Agent Robotics: Learning-from-observation

Katsu Ikeuchi
Applied Robotics Research
Microsoft

Learning-from-observation (LfO)

Task model
(Minsky's frame)

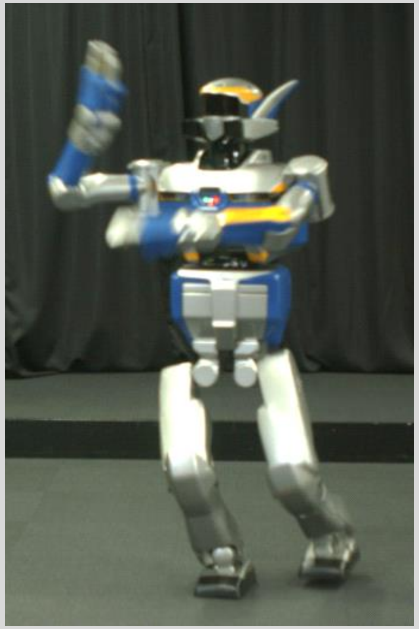
Encoding

Skill agents

What-to-do

Where-to-do

How-to-do



Not direct mimicking

Teaching mode

Execution mode

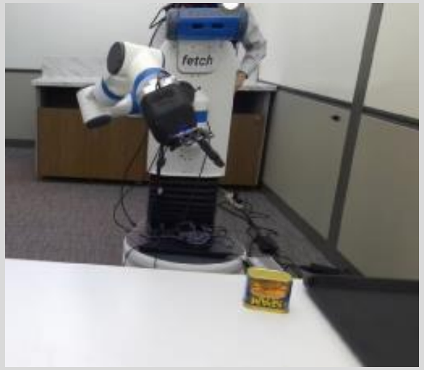
LfO (indirect mimicking)

Task models
What-to-do
where-to-do

Human demonstration

GPT-based encoder (cerebrum)

RL-trained skill agents (Cerebellum)



Robot execution

- **GPT-based encoder observes human demonstration and encode them as task models**
- **Skill agents corresponding to task models mimics the demonstration using where-to-do**

Pre-requisite for GPT-encoding

- GPT knows the collection of skill agents available

Library of grasp-skill agents

- Passive Form skill agent
- Passive Force skill agent
- Active Force skill agent


Closure theory

Library of manipulation-skill agents

- Pick skill agent
- Drawer-open skill agent
- Door-open skill agent
- Bring skill agent
- Drawer-adjust skill agent
- Door-adjust skill agent
- Place skill agent
- Drawer-close skill agent
- Door-close skill agent
- …



Kuhn-Tucker theory

GPT-Encoder (Ver23Sep): verbal + visual

Verbal instruction "Grasp a red cup"

Visual demonstrations

GPT

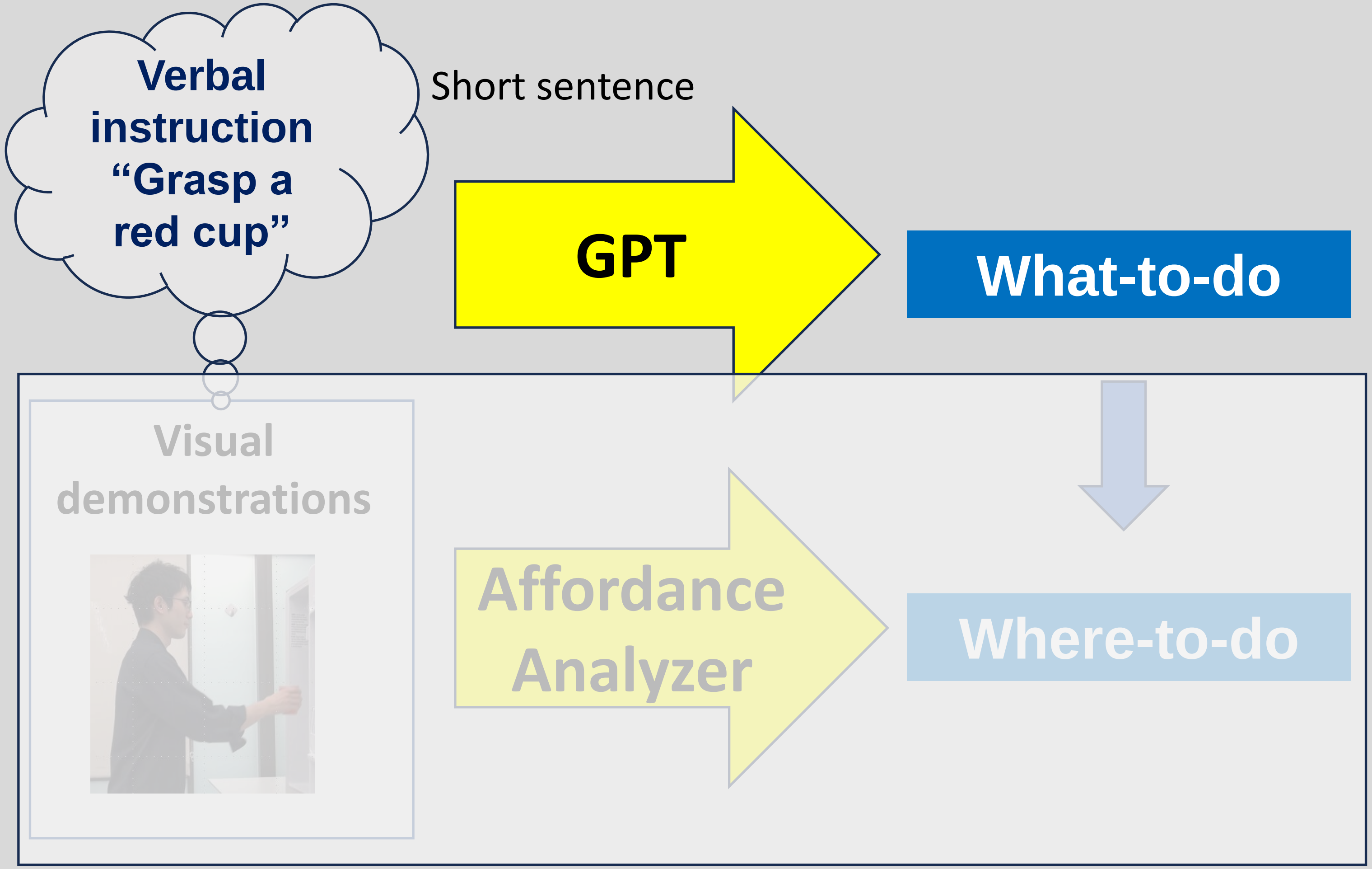
Affordance Analyzer

What-to-do

Where-to-do

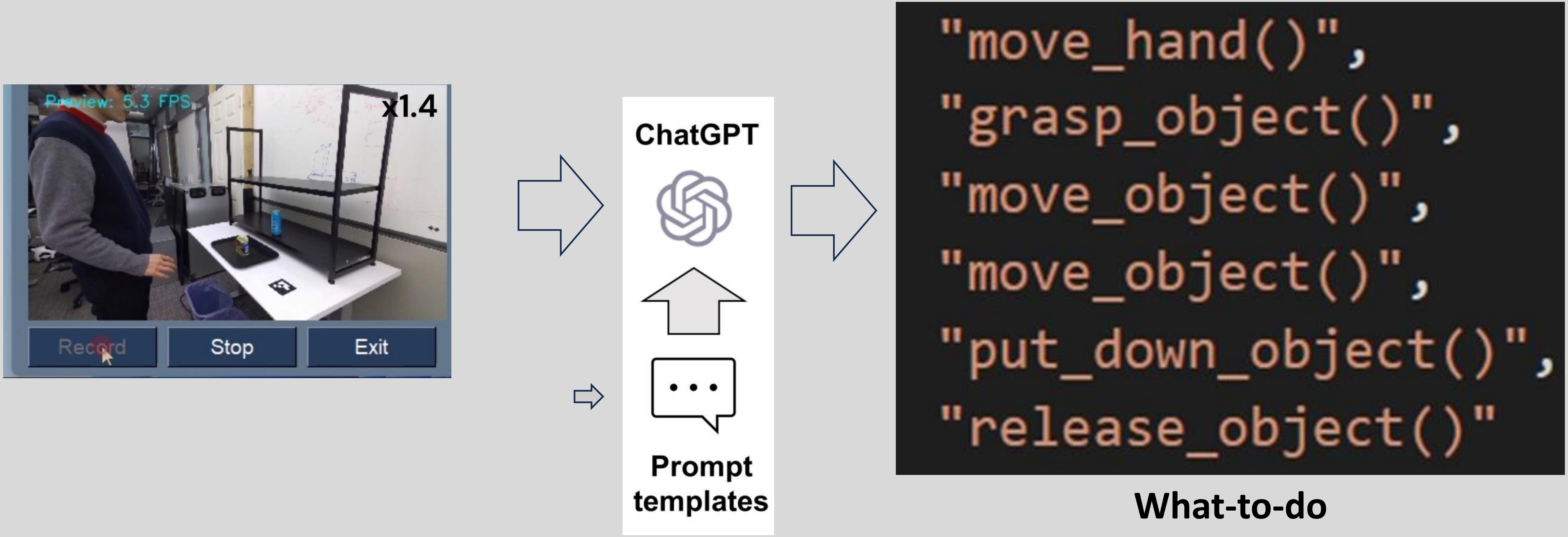
Wake2023chatgpt-IEEE-ACCESS

GPT-Encoder (Ver23Sep): verbal + visual



ChatGPT to get what-to-do

ChatGPT can generate a sequence of what to do (step-by-step action primitives) from natural language input



GPT-Encoder (Ver23Sep): verbal + visual

Verbal instruction "Grasp a red cup"

GPT

What-to-do

Visual demonstrations

Affordance Analyzer

Where-to-do

Stop & go teaching

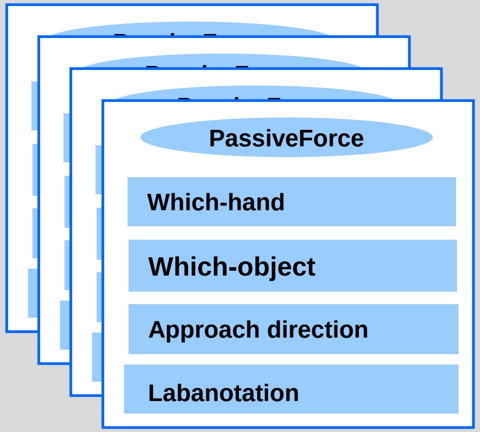
Affordance analyzer

GPT-output

Retrieved Task models

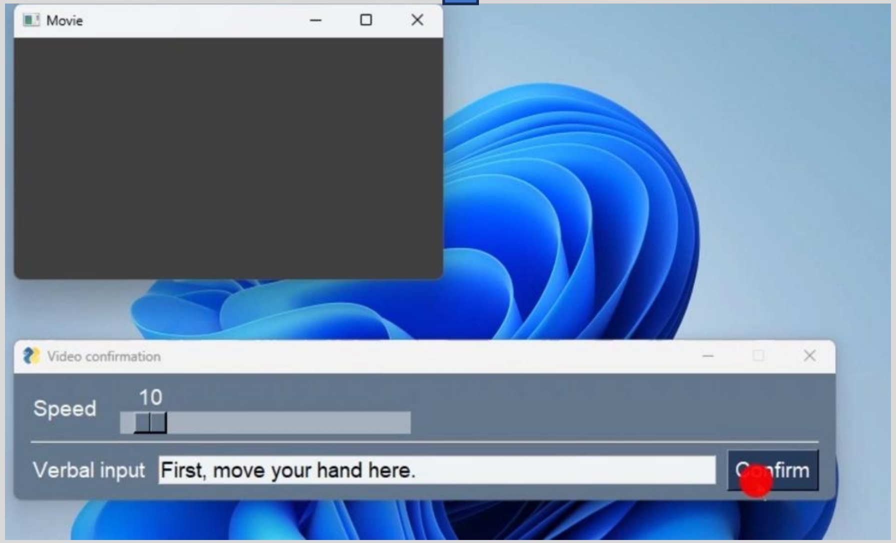
what-to-do	where-to-do
Move_hand()	**Where to move hand**
Grasp_object()	**From which direction to approach**
Move_object()	**To where to move the object**

what-to-do *where-to-do*



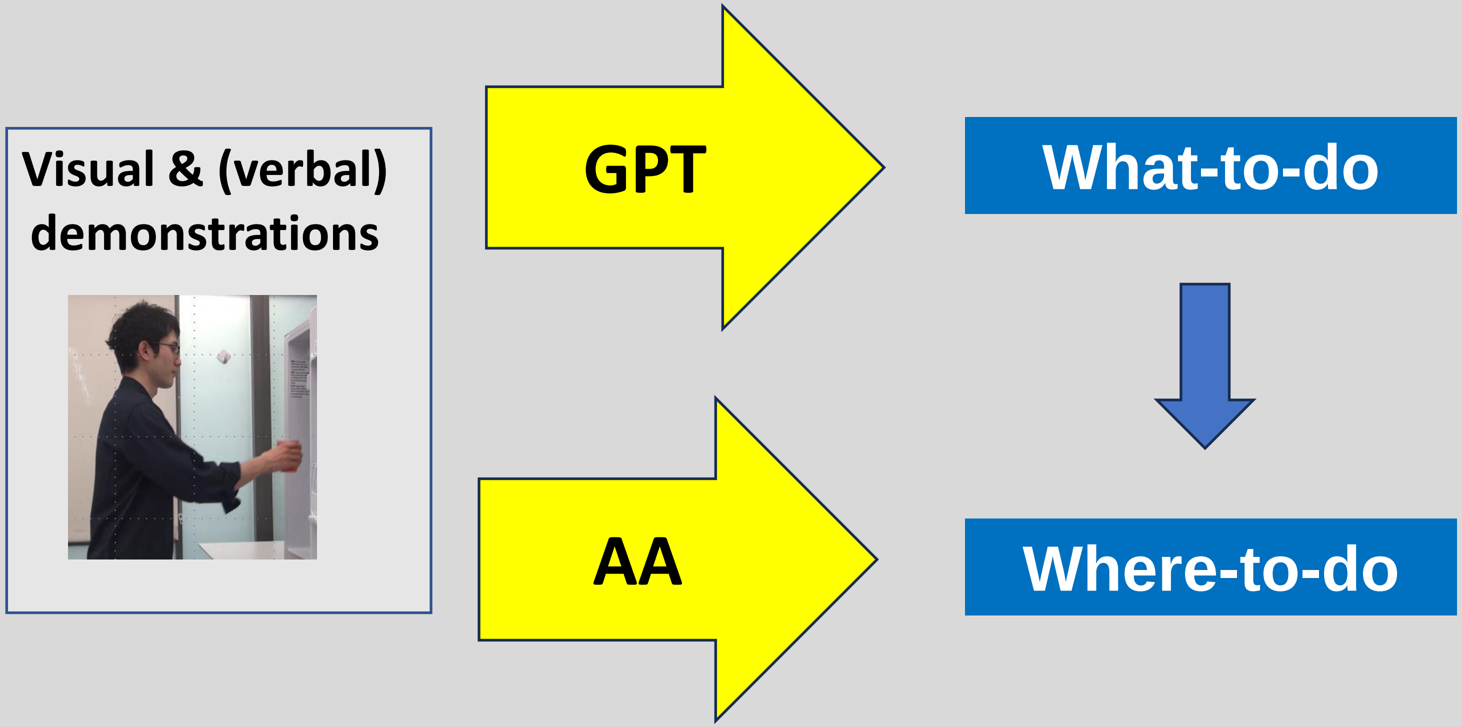
Abstract task models
(Minsky's frames)

Segmentation
Grounding



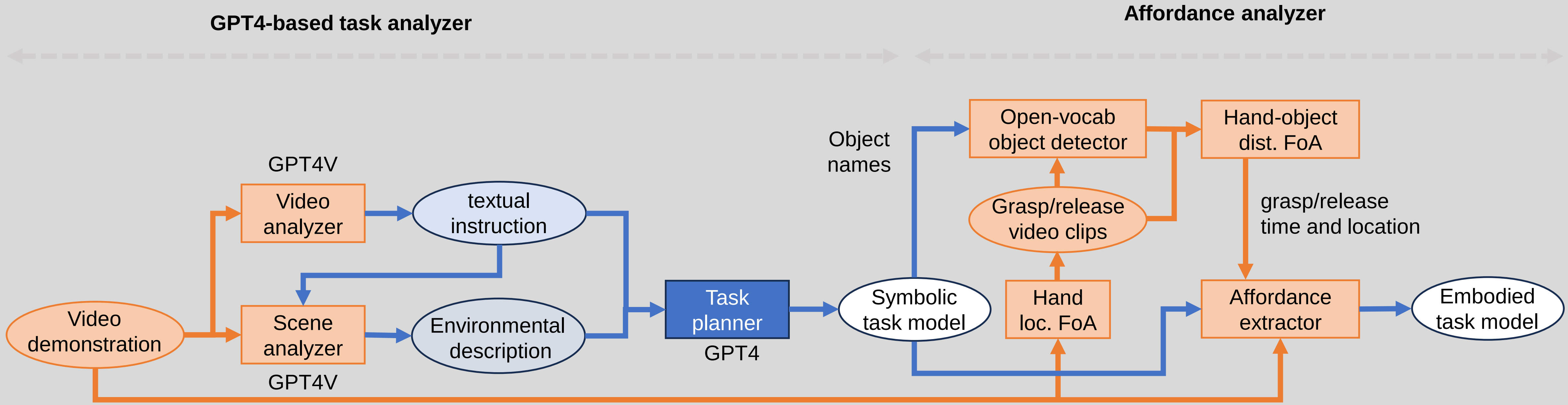
- **Where-to-do**
  - **Necessary parameters for execution**
  - **Defined to each what-to-do**
  - **Minsky's frame type design**
  - **Deamon to extract from input video**

GPT-Encoder (Ver23Nov): VLM + LLM (visual input only)



Wake2023gpt-arXiv

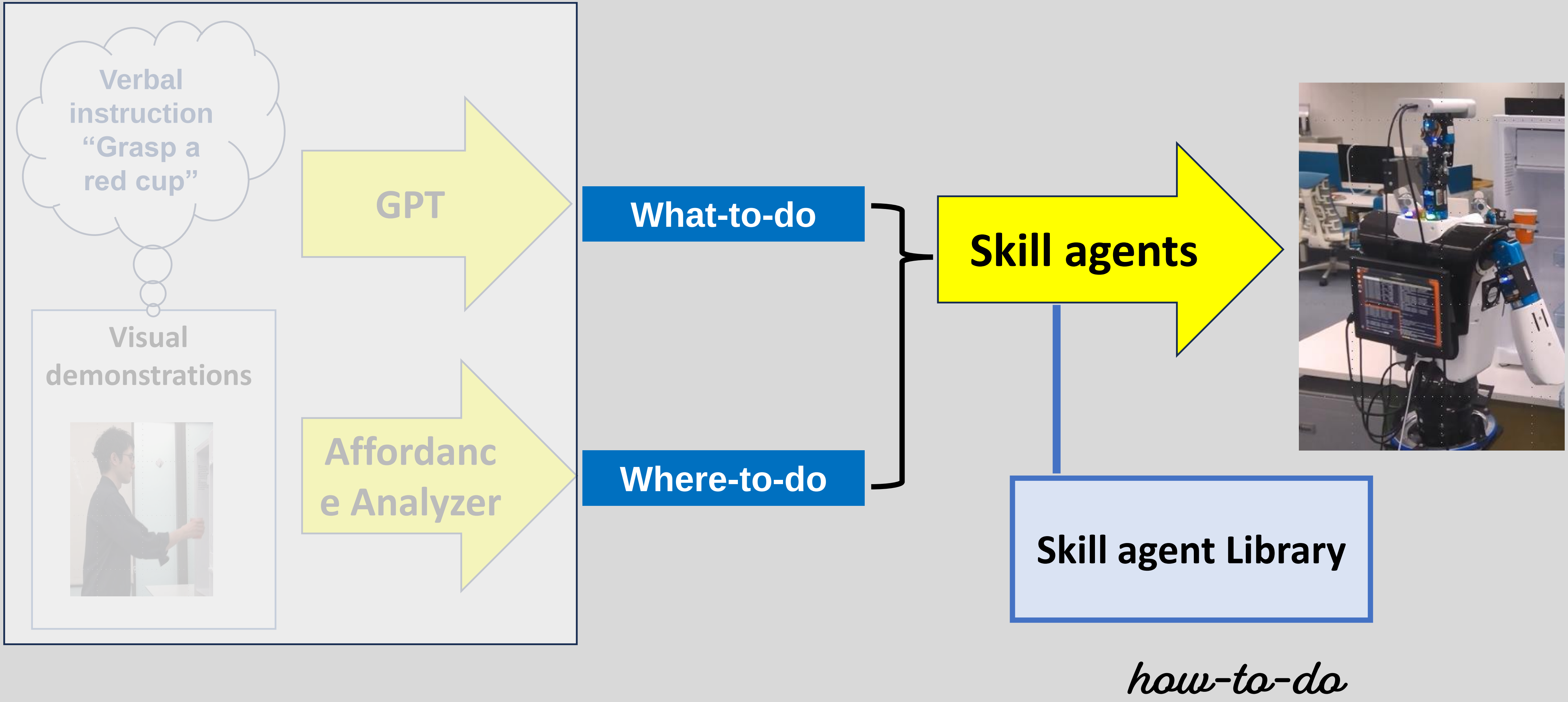
Task models without verbal input using GPT-4



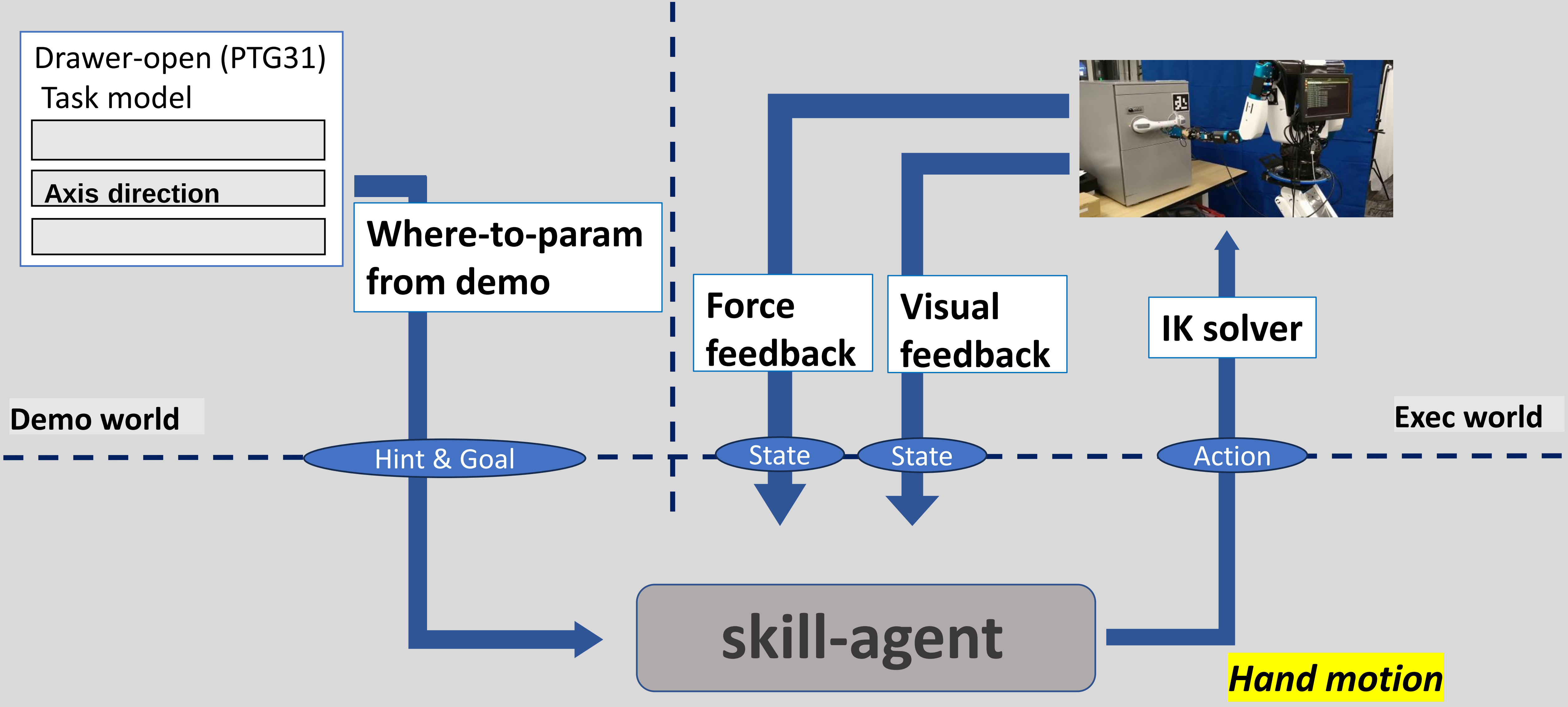
Blue components/lines are text-based information, and the red components are vision-related information

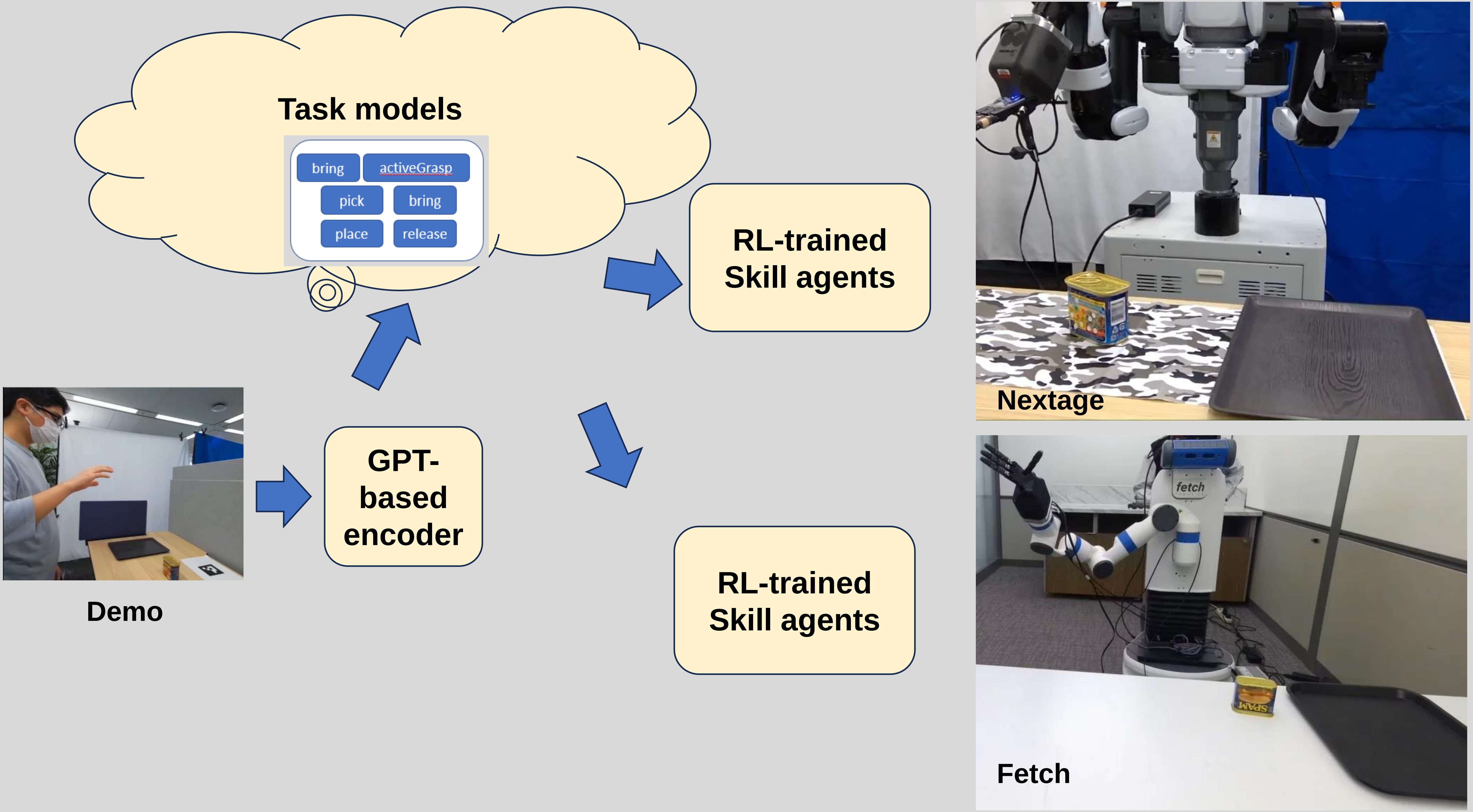
- Symbolic task planner: generated what-to-do
- Affordance analyzer: instantiate the tasks with affordance (i.e., skill parameters)

Skill-agents retrieved from the library



Skill agent to execute primitive actions





Task models
bring
activeGrasp
pick
bring
place
release
RL-trained Skill agents
GPT-based encoder
RL-trained Skill agents
Demo
Nextage
Fetch

Designing agent library

Library of skill agents

- A collection of reusable agents to execute primitive actions on different robot hardware
- Roughly corresponds to “verb” such as pickup or grasp
- Necessary and sufficient set to cover the action domain
  - Grasp library: given by Closure theory
  - Manipulation library: given by Kuhn-Tucker Theory

Library of Grasp-skill agents

Closure theory

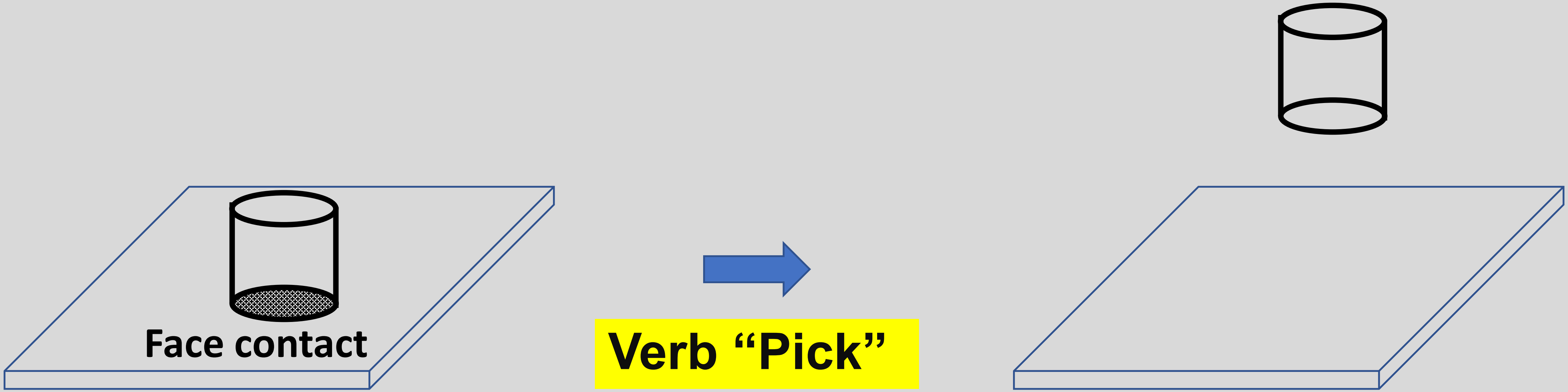
Library of Manipulation-skill agents

Kuhn-Tucker theory

Manipulation-skill agent

How to represent: face contact relation

Manipulation primitive action causes contact-state transition



"Pick" breaks the face contact

Ikeuchi1994Assembly-IEEE-TRA

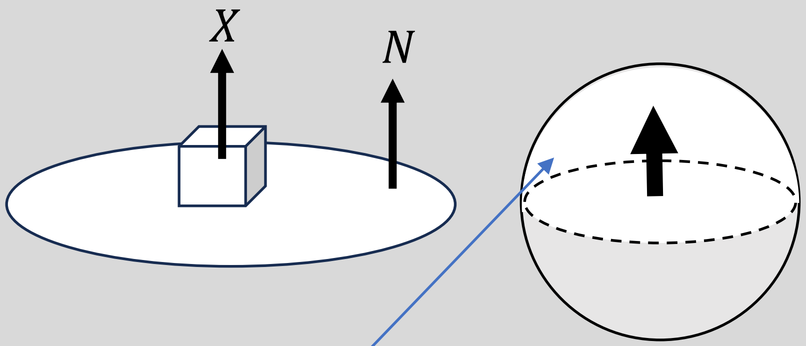
State of face contact

State of contact = movable directions

One directional contact

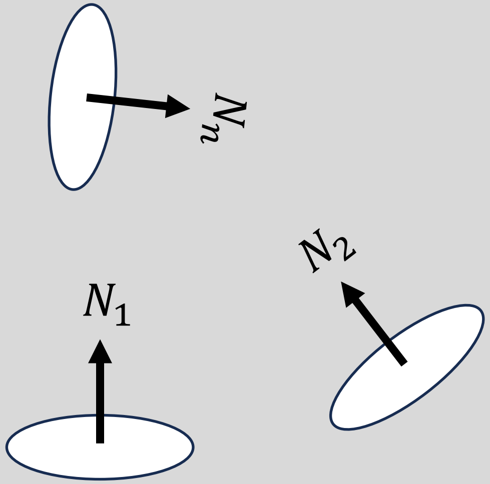
$$\boldsymbol{X} \cdot \boldsymbol{N} \geq 0$$

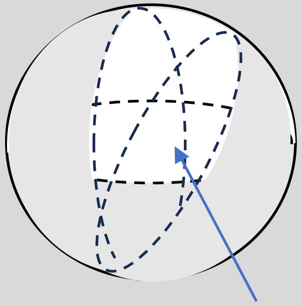
X: possible movable direction
N: Constraint Normal direction



Movable directions =
Northern hemisphere of Gaussian sphere

Multi directional contacts





Movable directions =
Polygonal area on
Gaussian sphere

$$\boldsymbol{X} \cdot \boldsymbol{N_1} \geq 0$$
$$\boldsymbol{X} \cdot \boldsymbol{N_2} \geq 0$$
$$\vdots$$
$$\boldsymbol{X} \cdot \boldsymbol{N_n} \geq 0$$

Solution of Simultaneous linear inequality equations

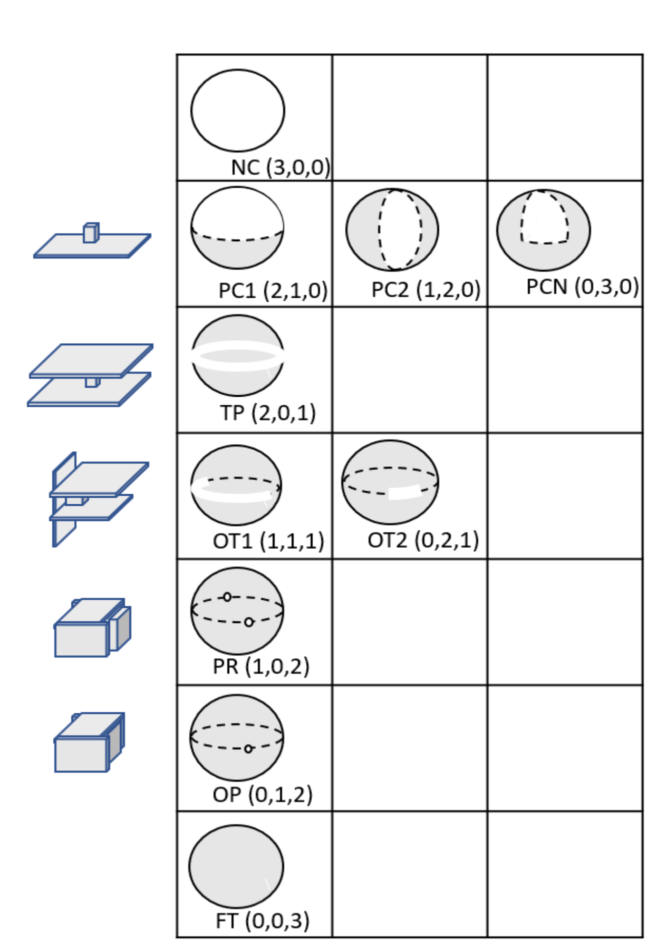
Kuhn-Tucker theory

$$X \cdot N_1 \geq 0$$
$$X \cdot N_2 \geq 0$$
$$\vdots$$
$$X \cdot N_n \geq 0$$

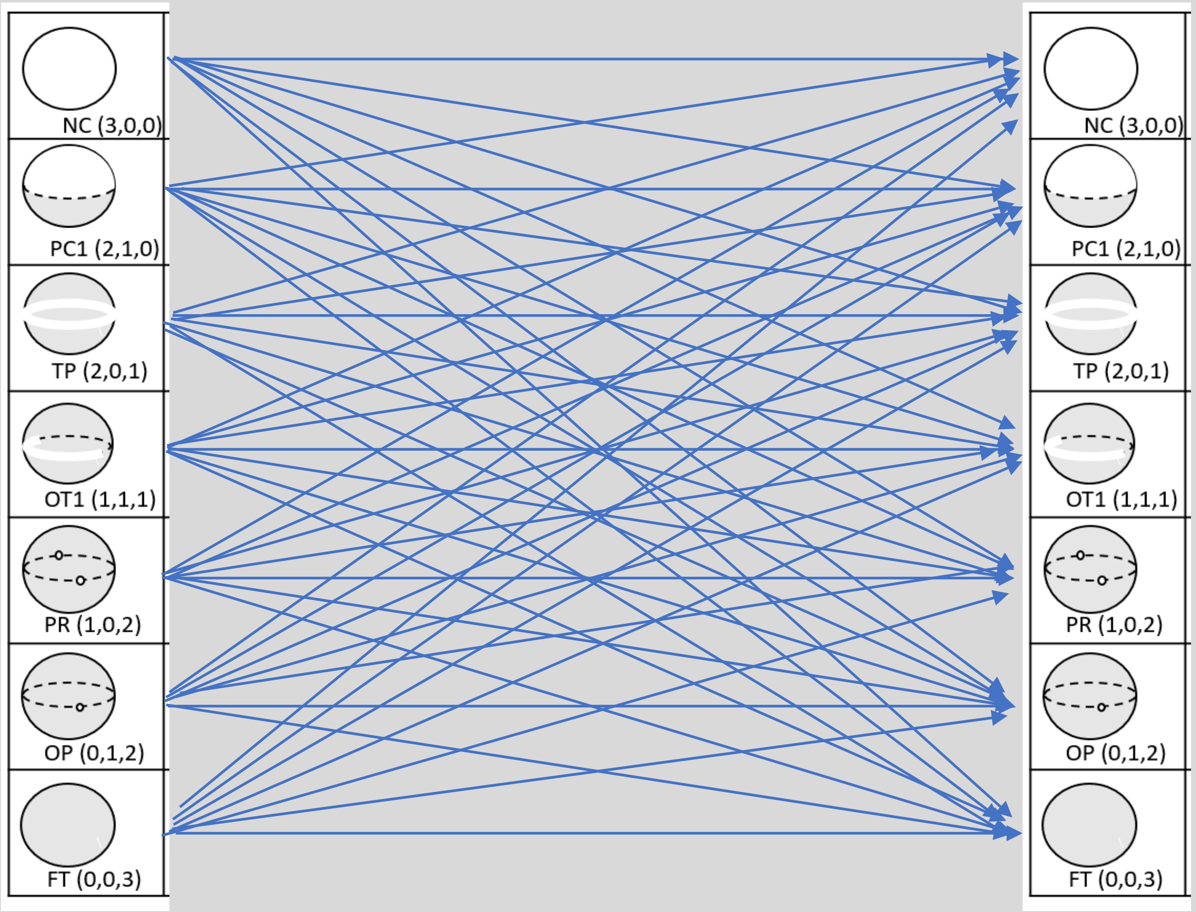
Simultaneous linear inequality equations

Kuhn-Tucker theory

Solution areas of the equations can be characterized into the following the classes



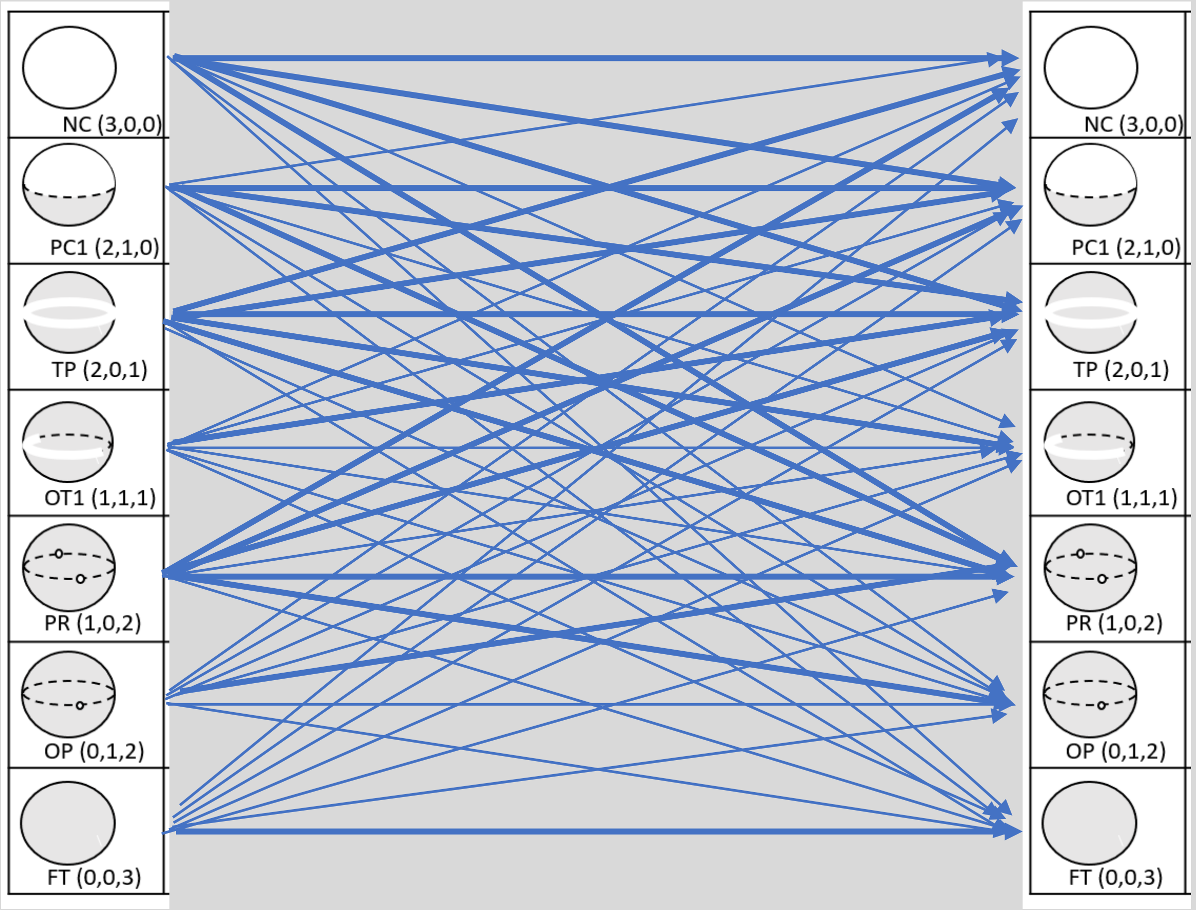
Possible transitions (possible primitive actions)



7 X 7 =49

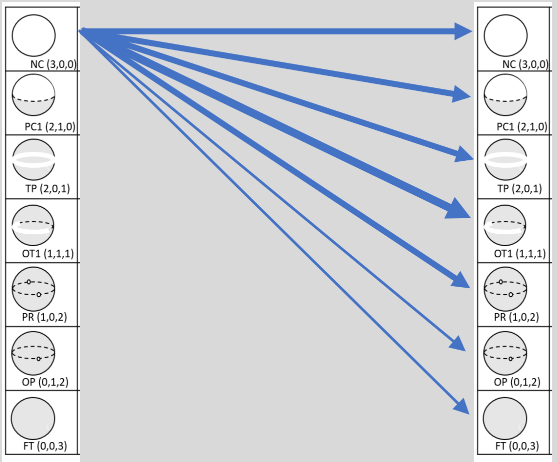
49 transitions???
49 primitive actions???

Physically Possible transitions



Physically possible:
20 transitions
20 primitive actions

Some examples From NC



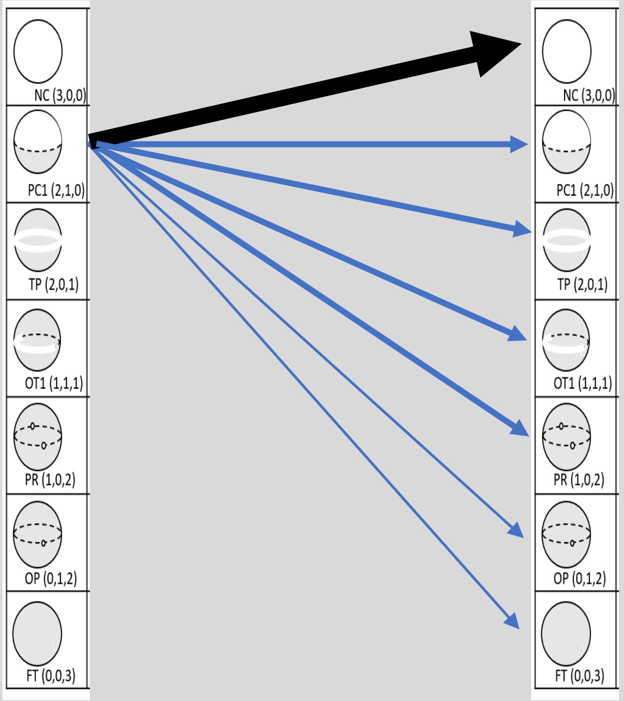
Bring	**NC:**	**NC:**	
Place	**NC:**	**PC:**	
	NC:	**TR:**	
	NC:	**OT:**	
insert	**NC:**	**PR:**	
	NC:	**OP:**	
	NC:	**FT:**	

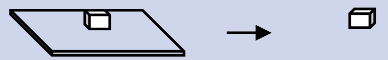
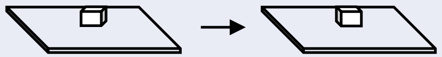
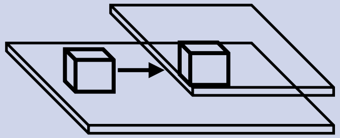
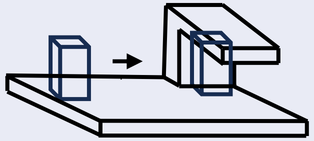
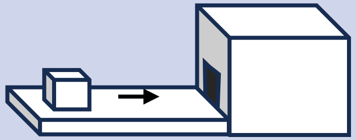
Often appear in YouTube

Often appear in YouTube

In industrial applications,
Yes, but in service robot
Not often

Some examples from PC

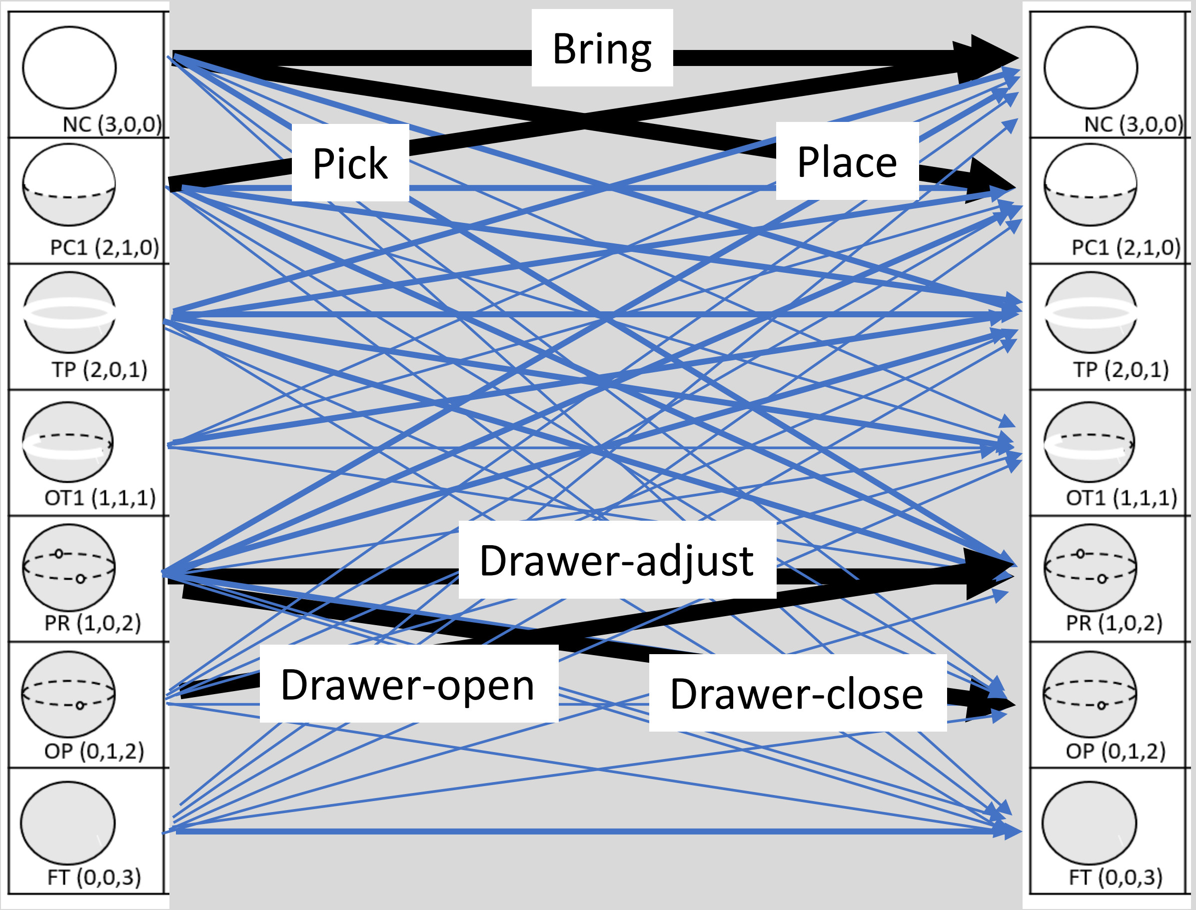


Pick	PC:	NC:	
Wipe	PC:	PC:	
	PC:	TR:	
	PC:	OT:	
	PC:	PR:	
	PC:	OP:	
	PC:	FT:	

Often appear in YouTube

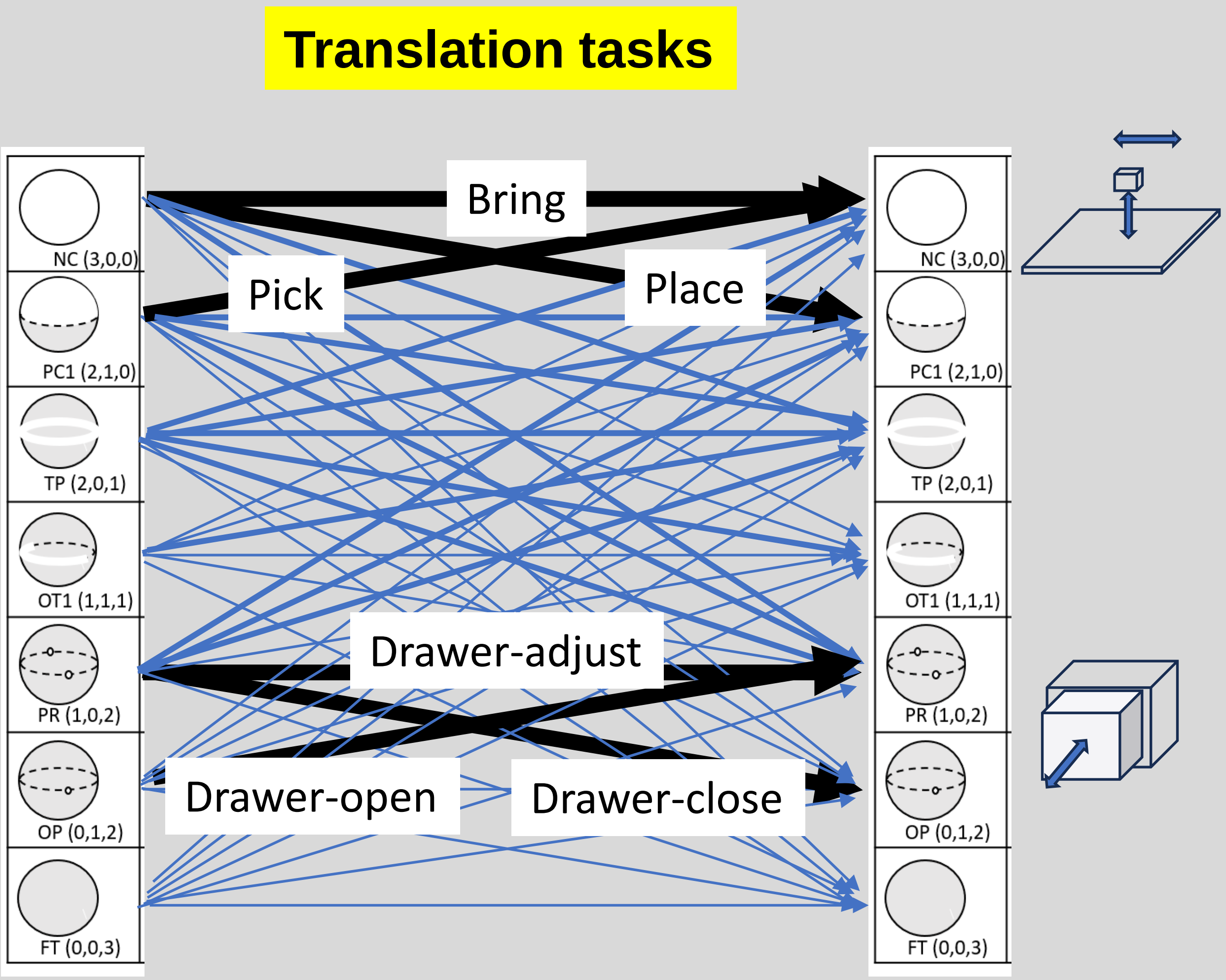
Discuss later

Frequently appeared transitions in YouTube

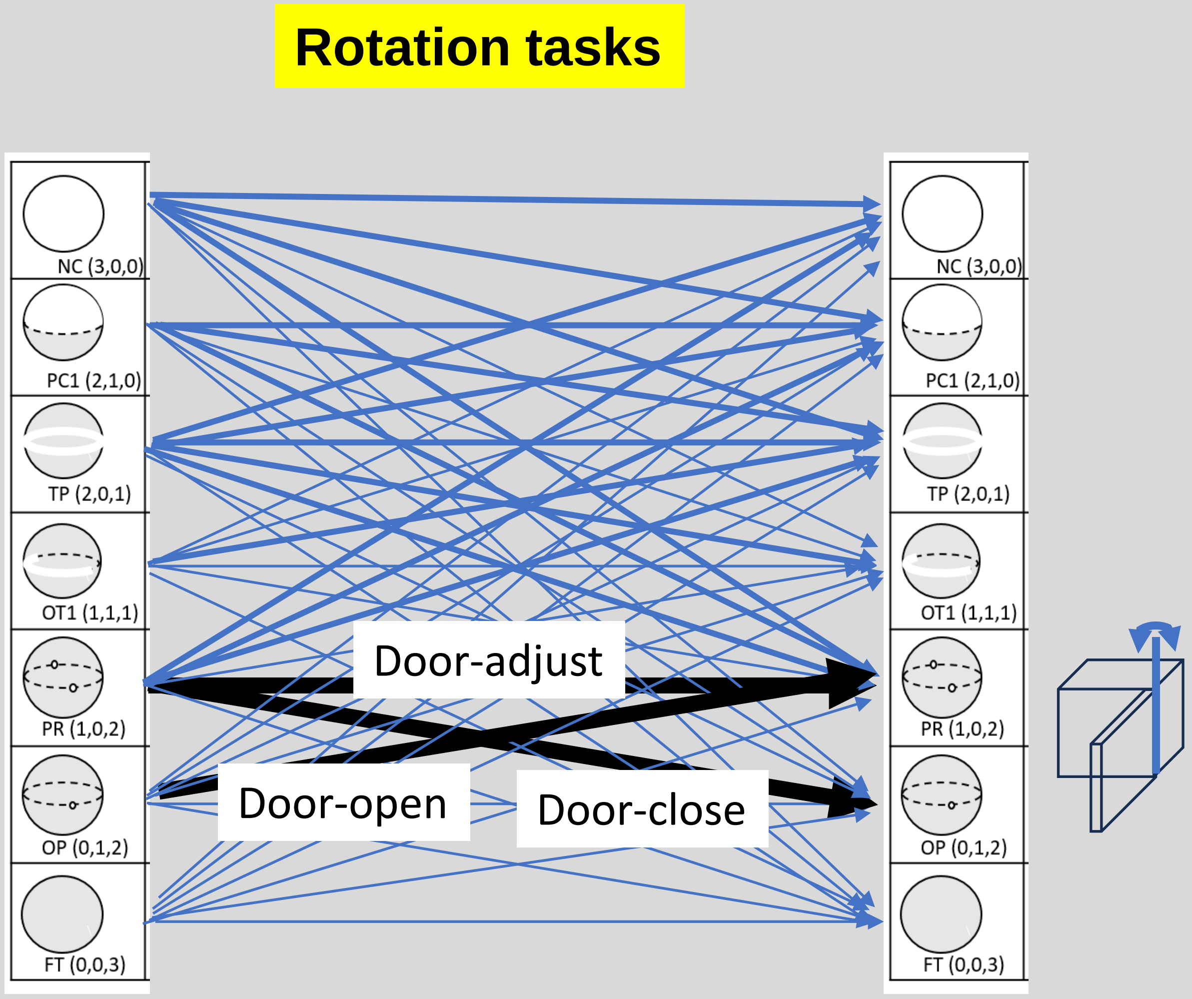


YouTube appear:
6 transitions
6 tasks

Physical manipulation agents



6 translation tasks



3 rotation tasks

Tool-env common sense:

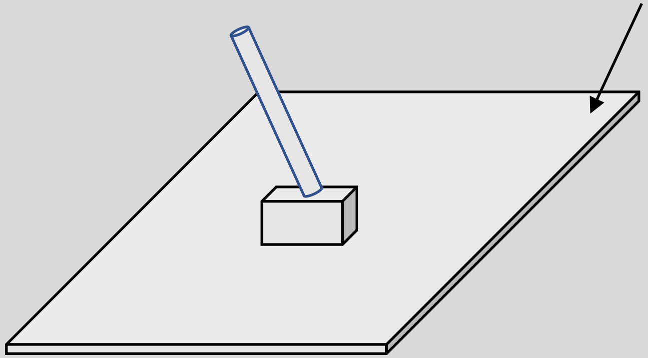
Common-sense: while wiping, do not detach from the table surface

Physical & Semantic constraints

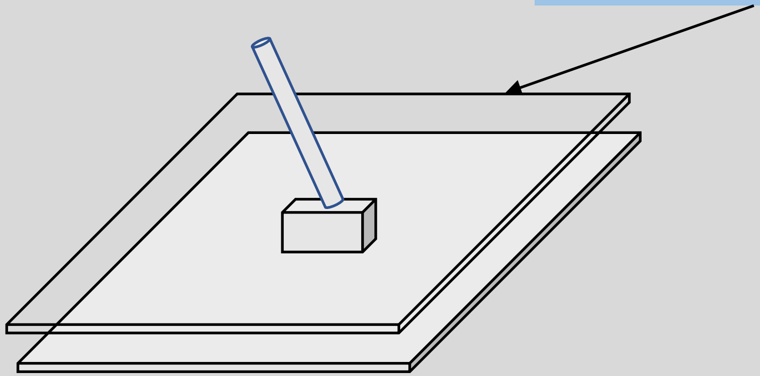
- **Physical constraint:** movable only upper directions due to the table surface

- **Semantic constraint:** for wiping, *not to detach* from the table surface

Physical surface



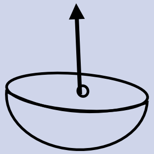
Semantic surface (common-sense representation)



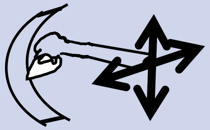
Ikeuchi2023semantic-IJRR

Semantic constraints extracted from YouTube cocking video

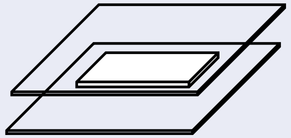
Semantic Ping

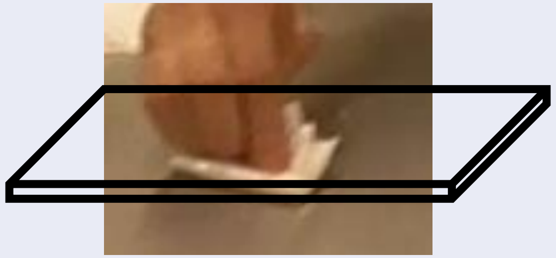


Semantic Sphere

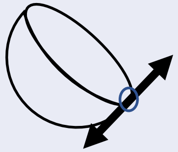


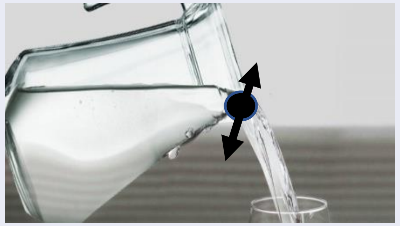
Semantic Walls



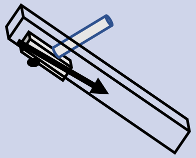


Semantic Hinge



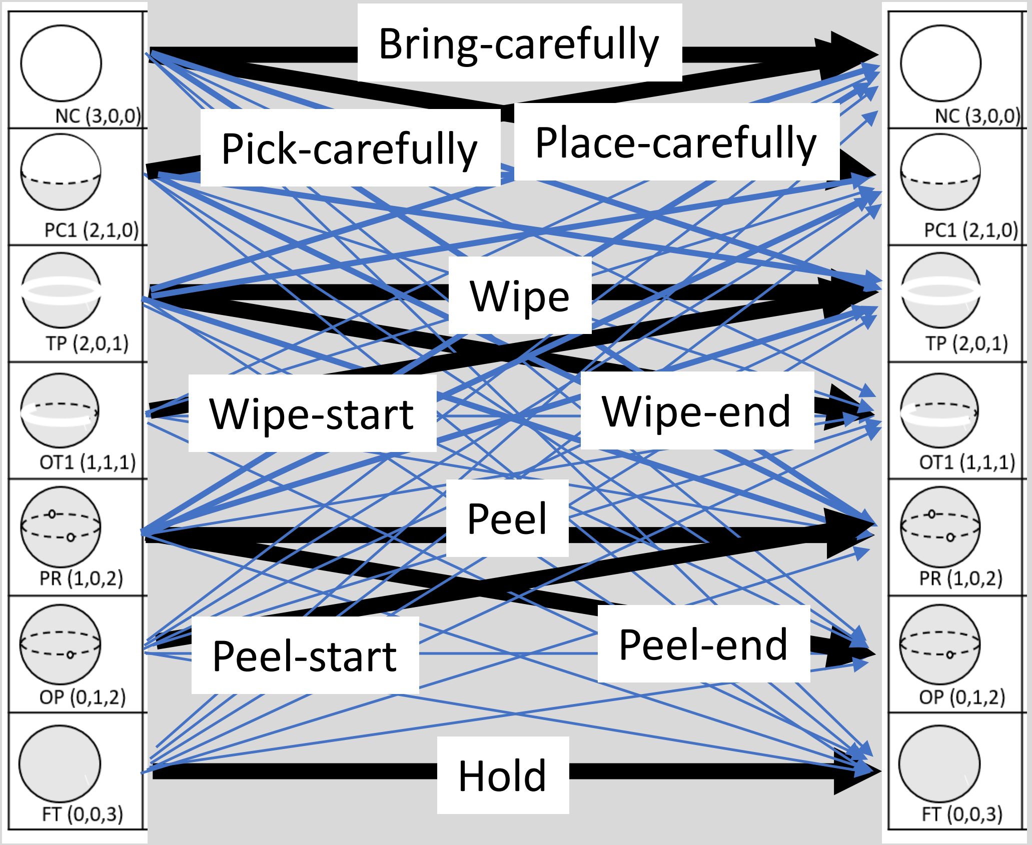


Semantic Tube



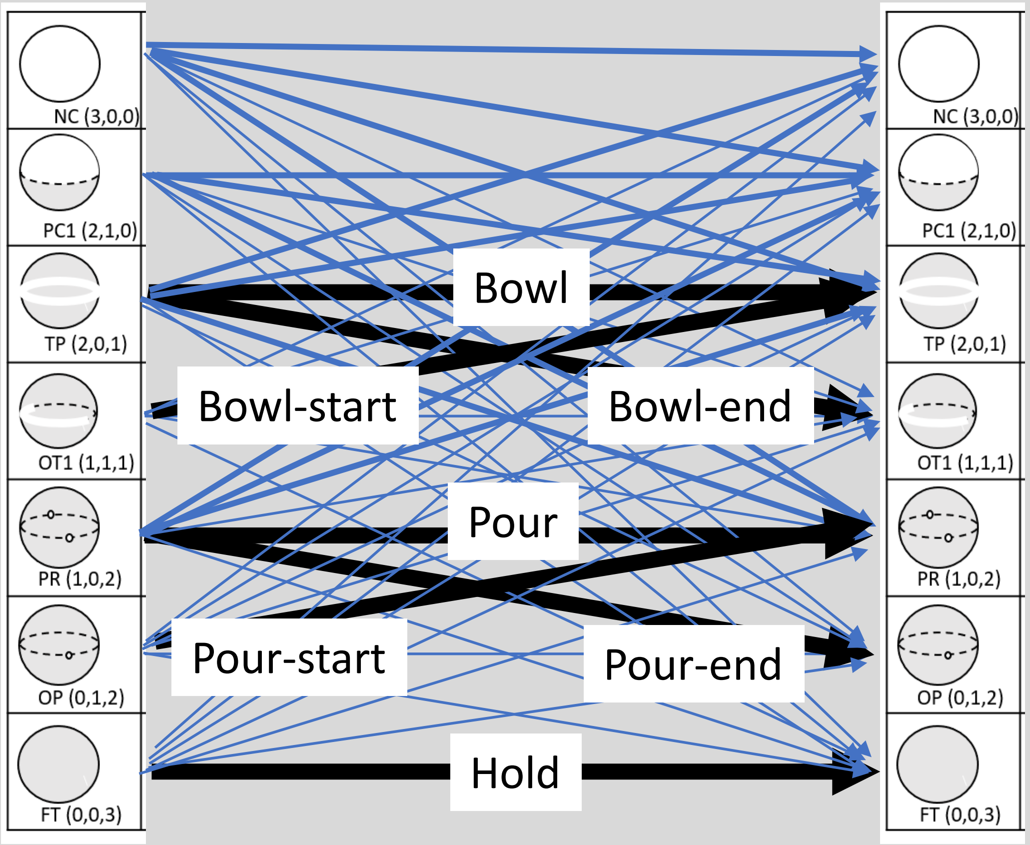
Semantic manipulation agents

Translation tasks



10 Translation tasks

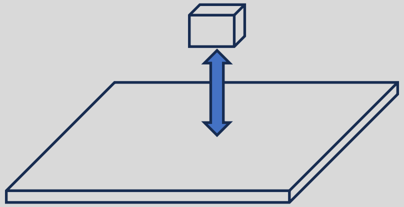
Rotation tasks



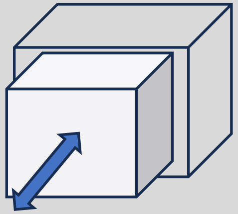
6 rotation tasks

Agents in the current library

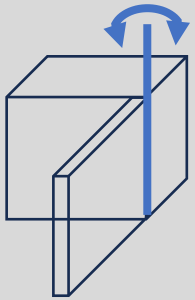
- **PTG1**
  - **Picking**
  - **Placing**
  - **Bringing**



- **PTG3**
  - **DrawerOpening**
  - **DrawerClosing**
  - **DrawerAdjusting**



- **PTG5**
  - **DoorOpening**
  - **DoorClosing**
  - **Door Adjusting**



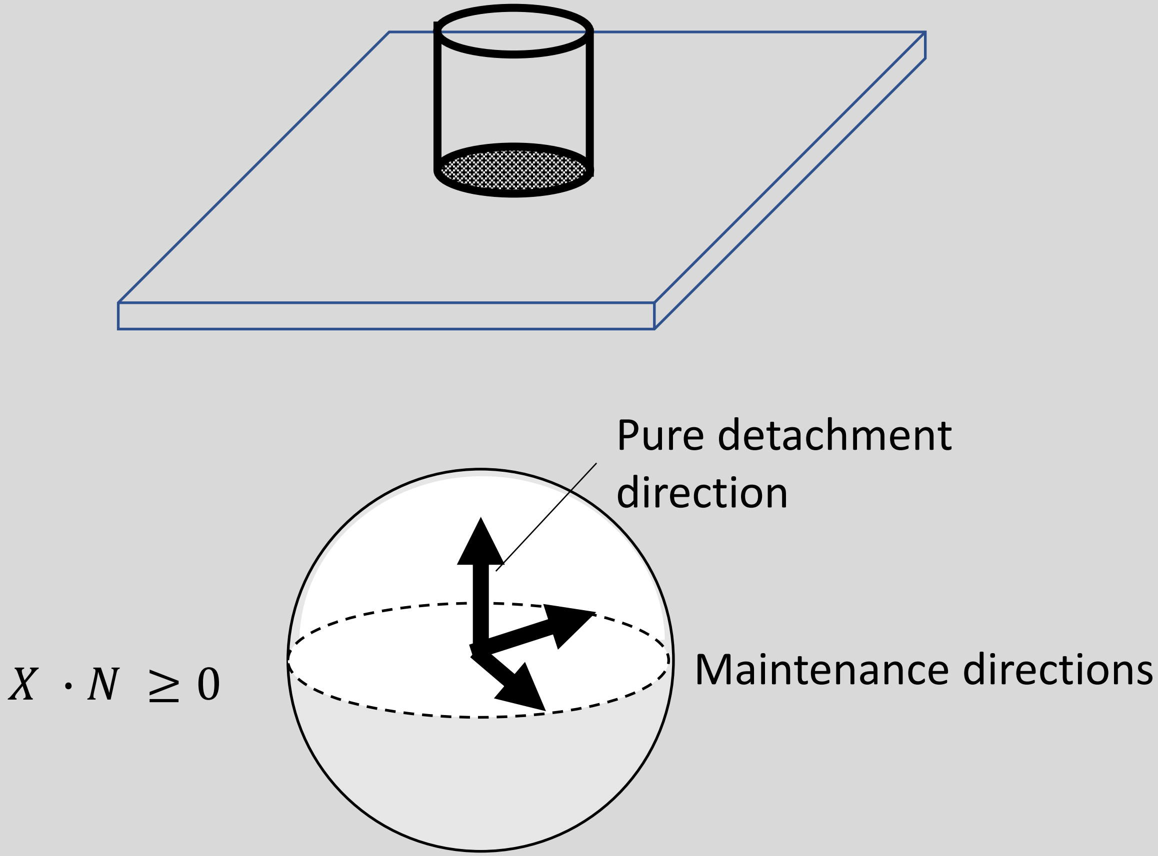
- **STG1**
  - **Bring-carefully**

- **STG2**
  - **Wiping**

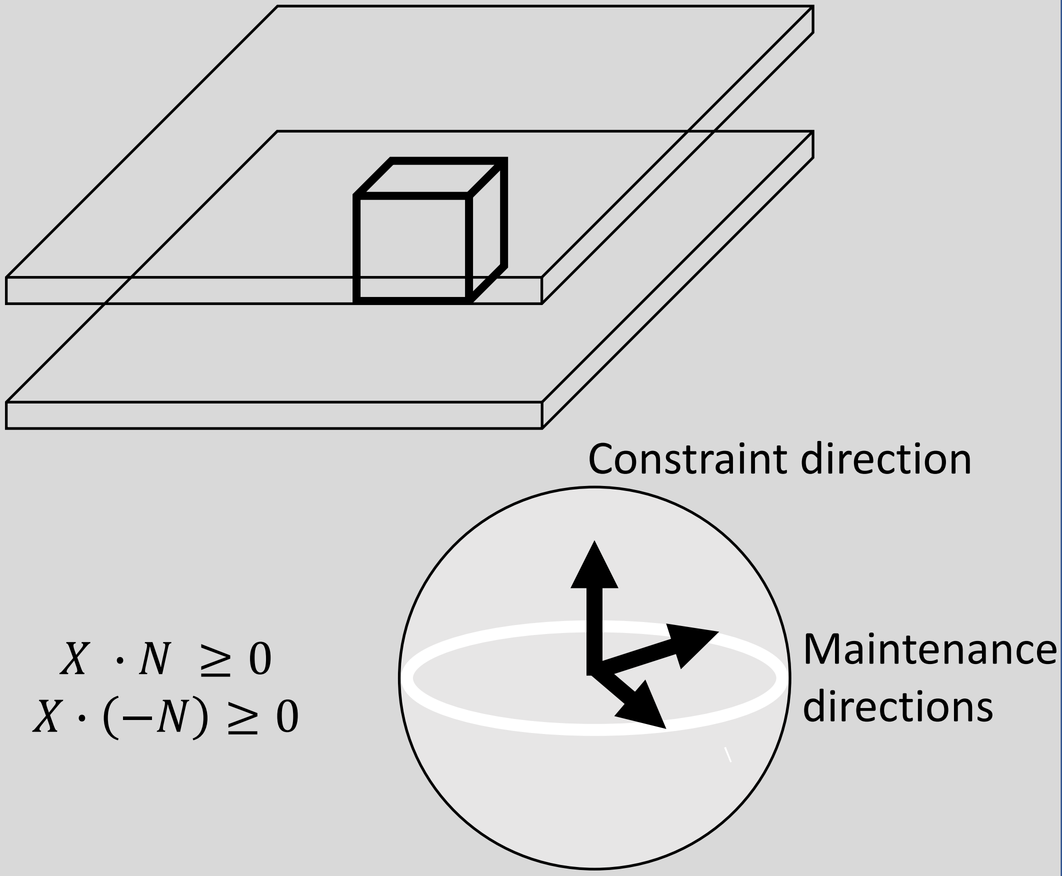
Takamatsu2023Designing-arXiv

RL training of agents

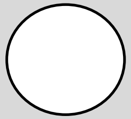
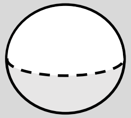
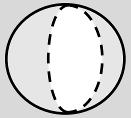
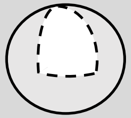
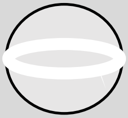
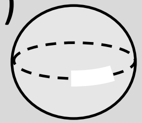
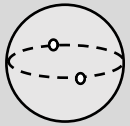
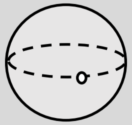
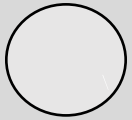
Maintain/Detach/Constraint dimension



Maintenance = 2 DoFs
Detachment = 1 DoFs (2,1,0)
Constraint = 0 DoFs

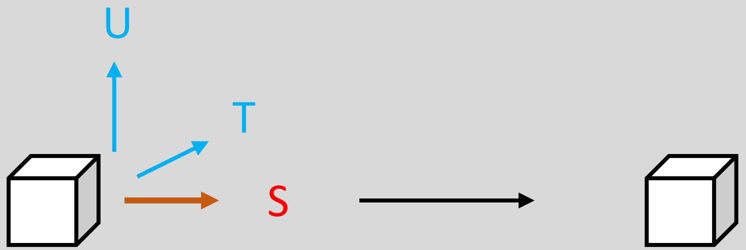


Maintenance = 2 DoFs
Detachment = 0 DoFs (2, 0, 1)
Constraint = 1 DoFs

State name	DOFs	Admissible translation directions on the Gaussian sphere & Dimensions
NC Non-contact translation	3	NC (M=3, D=0, C=0) 
PC Partial contact translation	2.5	PC1(M=2,D=1, C=0)  PC2(M=1, D=2, C=0)  PCN (M=0,D=3, C=0) 
TR Translation contact translation	2	TR(M=2, D=0, C=1) 
OT One-way translation contact translation	1.5	OT1(M=1, D=1, C=1)  OT2(M=0, D=2, C=1) 
PR Prismatic contact translation	1	PR(M=1, D=0, C=2) 
OP One-way prismatic contact translation	0.5	OP(M=0, D=1, C=2) 
FT Fully contact translation	0	FT(M=0, D=0, C=3) 

Dimension transition provides control laws

Bring



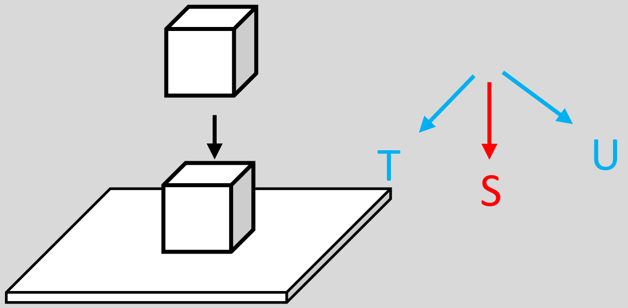
S: motion direction
T: perpendicular to motion
U:Perpendicular to motion

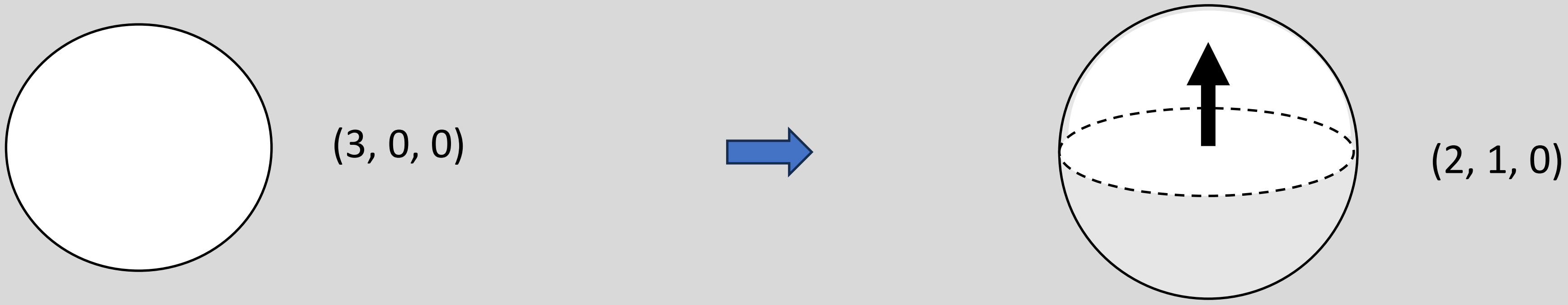
(M, D, C) = (3, 0, 0) → (M, D, C) = (3, 0, 0)

Motion direction (S): Maintenance to Maintenance → Position control
Perpendicular direction (T): Maintenance to Maintenance → Position control
Perpendicular direction (U): Maintenance to Maintenance → Position control

**If S = goal-s AND T = goal-t AND U = goal-u,
then reward**

Place

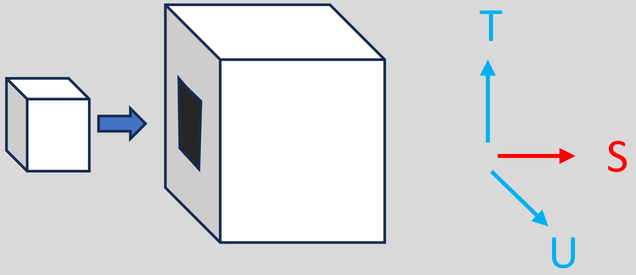


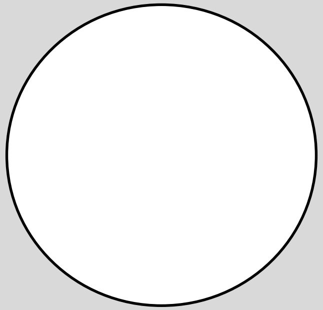


Motion direction (S): Maintenance to Detachment → force control
Perpendicular direction (T): Maintenance to Maintenance → position control
Perpendicular direction (U): Maintenance to Maintenance → position control

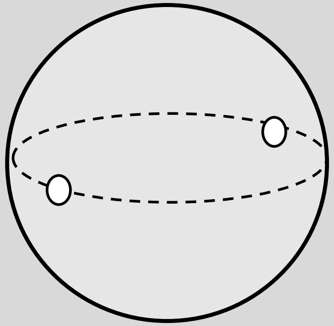
If F+s > delta-zero AND T = goal-t AND U = goal-u,
then reward

insert





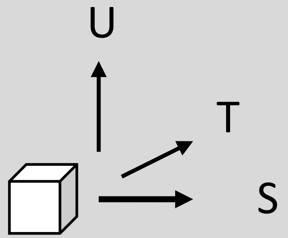
(M, D, C) = (3, 0, 0)



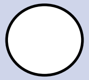
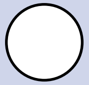
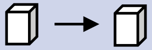
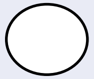
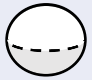
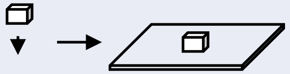
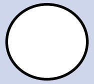
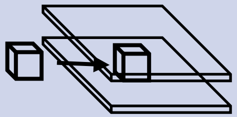
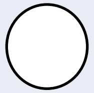
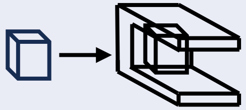
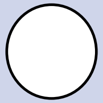
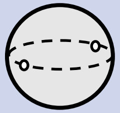
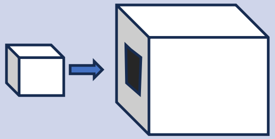
(M, D, C) = (1, 0, 2)

Motion direction (S): Maintenance to Maintenance → position control
Perpendicular direction (T): Maintenance to Constraint → visual control then force control
Perpendicular direction (U): Maintenance to Constraint → visual control then force control

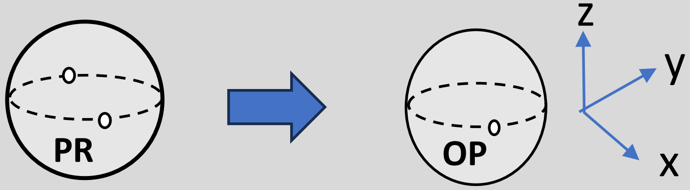
If Before Transition AND |T – feature-along-t-direction | > delta-gap, then penalty
If Before Transition AND |U – feature-along-u-direction | > delta-gap, then penalty
If AfterTransition AND F-t > delta-collision, then penalty
If AfterTrasnsition AND F-u > delta-collision, then penalty
If S = goal-s, then reward



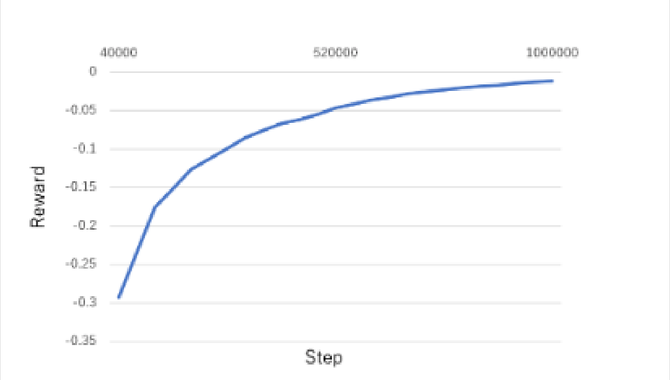
S: motion direction
T: perpendicular to motion
U:Perpendicular to motion

	Start	End	Example	position	Force	Vision	Control
Bring	NC: 3,0,0 	NC: 3,0,0 		S: (M-M) T: (M-M) U: (M-M)			If S = goal-s AND T = goal-t AND U = goal-u, then reward
Place	NC: 3,0,0 	PC: 2,1,0 		T: (M-M) U: (M-M)	S: (M-D)		If F+s > delta-zero AND T = goal-t AND U = goal-u, then reward
	NC: 3,0,0 	TR: 2,0,1 		S: (M-M) T: (M-M)		U: (M-C)	If BeforTransition AND |U – feature U| > delta-gap, then penalty If AfterTransition AND F-u > delta-collision, then penalty If S = goa-s AND T = goal-t, then reward
	NC: 3,0,0 	OT: 1,1,1 		S: (M-M)		T: (M-D) U: (M-C)	If BeforeTransition AND | T – feature t | > delta-gap, then penalty If BeforeTransition AND | U – feature u | > delta-gap, then penaly If AfterTransition AND F-t > delta-collision, then penalty If AfterTransition AND F-t < delta-zero, then penalty If After Transition AND F-u > delta-collision, then penalty If S = goal-s, then reward
insert	NC: 3,0,0 	PR: 1,0,2 		S: (M-M)		T: (M-C) U: (M-C)	If BeforeTransition AND |T – feature-t | > delta-gap, then penalty If Before Transition AND |U – feature-u | > delta-gap, then penalty If AfterTransition AND F-t > delta-collision, then penalty If AfterTrasnsition AND F-u > delta-collision, then penalty If S = goal-s, then reward

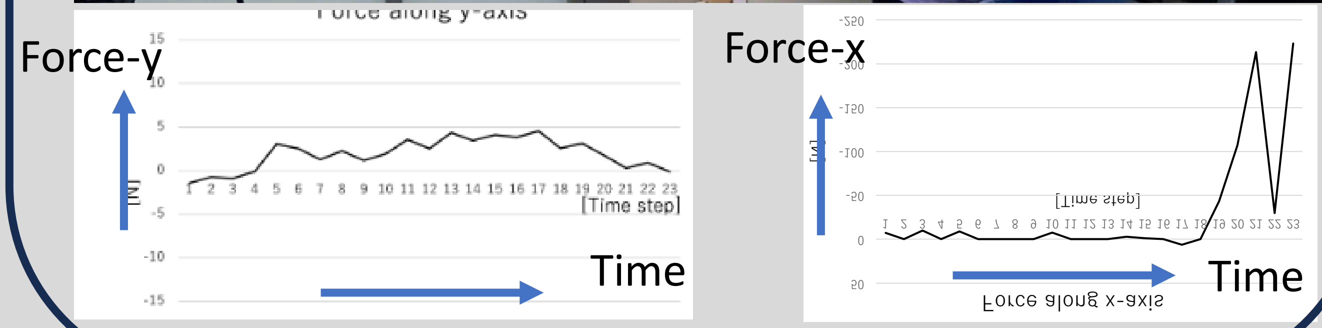
RL-trained PTG33 agent (Drawer-close)



If Force-y > delta-collision, penalty
If Force-z > delta-collision, penalty
If Force-x > delta-zero, reward

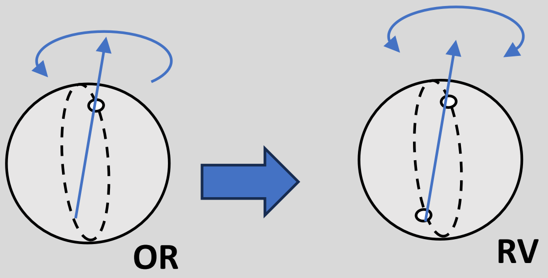


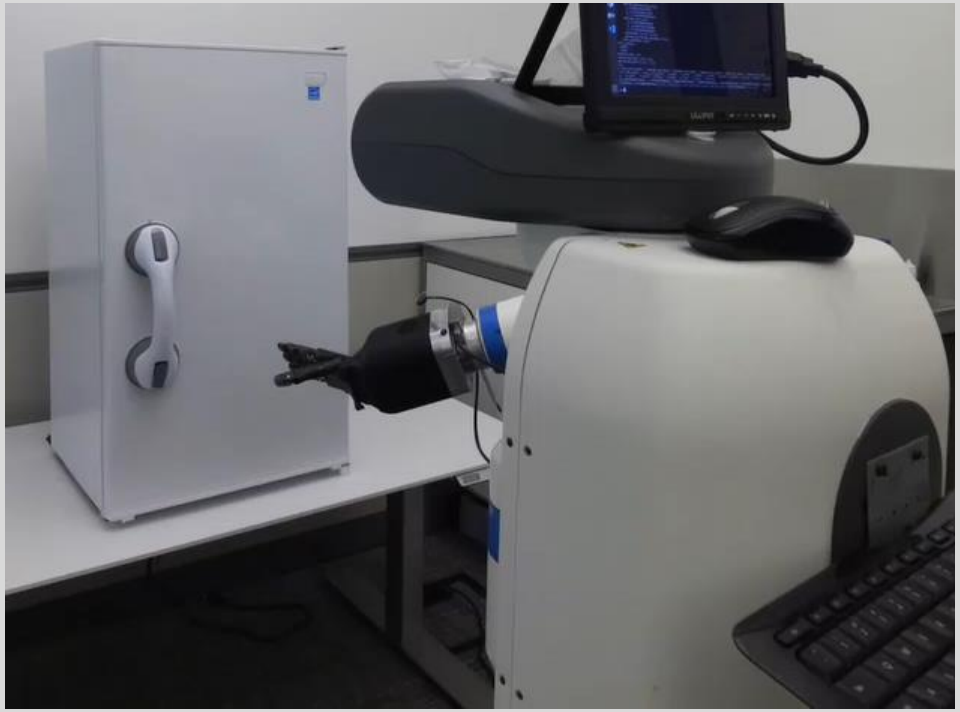
RL learning curve



RL-trained PTG51 agent (Door opening)

If Force-y > delta-collision, penalty
If Force-z > delta-collision, penalty
If X = Goal-x, reward

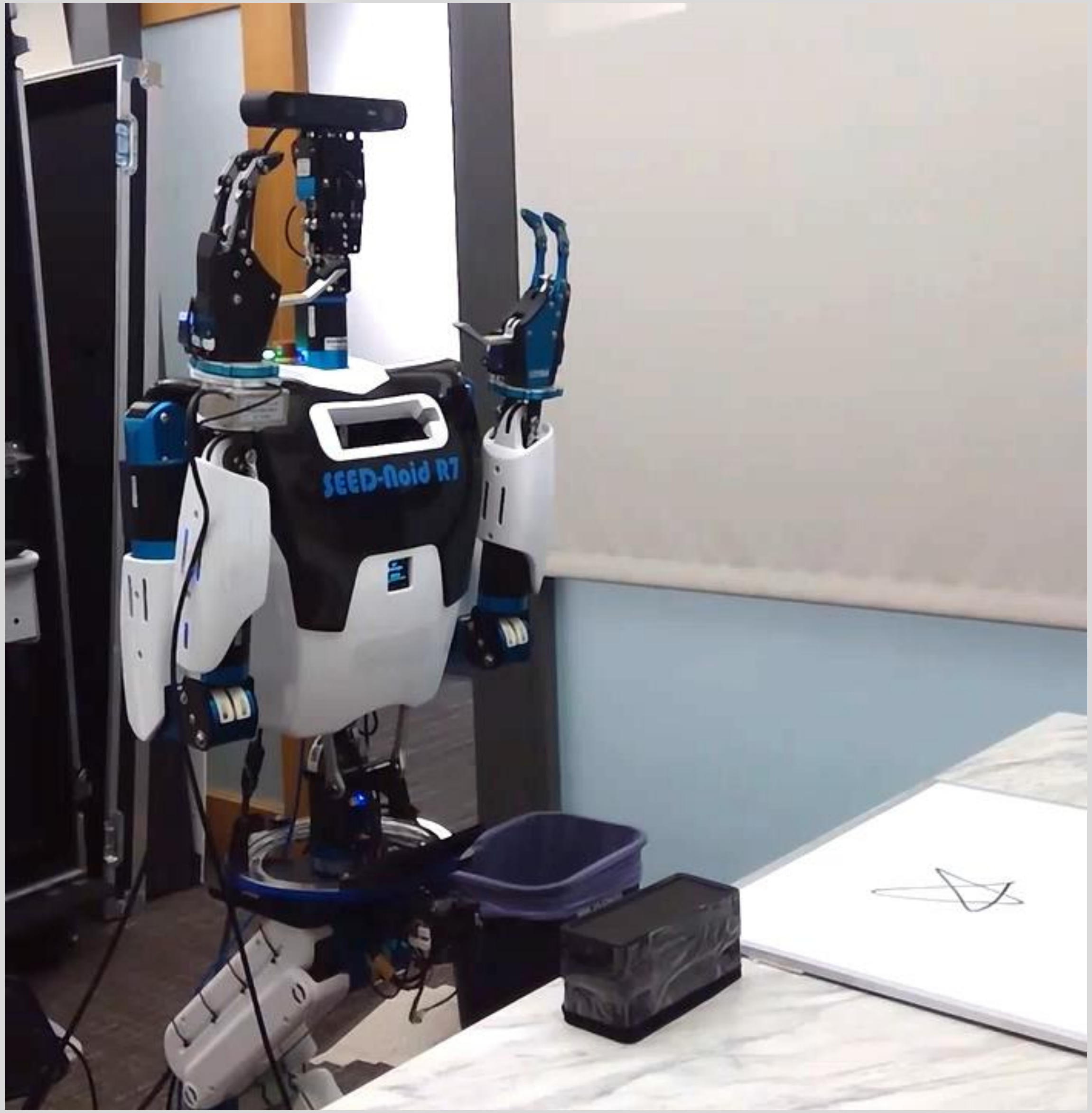


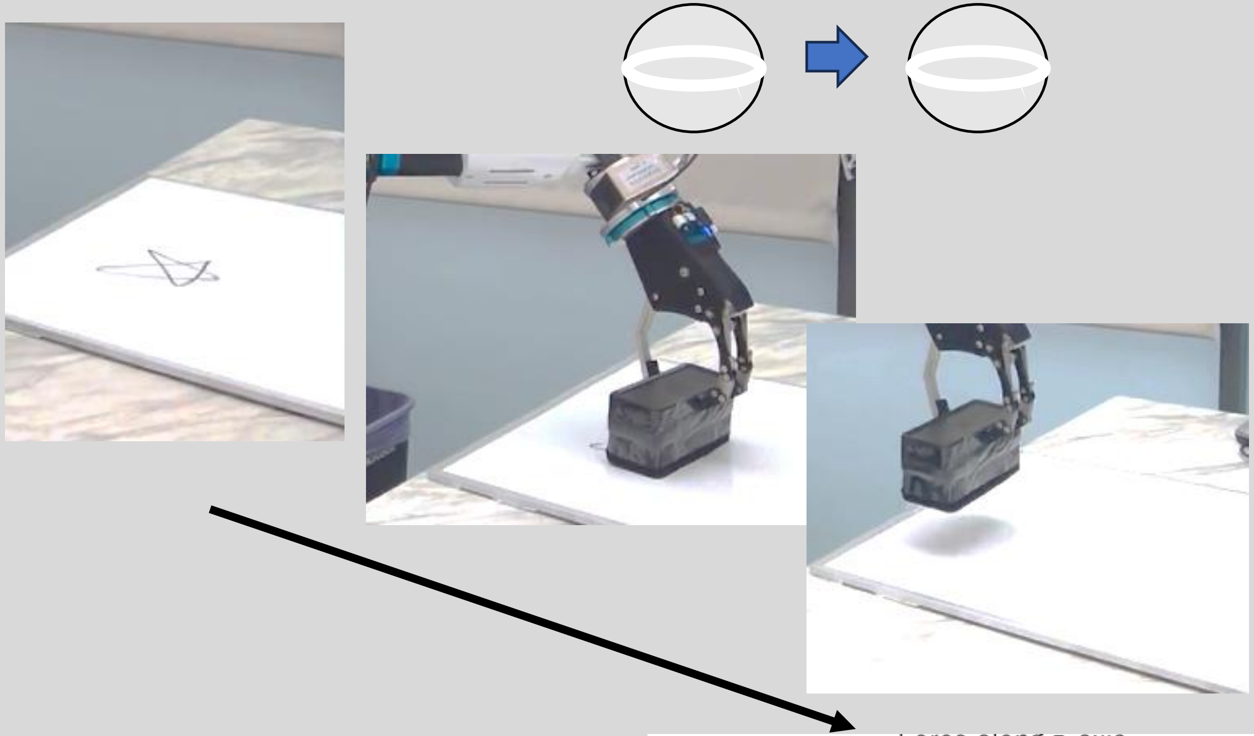


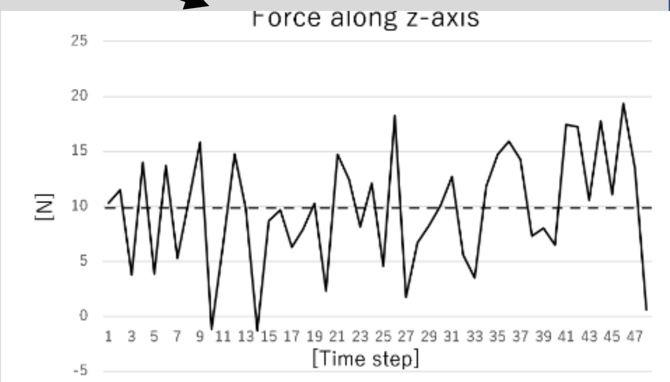
Nextage @Shinagawa × 3

Fetch @Redmond × 3

RL-trained STG2 agent (wipe)

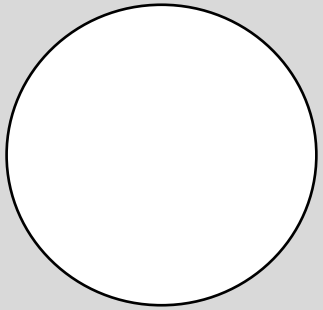




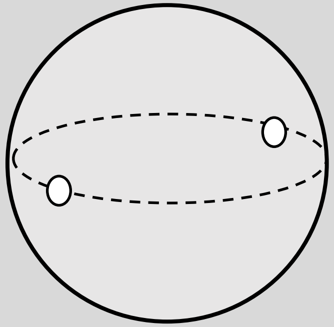


If Force-y > delta-collision, penalty
If Force-y > delta-zero, penalty
If X = Goal-x, reward

Insert (NC-PR)



Maintenance = 3
Detachment = 0
Constraint = 0



Maintenance = 1
Detachment = 0
Constraint = 2

If |T – feature-t | > delta-gap, then penalty
If |U – feature-u | > delta-gap, then penalty
If S = goal-S, then reward

Grasp-skill agent (Current version)

Current

Grasp types

- **Grasping depends on the goal of the task sequence**

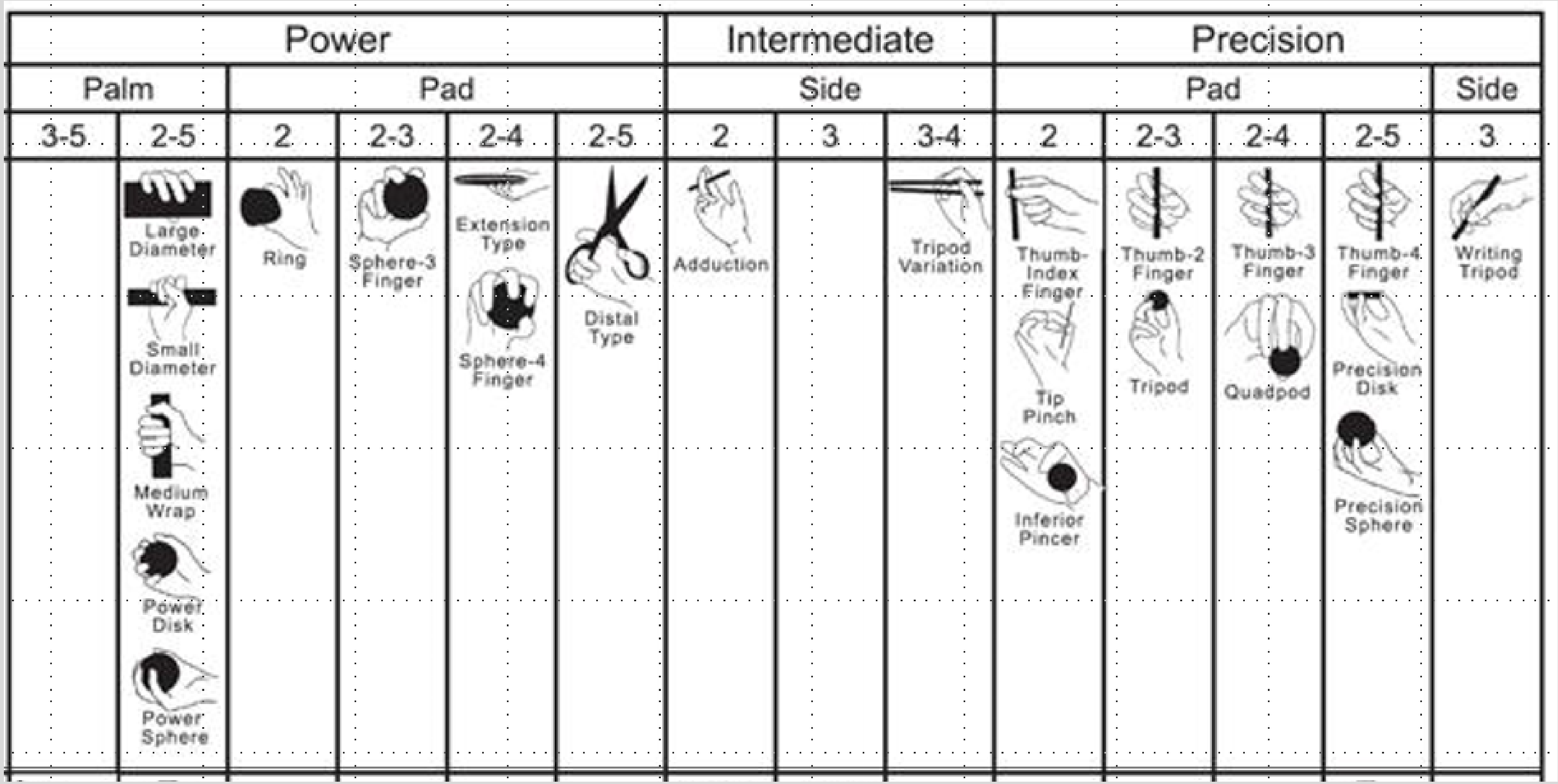
Need power to push

Need control to point

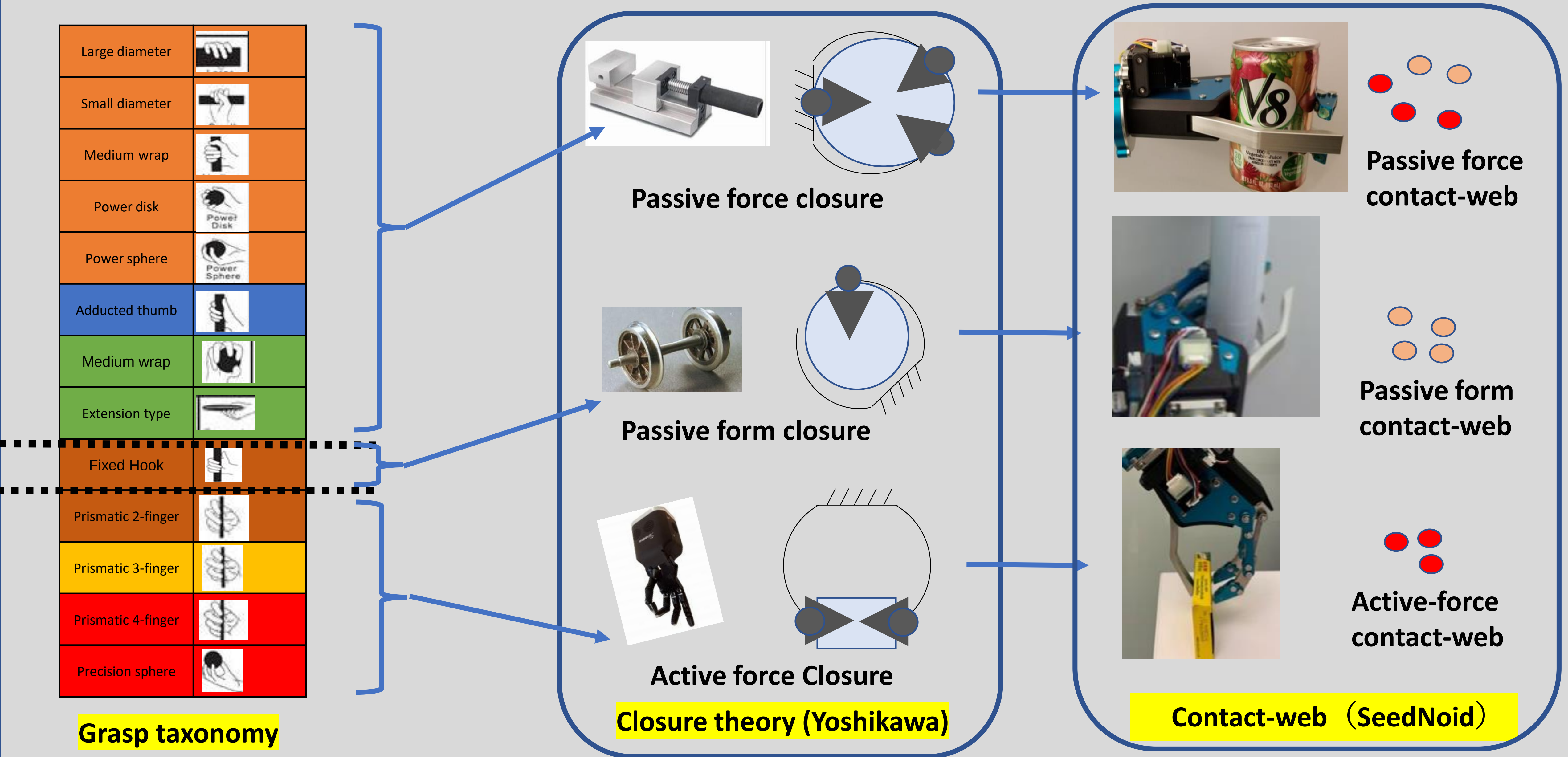
Need power and control to write

Grasp taxonomy in Robotics community

Felix et al



Purpose of task: from taxonomy to closure



Three grasp agents prepared

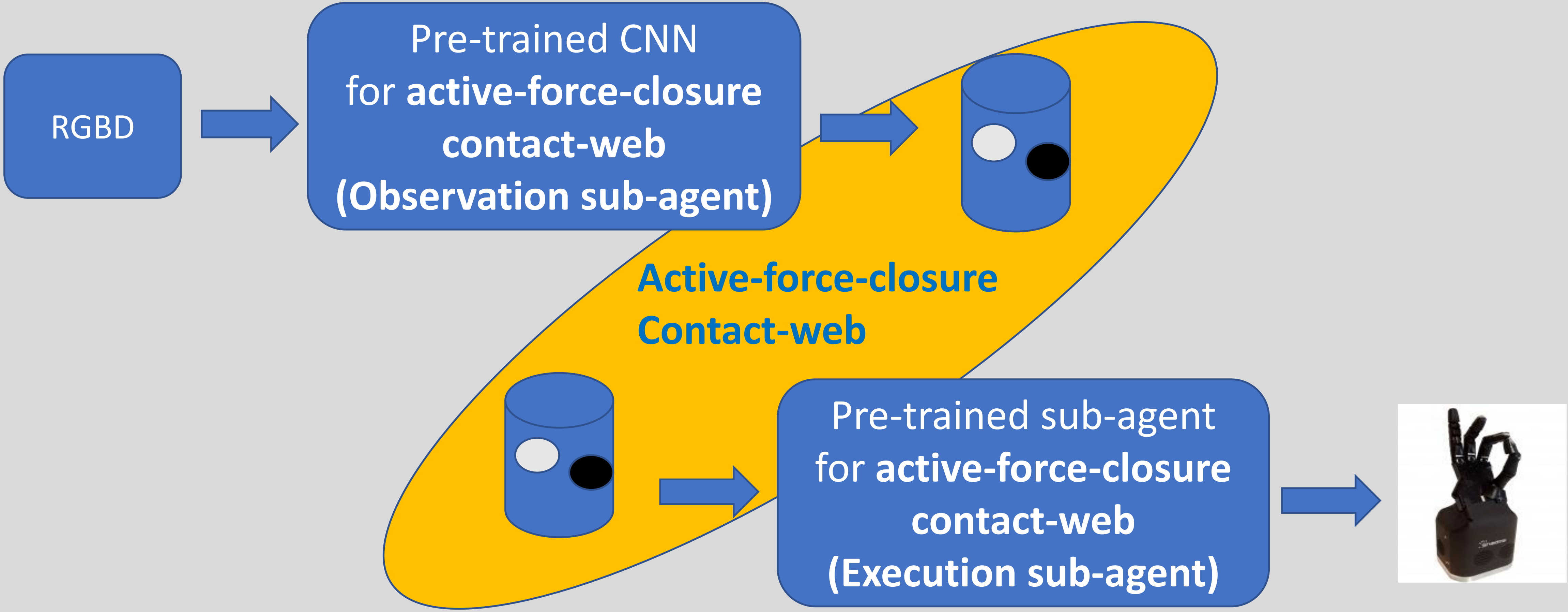
RGBD → Pre-trained CNN for **passive-force contact-web** → **Passive force contact-web** → Pre-trained agent for **passive-force contact-web** →

RGBD → Pre-trained CNN for **passive-form contact-web** → **Passive form contact-web** → Pre-trained agent for **passive-form contact-web** →

RGBD → Pre-trained CNN for **active-force contact-web** → **Active-force contact-web** → Pre-trained agent for **active-force contact-web** →

Saito&Sasabuchi2023Constraint-MDPI

Each contact-web based agent (end-2-end system)



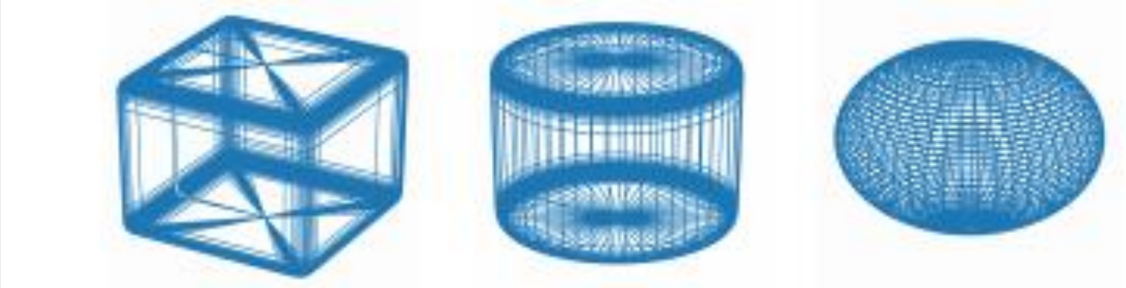
Super quadric

quadric $$\left(\frac{x}{a}\right)^2 + \left(\frac{y}{b}\right)^2 + \left(\frac{z}{c}\right)^2 = 1$$

super quadric $$\left(\frac{x}{a}\right)^\alpha + \left(\frac{y}{b}\right)^\beta + \left(\frac{z}{d}\right)^\gamma = 1$$

Size parameter: a, b, c

Shape parameter: α, β, γ

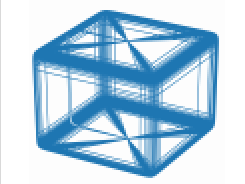


Shape variations due to parameters: α, β, γ

Training CNN

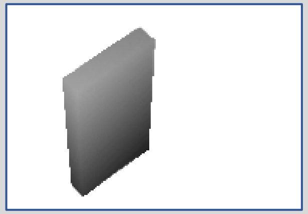
Training data

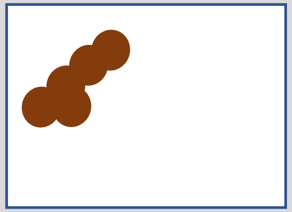
Size parameter: a, b, c   Shape parameter: α, β, γ



Super quadric

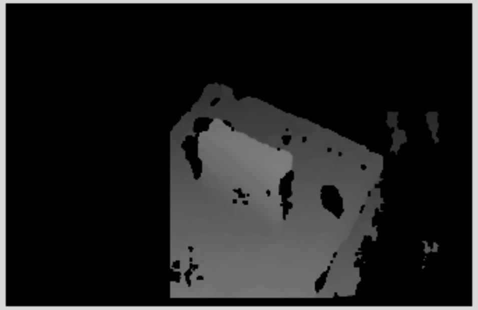
Depth image

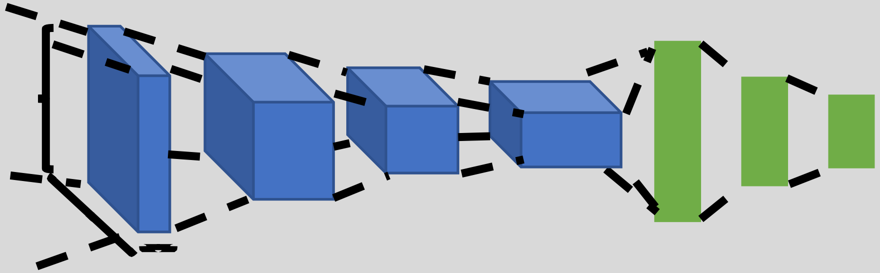


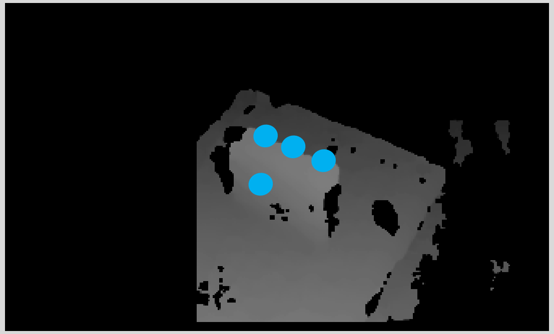


Contact web position

Trained CNN

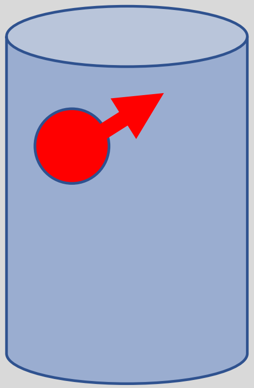




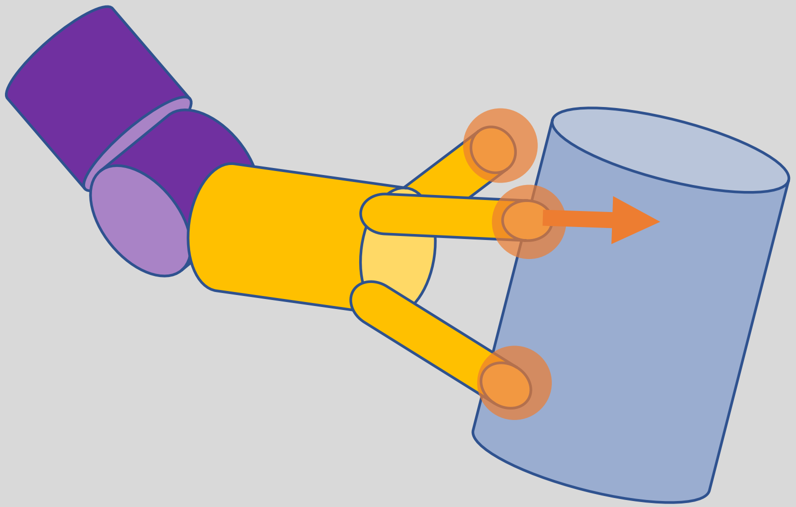


Reinforcement learning

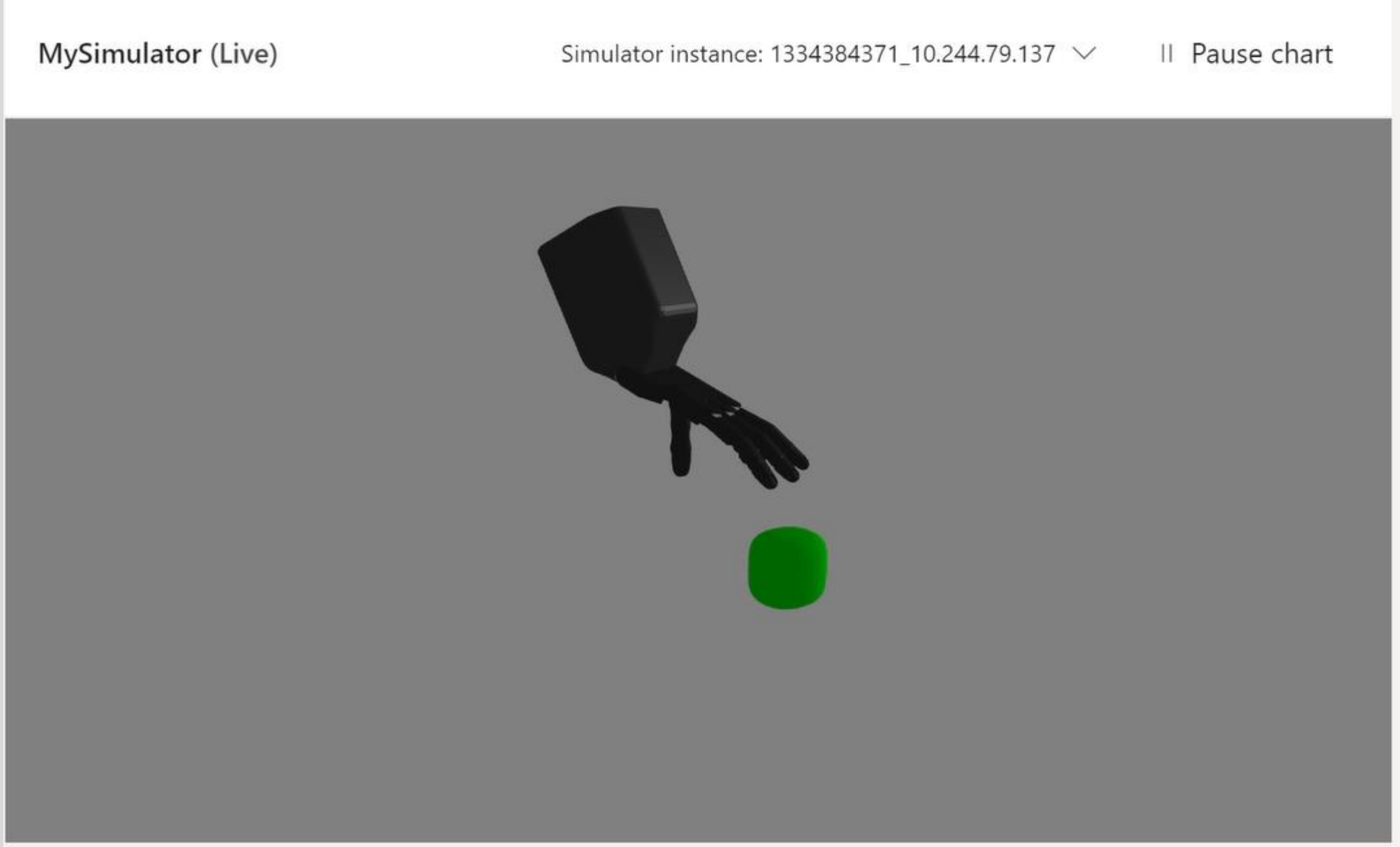
Hint information
- **Contact web**
- **Approach direction**



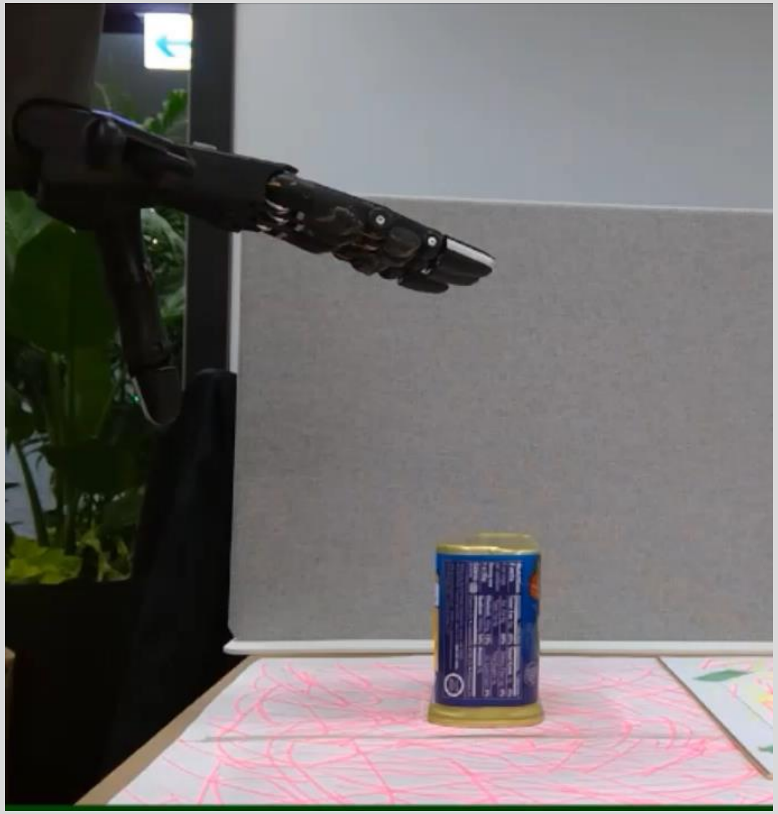
State
- **current finger positions**
- **current contact force direction**



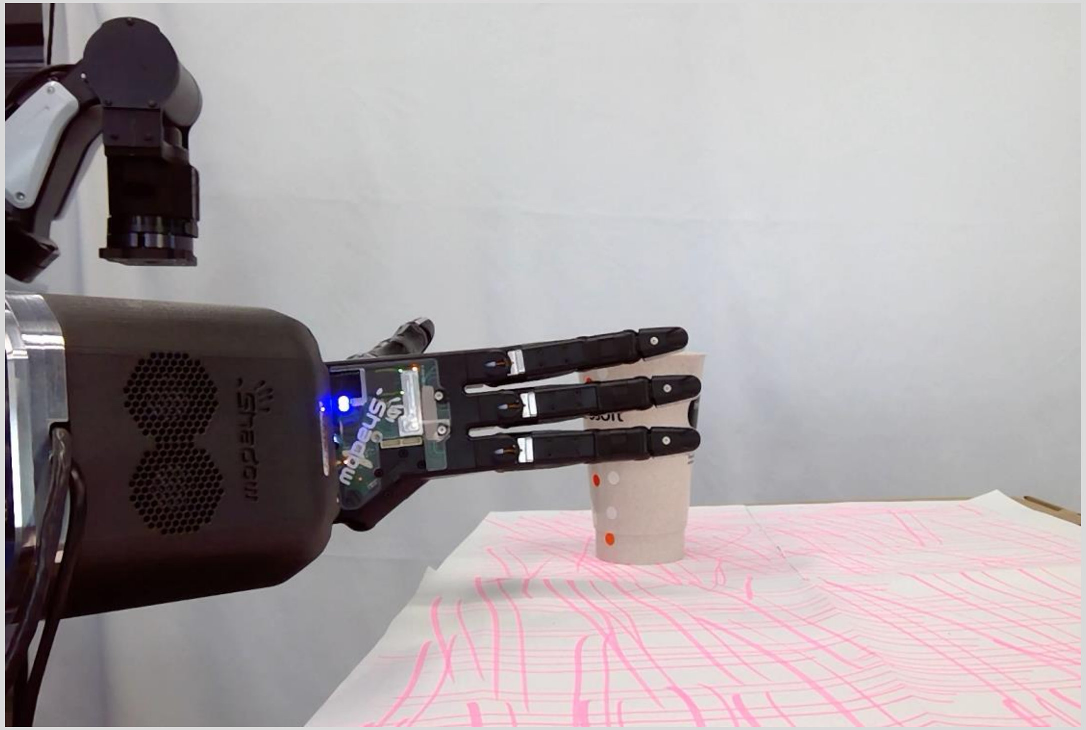
Train various objects of the same grasp



learn against different size and shapes



The same agent can grasp various shaped objects



Passive-force agent

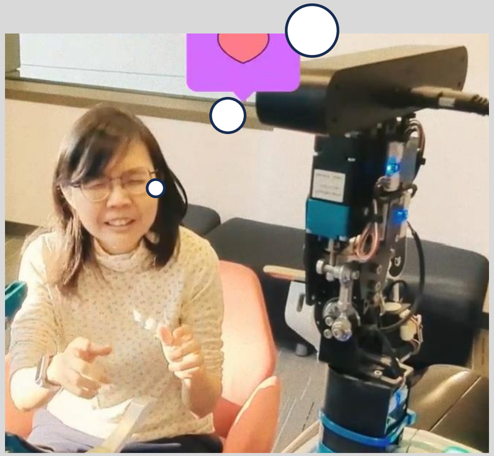
Active-force agent

The same object with two different agents

Errant robot project

Errand humanoid

Human order:
"Go to Kitchen and clean up the table"



ChatGPT interface inTSS

Skill library in TSS

- LfO generated program
- Symbolic map

Humanoid at Kitchen

Key components

Teaching

- Learning from observation for manipulation
- HoloLens based map-making & navigation

Execution

- ChatGPT-based program generation
- TSS execution platform with skill libraries

Microsoft

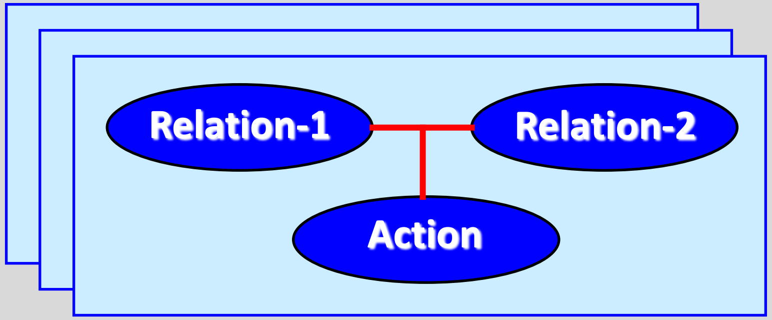
Project G.U. 自由

Adaptive Prompts for Onsite Teaching: Errand-Running Robots using LLM powered w/ HoloLens

Diachronic discussion

Learning-from-observation

Learning-from-observation



Hardware independent essence of the task

Observation

Direct mimicking does not work

Performance

How to obtain the essence?

Top down approach

- Design the task models based on Robotics theories

Bottom up approach

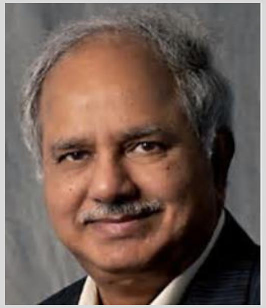
- Learn everything from scratch

Kanade's principle:

Do not apply learning approaches to those you can solve without them

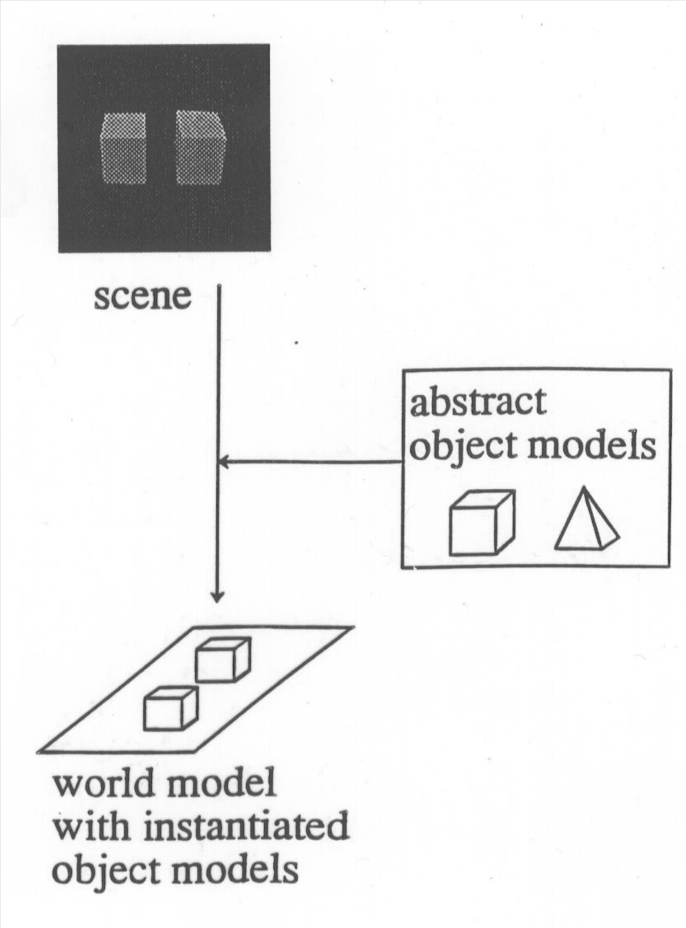
Top-down LfO: starting point

We started this effort 30 years ago at CMU!

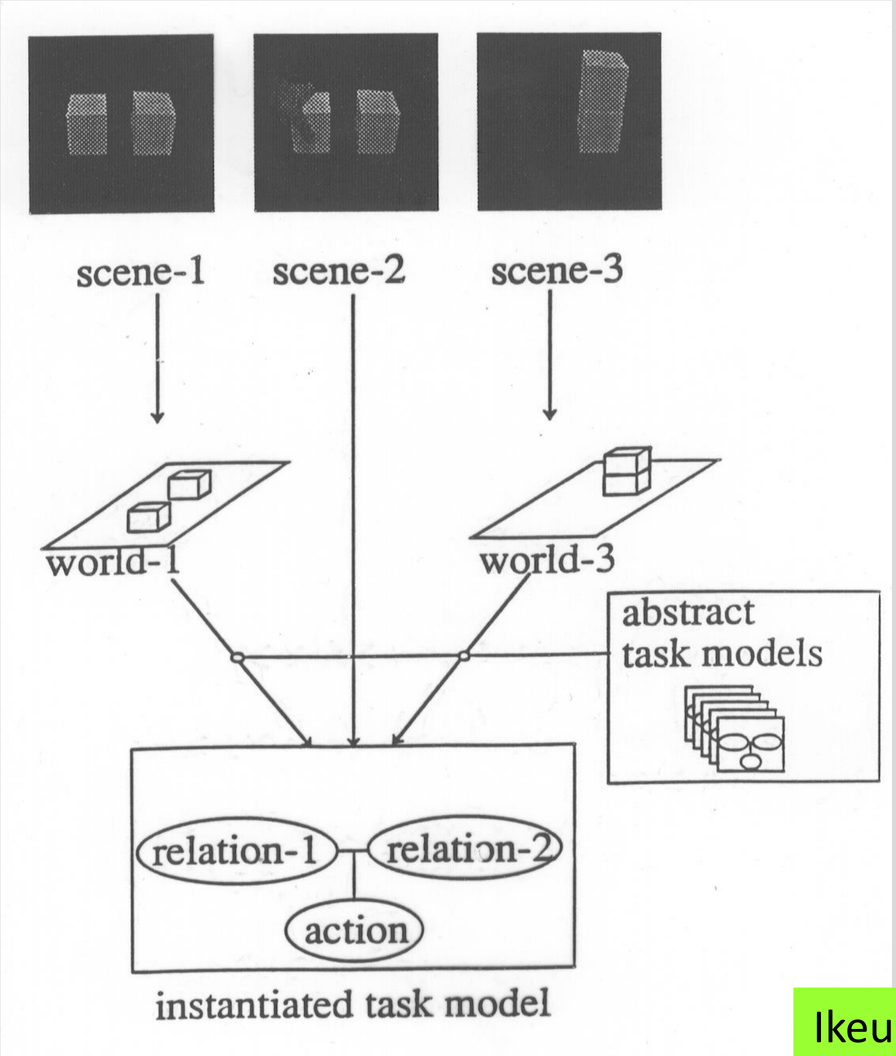


Ikeuchi & Reddy *CMU-RI* 89

Object recognition & Task recognition

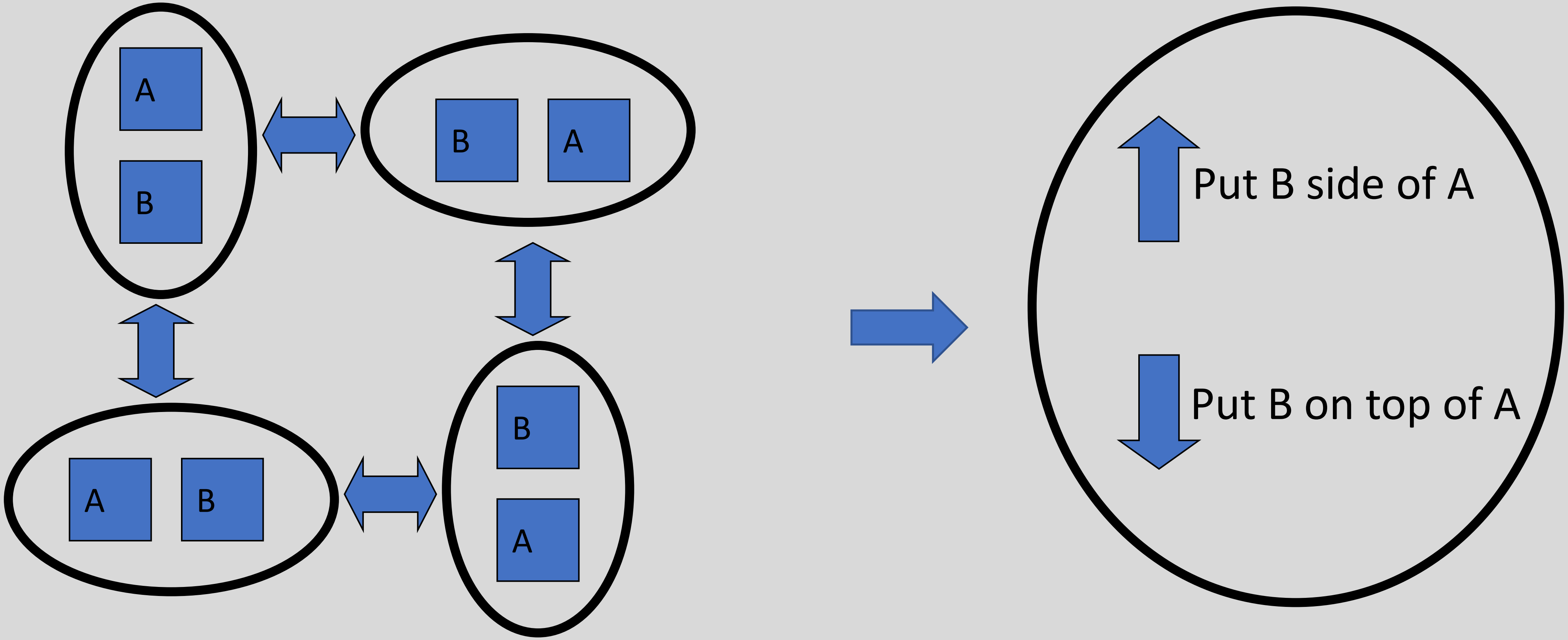


Marr *82*



Ikeuchi, Suehiro TRO94

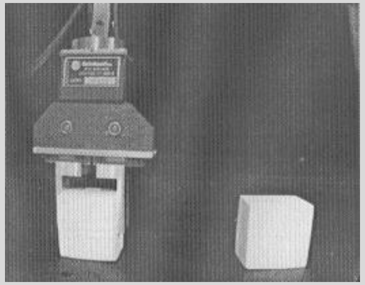
Key Idea: Essence = State transition



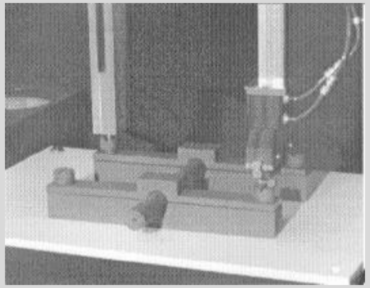
State transition **Necessary skills**

Domains explore the possible sets of states

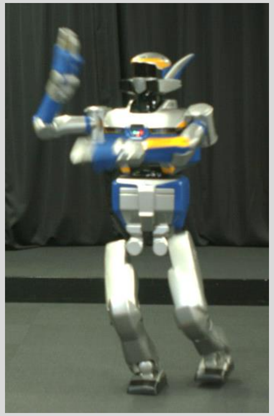
Two blocks



Machine parts

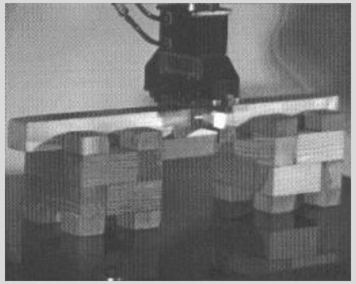


Dance

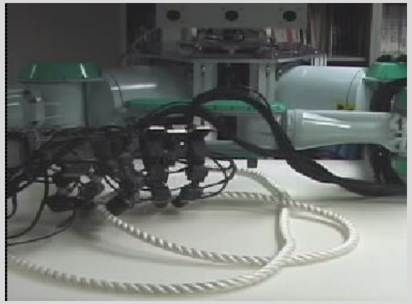


Household

1988 1990 2000 2010 2020



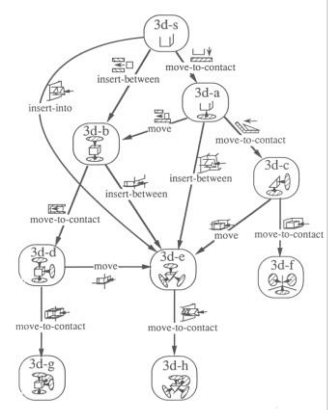
Polyhedron



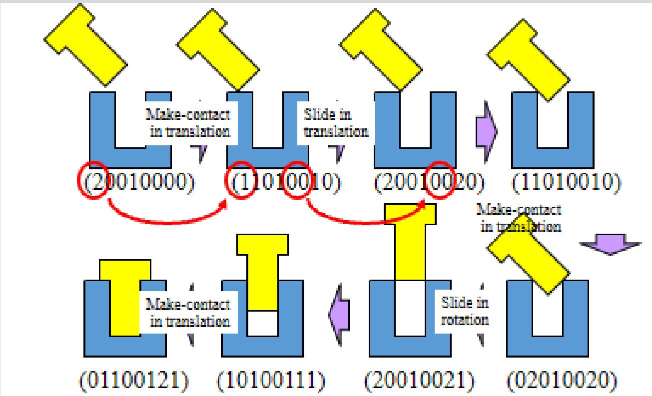
Rope

Gesture

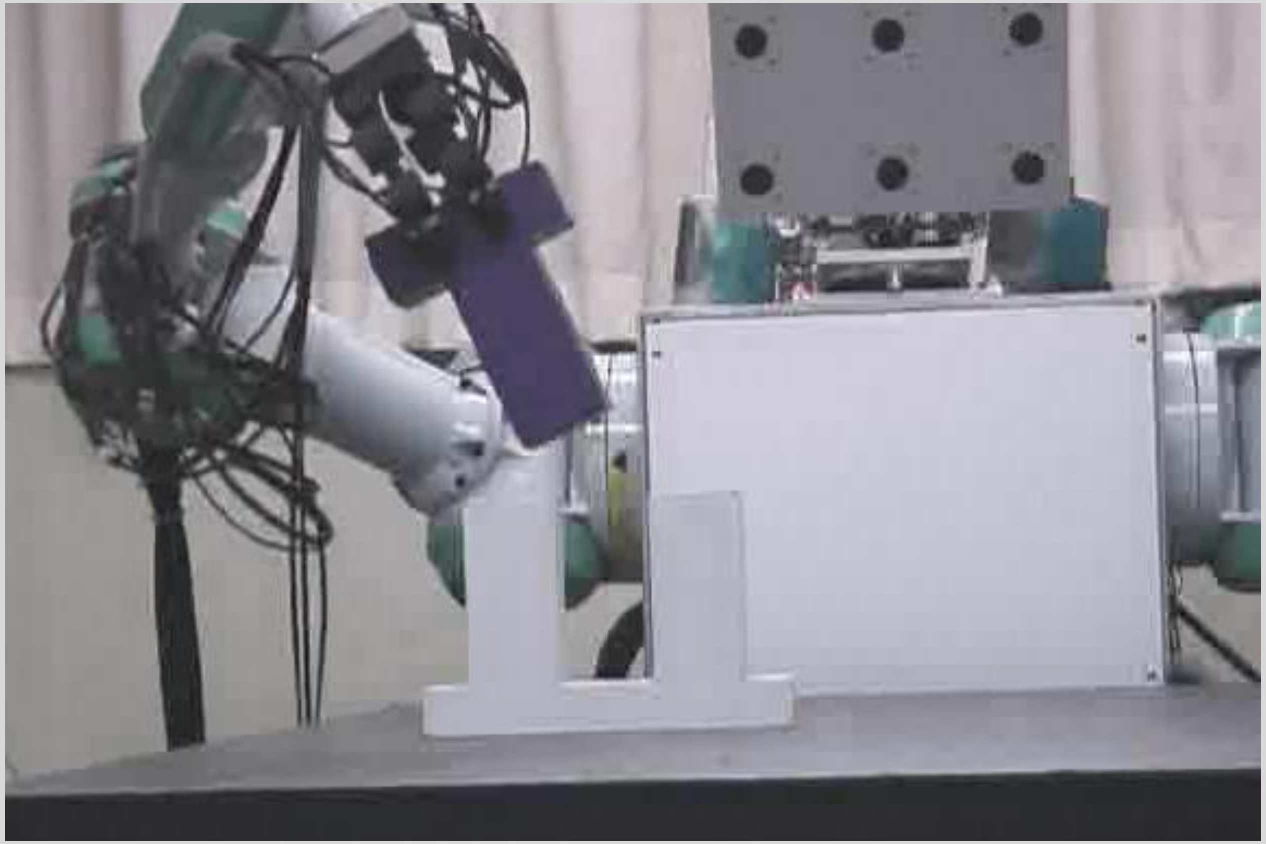
Polyhedral world: state = face-contact



Face-contact states

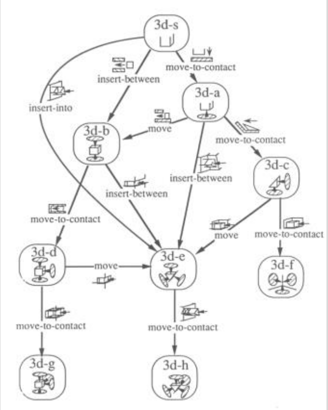


Extracted state transitions & required skills



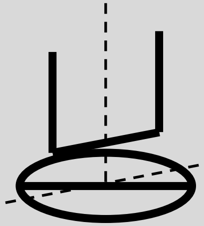
Takamatsu et.al. “assembly tasks” IJRR2007

Machine parts: States = Parts mating

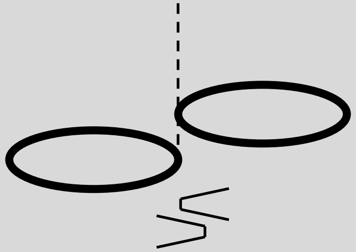


Polyhedral rep

+

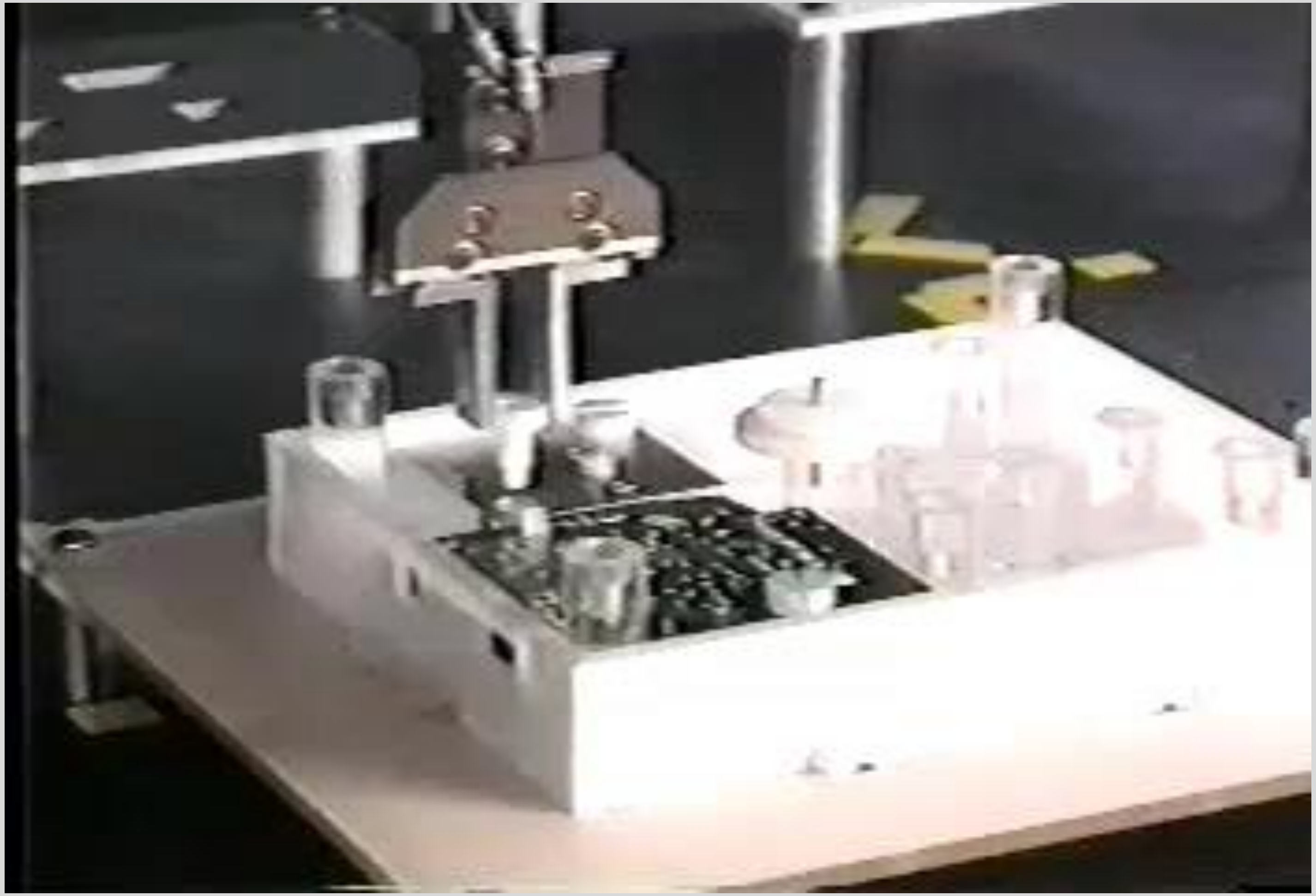


Screw-align



Gear-align

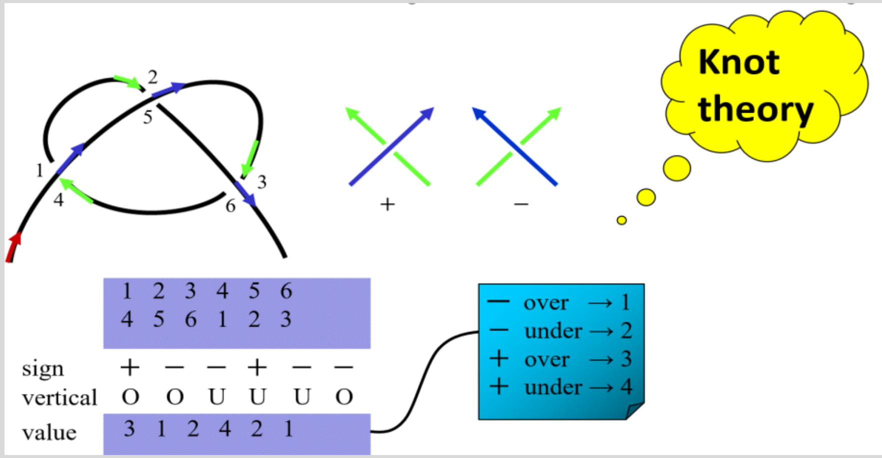
Parts mating



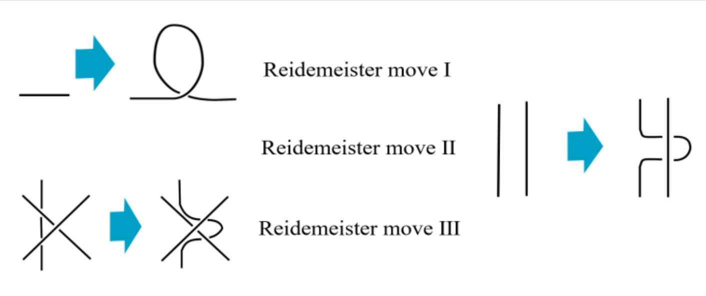
Miura & Ikeuchi Task PAMI98

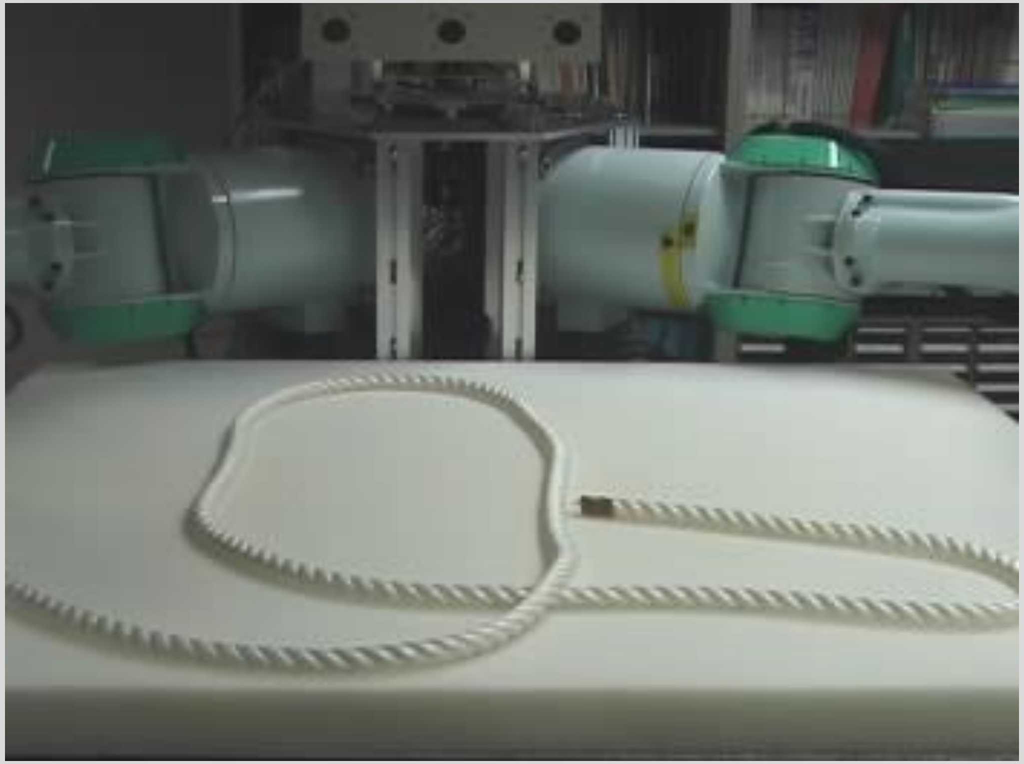
Knot-tying: Status = P-data in the knot theory

P-data representation



Reidemister move (action primitives)





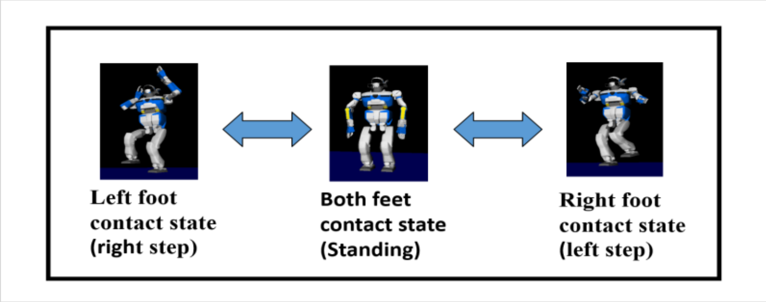
Execution mode

States = P data in the Knot theory

Takamatsu et. al., “Knot-tying,” TRA 2006

Human dance: State = Key pose & foot contact

Key pose (Labanotation)



Foot contact

Nakaoka et. al. ICRA2003

Synchronic discussion

Learning-from-observation

Imitation learning vs LfO

Human demonstration

- Pick up a dish

<bring trajectory>

- Pick up a sponge

<bring trajectory>

- Wipe

<bring trajectory>

- Place the dish

<bring trajectory>

- Place the sponge

Imitation learning = mimic all trajectories

- **It works only when the object and the environment are exactly same because the system mimics all the trajectories**
- **Error by demonstrator will be mimicked -> fatigue of the operator**

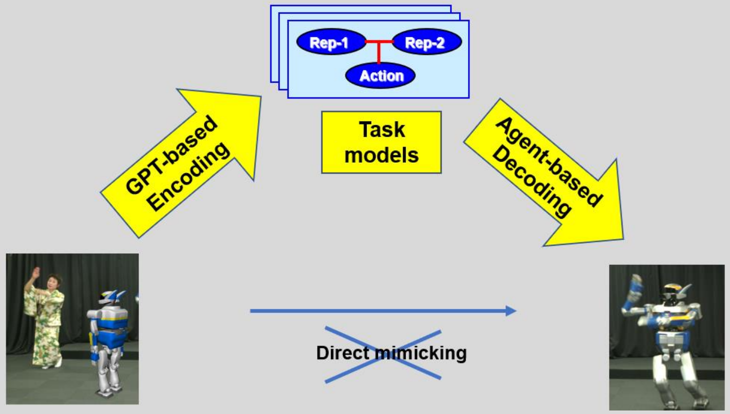
Learning from observation

- **Only mimic where it important**

LfO = symbolic teleoperation

- Task-encoding based on GPT: what-to-do & where-to-do

- Task-decoding based agents: how-to-do



Symbolic representation

Human demo

Forward pretrained model (GPT-base task encoding)

Cerebrum (?)

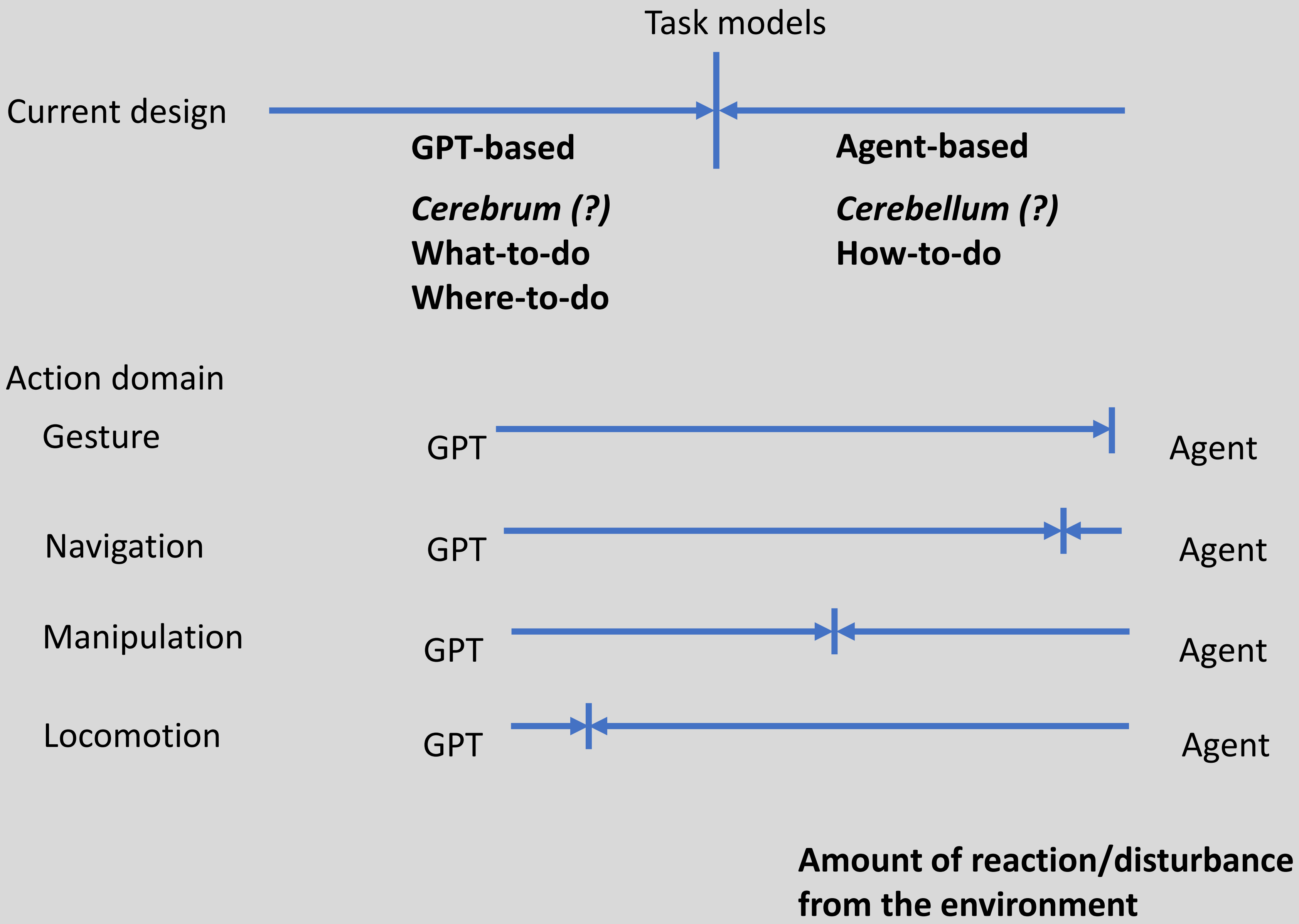
Pretrained skill agents (RL-based task decoding)

Cerebellum (?)

Robot exec

?

Cerebrum vs Cerebellum



Defense of only vision (not force)

- In children's imitation, the connection with the mother is limited to the visual world only
- Visual observation is done through Piaget's *schema*; not entire actions
  - in LfO, Affordance analyzer using Minsky's frame (Task model)
- Force information is not shared; force feedback is learned separately through *circular reactions*

(Reinforcement learning (?))

LfO and Piaget's theory

- Sensormotor stage
  - Physical sensations
  - Coordinating their body

2

- Preoperational stage
  - Symbolic thought
  - Ego-centric view

7

- Concrete operational stage
  - Logical thought
  - Decentering view

11

- Formal operational stage
  - Scientific reasoning

- **Circular reactions**
  - **Repeat same actions**
    - **→ Hand-eye calibration (?)**
    - **→ Reinforcement learning (?)**
- **Imitation behavior**
  - **Hand actions & face expressions**
    - **→ Learning-from-observation (?)**

Summary

- Learning from observation
    - GPT-based encoding
    - TSS-skill library
        - Manipulation agent library
        - Grasp agent library
- Diachronic discussion
- Synchronic discussion

Recent publication & Team

GPT-encoder

- *Wake et. al. : arXiv:2311.12015 (2023)*
- *GPT-4V(ision) for robotics*

Task/skill model design

- *Ikeuchi et.al.: IJRR (2024)*
- *Semantic constraints to represent common sense*

• *Skill model training*

- *Takamatsu et. al.: arXiv:2403.02316(2024)*
- *Designing library of skill-agents for hardware-level reusability*

• *TSS-Platform*

- *Sasabuchi el.al.: IEEE RAL (2020)*
- *Task-oriented motion mapping on robots*

Katsu Ikeuchi

Jun Takamatsu

Kazu Sasabuchi

Naoki Wake

Atsu Kanehira