#4 Define output format :**
solely 6 directions

# Experiments

**Hypothesis : Successful disassembly within less trials, Reduced computation time**

**1. Verify VLM's 3D spatial reasoning**

Actions sampled, planning time

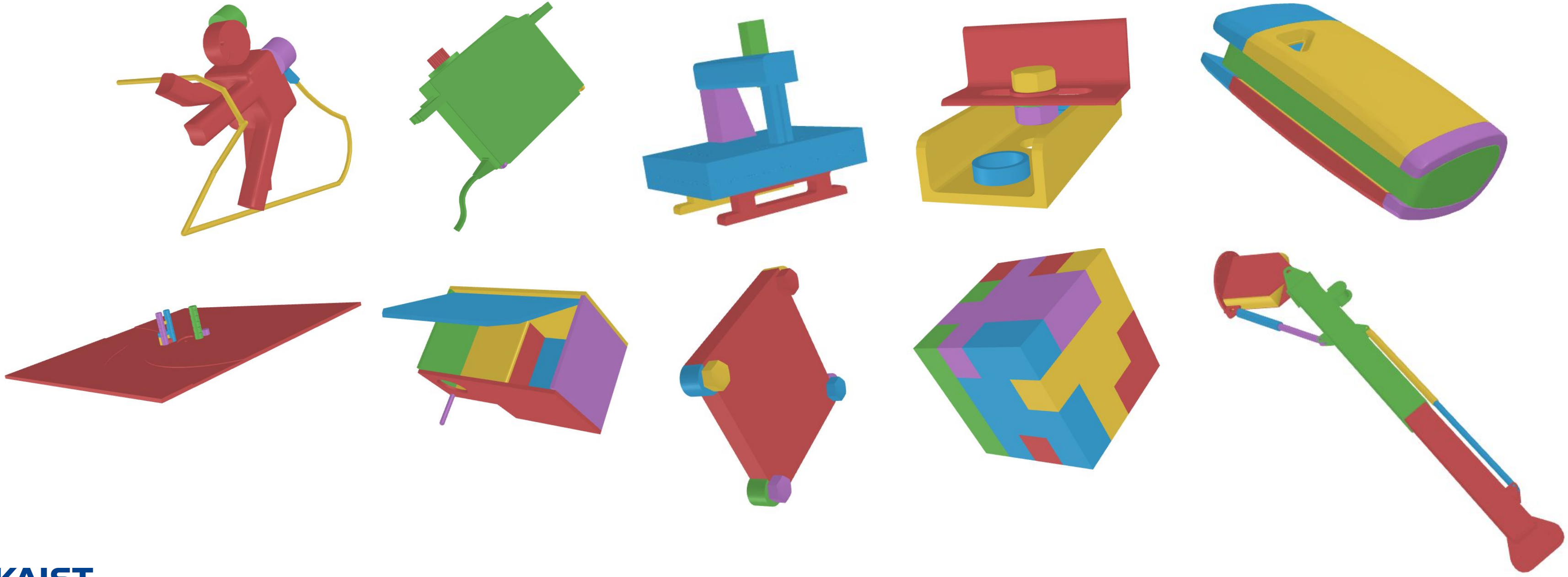
**2. VLM capability on narrow-passage problem**

Actions sampled & time taken

under different passage narrowness

# **Experiments** | Settings

Dataset : **46 assemblies from ASAP (~250 part motion plans)**



# Experiments

**Hypothesis : Successful disassembly within less trials, Reduced computation time**

**1. Verify VLM's 3D spatial reasoning**
Actions sampled, planning time

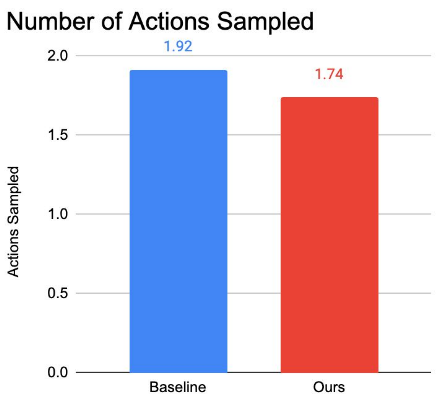
**2. VLM capability on narrow-passage problem**
Actions sampled & time taken
under different passage narrowness

# **Experiment 1** | Verifying VLM's 3D reasoning capability

<u>Q: "Can VLM perform 3D spatial reasoning from image snapshots?"</u>

- Actions sampled: avg. number of actions sample per disassembly
- Planning time: avg. time taken per disassembly

Successful disassembly within fewer actions sampled (& took less time)




| Baseline | Ours | Improvement |
|---|---|---|
| 1.9±1.28 | 1.74 ±1.32 | **+9.90%** |

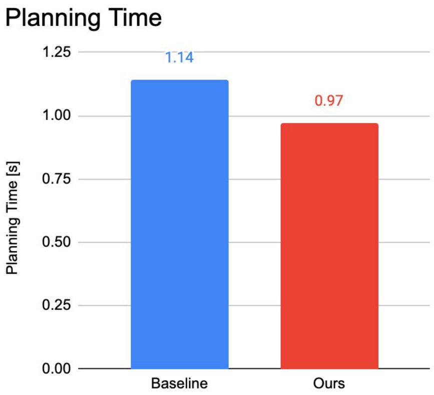
KAIST

# **Experiment 1** | Verifying VLM's 3D reasoning capability

<u>Q: "Can VLM perform 3D spatial reasoning from image snapshots?"</u>

- Actions sampled: avg. number of actions samples per disassembly
- Planning time: avg. time taken per disassembly

Successful disassembly within fewer actions sampled (& took less time)




| Baseline | Ours | Improvement |
| --- | --- | --- |
| 1.14±1.63 | 0.970±1.45 | **+17.50%** |

KAIST

# Experiments

**Hypothesis : Successful disassembly within less trials, Reduced computation time**

**1. Verify VLM's 3D spatial reasoning**

Actions sampled, planning time

**2. VLM capability on narrow-passage problem**

Actions sampled & time taken

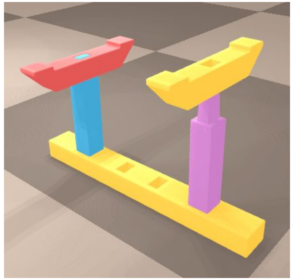
under different passage narrowness

# **Experiment 2** | Motivation

Spatial reasoning shall shine, especially under **disassemblies with tighter constraints** (i.e. narrower passages)

Given access to VLM with ‘good’ spatial reasoning capabilities:

- Random sampling will result in even lower success rate
- Performance improvement will be more significant



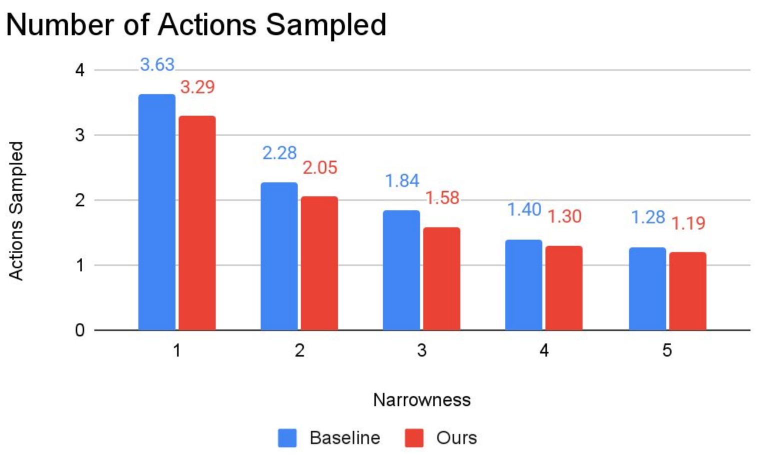
Only +Z force is allowed

Disassembly motion with a narrow passage.

# **Experiment 2** | Capability on narrow-passage problem

Q: “Does VLM perform better / worse on tasks with **narrow passages**?”

- Measured actions sampled, planning time
- **Narrowness** of a passage: how many out of 6 actions leads to success?
  - Lower values → more difficult to disassemble
- **Number of actions tried reduced** across different difficulties



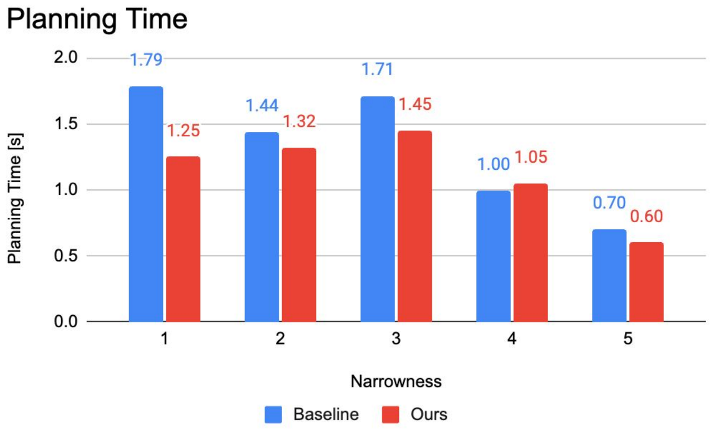

| Narrowness | Baseline | Ours | Improvement |
|---|---|---|---|
| 1 | 3.63±1.72 | 3.29±1.94 | **+10.4%** |
| 2 | 2.28±1.15 | 2.05±1.48 | **+11.0%** |
| 3 | 1.84±0.90 | 1.58±0.77 | **+16.7%** |
| 4 | 1.40±0.62 | 1.30±0.60 | **+7.70%** |
| 5 | 1.28±0.45 | 1.19±0.40 | **+7.60%** |

KAIST

# **Experiment 2** | Capability on narrow-passage problem

Q: "Does VLM perform better / worse on tasks with **narrow passages**?"

- Measured actions sampled, planning time
- **Narrowness** of a passage: how many out of 6 actions leads to success?
  - Lower values → more difficult to disassemble
- **Planning time had much higher variance** (simulation cost is inconsistent)



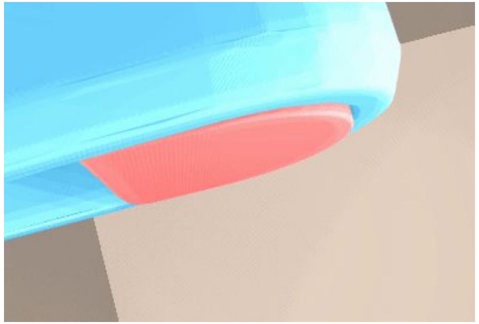

| Narrowness | Baseline | Ours | Improvement |
|---|---|---|---|
| 1 | 1.79±1.98 | 1.25±1.15 | **+42.8%** |
| 2 | 1.44±1.82 | 1.32±2.23 | **+8.95%** |
| 3 | 1.71±2.06 | 1.45±1.62 | **+18.0%** |
| 4 | 1.00±1.48 | 1.05±1.58 | **-5.34%** |
| 5 | 0.70±1.20 | 0.60±0.92 | **+16.7%** |

Avg. time per simulation = 0.71**±1.82** s

KAIST

# **Experiment 2** | Outlier Cases

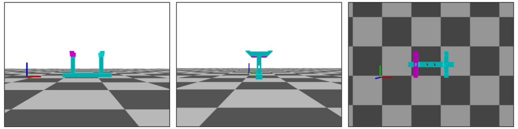
- Some simulations took unreasonably long computation time
  - Caused high variance in planning time, independent of our VLM method



A simulation taking ~60s to determine failure

# **Conclusion** | Limitations & Future Works

**Prediction Accuracy**



- 3 principal views may be insufficient (prone to occlusion)
- Sophisticated prompt engineering may improve prediction accuracy

VLM **Inference Overhead**

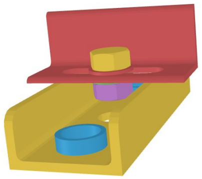
- Used GPT-4o API for testing (~2-3s overhead per query)
- Try faster models
- Try open-sourced model to deploy in-house

ChatGPT API

Costs ~1¢ per query

Further Improvements in **Running Time**

- Try VLM on other assembly problems, e.g. part selection (Q. which part should be disassembled first?)



KAIST

# **Conclusion** | Key Takeaways

VLM can perform 3D spatial reasoning on part motion planning

- **Achieves higher action success rate** compared to random sampling method

Reduced number of call is **related** to less overall computation time

- Yet, **variance within a simulation** is also a significant factor

# Thank you!

# Schedule & Roles

O : main

V : support

| | | Geumyoung Jung | Jaemin Kim |
|---|---|---|---|
| Have Done | Analyze previous work | O | O |
| | Test Reference code | O | O |
| | VLM background research & prompt engineering | O | V |
| | Integrating VLM into pipeline | V | O |
| | Testing & collecting results | O | O |
| | Prepare final presentation | O | O |