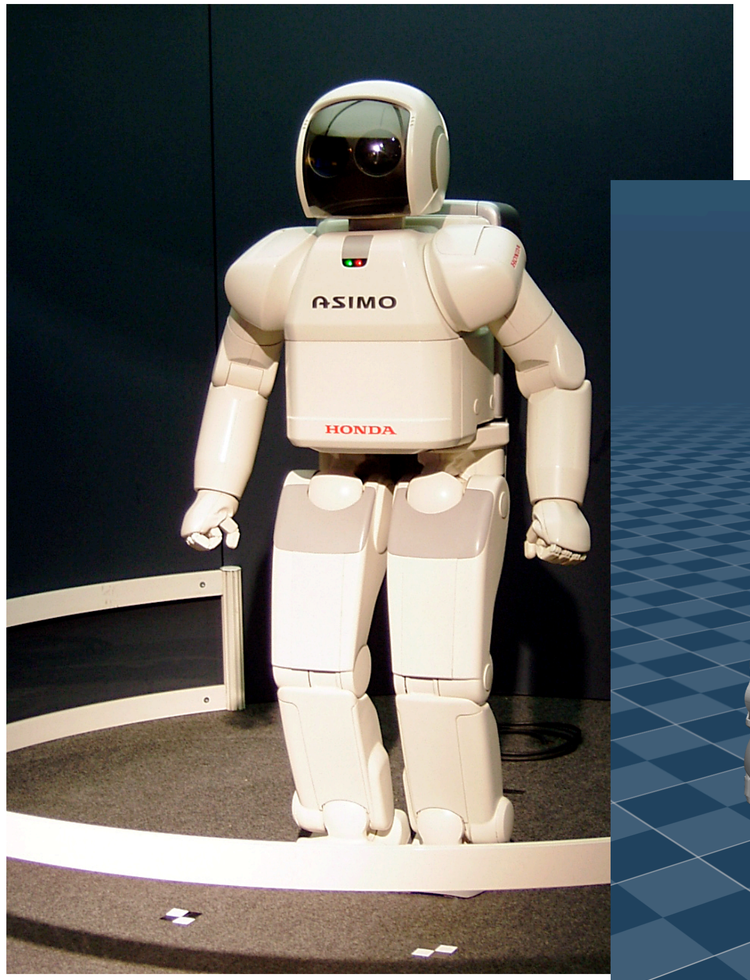
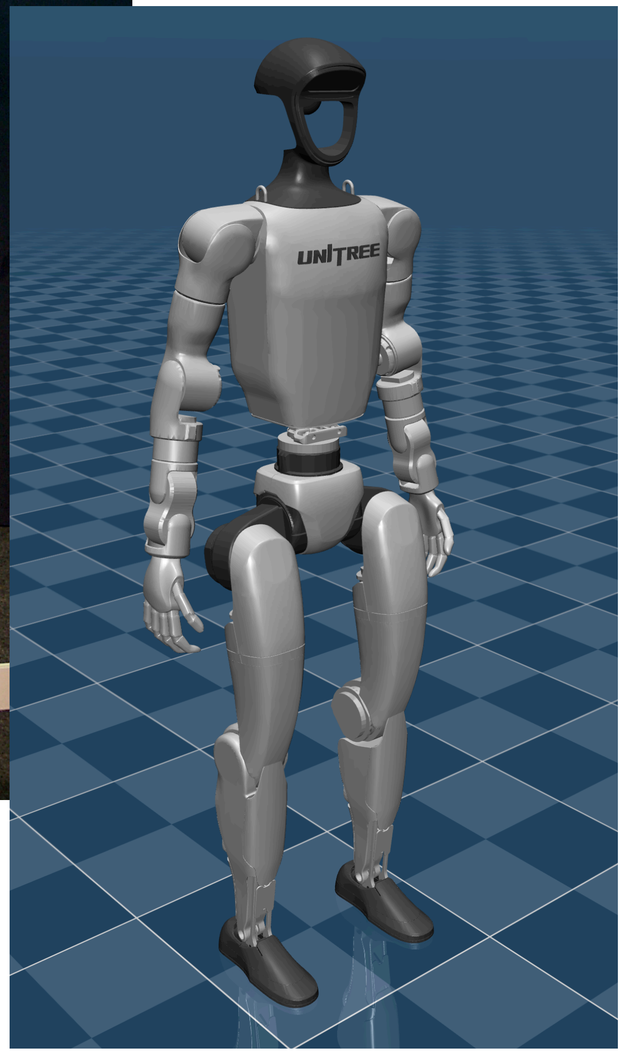
ADVANCES IN
COMPUTER
VISION

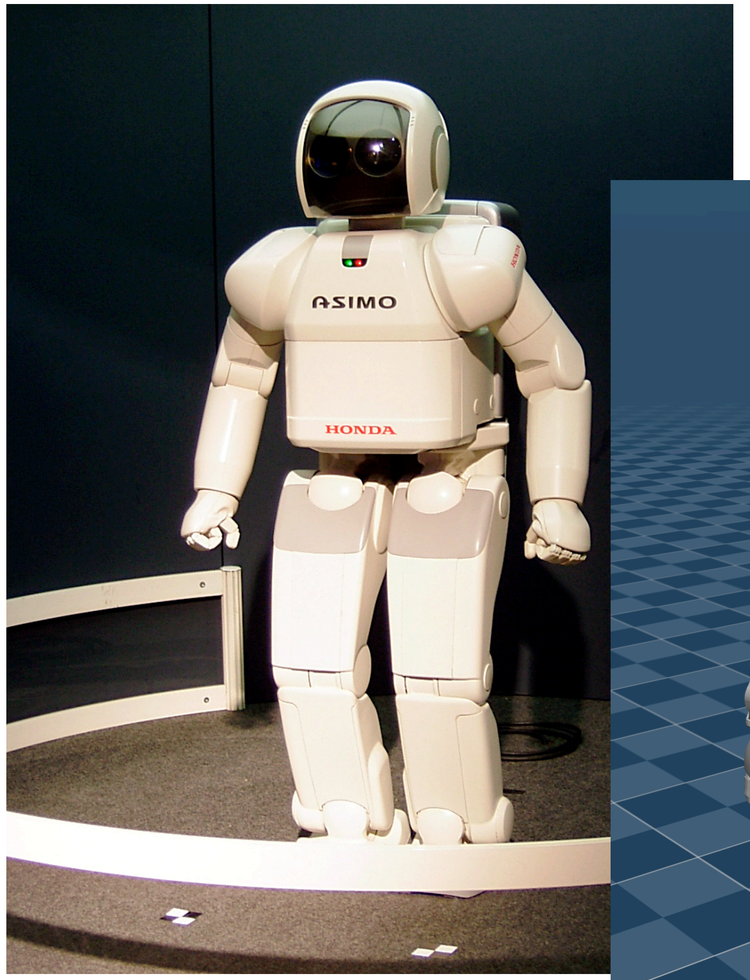
Module 3:
Vision for Embodied Agents

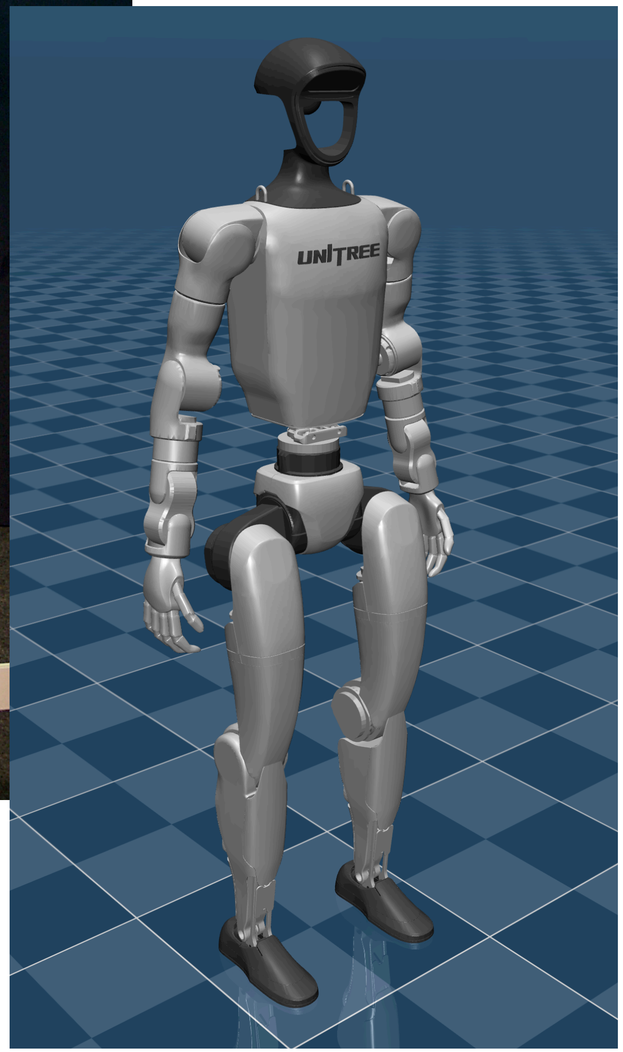
Prof. Vincent Sitzmann

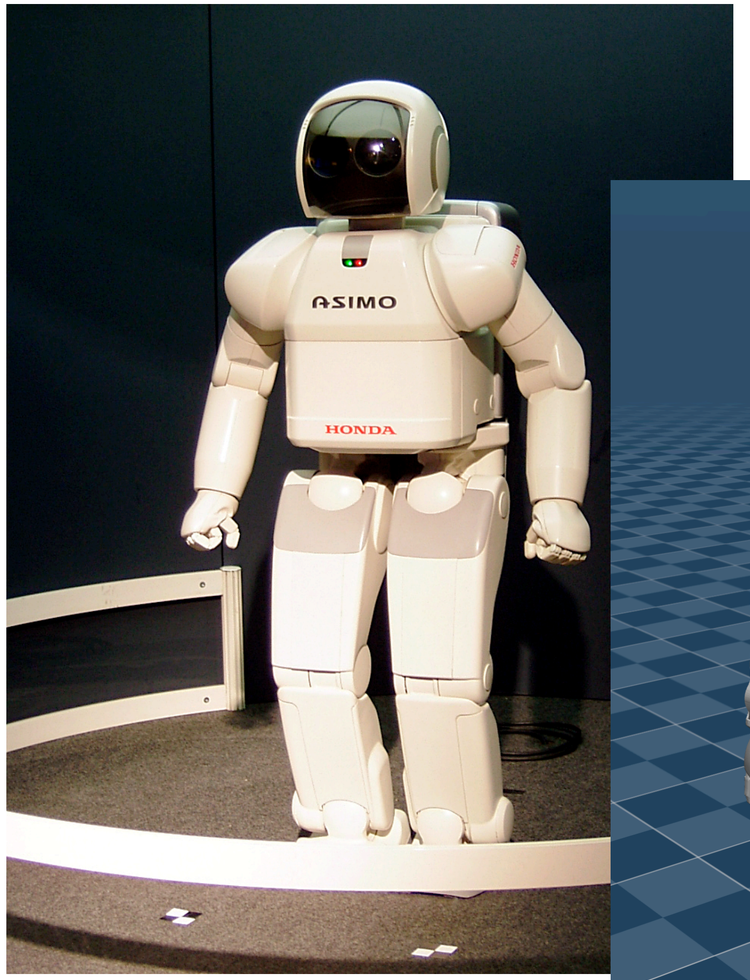

ASIMO
HONDA

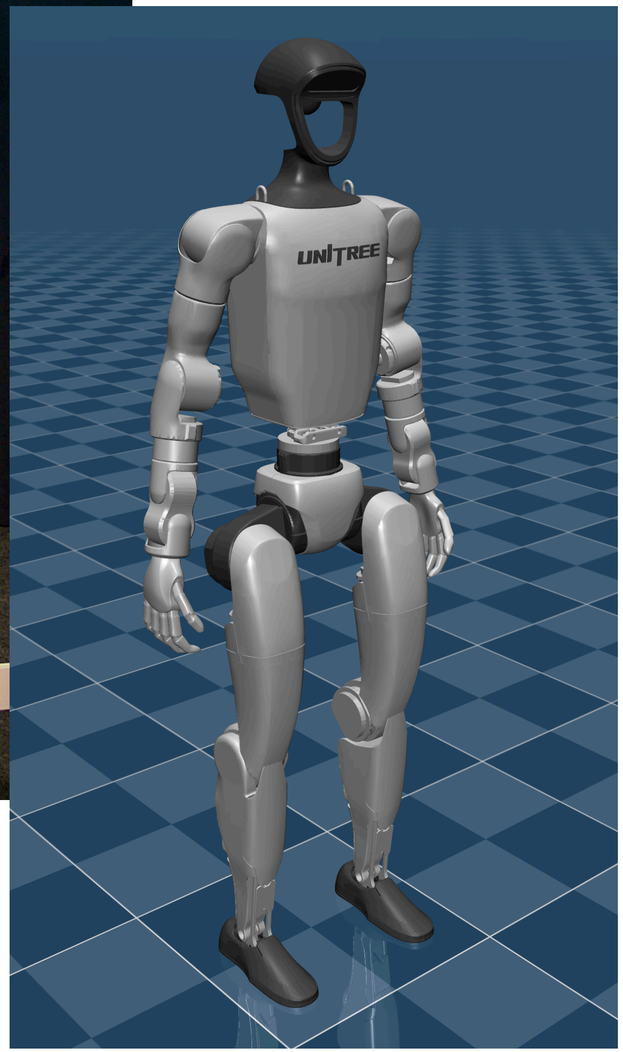

UNITREE

What is an “embodied agent”?

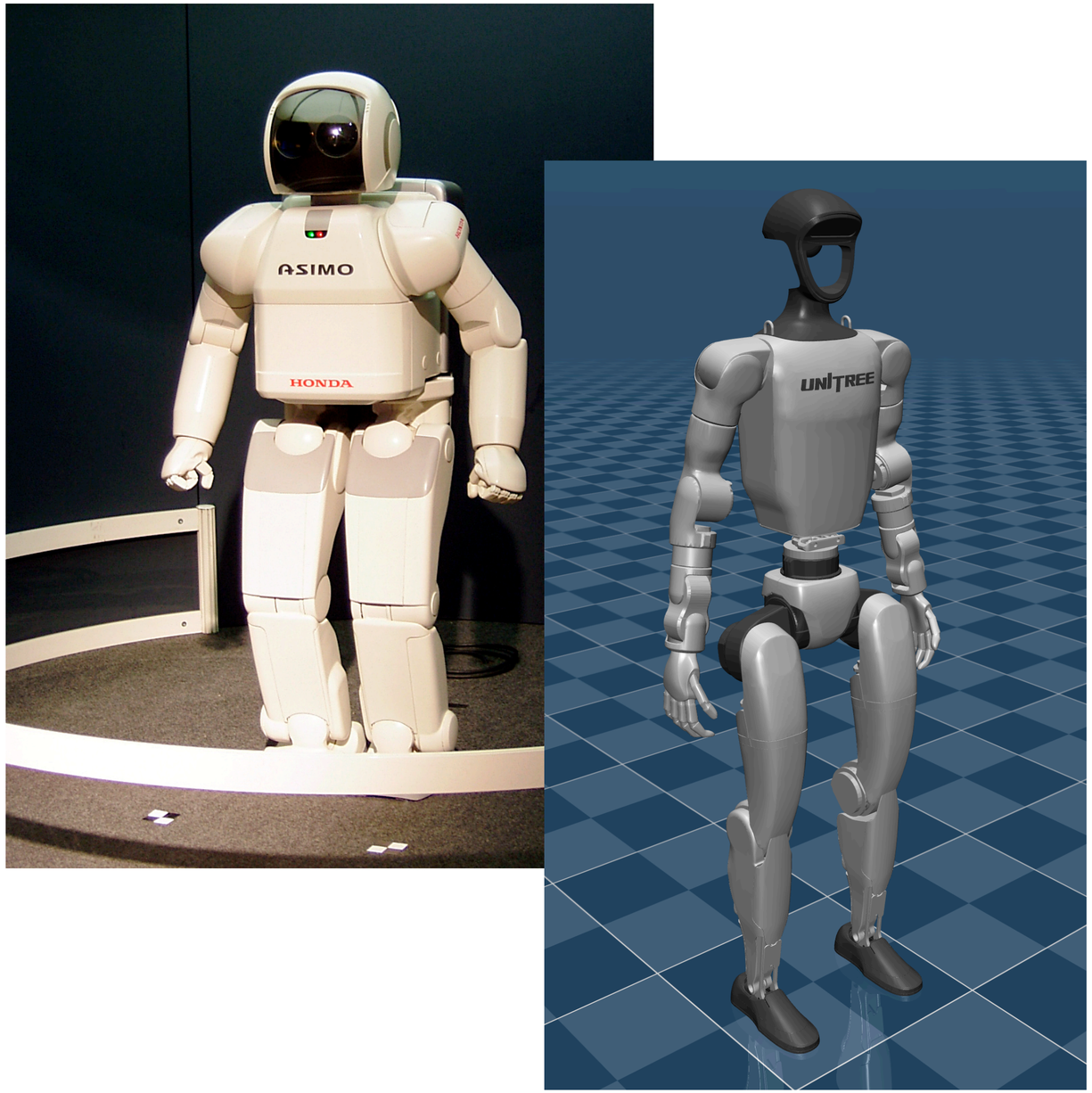


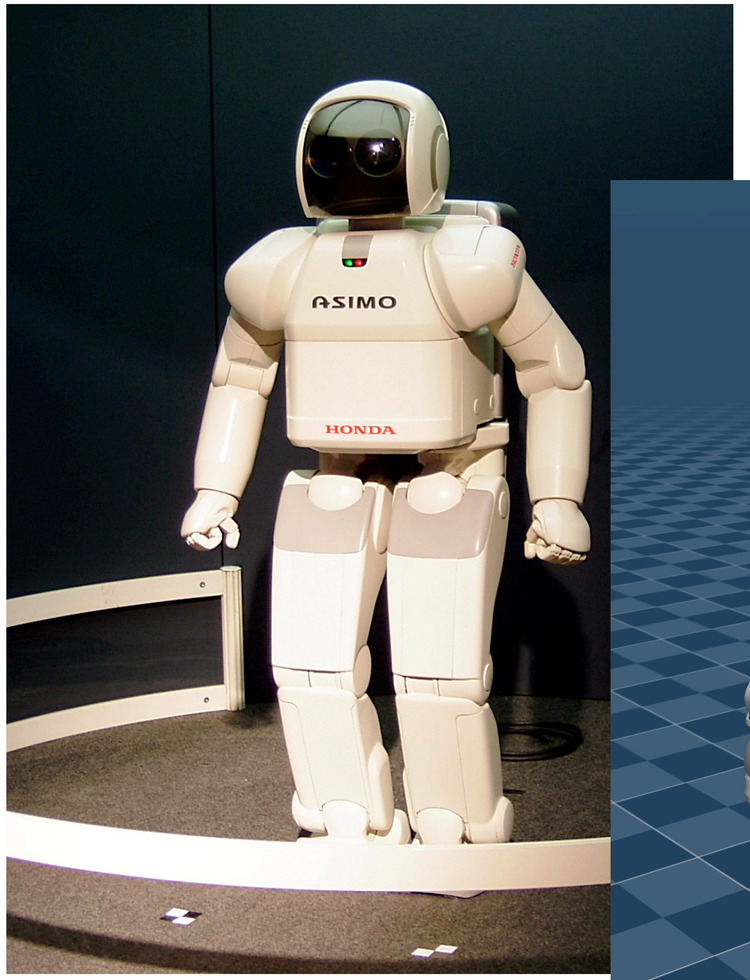


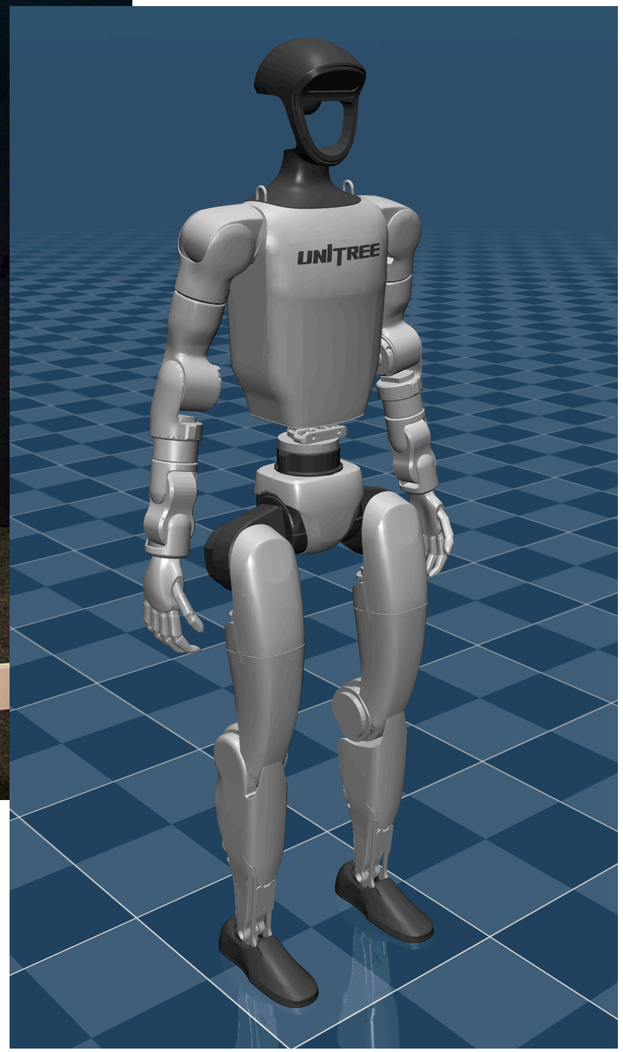

ASIMO
HONDA


UNITREE

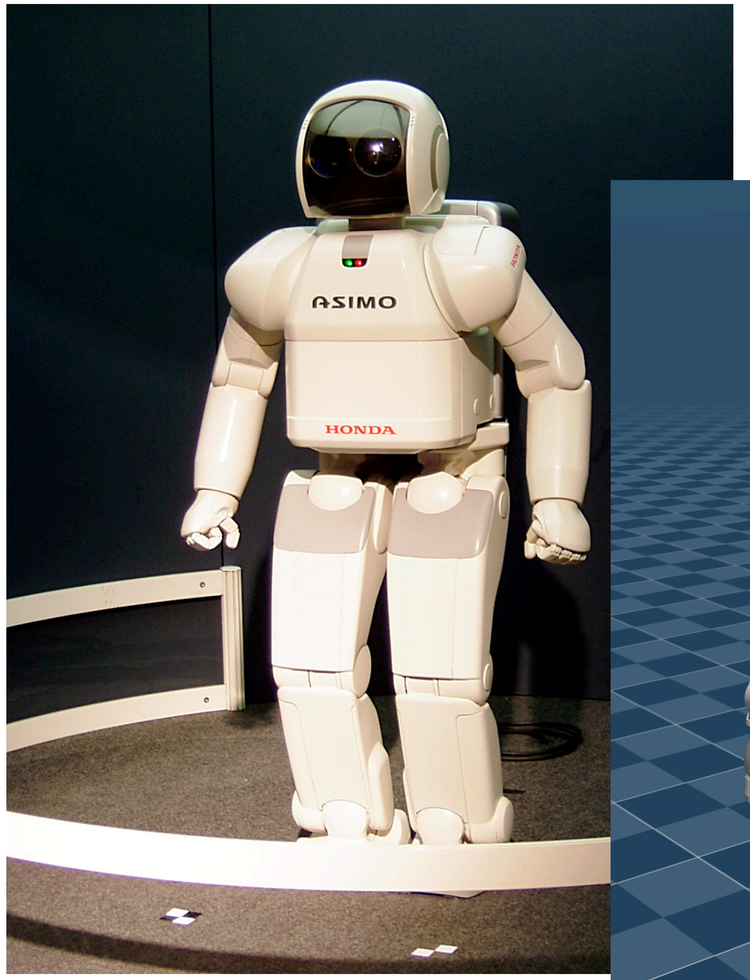
Anything that has to make decisions and act independently

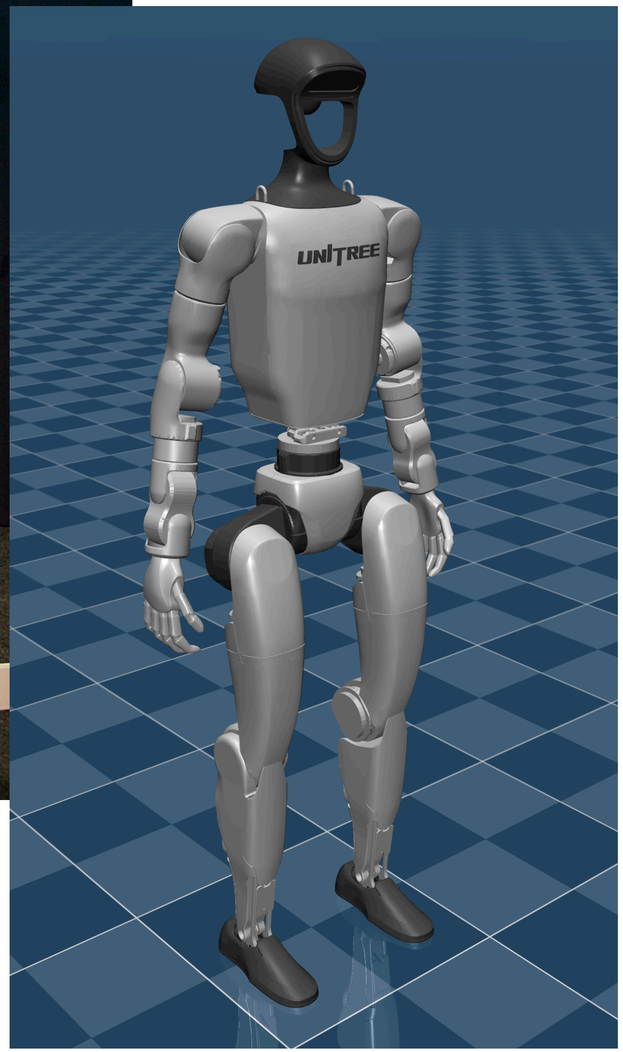



ASIMO
HONDA


UNITREE

Similar to “robot”, but has the connotation that it can also be embodied in a simulation





CSAIL Embodied Intelligence

Home | People | Labs | Seminars | Calendar | Contact

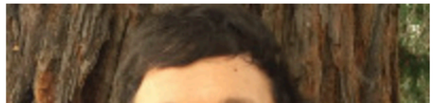
PIs

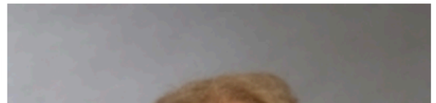
Ted Adelson
adelson [at] csail.mit.edu

Pulkit Agrawal
pulkitag [at] csail.mit.edu

Jacob Andreas
jda [at] mit.edu

Sara Beery
beery [at] mit.edu





What are common problems with embodied agents specifically?

Closely related to lots of vision / perception problems!

What are common problems with embodied agents specifically?

- **Environment state estimation:** what am I looking at, what are the physical properties of things around me

Closely related to lots of vision / perception problems!

What are common problems with embodied agents specifically?

- **My own state estimation:** Where am I, what pose is my body in

Closely related to lots of vision / perception problems!

What are common problems with embodied agents specifically?

- **Simulation:** what will happen next, what would happen if I took action X

Closely related to lots of vision / perception problems!

What are common problems with embodied agents specifically?

- **Planning:** what actions do I have to take to accomplish goal X

Closely related to lots of vision / perception problems!

What are common problems with embodied agents specifically?

- **Control:** Given intended trajectory and my initial action plan, how do I follow the intended trajectory while regulating velocity, acceleration etc

Closely related to lots of vision / perception problems!

https://commons.wikimedia.org/wiki/File:Robot_vacuum_cleaner_-_Japan_-_2024_June_24.webm
User Nesnad

https://commons.wikimedia.org/wiki/File:Robot_vacuum_cleaner_-_Japan_-_2024_June_24.webm
User Nesnad

Vision for Self-State Estimation & Control

Prof. Vincent Sitzmann

What do you think is your visual representation of your own embodiment?

Look at your hand and think about how it will move when you activate different muscles?

1. Scan Scene

2. Train NeRF
and Distill Features

3. Language-Guided
Manipulation

Distilled Feature Fields Enable Few-Shot Manipulation, Shen et al., 2023

What is missing in the 3D Scene Repr.?

1. Scan Scene

2. Train NeRF
and Distill Features

3. Language-Guided
Manipulation

Distilled Feature Fields Enable Few-Shot Manipulation, Shen et al., 2023

What is missing in the 3D Scene Repr.?

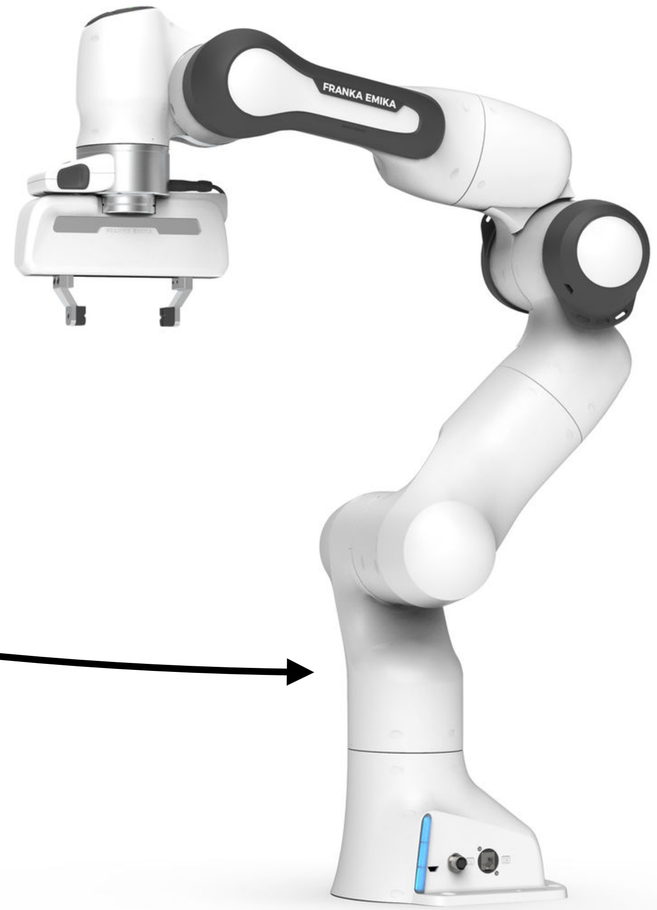
1. Scan Scene

2. Train NeRF
and Distill Features

3. Language-Guided
Manipulation

Distilled Feature Fields Enable Few-Shot Manipulation, Shen et al., 2023

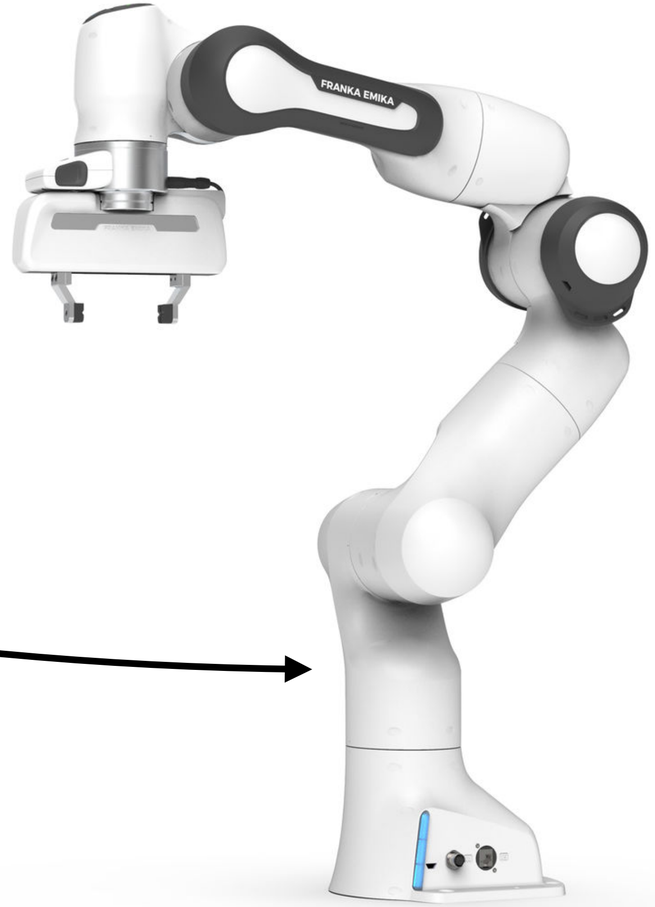
How do Robots Work Today?



Discrete Joints with precise sensors

Rigid Segments

How do Robots Work Today?



Discrete Joints with precise sensors

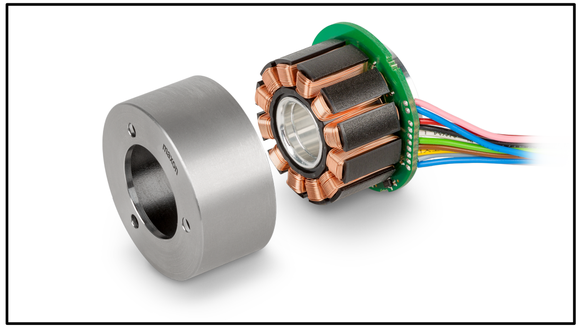
Rigid Segments

Manipulator State Estimation is Almost Exclusively “Blind”!

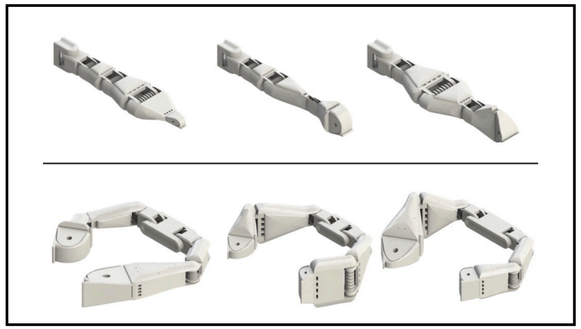
This makes robots expensive!

Shadow Robot Hand
$150,000 - $200,000

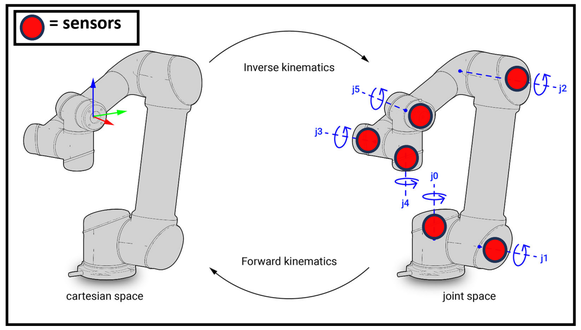
Universal Robot UR5
$32,000 - $35,000



precision sensors and motors

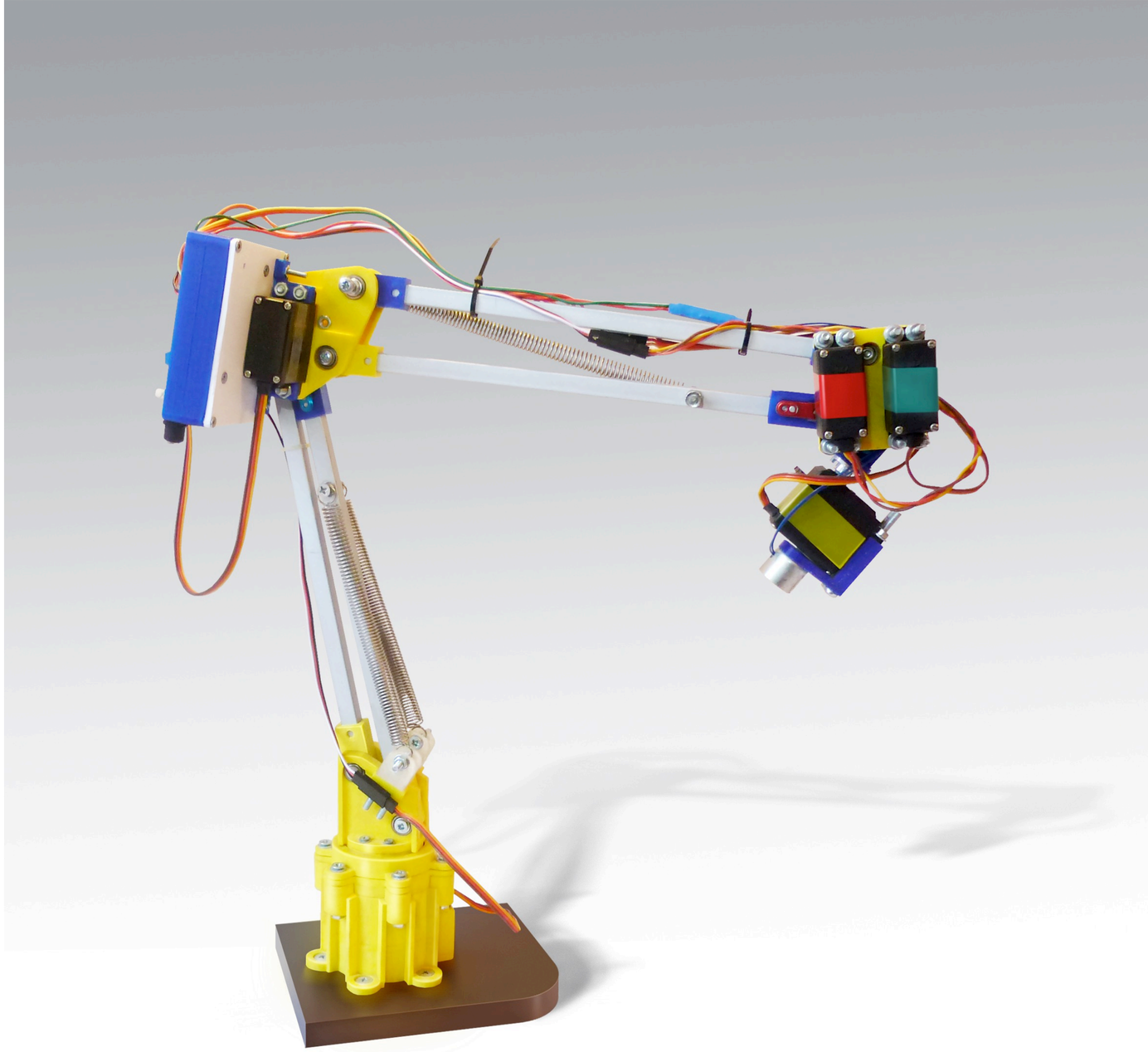


limited design space



rigidity assumption

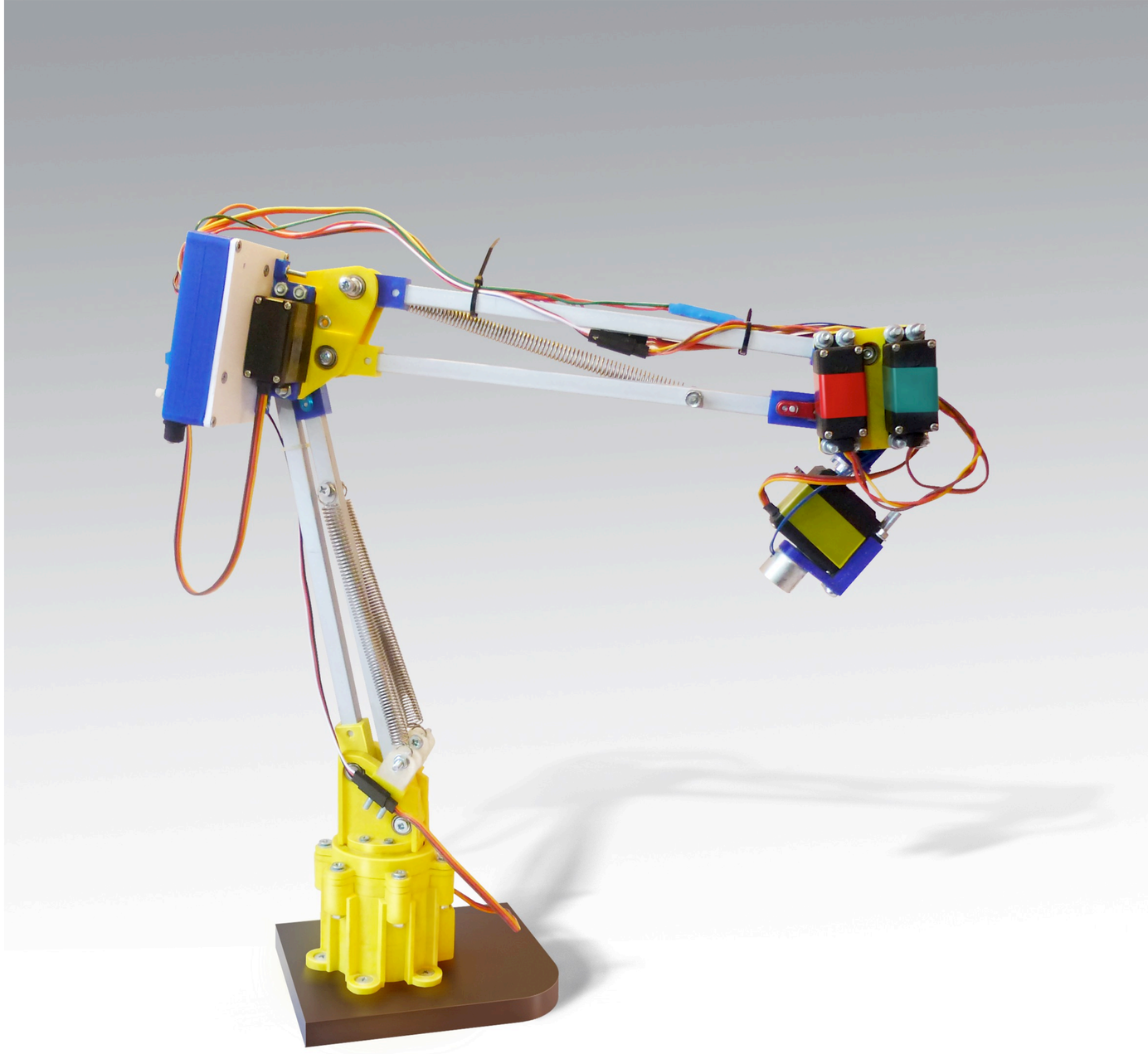
Credit: Shadow Robot Company, Universal Robot



TERTIARM

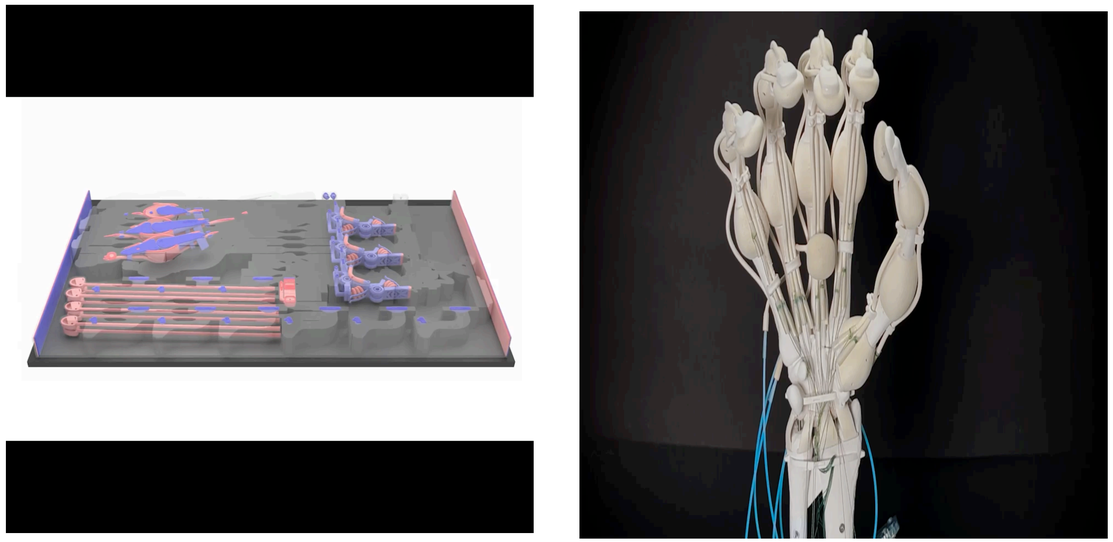
Could we use this robot today?

Yes and no;

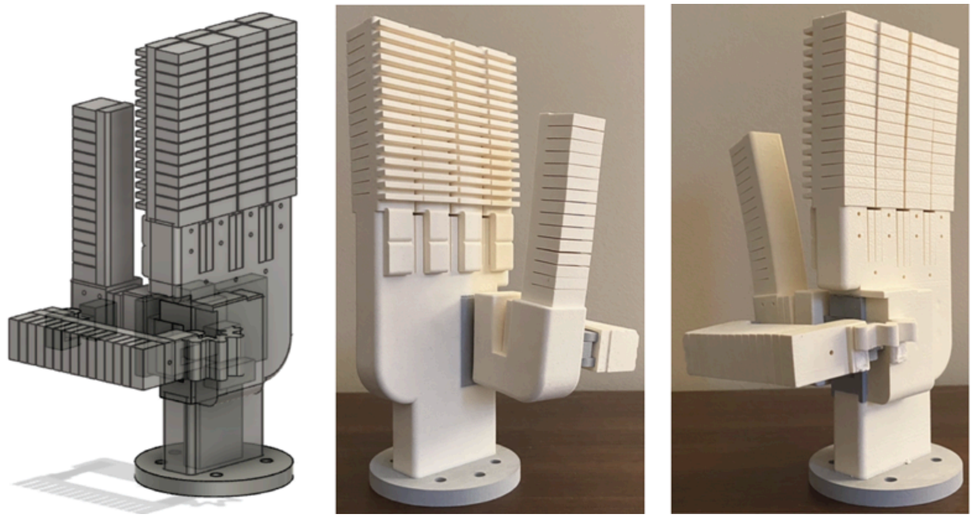


TERTIARM

Existing cheap, 3D-printed, soft, etc robots are practically useless: can’t model and control them!



3D printing of a pneumatic hand
MIT, Inkbit, ETH Zurich, Nature 2023

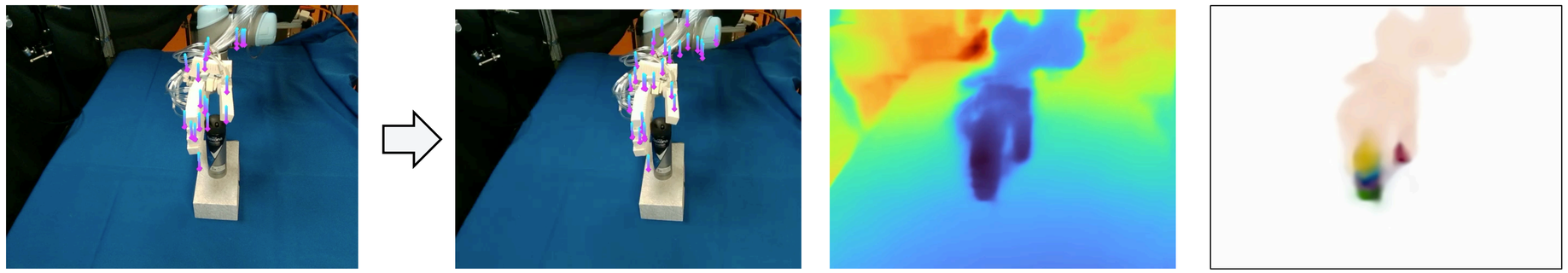


Directly 3D printed multi-material hand
Matusik et al., ICRA 2024

Robot State Estimation & Control From Vision Alone

Unifying 3D Representation and Control of Diverse Robots with a Single Camera, Li et al. 2024

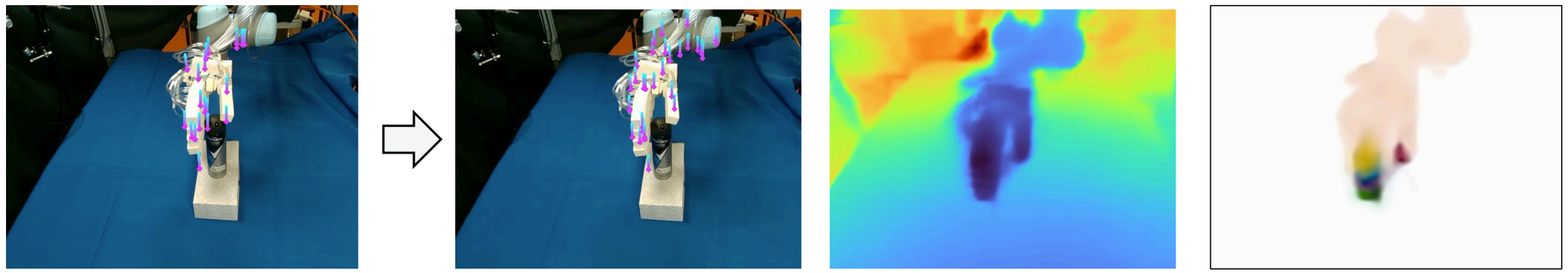
Desired Motion | **Predicted Motion** | **Predicted Depth** | **Predicted Jacobian**



Robot State Estimation & Control From Vision Alone

Unifying 3D Representation and Control of Diverse Robots with a Single Camera, Li et al. 2024

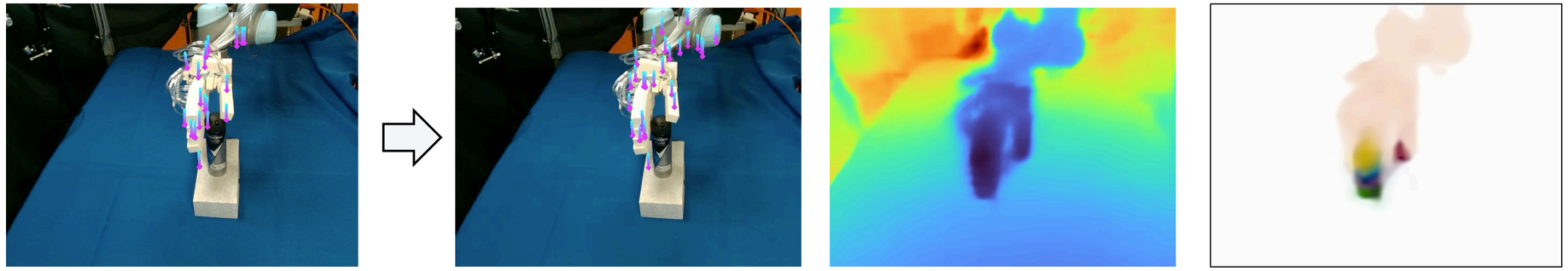
Desired Motion | **Predicted Motion** | **Predicted Depth** | **Predicted Jacobian**



Robot State Estimation & Control From Vision Alone

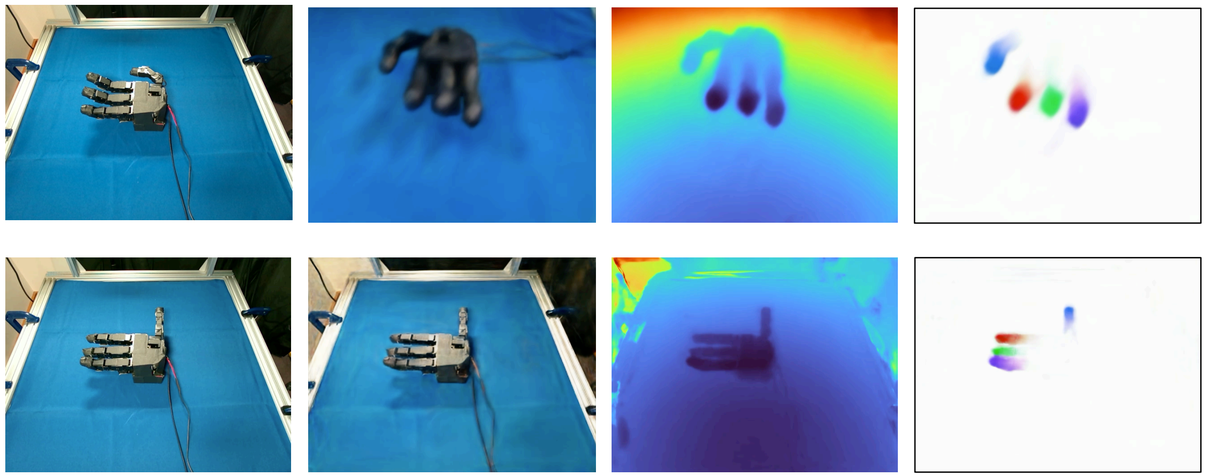
Unifying 3D Representation and Control of Diverse Robots with a Single Camera, Li et al. 2024

Desired Motion | **Predicted Motion** | **Predicted Depth** | **Predicted Jacobian**



Rigid System

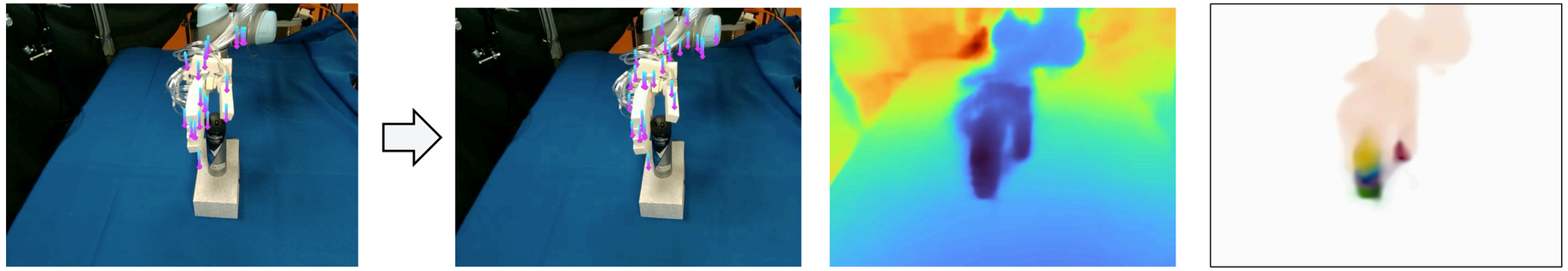
Input RGB | **Predicted RGB** | **Predicted Depth** | **Predicted Jacobian**



Robot State Estimation & Control From Vision Alone

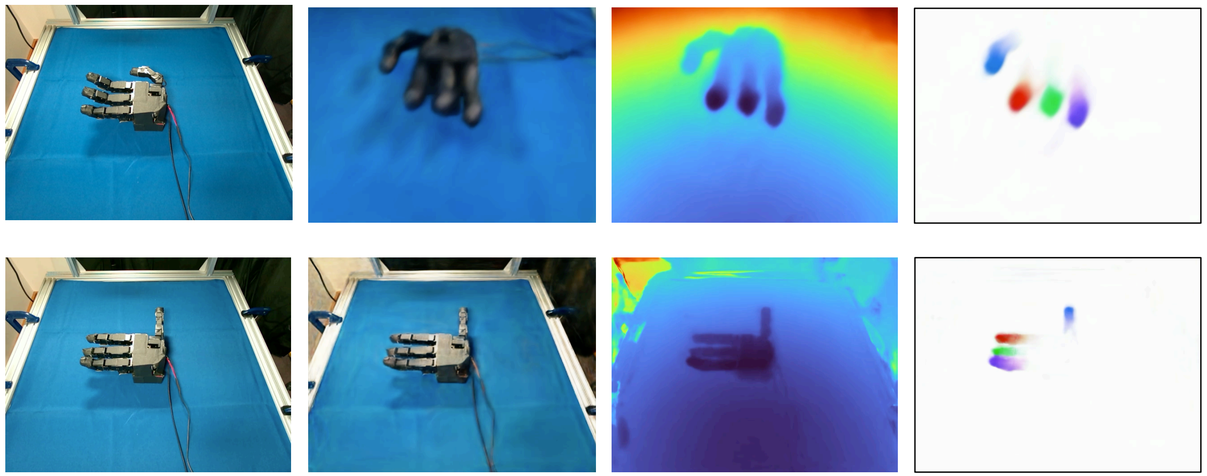
Unifying 3D Representation and Control of Diverse Robots with a Single Camera, Li et al. 2024

Desired Motion | **Predicted Motion** | **Predicted Depth** | **Predicted Jacobian**



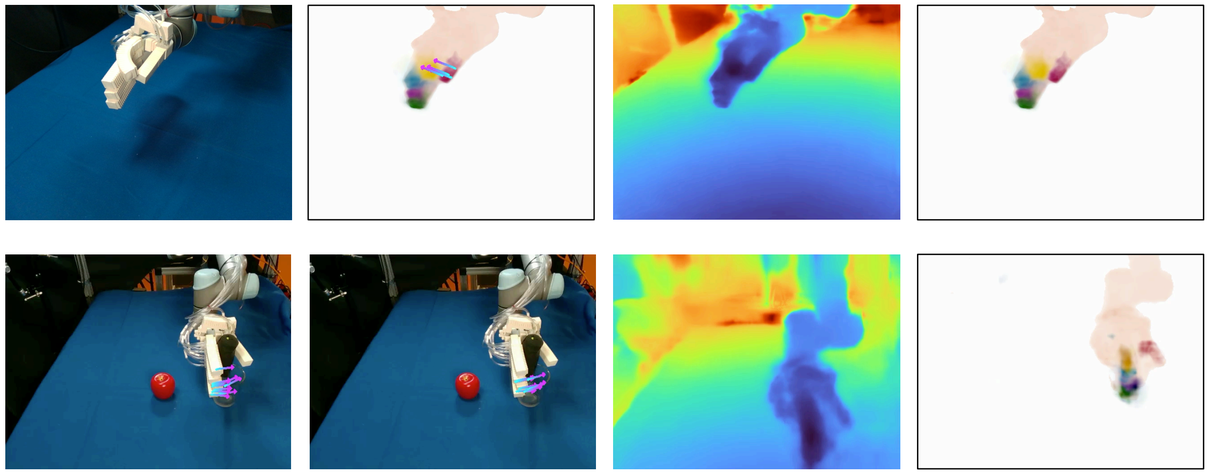
Rigid System

Input RGB | **Predicted RGB** | **Predicted Depth** | **Predicted Jacobian**

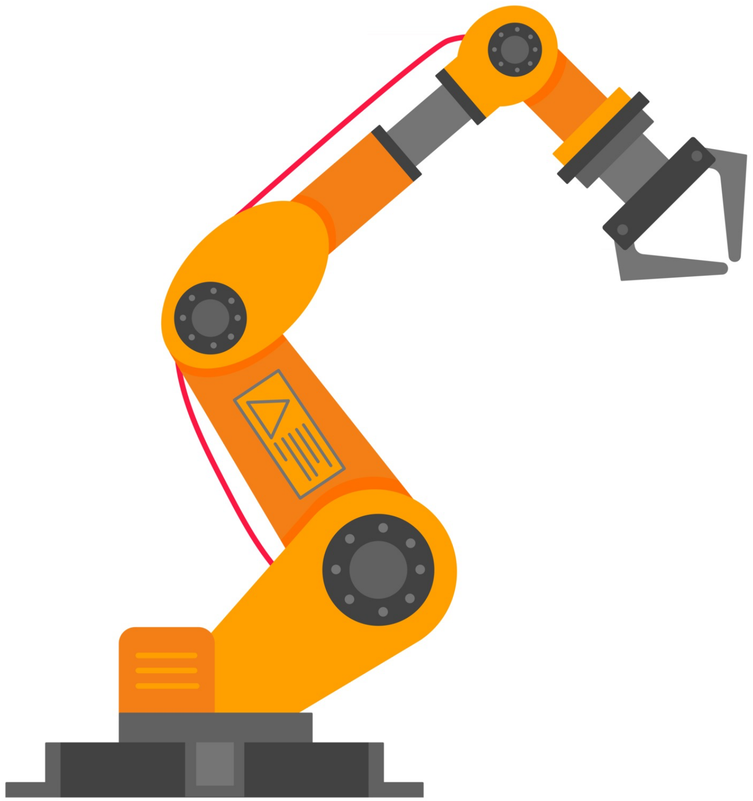


Soft System

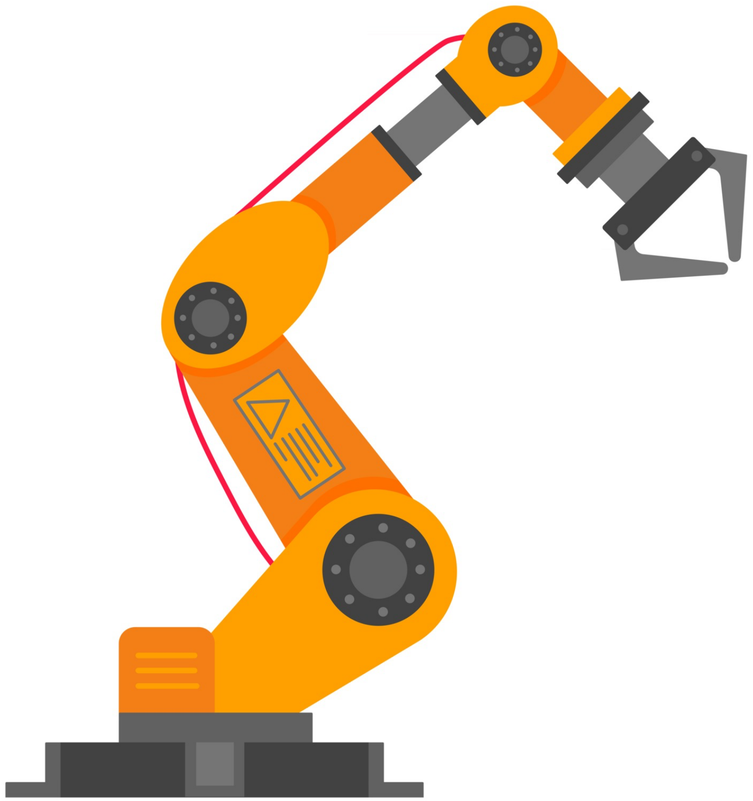
Input RGB / Motion | **Predicted Motion** | **Predicted Depth** | **Predicted Jacobian**



Background: The Body Jacobian

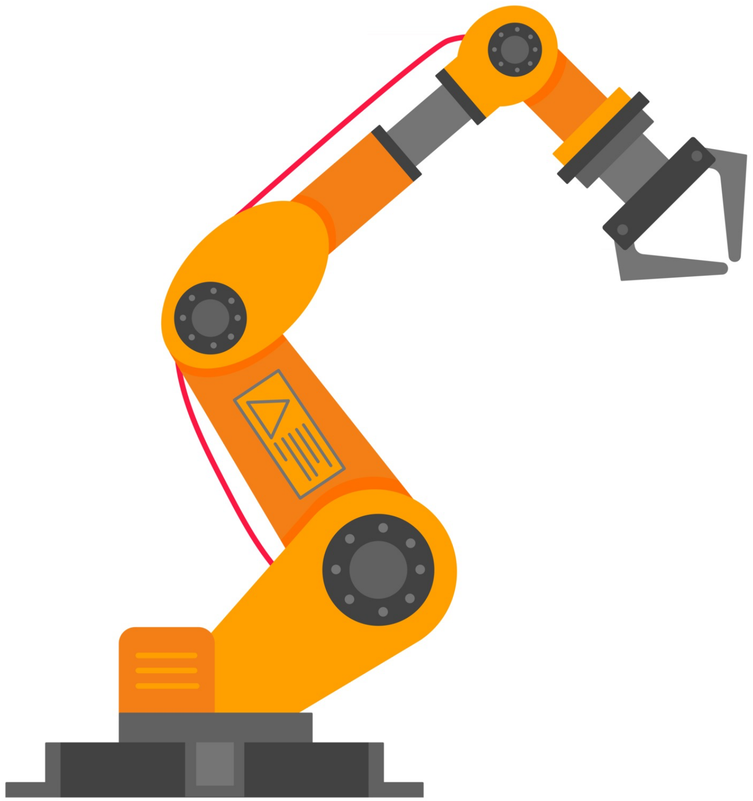


Background: The Body Jacobian



Robot with points in 3D $\mathbf{x}$ and n joint angles $\mathbf{q}$.

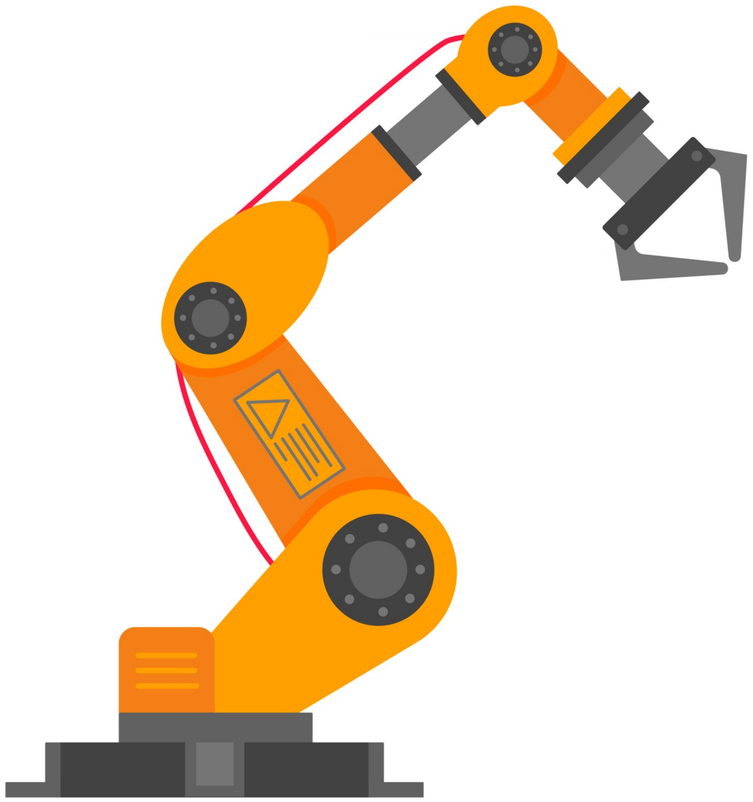
Background: The Body Jacobian



Robot with points in 3D $\mathbf{x}$ and n joint angles $\mathbf{q}$.

For *any point* $\mathbf{x}$ on the robot, we can find the *Body Jacobian,* defined as the $\mathbb{R}^{n\times 3}$ matrix $\mathbf{J}(\mathbf{q})$:

Background: The Body Jacobian

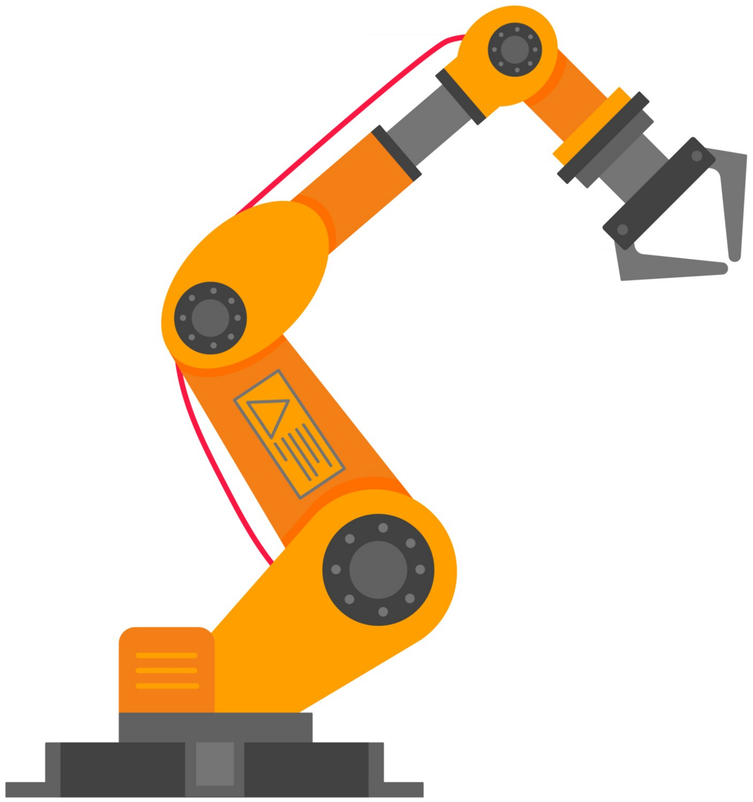


Robot with points in 3D $\mathbf{x}$ and n joint angles $\mathbf{q}$.

For *any point* $\mathbf{x}$ on the robot, we can find the *Body Jacobian,* defined as the $\mathbb{R}^{n\times 3}$ matrix $\mathbf{J}(\mathbf{q})$:

$$\begin{bmatrix} \dot{x} \\ \dot{y} \\ \dot{z} \end{bmatrix} = \mathbf{J}(\mathbf{q}) \cdot \begin{bmatrix} \dot{q}_1 \\ \dot{q}_2 \\ \vdots \\ \dot{q}_n \end{bmatrix}$$

Background: The Body Jacobian



Robot with points in 3D $\mathbf{x}$ and n joint angles $\mathbf{q}$.

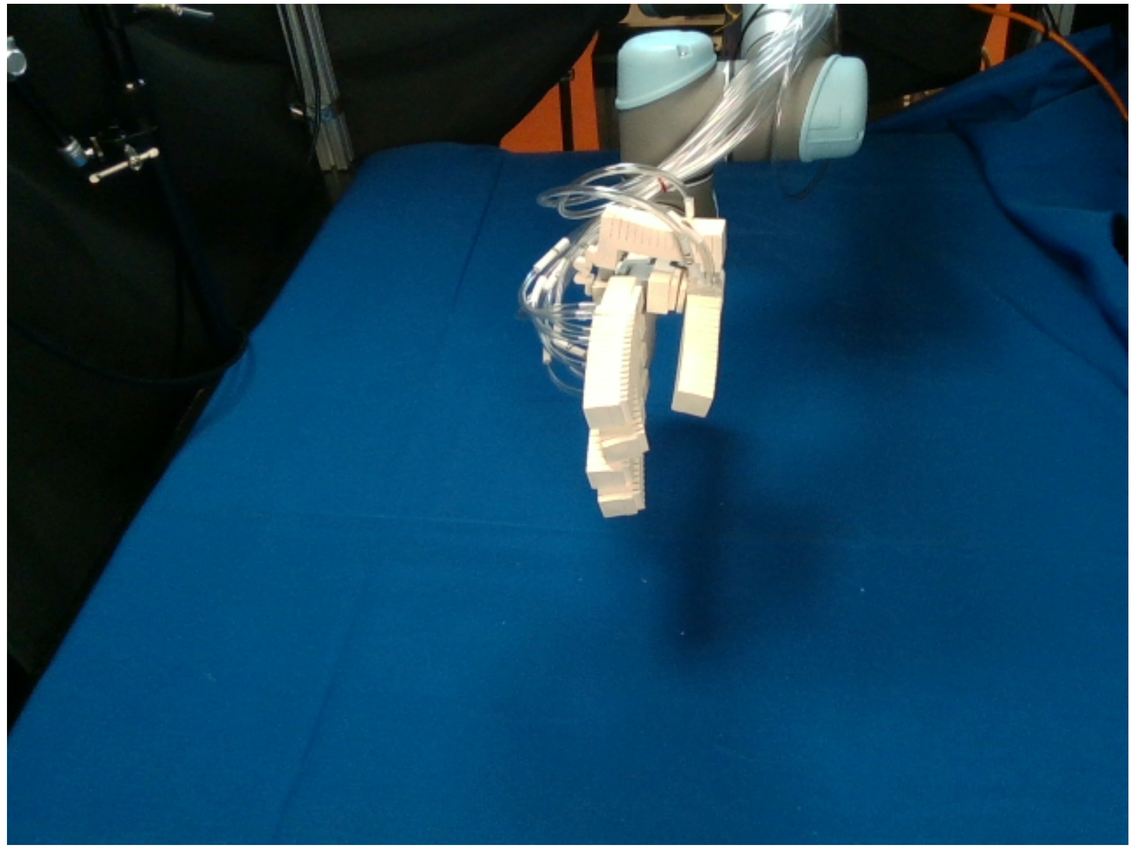
For *any point* $\mathbf{x}$ on the robot, we can find the *Body Jacobian,* defined as the $\mathbb{R}^{n\times 3}$ matrix $\mathbf{J}(\mathbf{q})$:

$$\begin{bmatrix} \dot{x} \\ \dot{y} \\ \dot{z} \end{bmatrix} = \mathbf{J}(\mathbf{q}) \cdot \begin{bmatrix} \dot{q}_1 \\ \dot{q}_2 \\ \vdots \\ \dot{q}_n \end{bmatrix}$$

Proposal: Learn to Reconstruct a Jacobian Field, i.e., a function that maps every 3D point to its Body Jacobian!

Unifying 3D Representation and Control of Diverse Robots with a Single Camera, Li et al. 2024

Input: Single Image

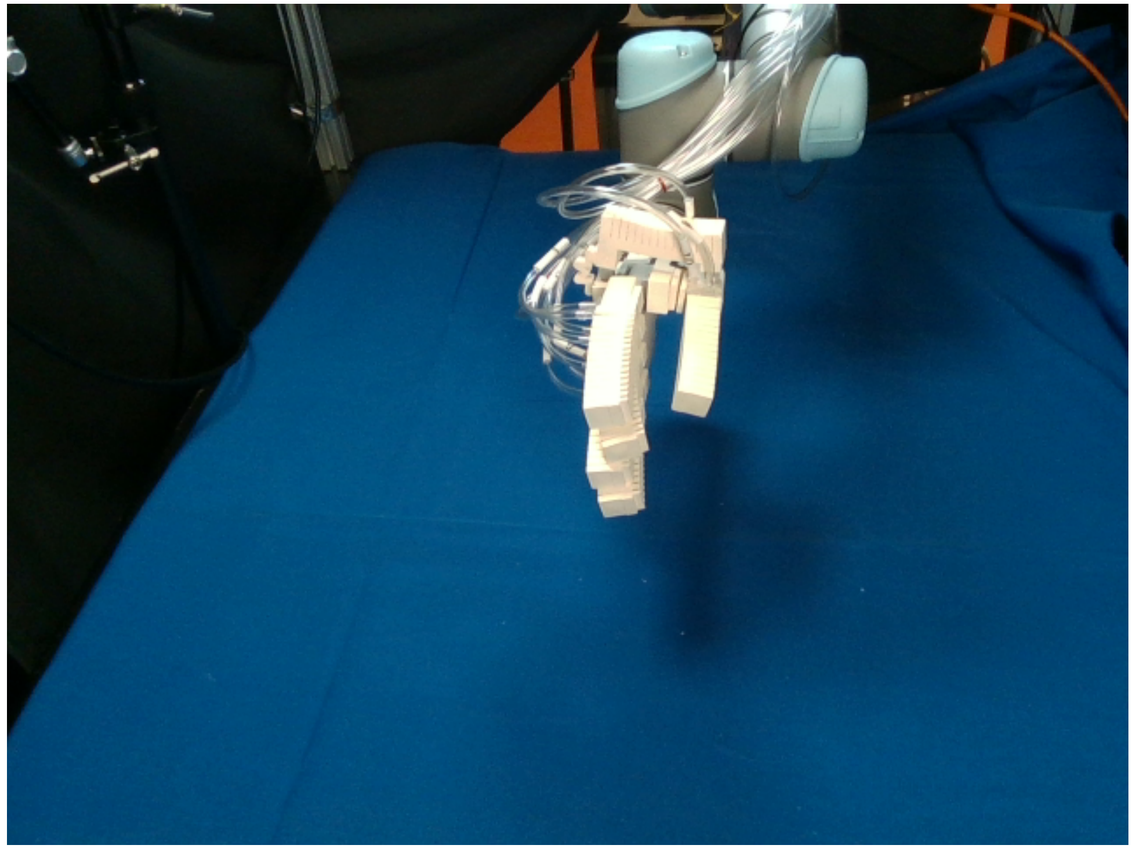


Neural Jacobian Field $\mathbf{J}(\mathbf{x}, \mathbf{I})$

Proposal: Learn to Reconstruct a Jacobian Field, i.e., a function that maps every 3D point to its Body Jacobian!

Unifying 3D Representation and Control of Diverse Robots with a Single Camera, Li et al. 2024

Input: Single Image

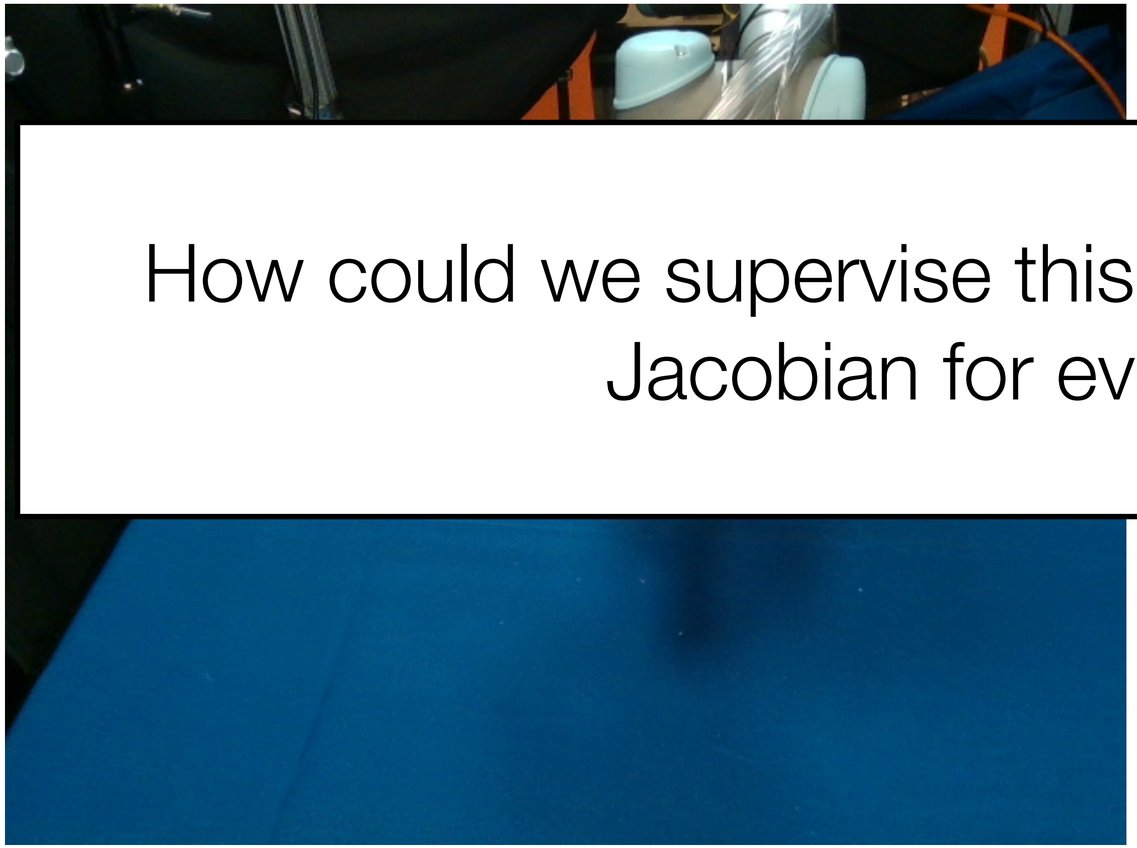


Neural Jacobian Field $\mathbf{J}(\mathbf{x}, \mathbf{I})$

Proposal: Learn to Reconstruct a Jacobian Field, i.e., a function that maps every 3D point to its Body Jacobian!

Unifying 3D Representation and Control of Diverse Robots with a Single Camera, Li et al. 2024

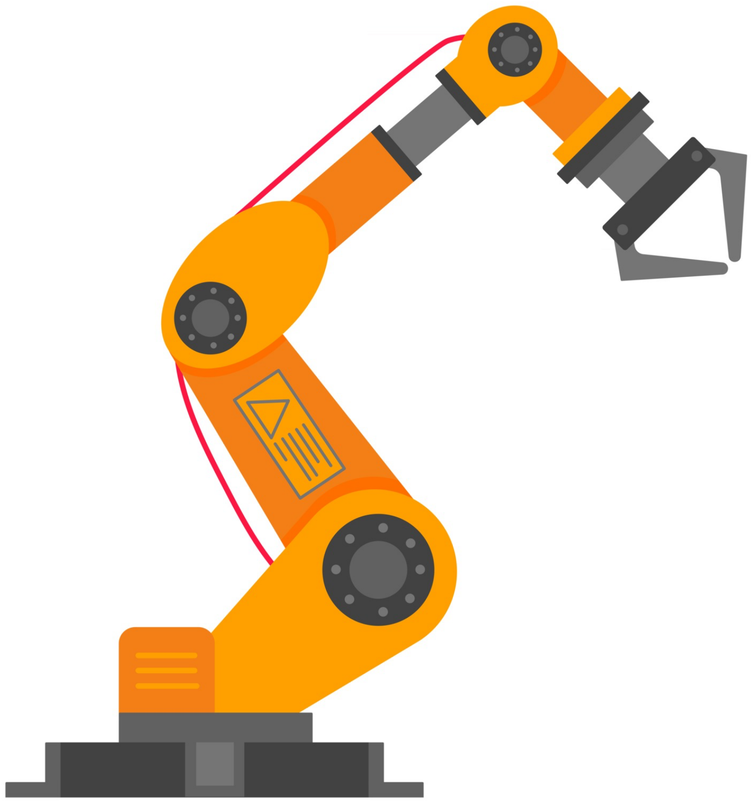
Input: Single Image



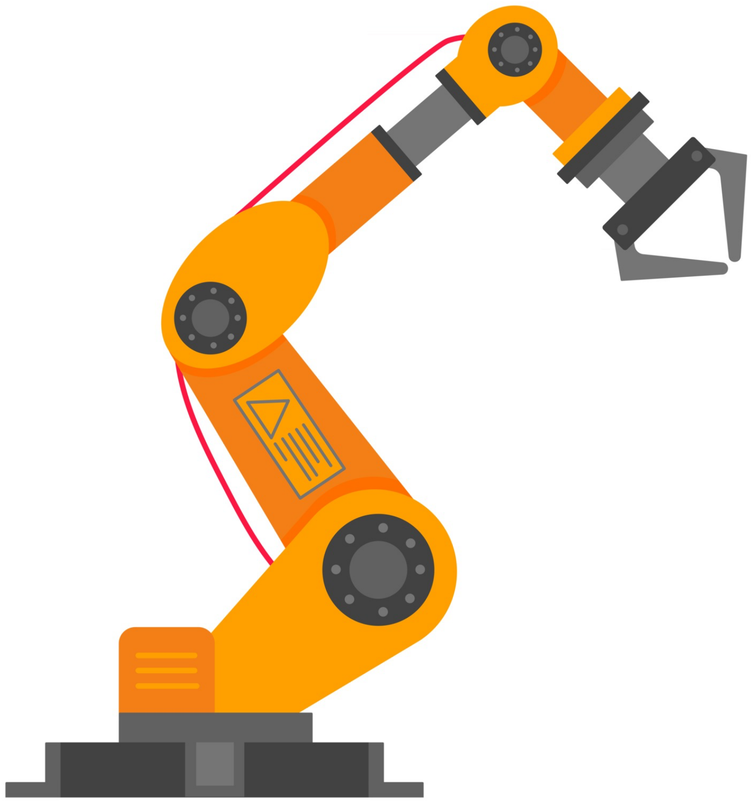
Neural Jacobian Field $\mathbf{J}(\mathbf{x}, \mathbf{I})$

How could we supervise this? We don't *have* the ground-truth Jacobian for every point in space…

Background: The Body Jacobian

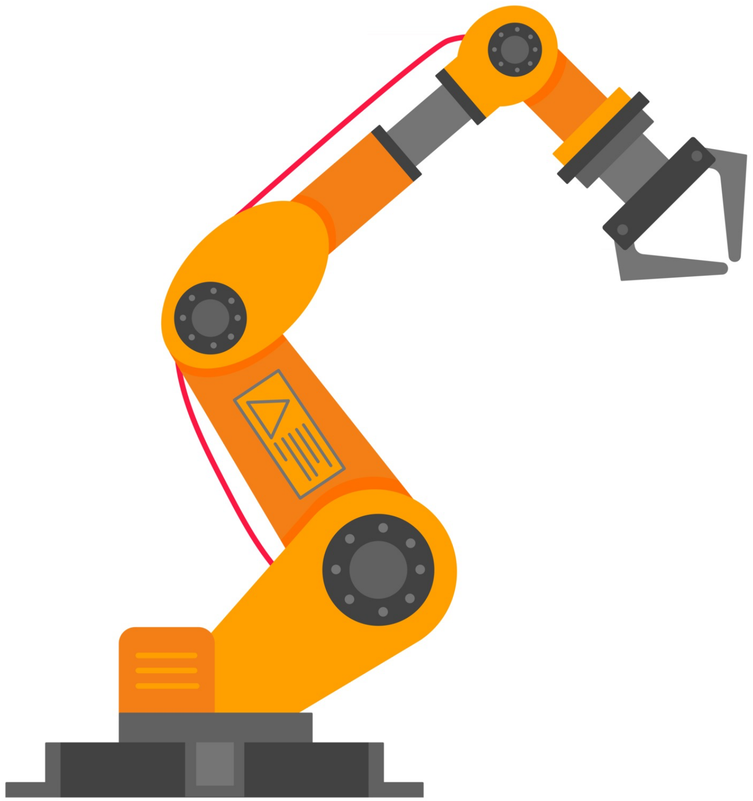


Background: The Body Jacobian



Robot with points in 3D $\mathbf{x}$ and n joint angles $\mathbf{q}$.

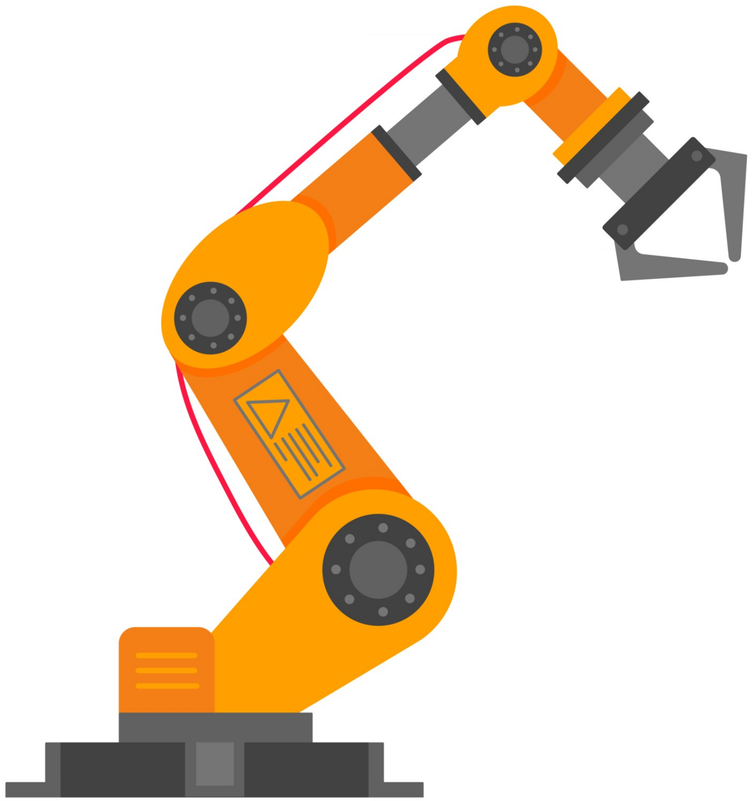
Background: The Body Jacobian



Robot with points in 3D $\mathbf{x}$ and n joint angles $\mathbf{q}$.

For *any point* $\mathbf{x}$ on the robot, we can find the *Body Jacobian,* defined as the $\mathbb{R}^{n\times 3}$ matrix $\mathbf{J}(\mathbf{q})$:

Background: The Body Jacobian

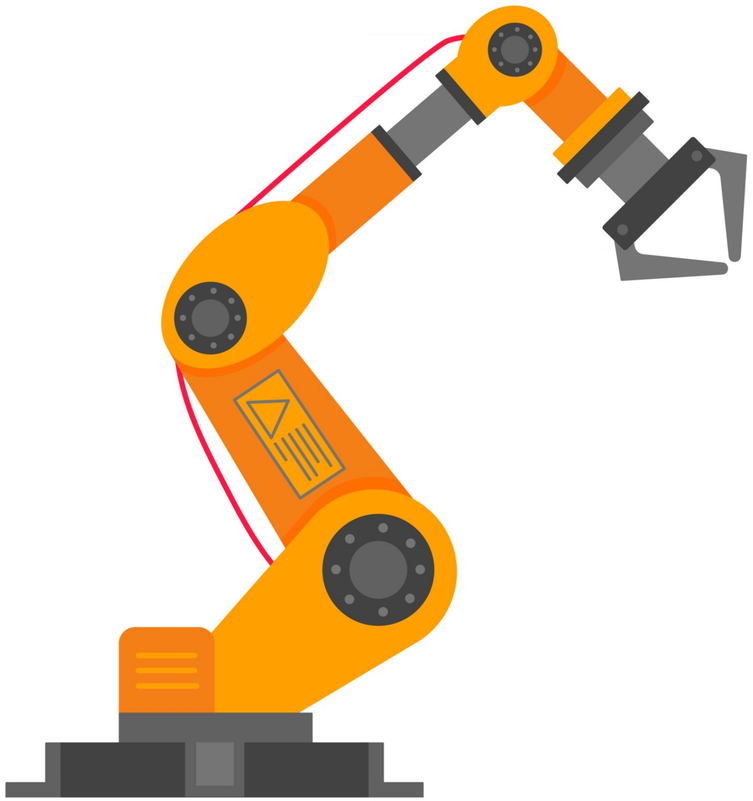


Robot with points in 3D $\mathbf{x}$ and n joint angles $\mathbf{q}$.

For *any point* $\mathbf{x}$ on the robot, we can find the *Body Jacobian,* defined as the $\mathbb{R}^{n\times 3}$ matrix $\mathbf{J}(\mathbf{q})$:

$$\begin{bmatrix} \dot{x} \\ \dot{y} \\ \dot{z} \end{bmatrix} = \mathbf{J}(\mathbf{q}) \cdot \begin{bmatrix} \dot{q}_1 \\ \dot{q}_2 \\ \vdots \\ \dot{q}_n \end{bmatrix}$$

Background: The Body Jacobian



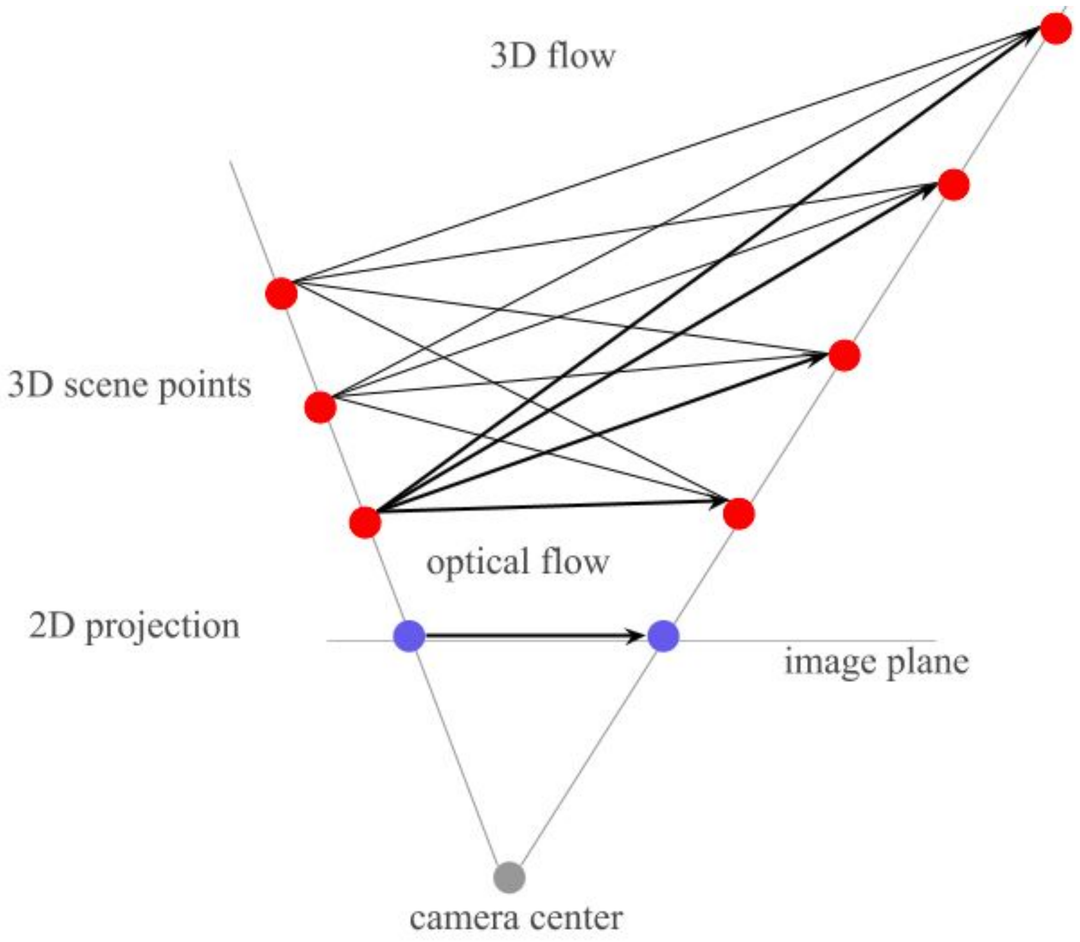
Robot with points in 3D $\mathbf{x}$ and n joint angles $\mathbf{q}$.

For *any point* $\mathbf{x}$ on the robot, we can find the *Body Jacobian,* defined as the $\mathbb{R}^{n\times 3}$ matrix $\mathbf{J}(\mathbf{q})$:

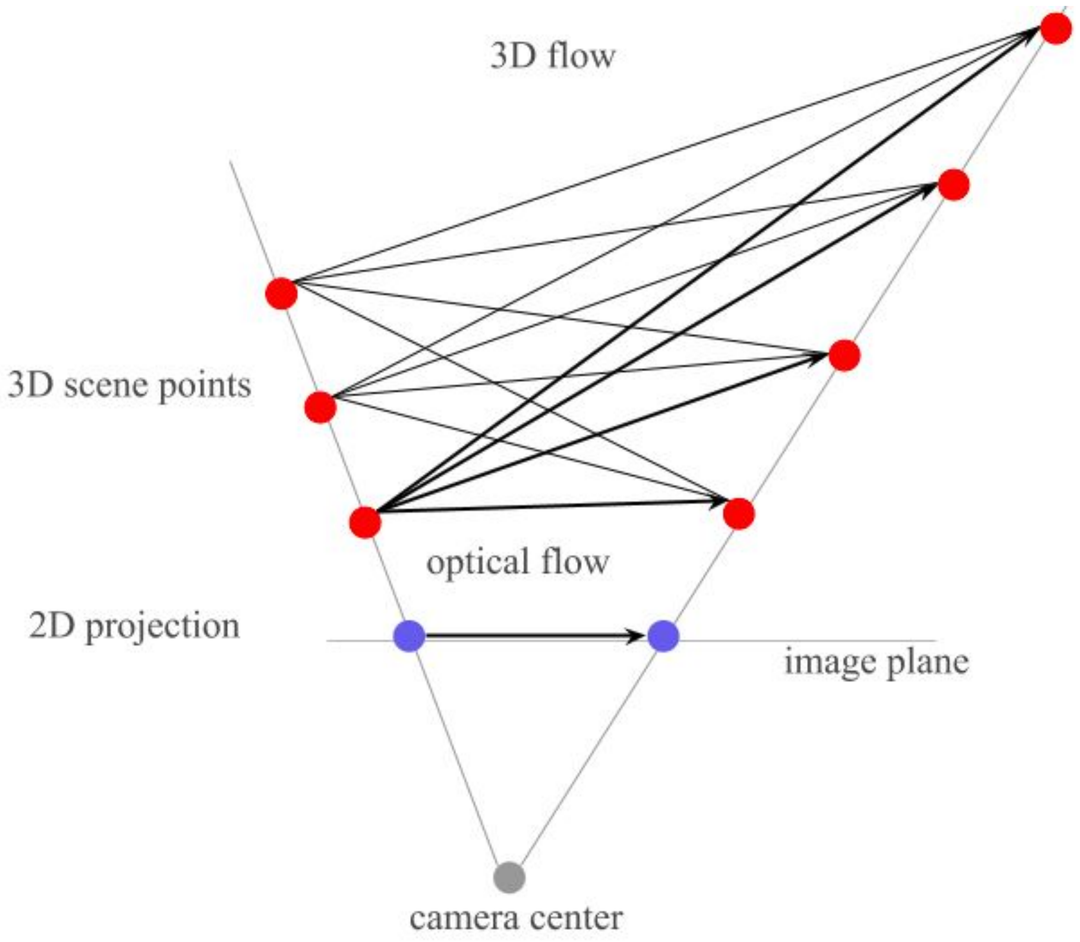
$$\begin{bmatrix} \dot{x} \\ \dot{y} \\ \dot{z} \end{bmatrix} = \mathbf{J}(\mathbf{q}) \cdot \begin{bmatrix} \dot{q}_1 \\ \dot{q}_2 \\ \vdots \\ \dot{q}_n \end{bmatrix}$$

Insight: this equation relates *scene flow* to robot actions!

Reminder: Scene Flow and Optical Flow

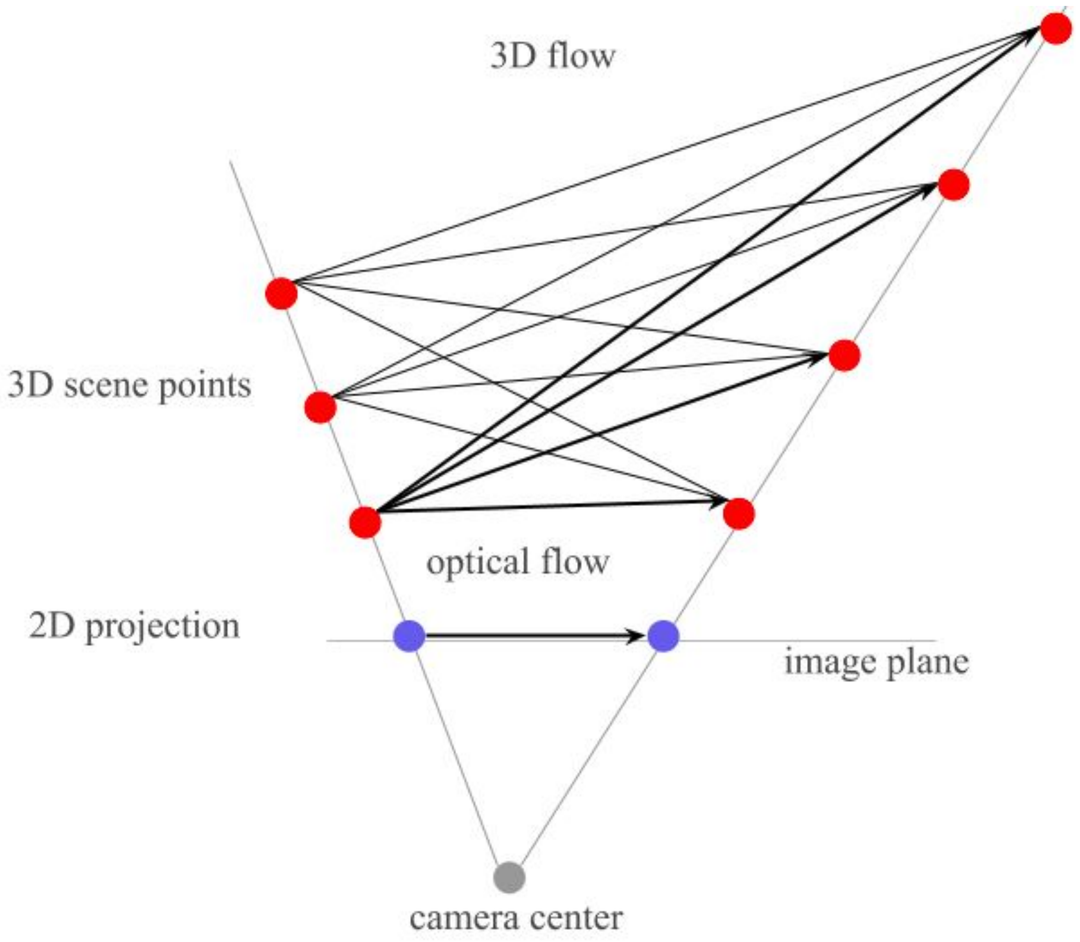


Reminder: Scene Flow and Optical Flow



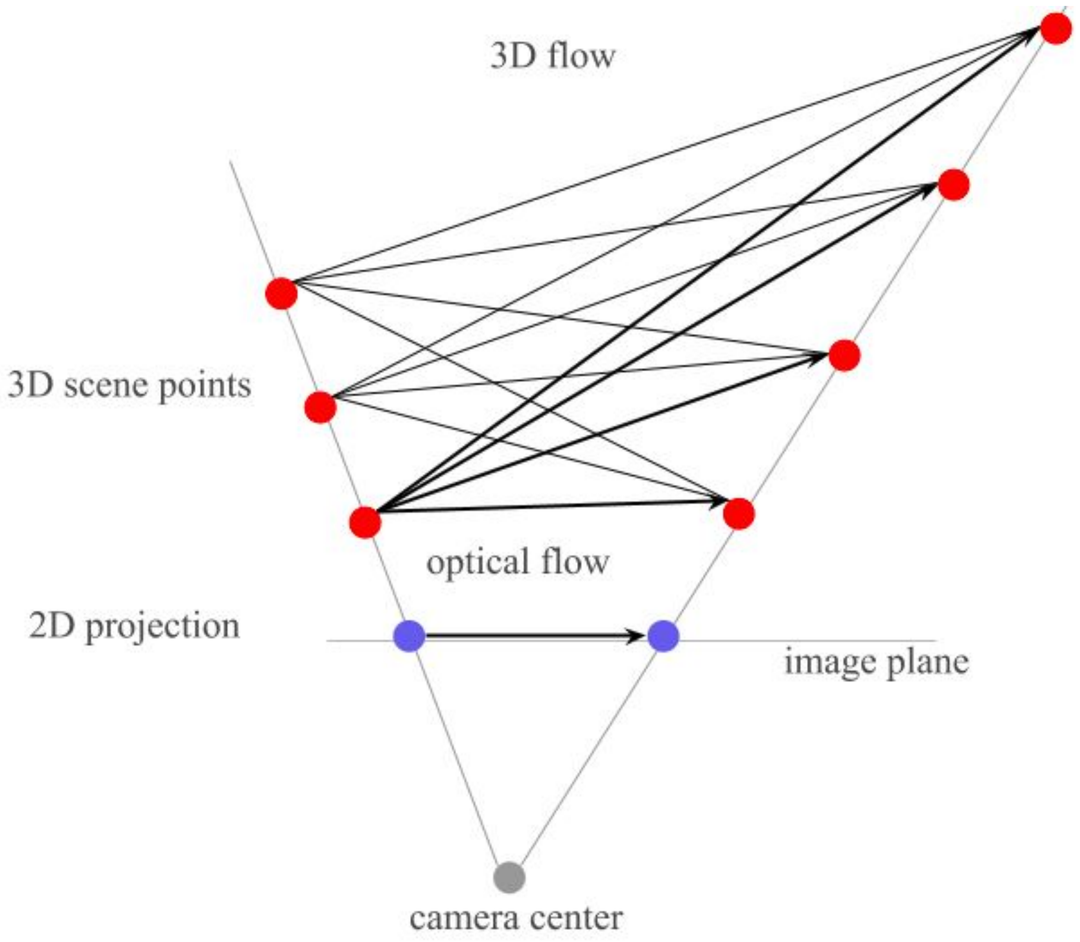
Scene Flow: Motion field of 3D vectors in 3D

Reminder: Scene Flow and Optical Flow



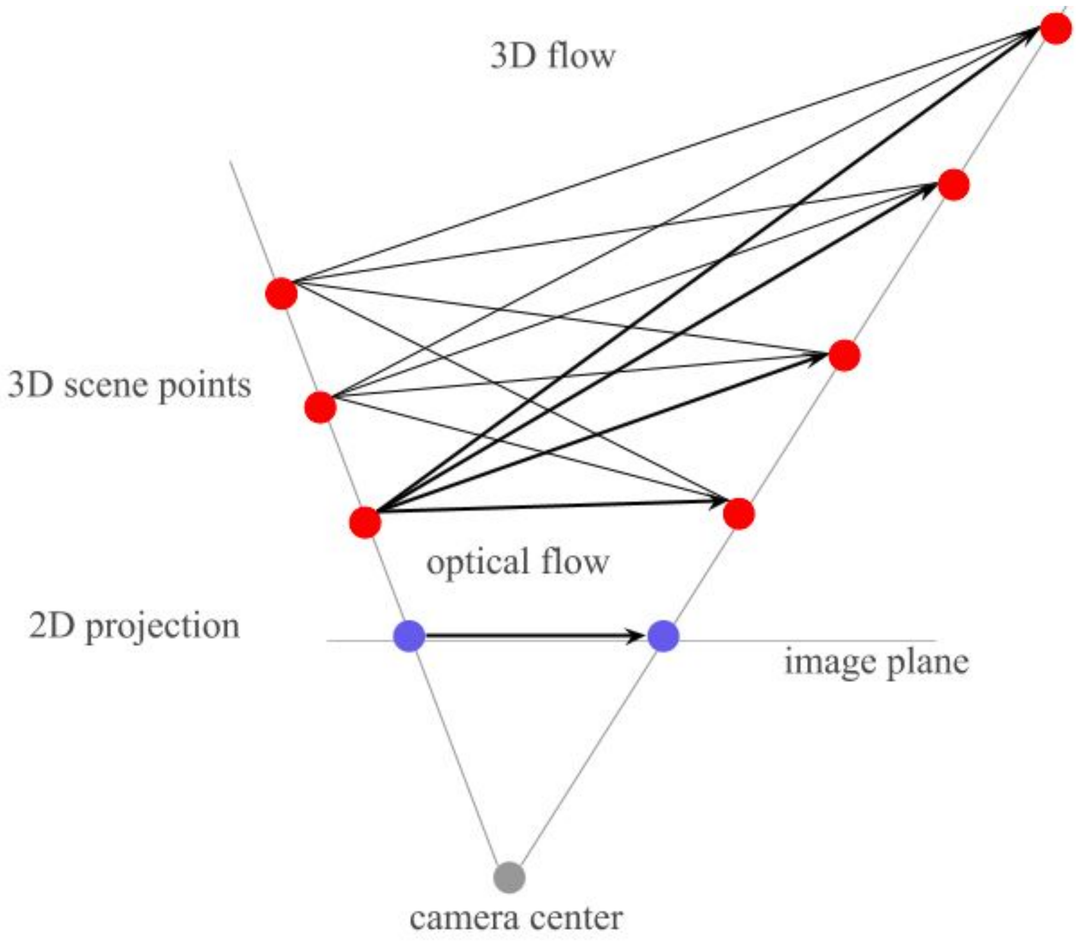
Projecting 3D Scene Flow to the image plane yields **optical flow**

Reminder: Scene Flow and Optical Flow



Idea: we can simply supervise the Jacobian by enforcing that the projected, predicted scene flow matches the optical flow $\mathbf{u}$ from all views:

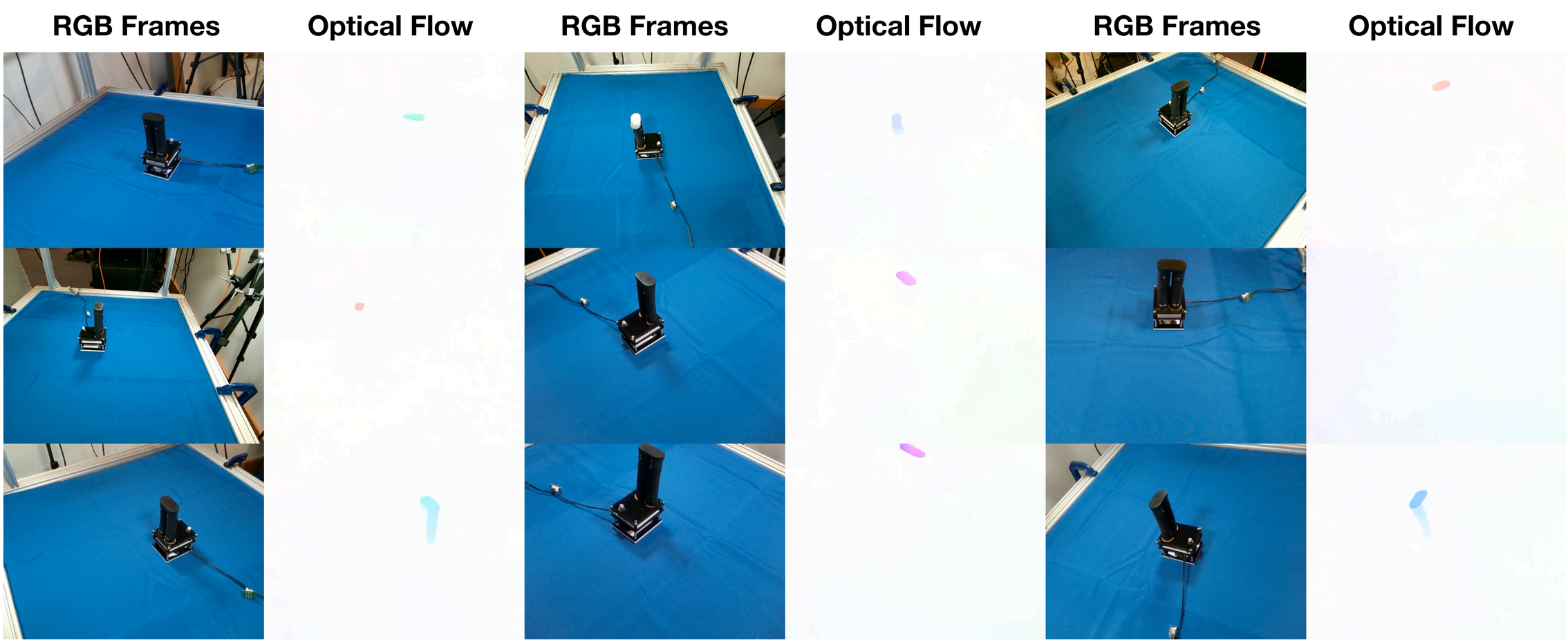
Reminder: Scene Flow and Optical Flow



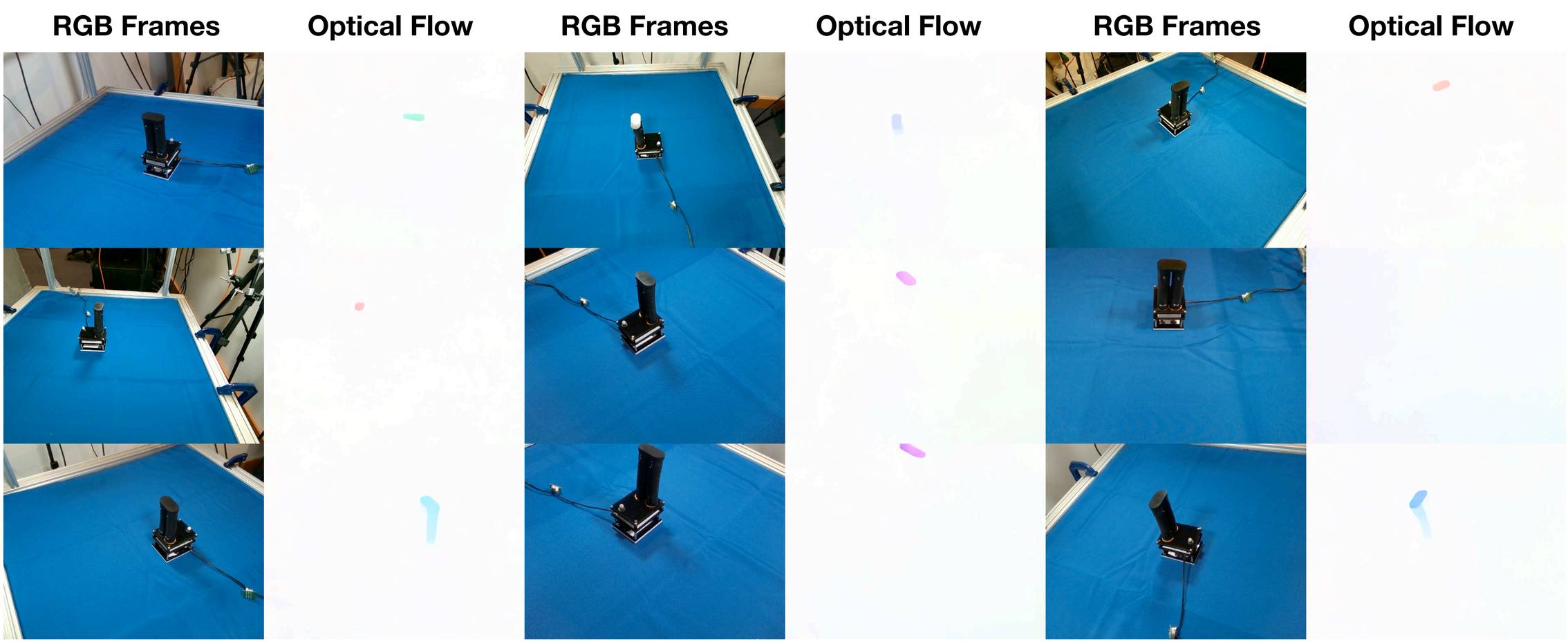
Idea: we can simply supervise the Jacobian by enforcing that the projected, predicted scene flow matches the optical flow $\mathbf{u}$ from all views:

$$\|\pi(\mathbf{J}\Delta\mathbf{q}) - \mathbf{u}\|_2^2$$

Training Data: Multi-View Video plus (random) Actions

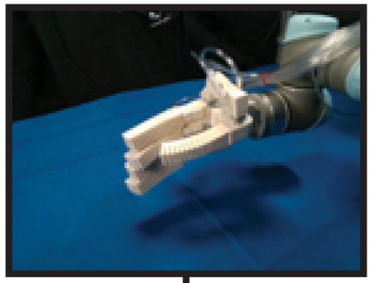


Training Data: Multi-View Video plus (random) Actions



Self-Supervised Training

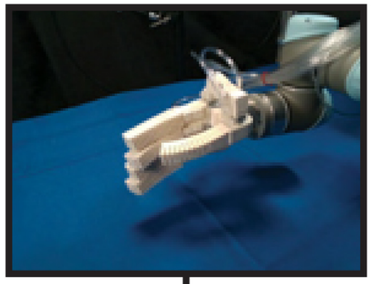
Single Input Image



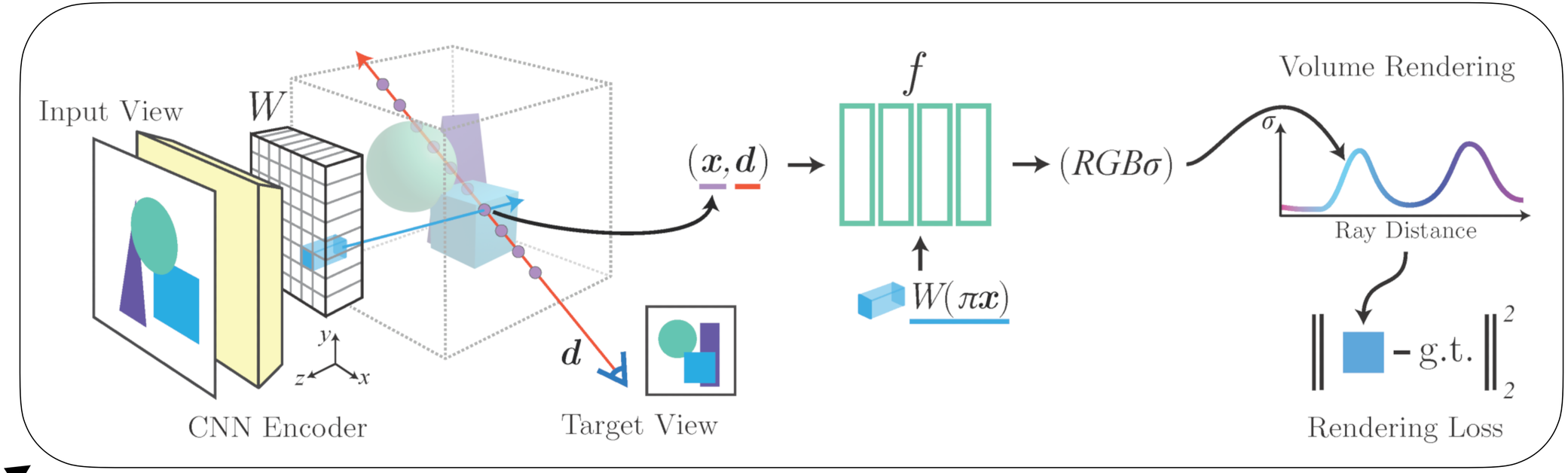
Neural 3D Reconstruction

Self-Supervised Training

Single Input Image



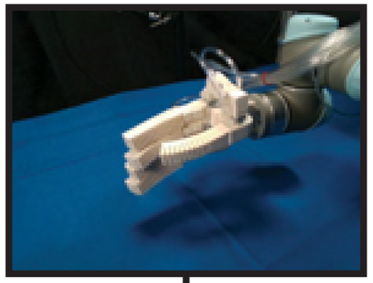
Neural 3D Reconstruction



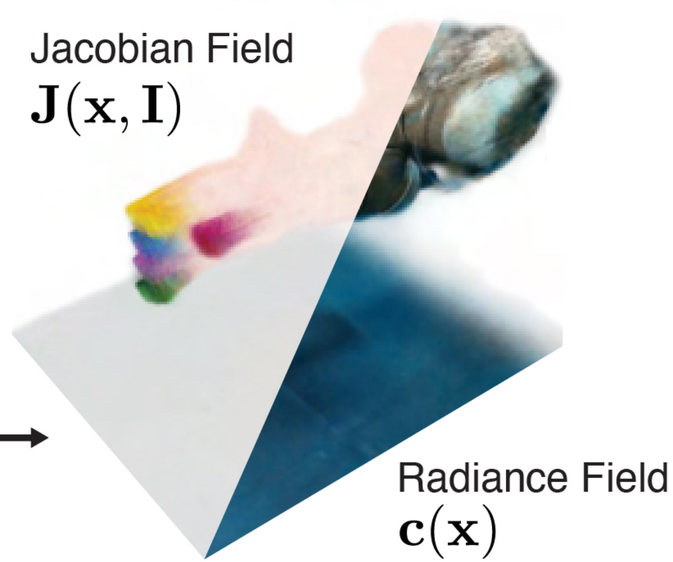
We use pixelNeRF (Yu et al. 2021) - could also use pixelSplat (Charatan et al. 2024), SplatterImage (Szymanowicz et al. 2024), …

Self-Supervised Training

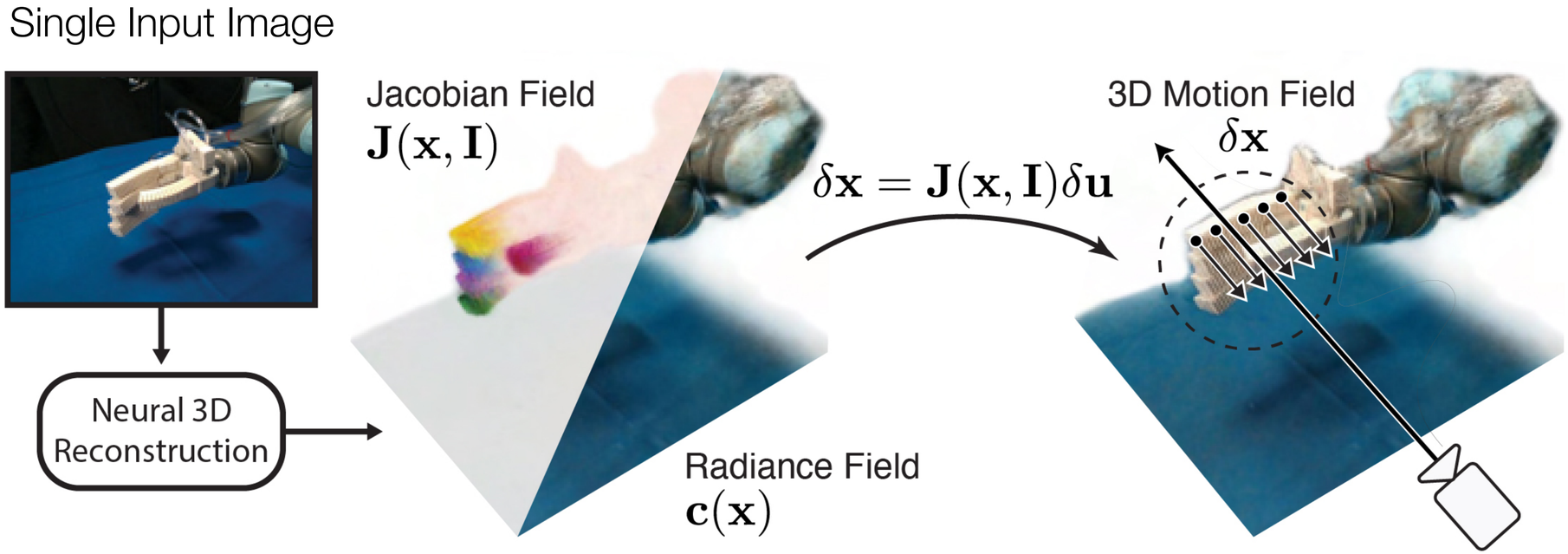
Single Input Image



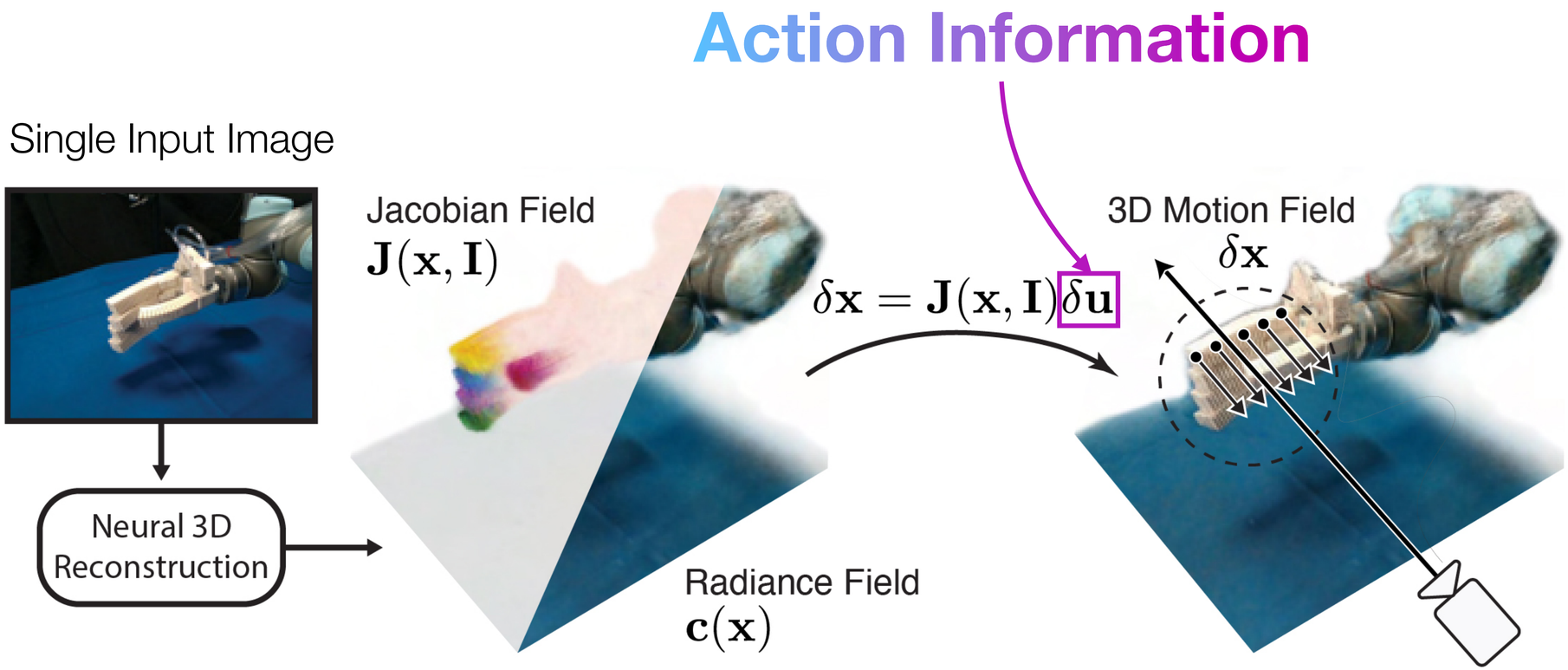
Neural 3D Reconstruction



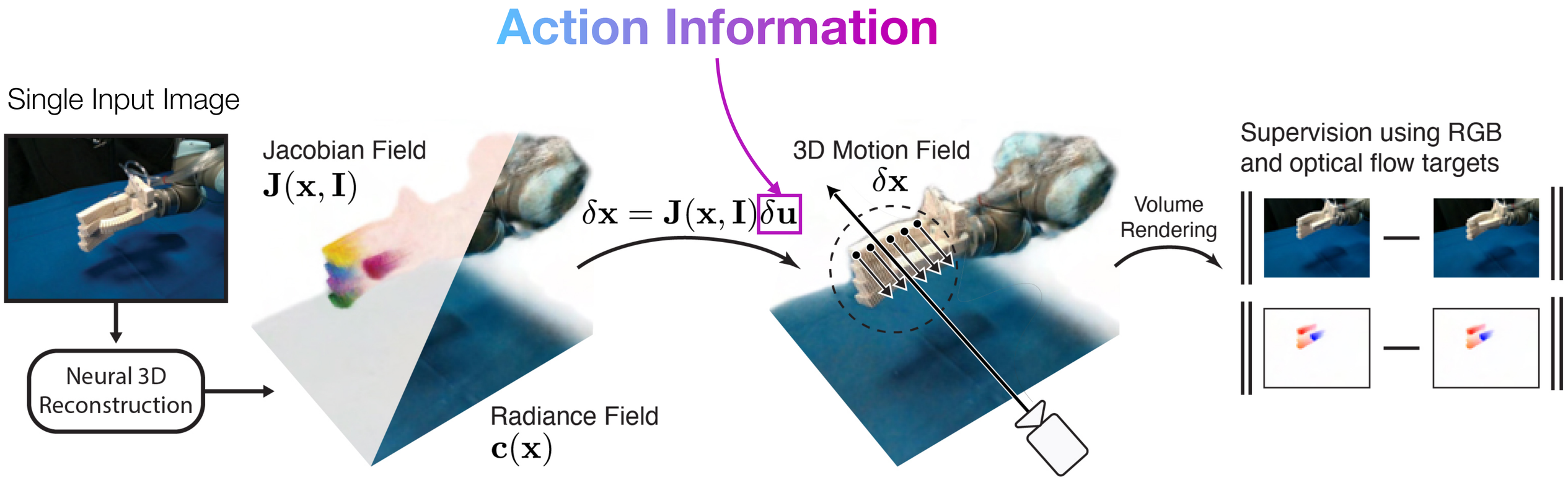
Self-Supervised Training



Self-Supervised Training

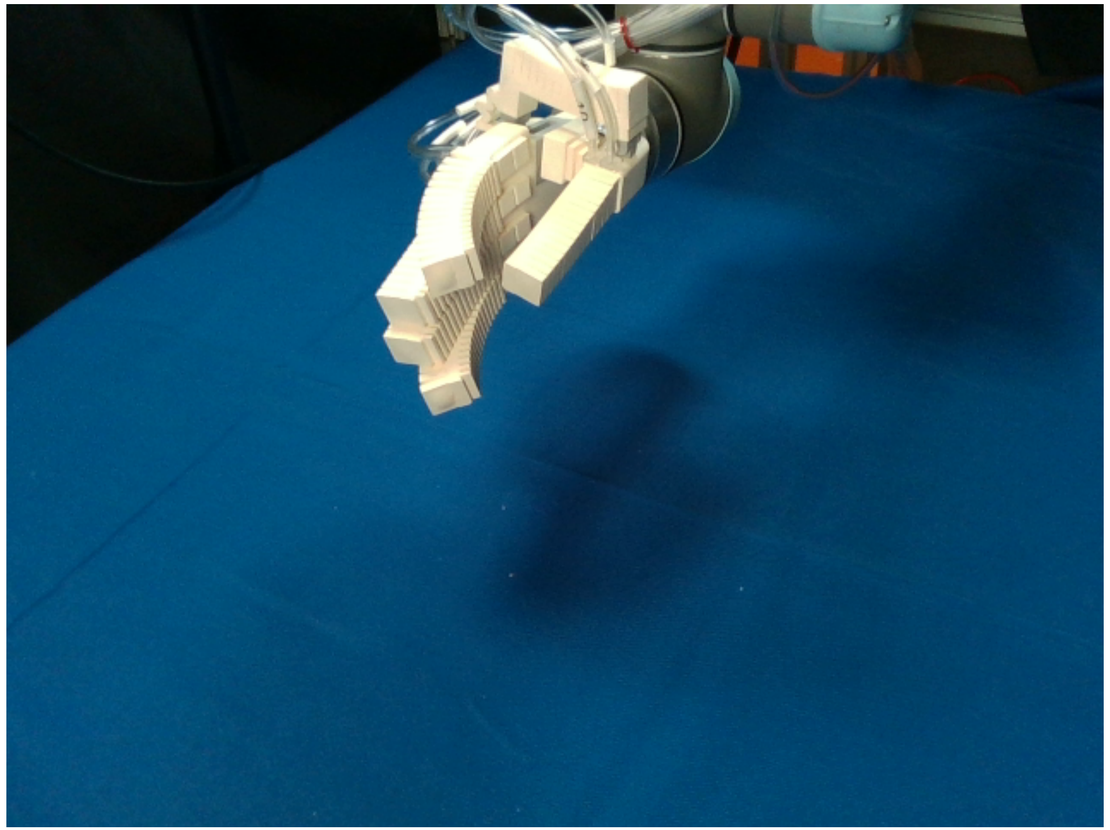


Self-Supervised Training

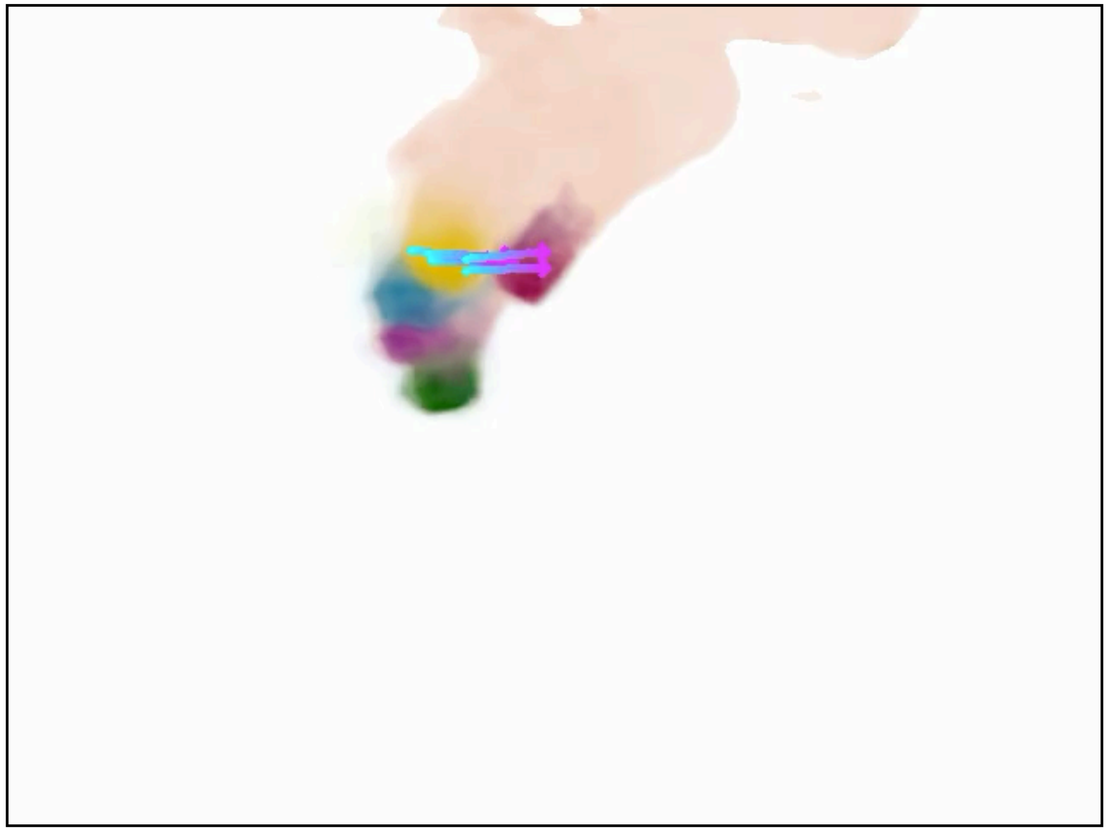


3D-Printed, Pneumatic Hand

Input RGB + Command

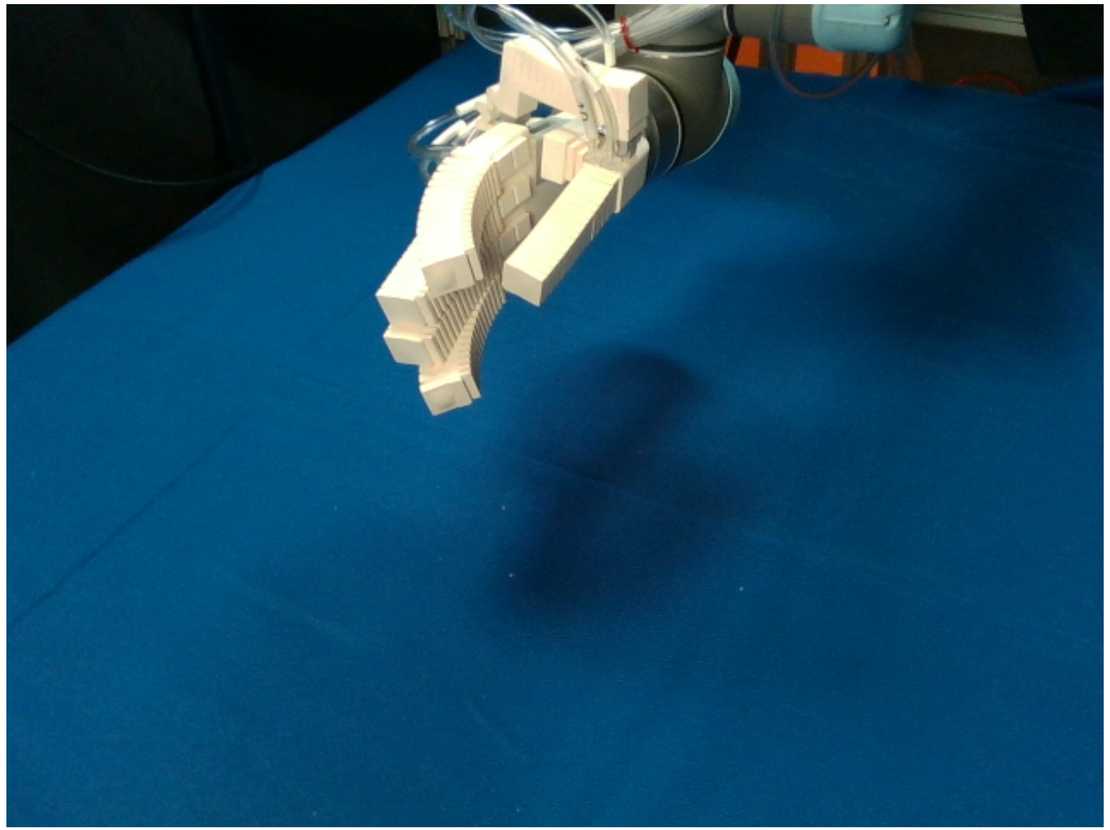


Predicted 3D Motion

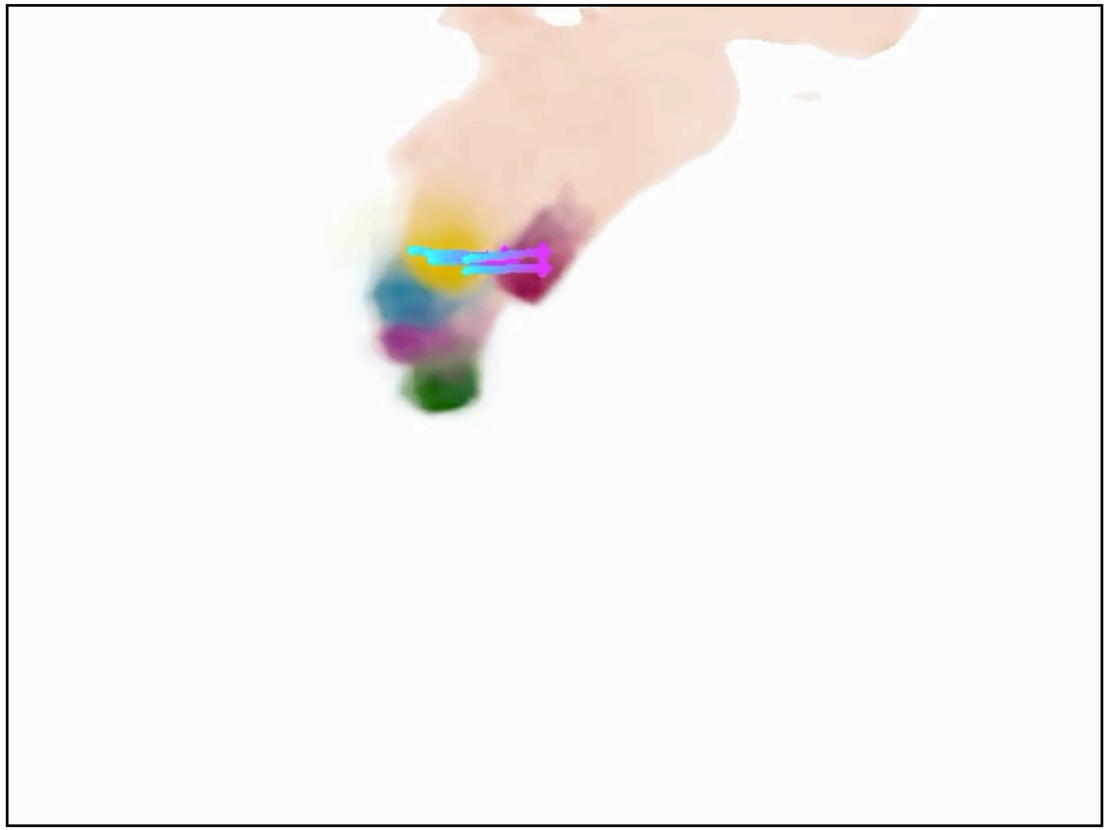


3D-Printed, Pneumatic Hand

Input RGB + Command

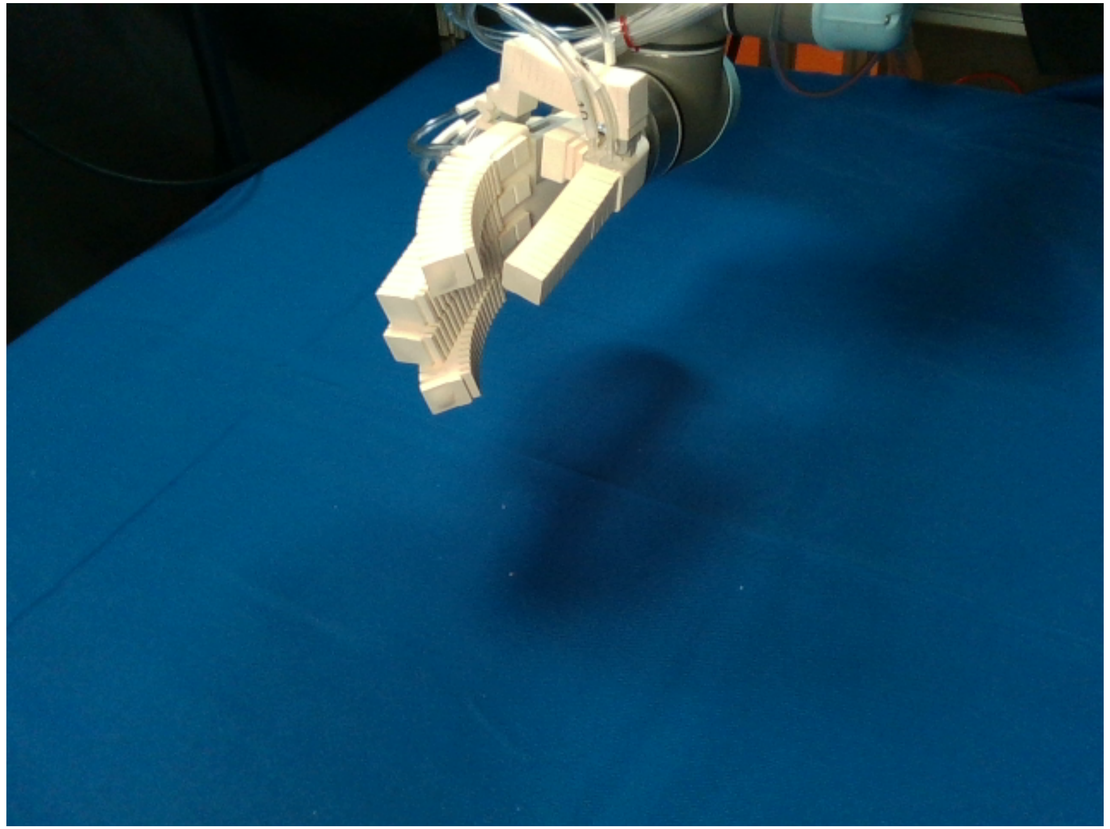


Predicted 3D Motion

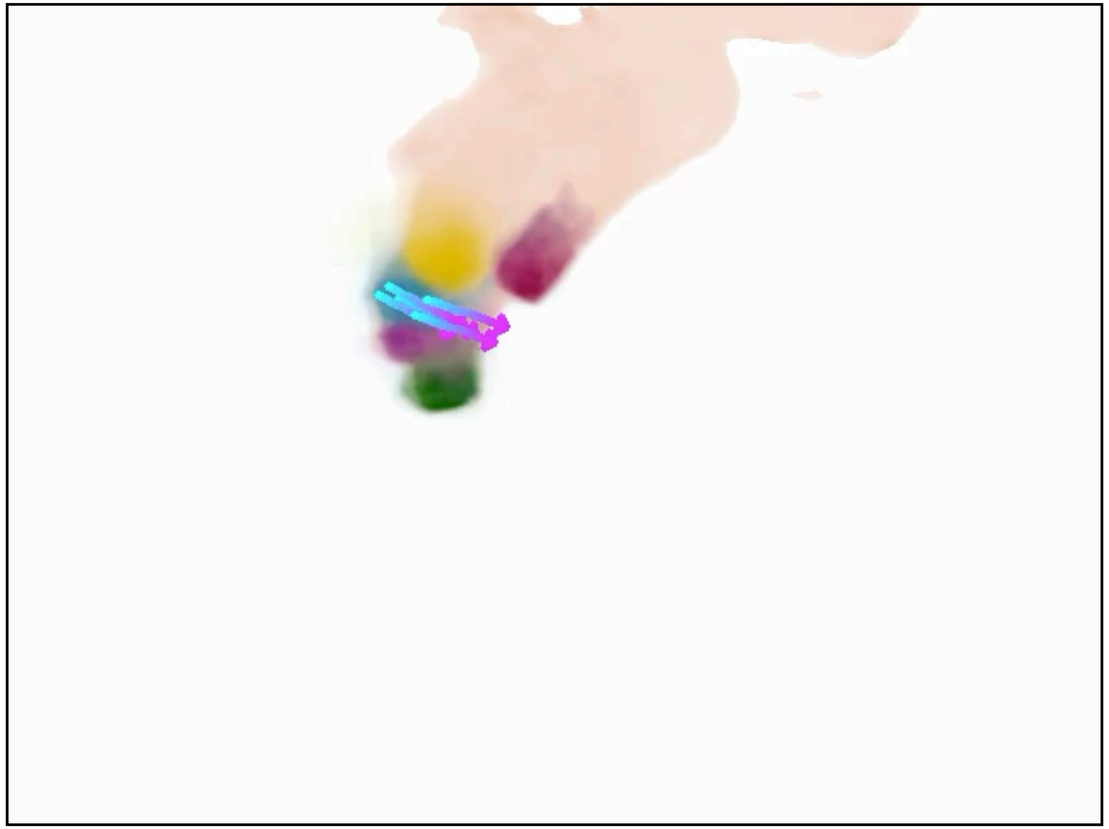


3D-Printed, Pneumatic Hand

Input RGB + Command

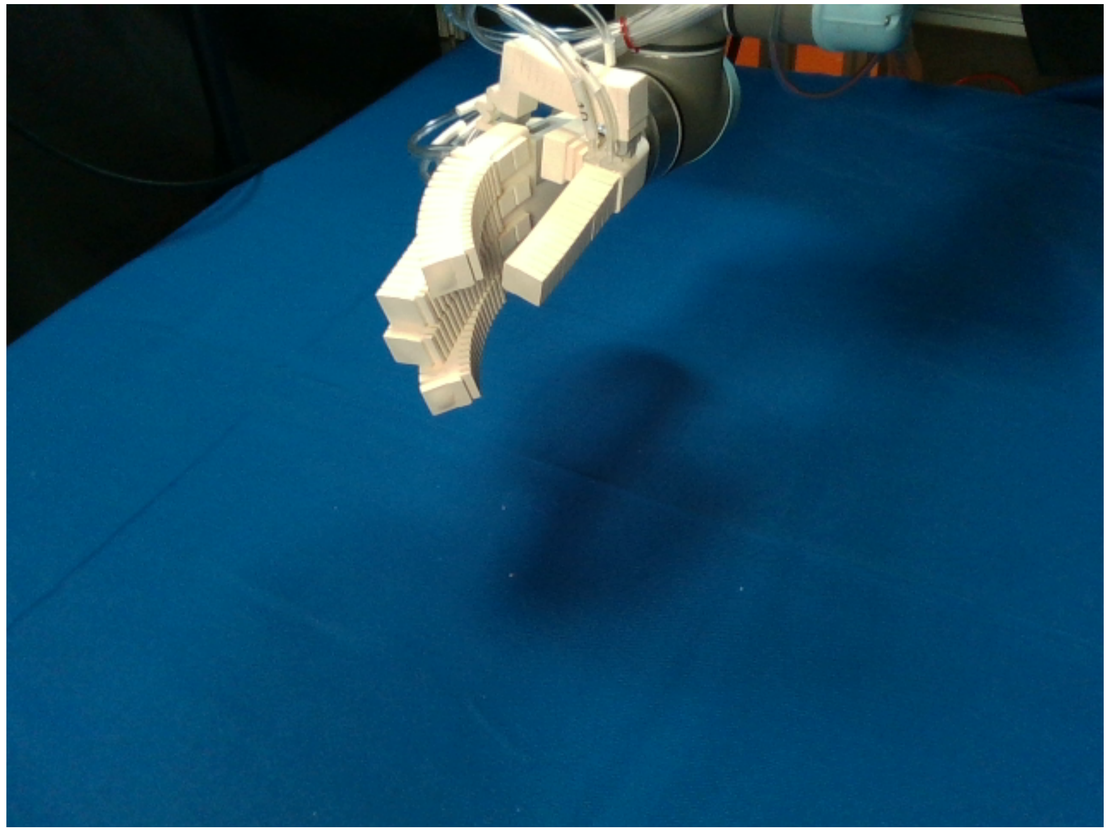


Predicted 3D Motion



3D-Printed, Pneumatic Hand

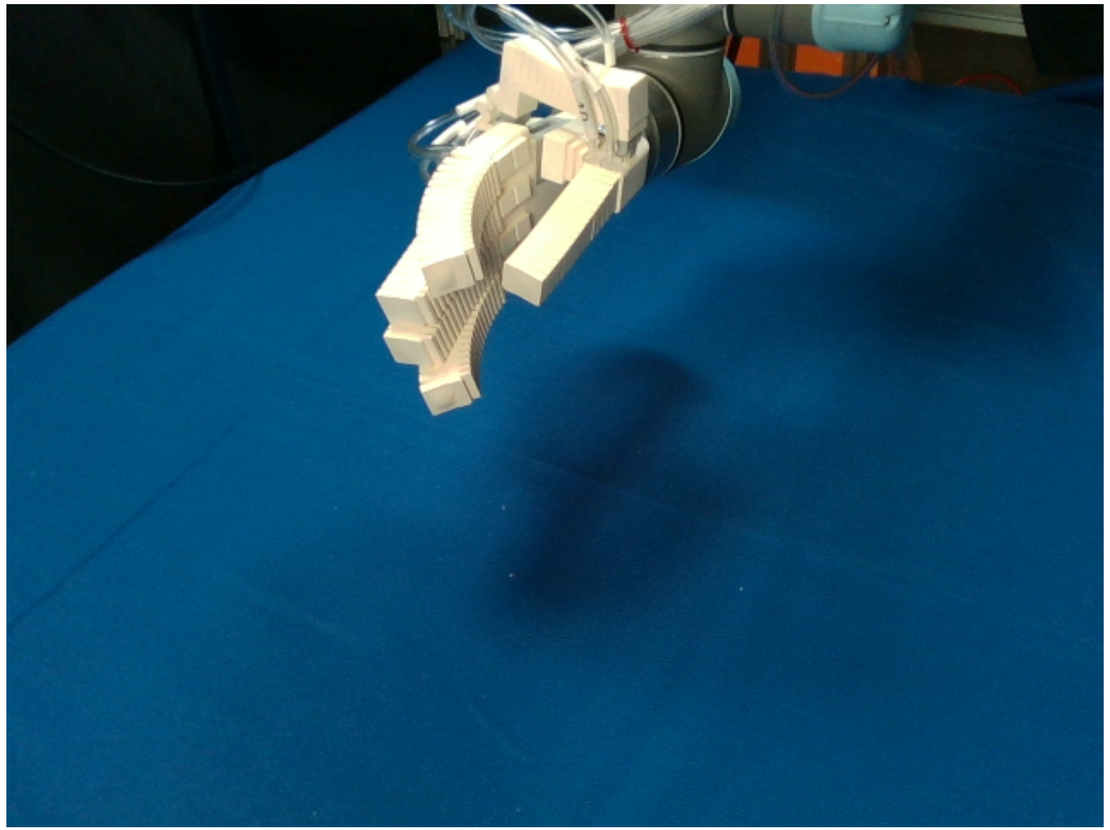
Input RGB + Command



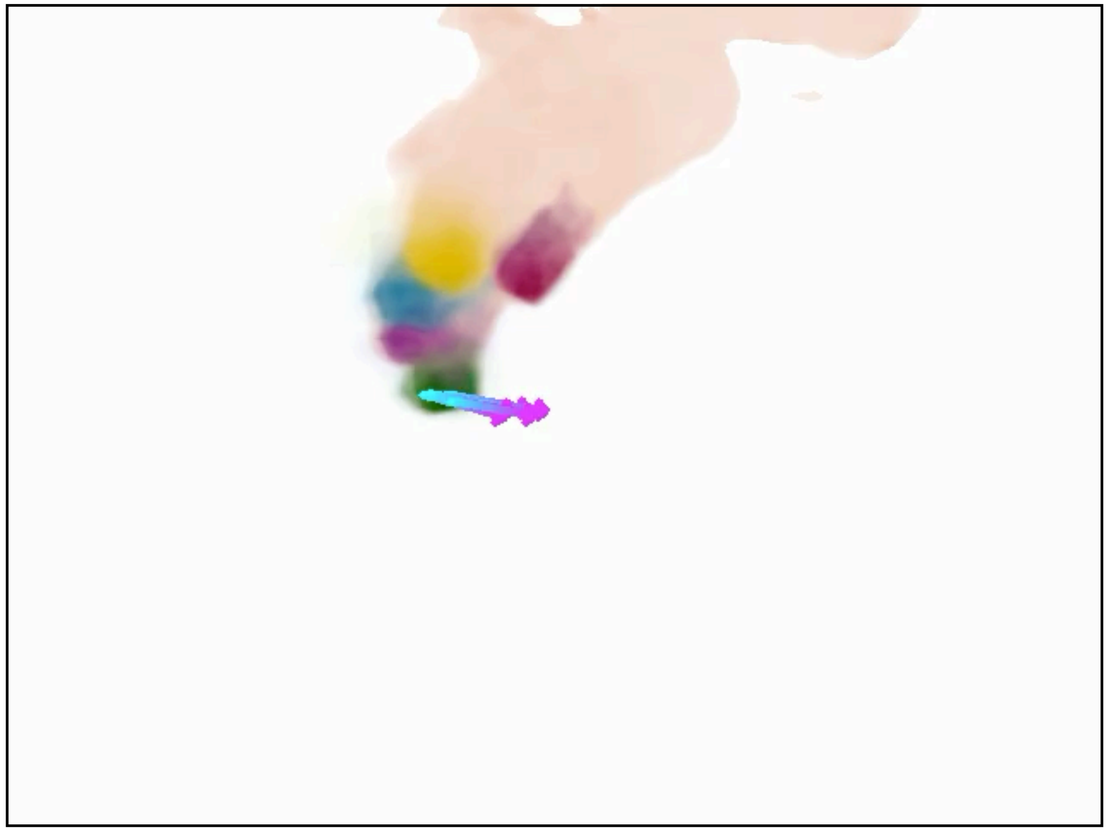
Predicted 3D Motion

3D-Printed, Pneumatic Hand

Input RGB + Command

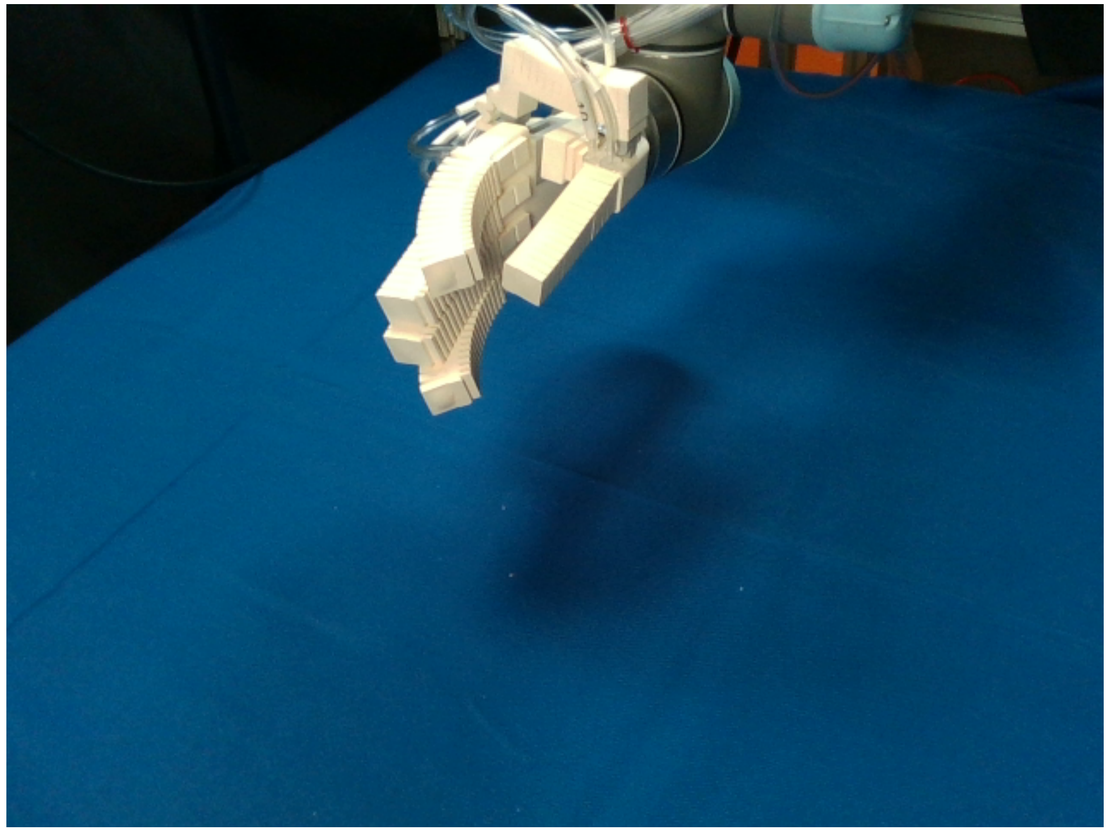


Predicted 3D Motion



3D-Printed, Pneumatic Hand

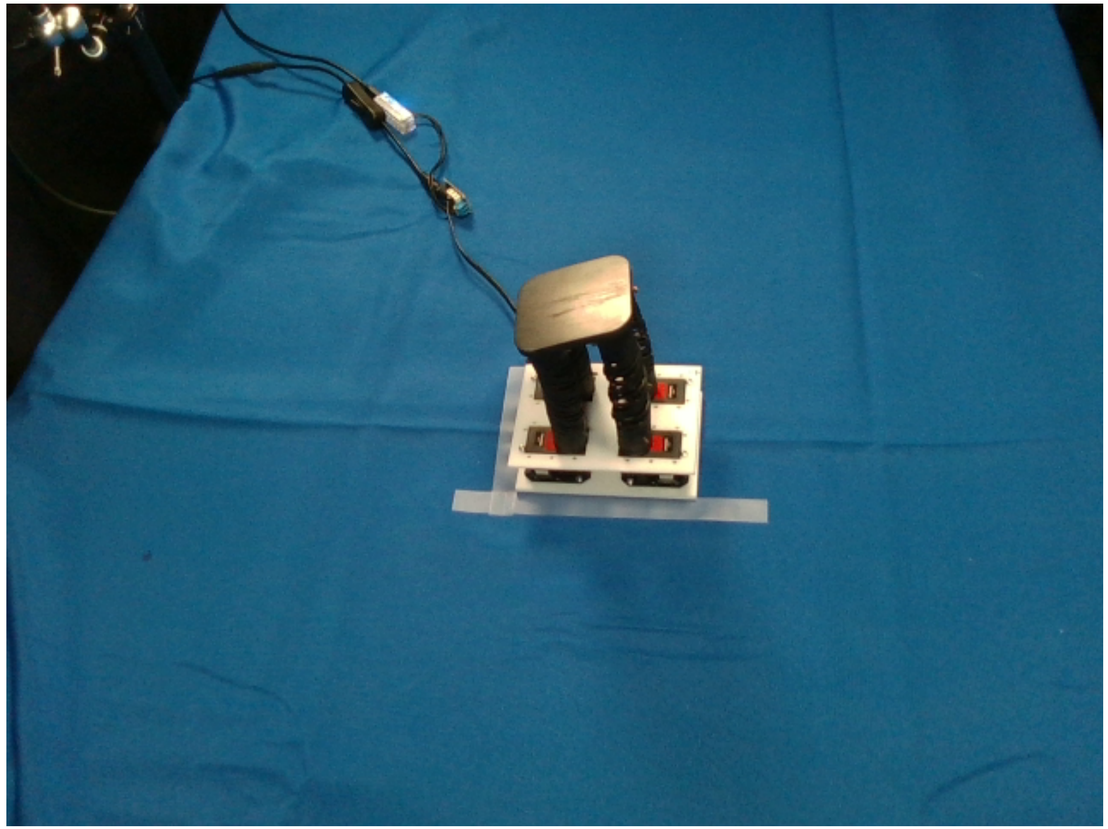
Input RGB + Command



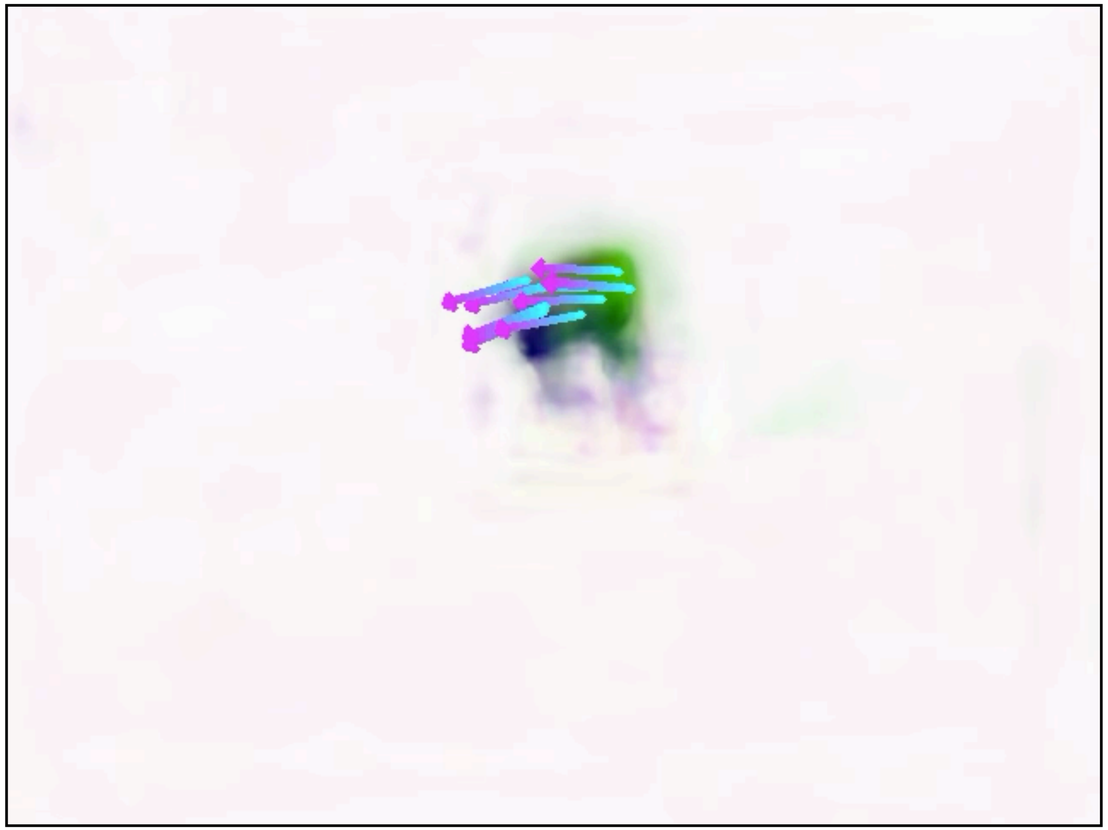
Predicted 3D Motion

Handed Shearing Auxetics

Input RGB + Command

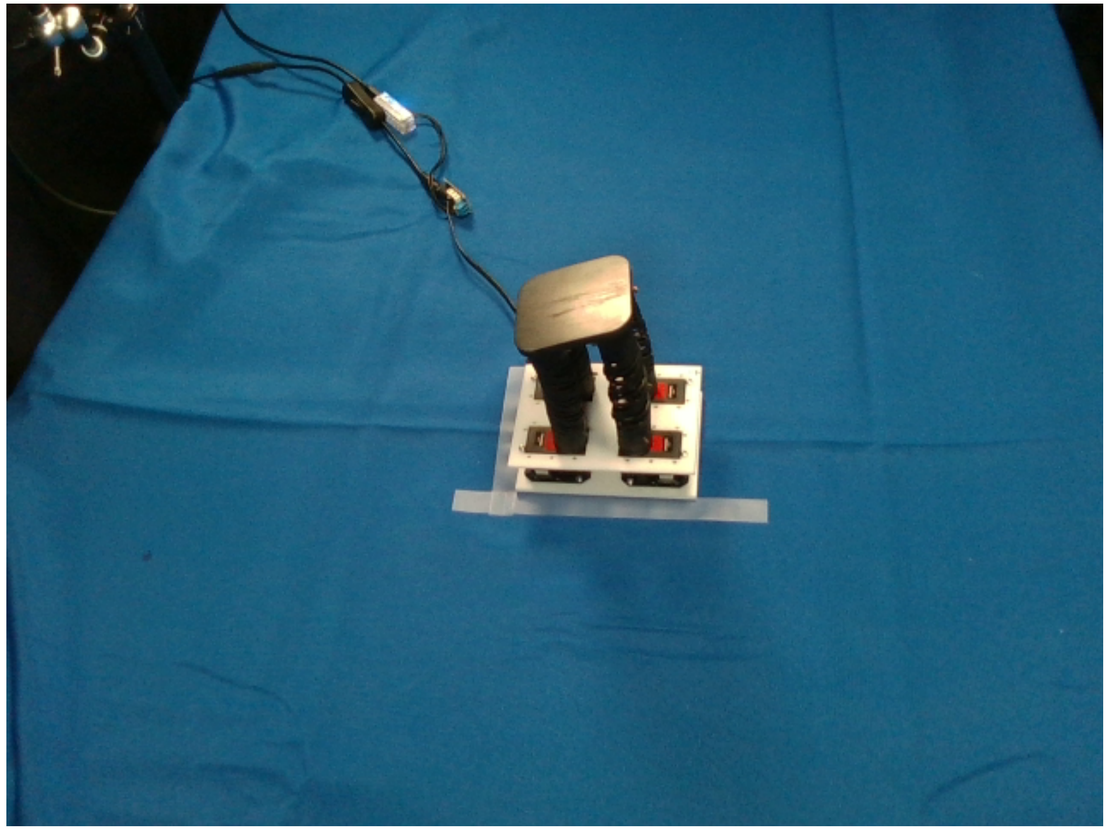


Predicted 3D Motion

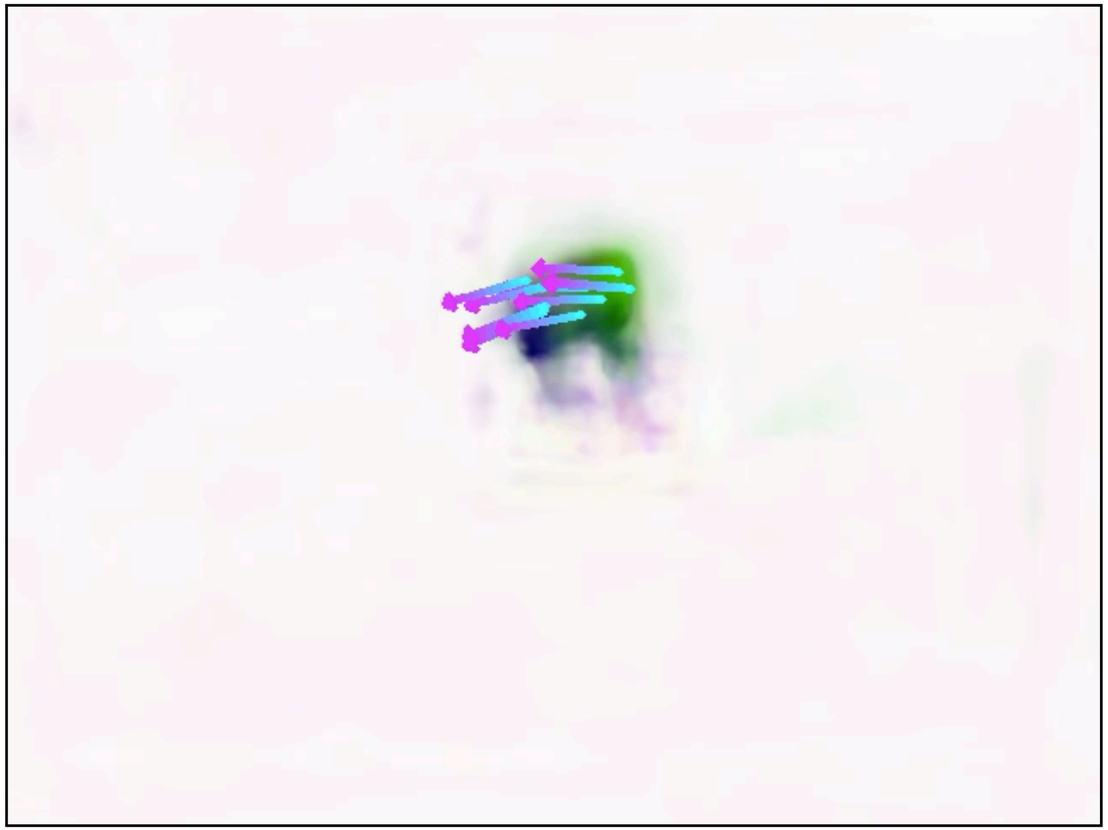


Handed Shearing Auxetics

Input RGB + Command

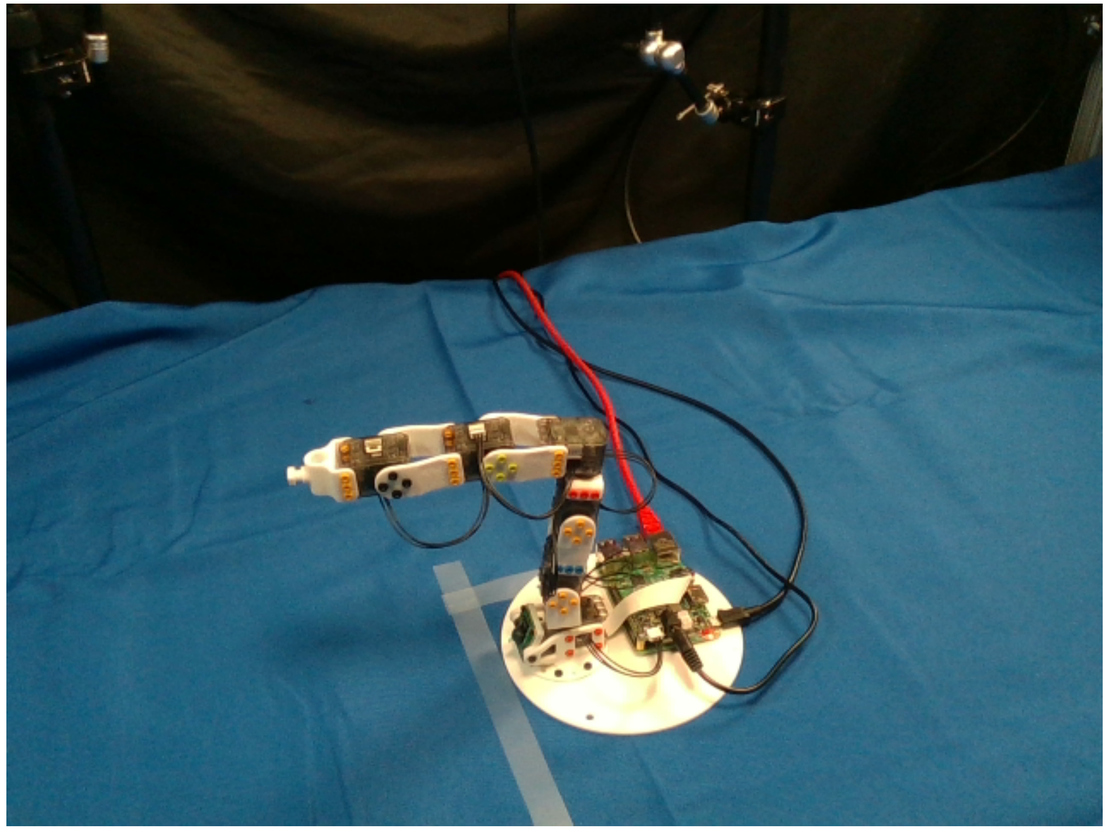


Predicted 3D Motion

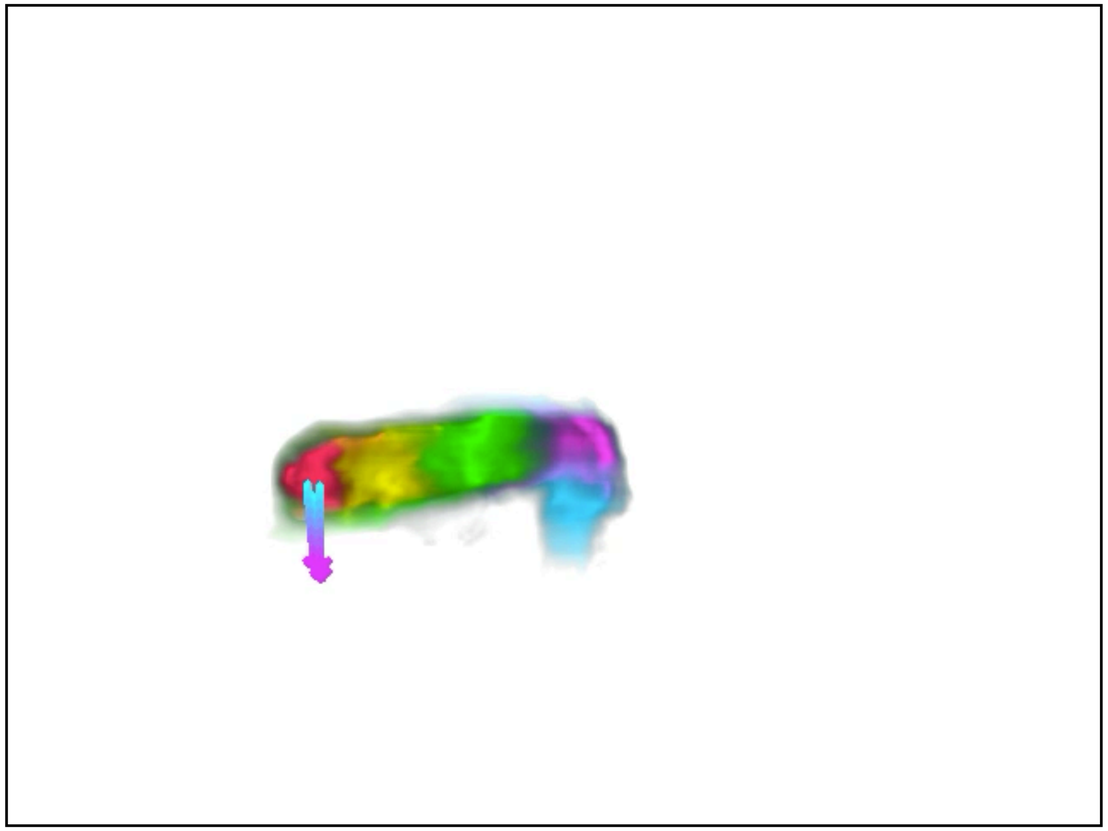


$270 3D-printed robot

Input RGB + Command



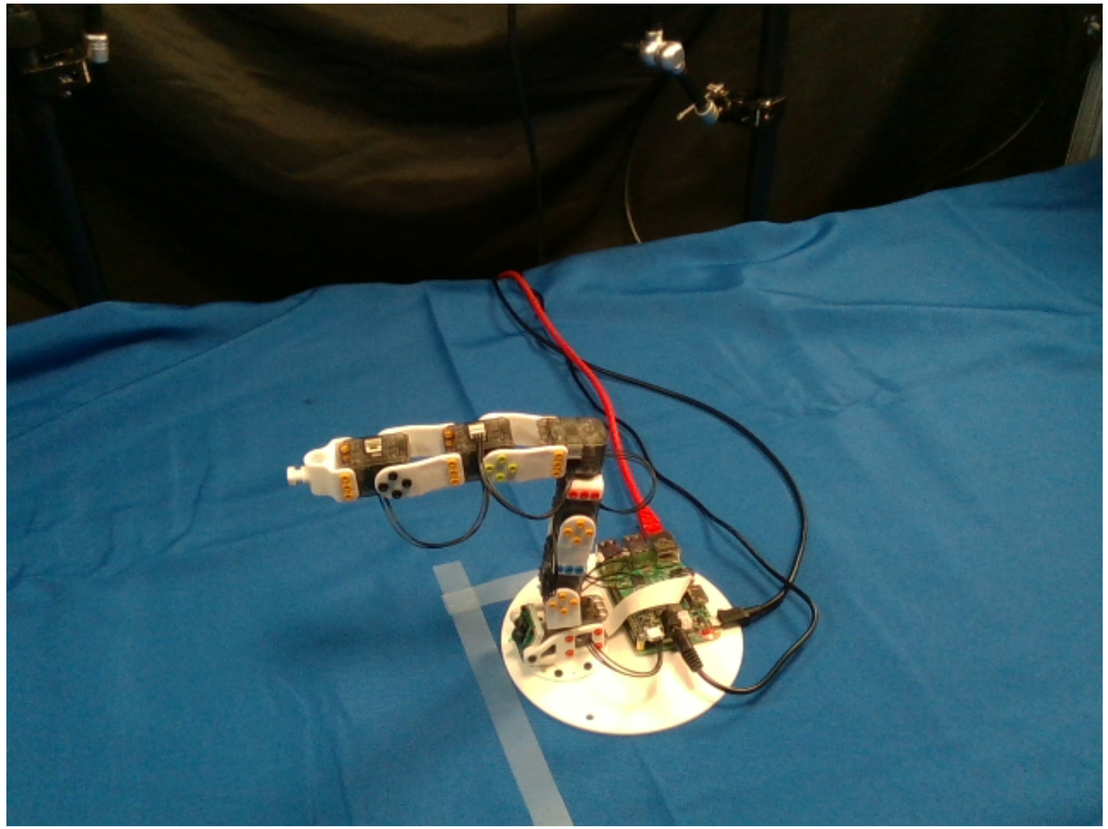
Predicted 3D Motion



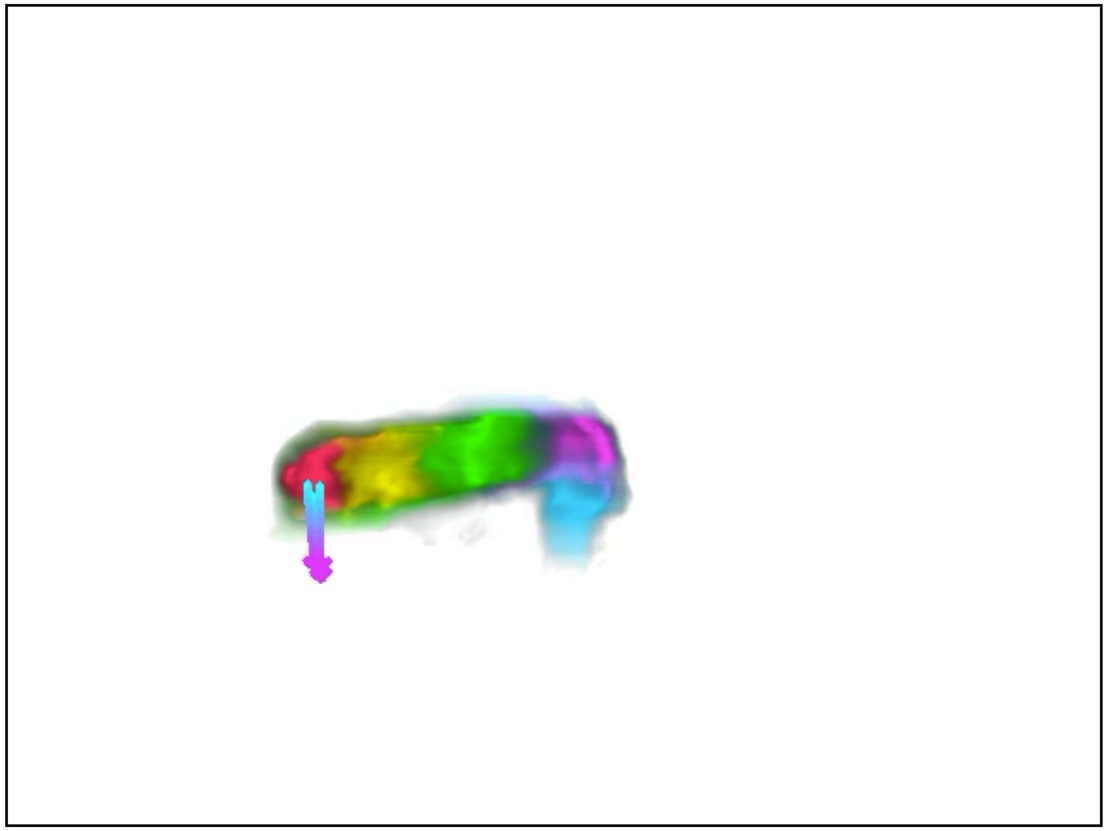
3D Predicted Motion

$270 3D-printed robot

Input RGB + Command



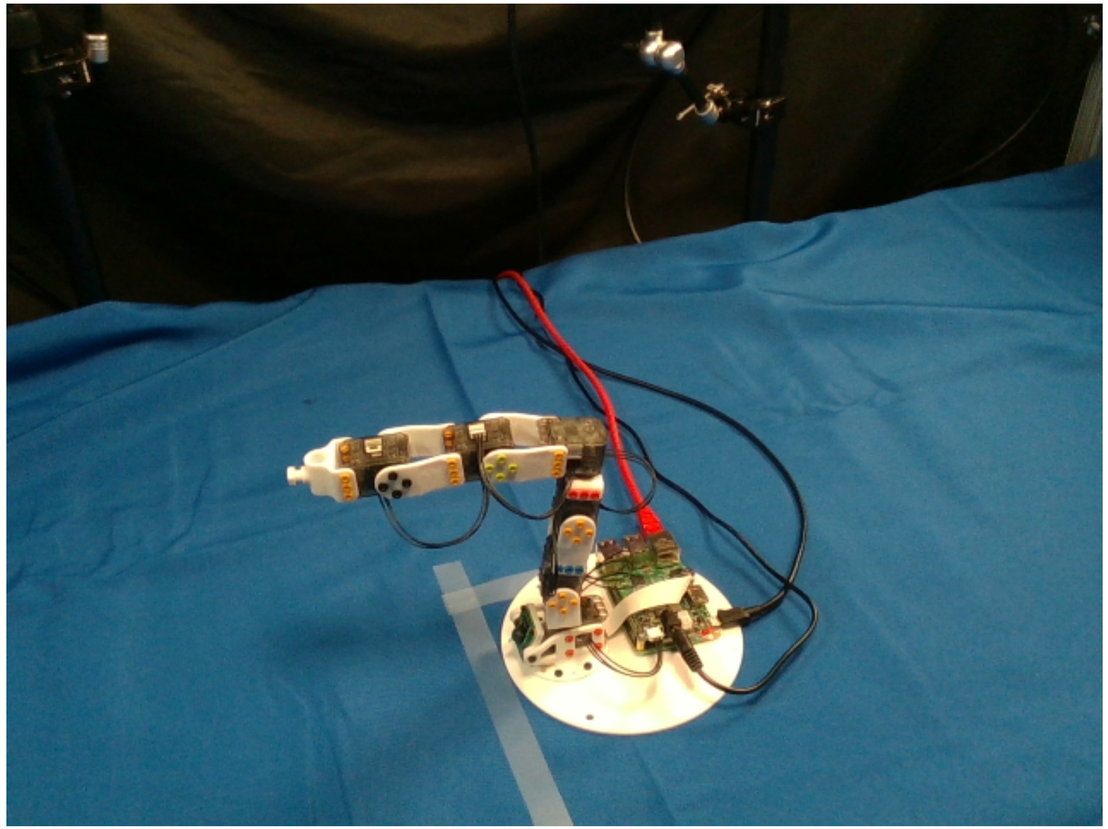
Predicted 3D Motion



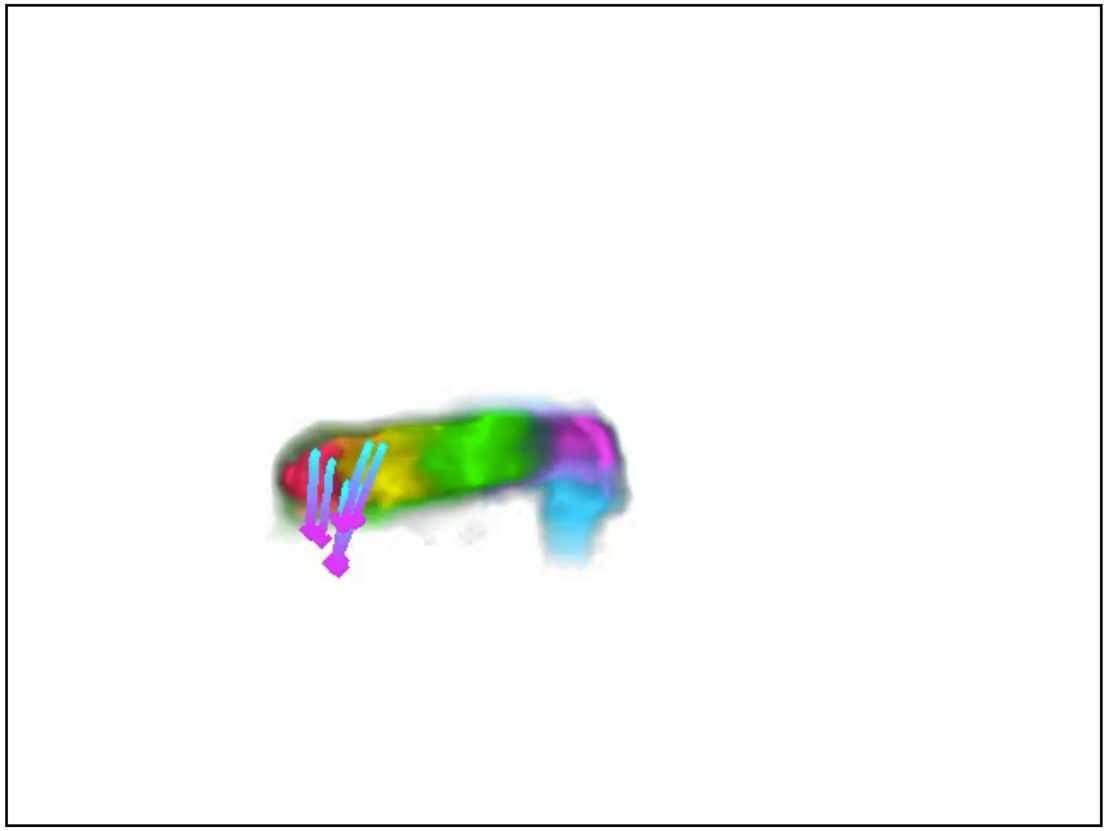
3D Predicted Motion

$270 3D-printed robot

Input RGB + Command



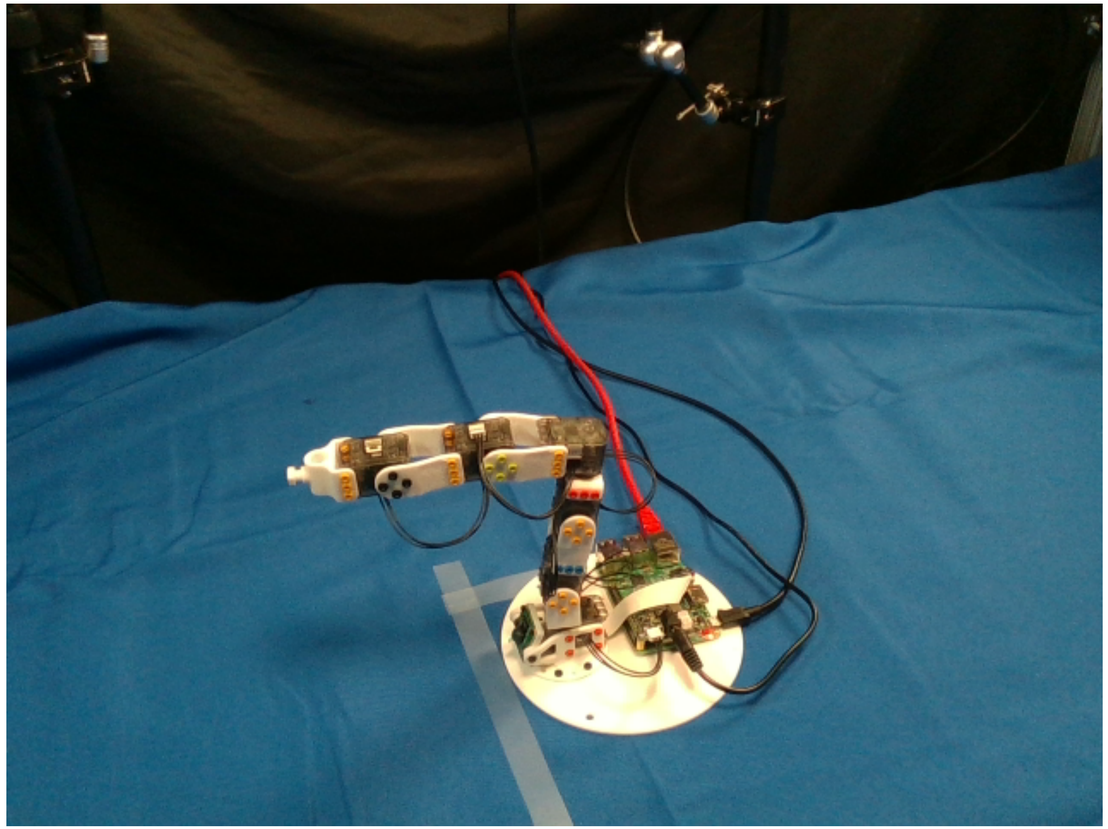
Predicted 3D Motion



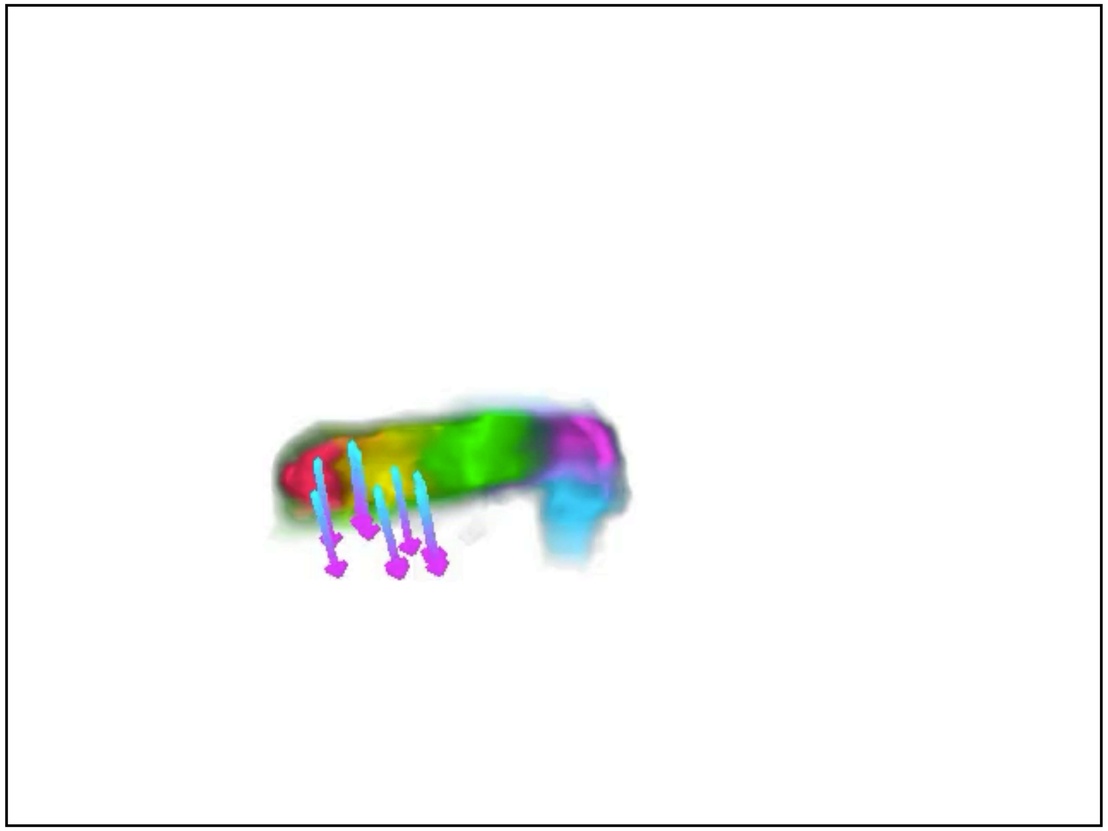
3D Predicted Motion

$270 3D-printed robot

Input RGB + Command



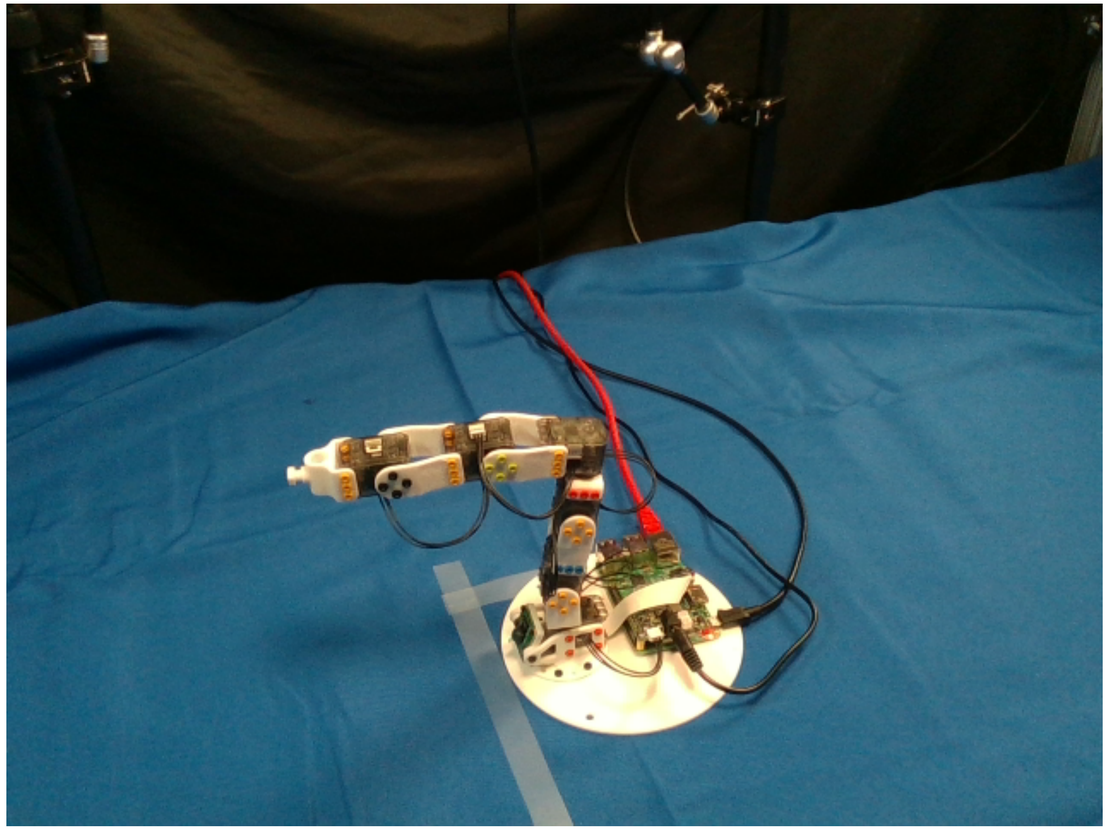
Predicted 3D Motion



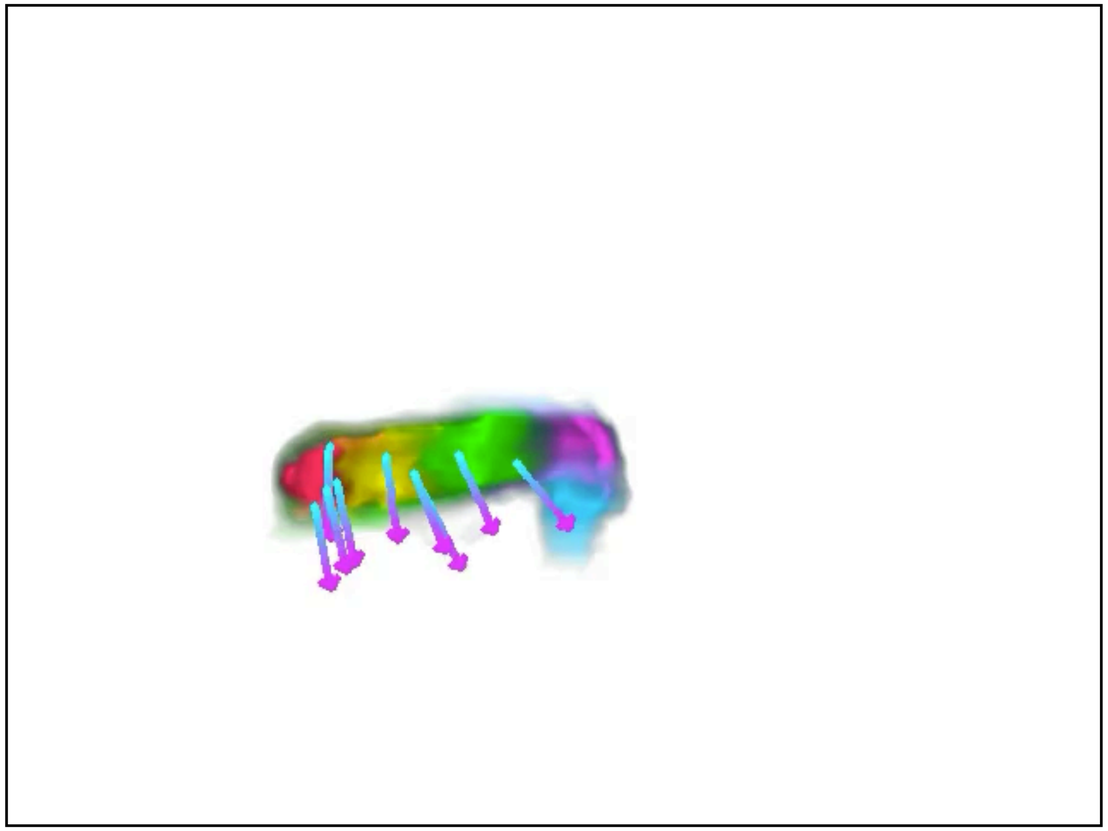
3D Predicted Motion

$270 3D-printed robot

Input RGB + Command



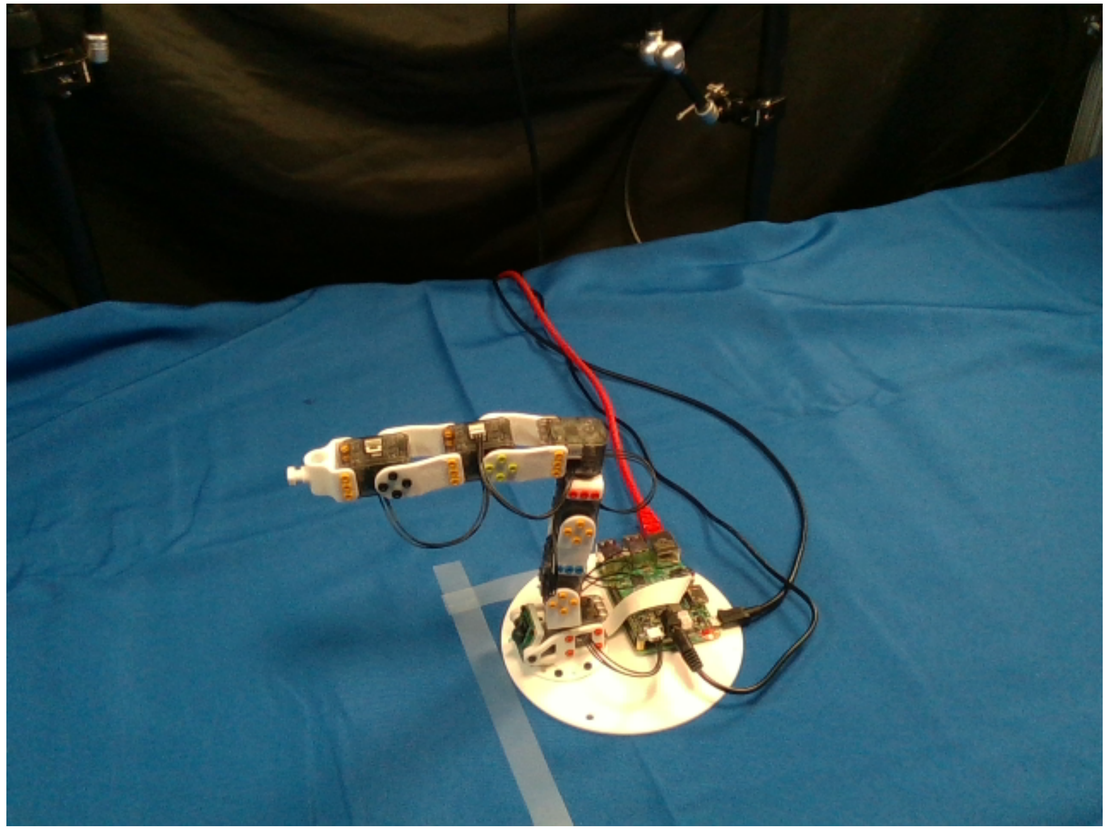
Predicted 3D Motion



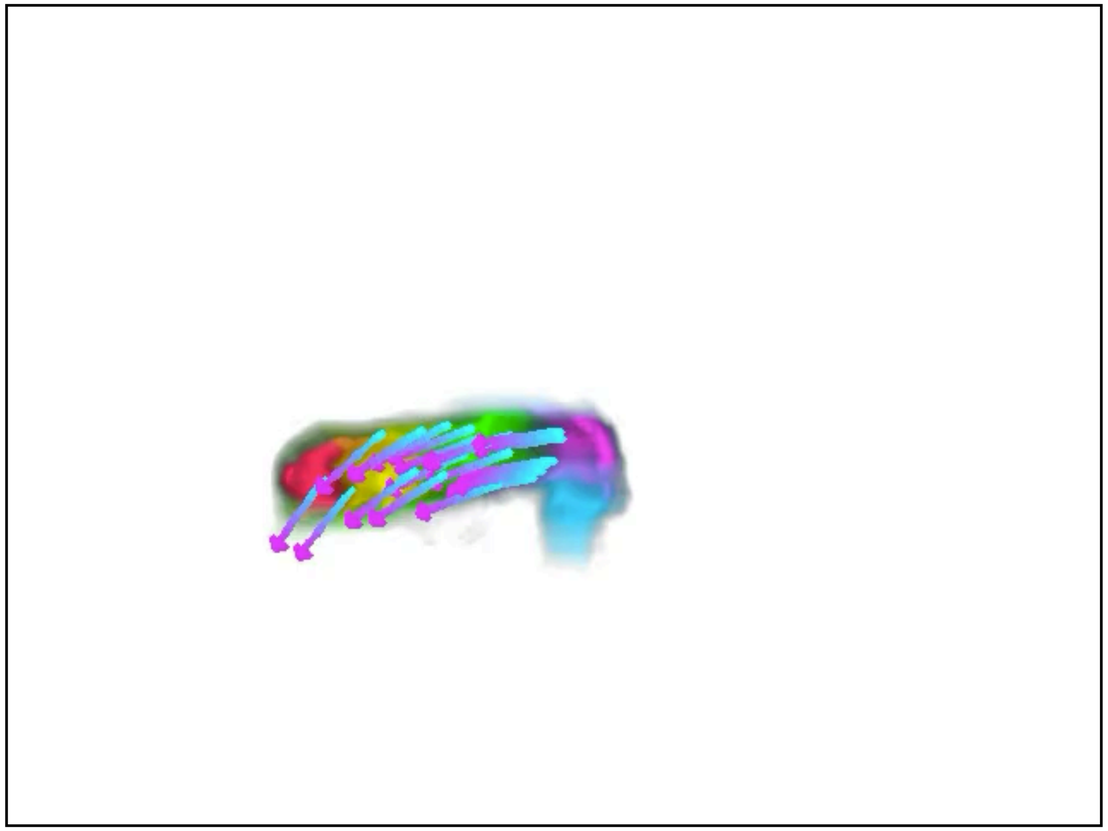
3D Predicted Motion

$270 3D-printed robot

Input RGB + Command

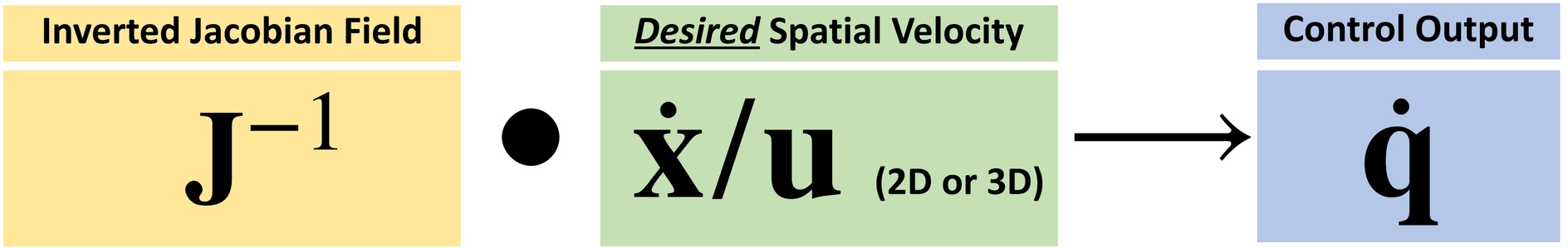


Predicted 3D Motion



3D Predicted Motion

How can we *control* the robot using Jacobian Fields?



How can we *control* the robot using Jacobian Fields?

Inverted Jacobian Field

$$\mathbf{J}^{-1}$$

$\bullet$

Desired **Spatial Velocity**

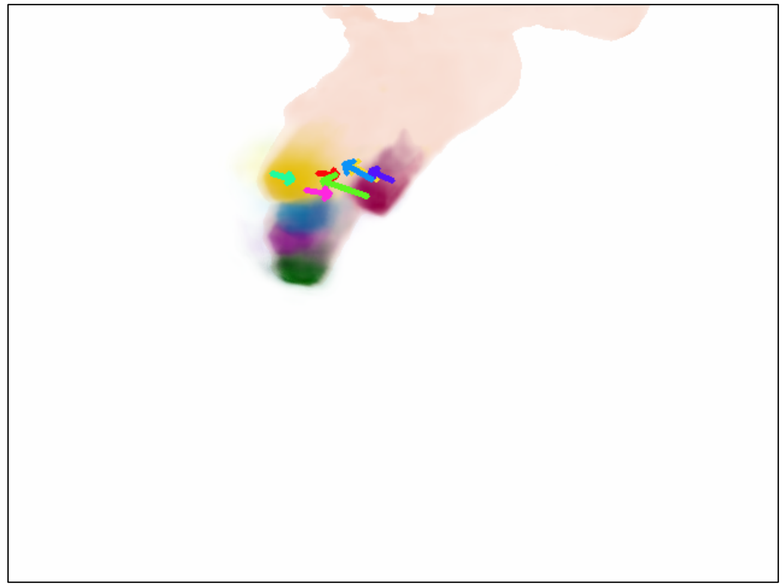
$$\dot{\mathbf{x}}/\mathbf{u}$$ **(2D or 3D)**

$\longrightarrow$

Control Output

$$\dot{\mathbf{q}}$$

Desired **Spatial Velocity (2D)**



How can we *control* the robot using Jacobian Fields?

Inverted Jacobian Field

$$\mathbf{J}^{-1}$$

$\bullet$

Desired **Spatial Velocity**

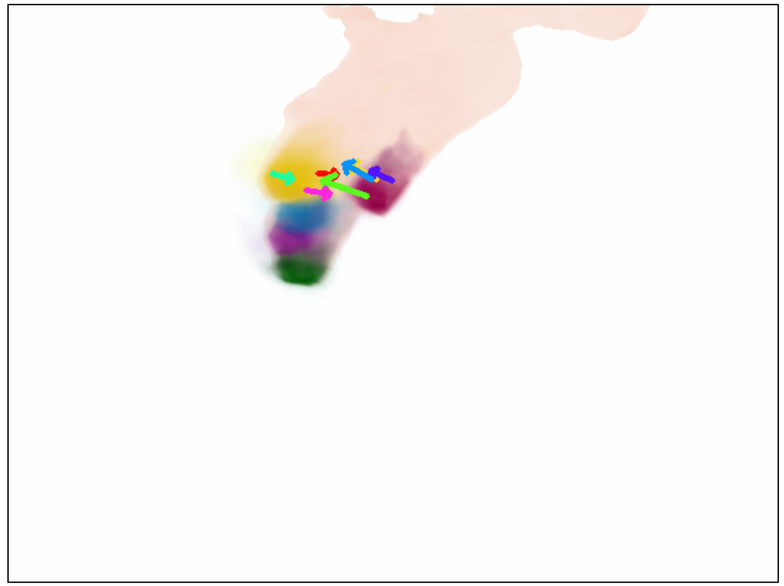
$$\dot{\mathbf{x}}/\mathbf{u}$$ **(2D or 3D)**

$\longrightarrow$

Control Output

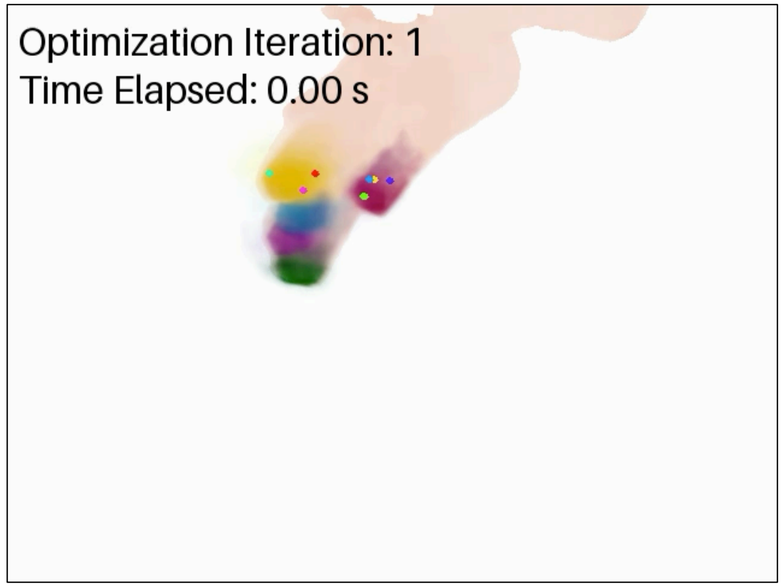
$$\dot{\mathbf{q}}$$

Desired **Spatial Velocity (2D)**



$\longrightarrow$

Optimization Solution



How can we *control* the robot using Jacobian Fields?

Inverted Jacobian Field

$$\mathbf{J}^{-1}$$

$\bullet$

Desired **Spatial Velocity**

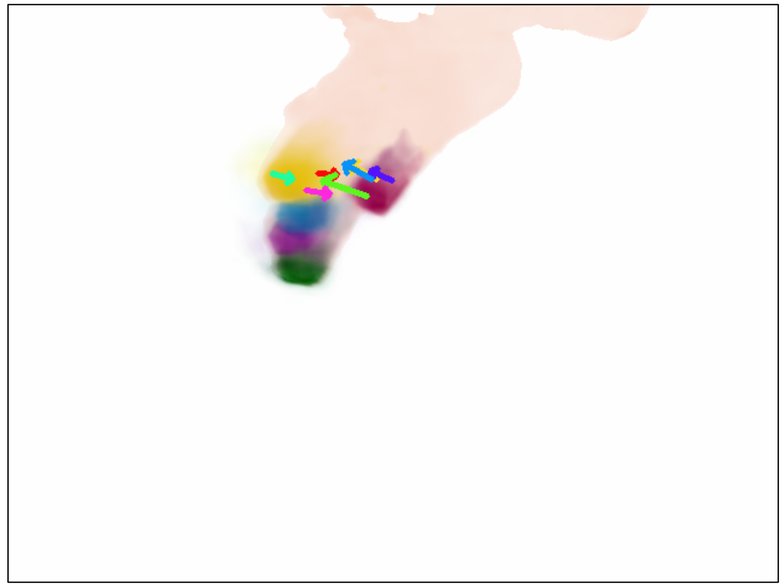
$$\dot{\mathbf{x}}/\mathbf{u}$$ **(2D or 3D)**

$\longrightarrow$

Control Output

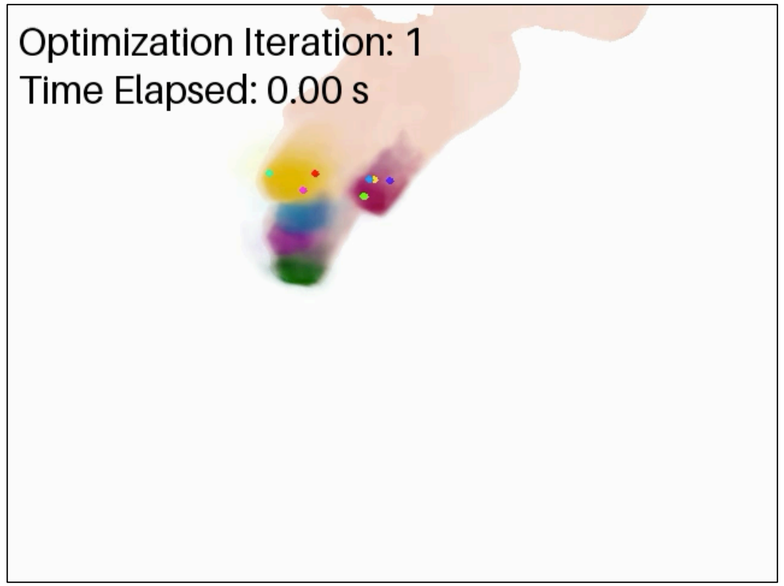
$$\dot{\mathbf{q}}$$

Desired **Spatial Velocity (2D)**

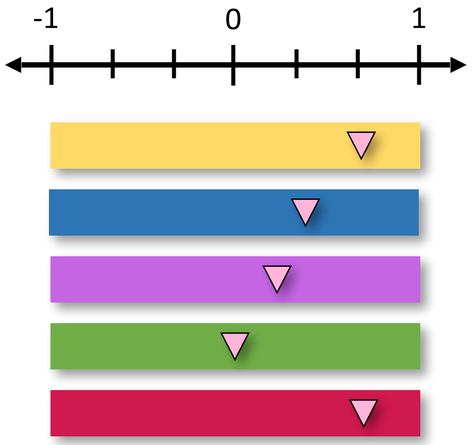


$\longrightarrow$

Optimization Solution



Control Output



How can we *control* the robot using Jacobian Fields?

Inverted Jacobian Field

$$\mathbf{J}^{-1}$$

$\bullet$

Desired **Spatial Velocity**

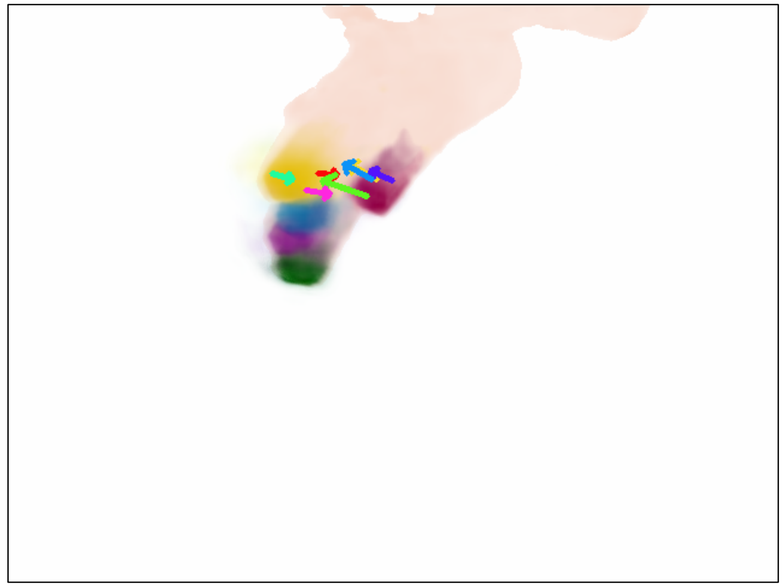
$$\dot{\mathbf{x}}/\mathbf{u}$$ **(2D or 3D)**

$\longrightarrow$

Control Output

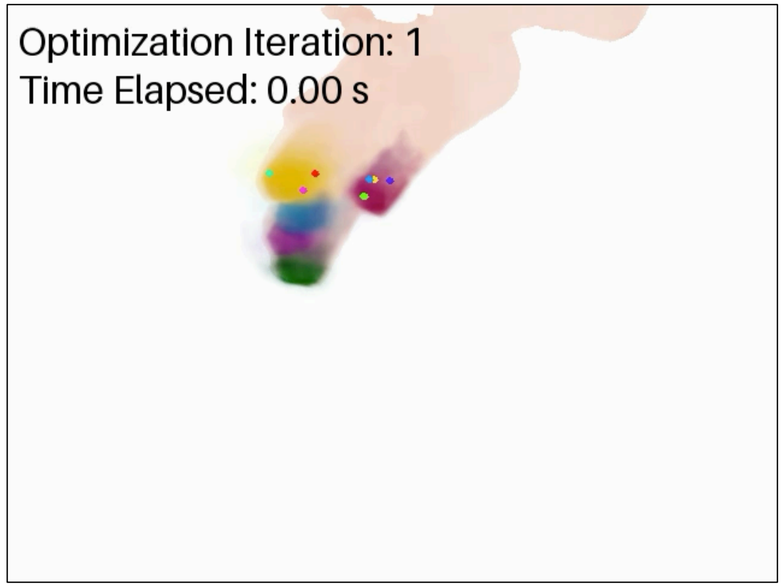
$$\dot{\mathbf{q}}$$

Desired **Spatial Velocity (2D)**

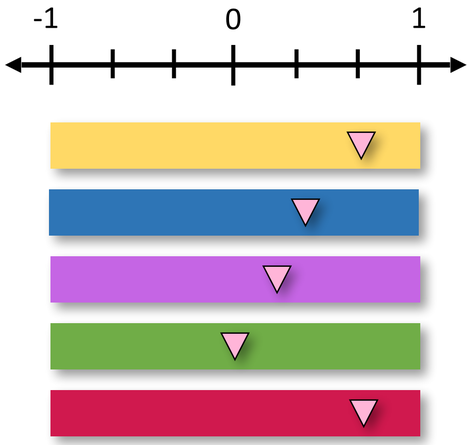


$\longrightarrow$

Optimization Solution



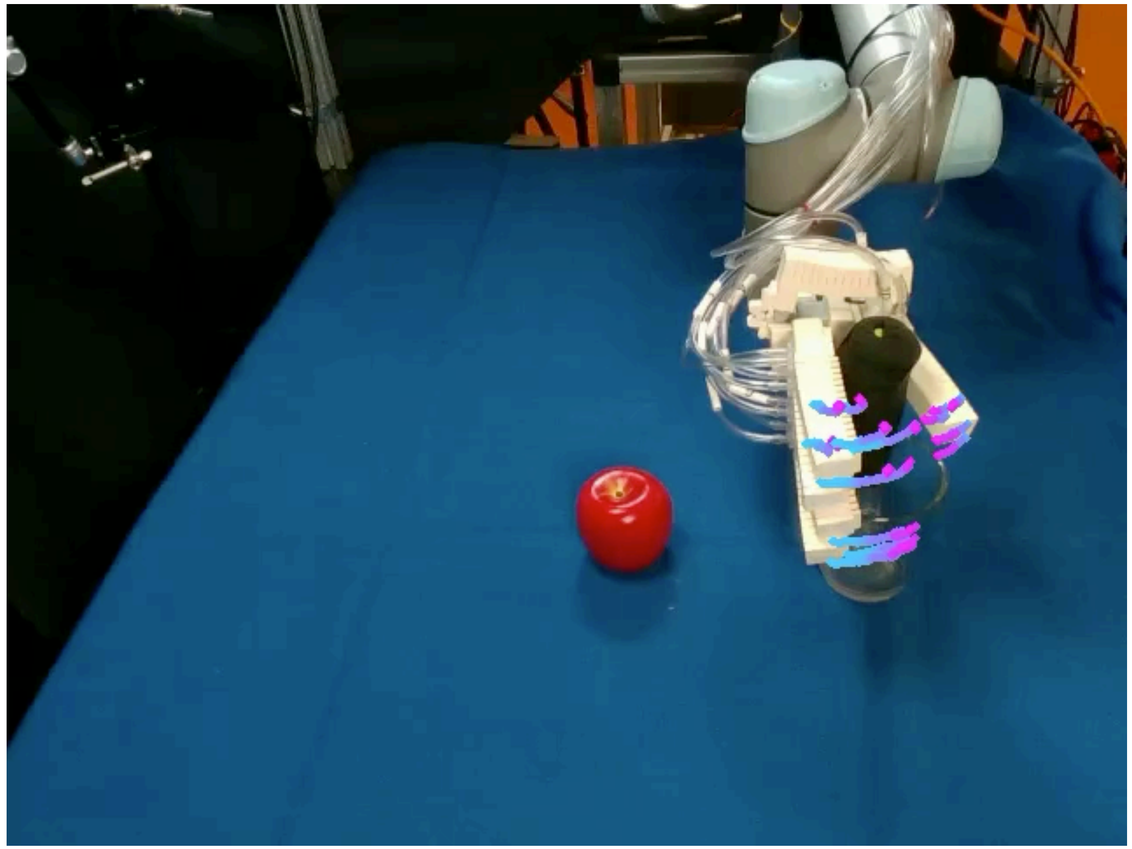
Control Output



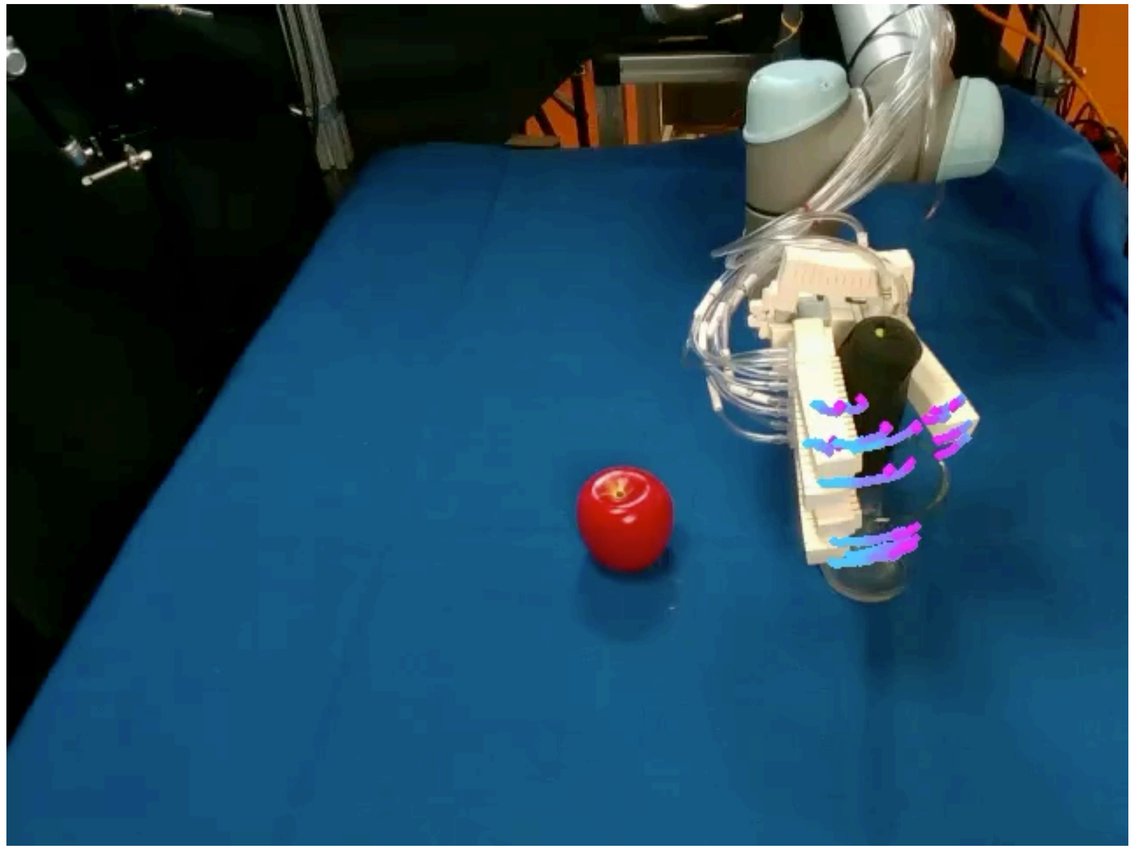
Manipulation results: Pneumatic Hand

Task: grasp tool and push apple

Motion Plan



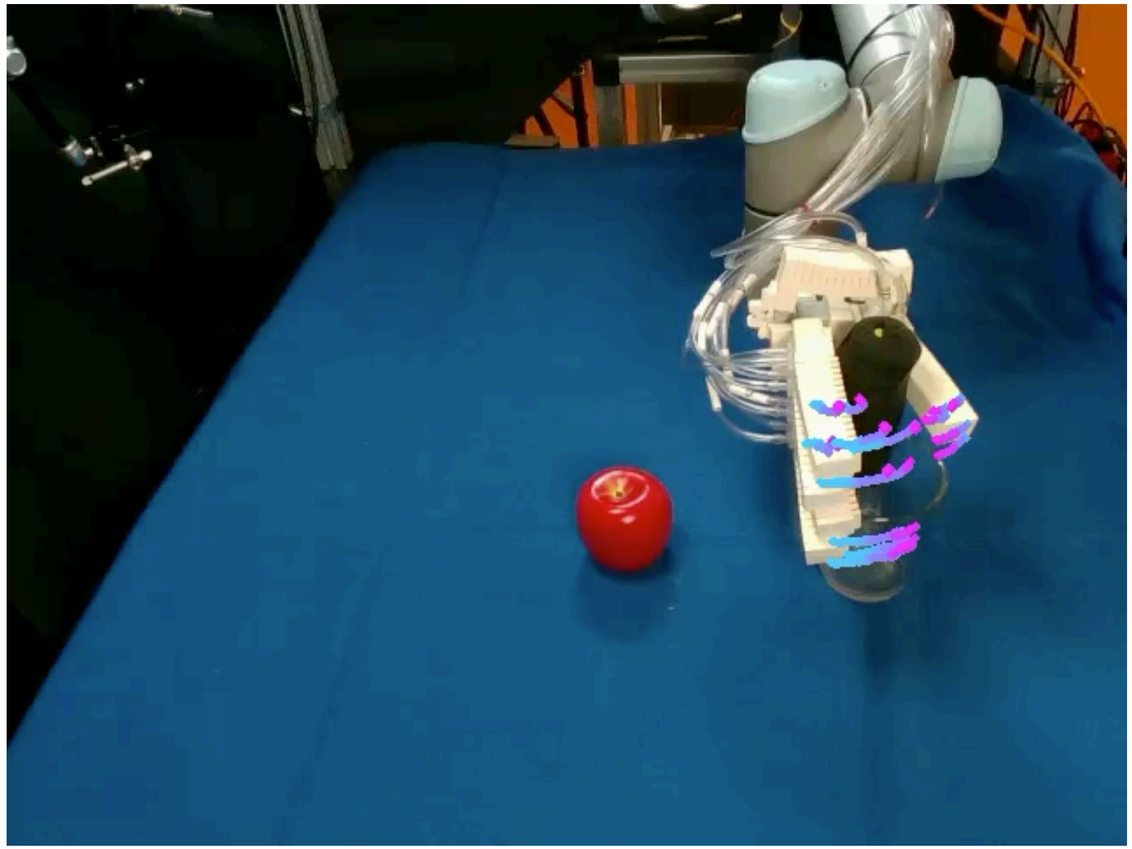
Execution



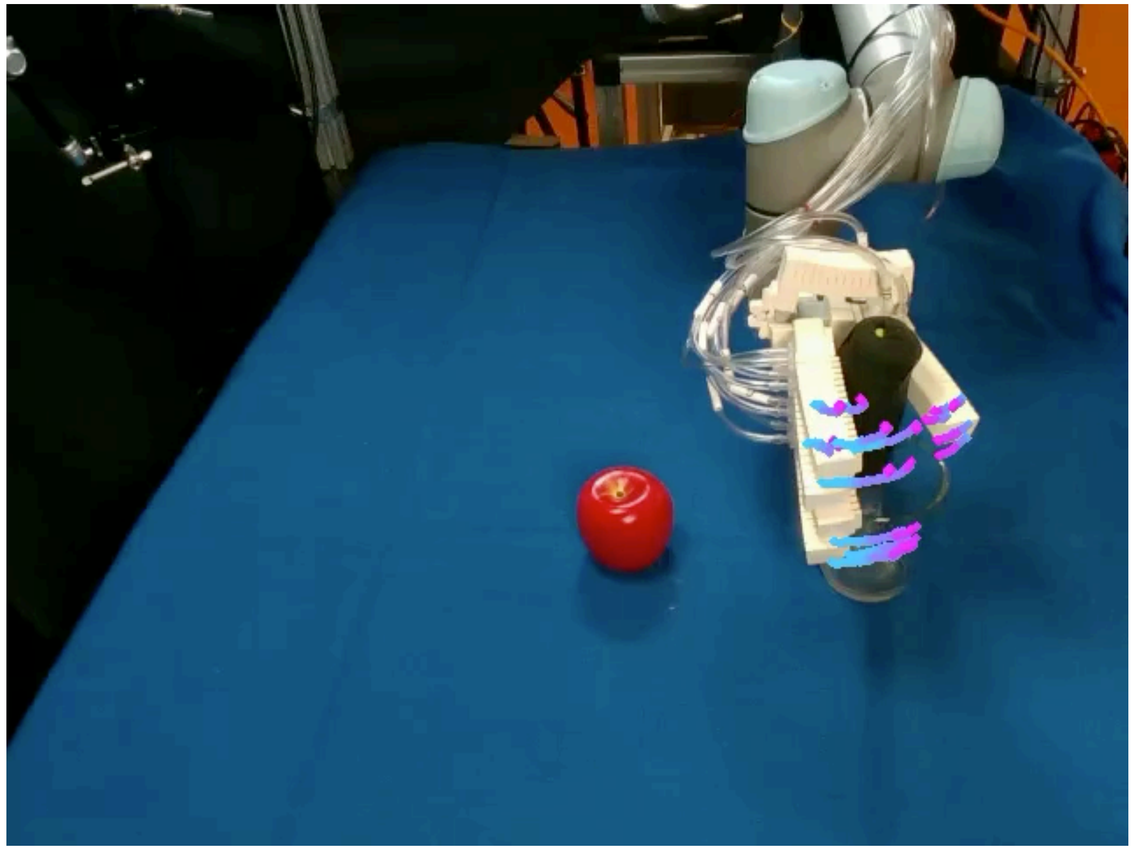
Manipulation results: Pneumatic Hand

Task: grasp tool and push apple

Motion Plan



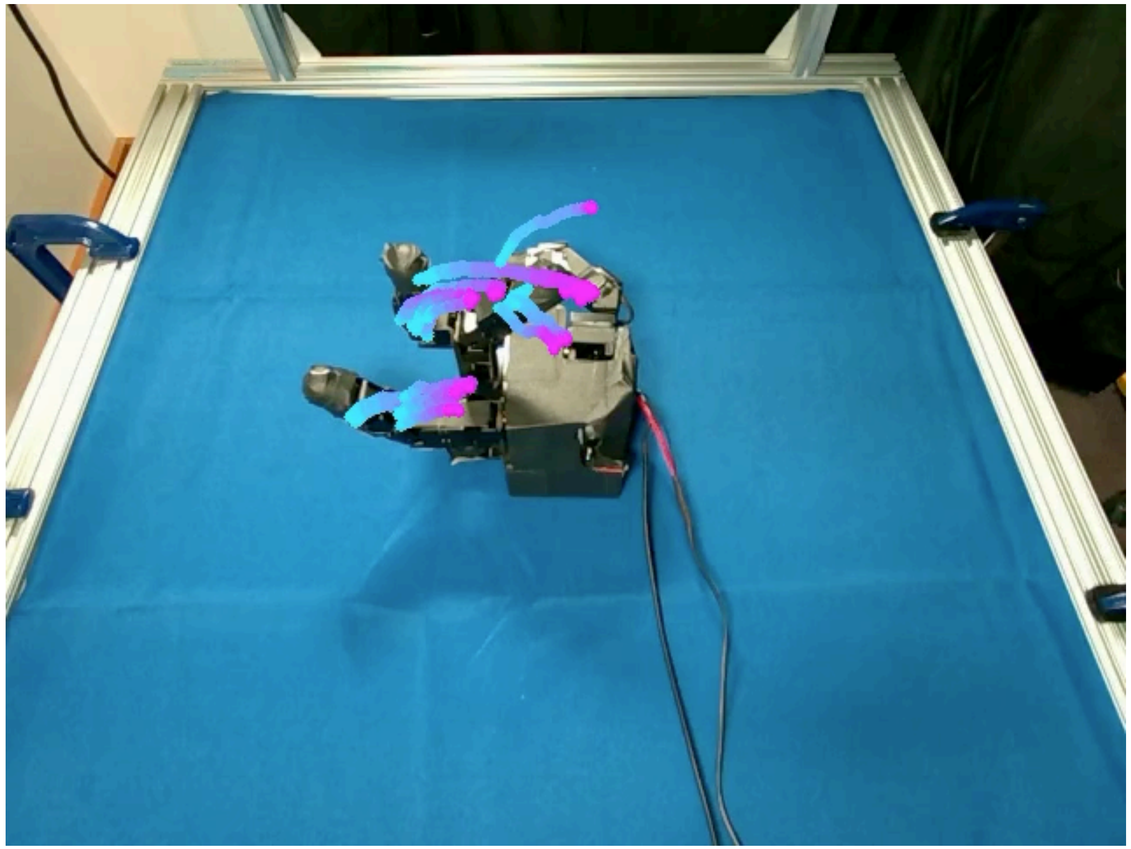
Execution



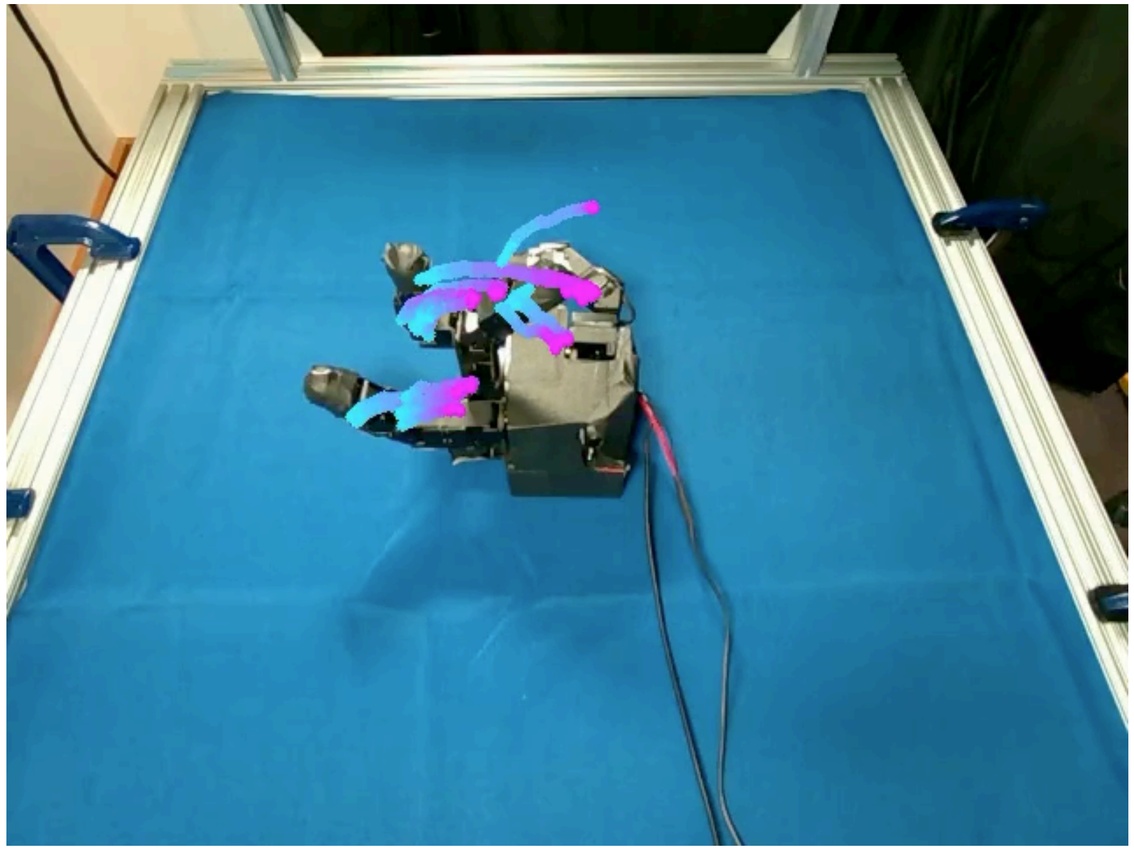
Manipulation results: Allegro Hand

Task: close fingers to achieve a pose

Motion Plan



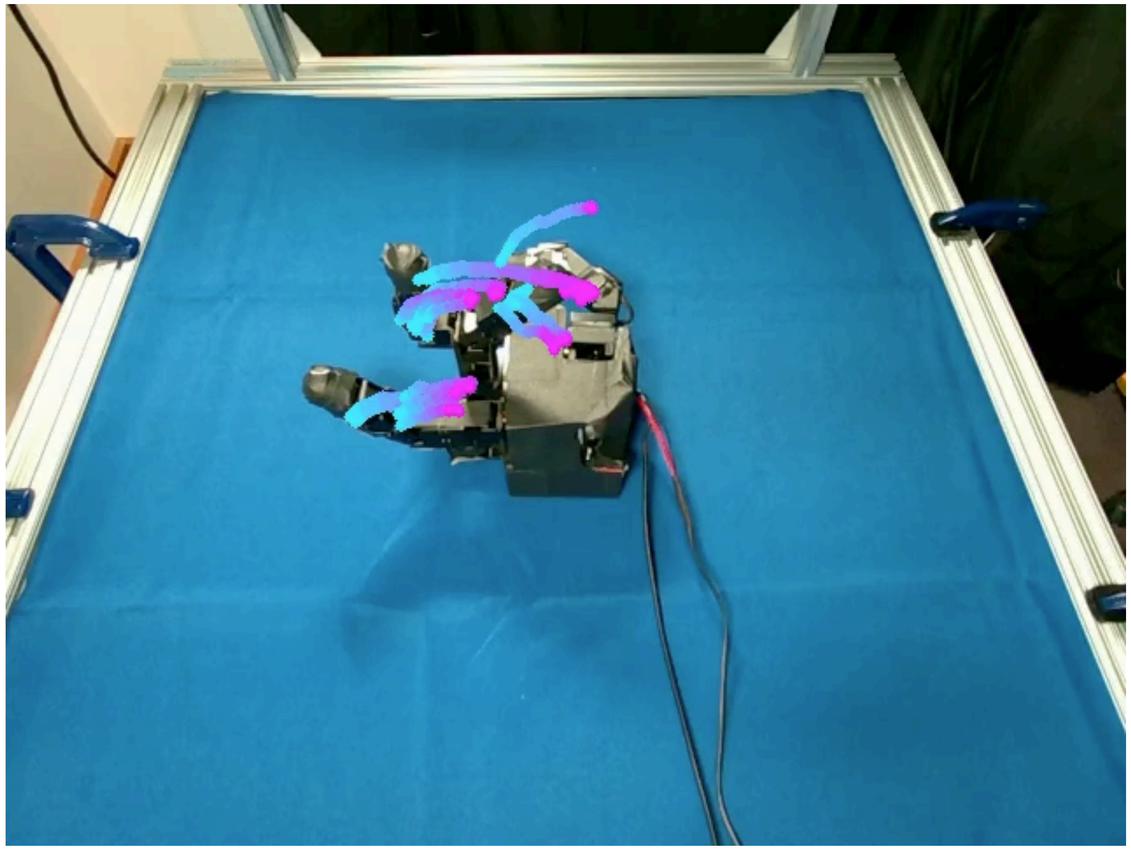
Execution



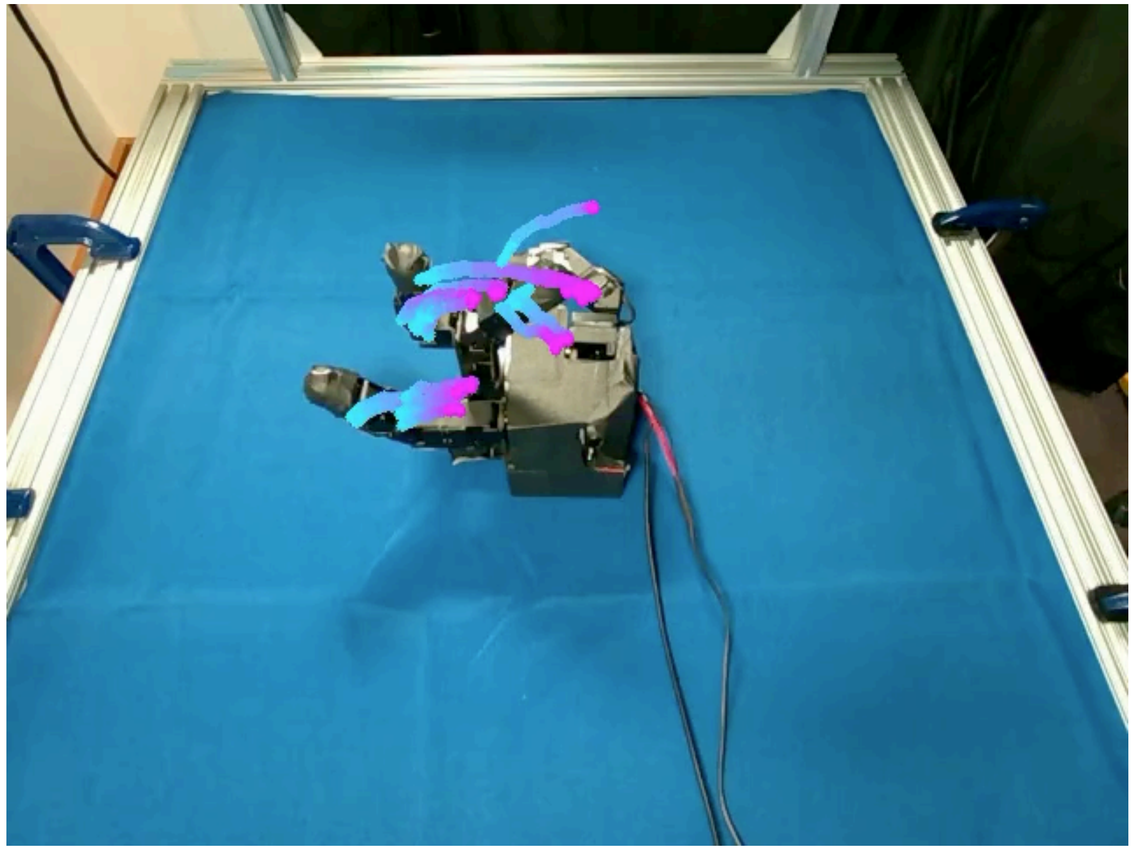
Manipulation results: Allegro Hand

Task: close fingers to achieve a pose

Motion Plan



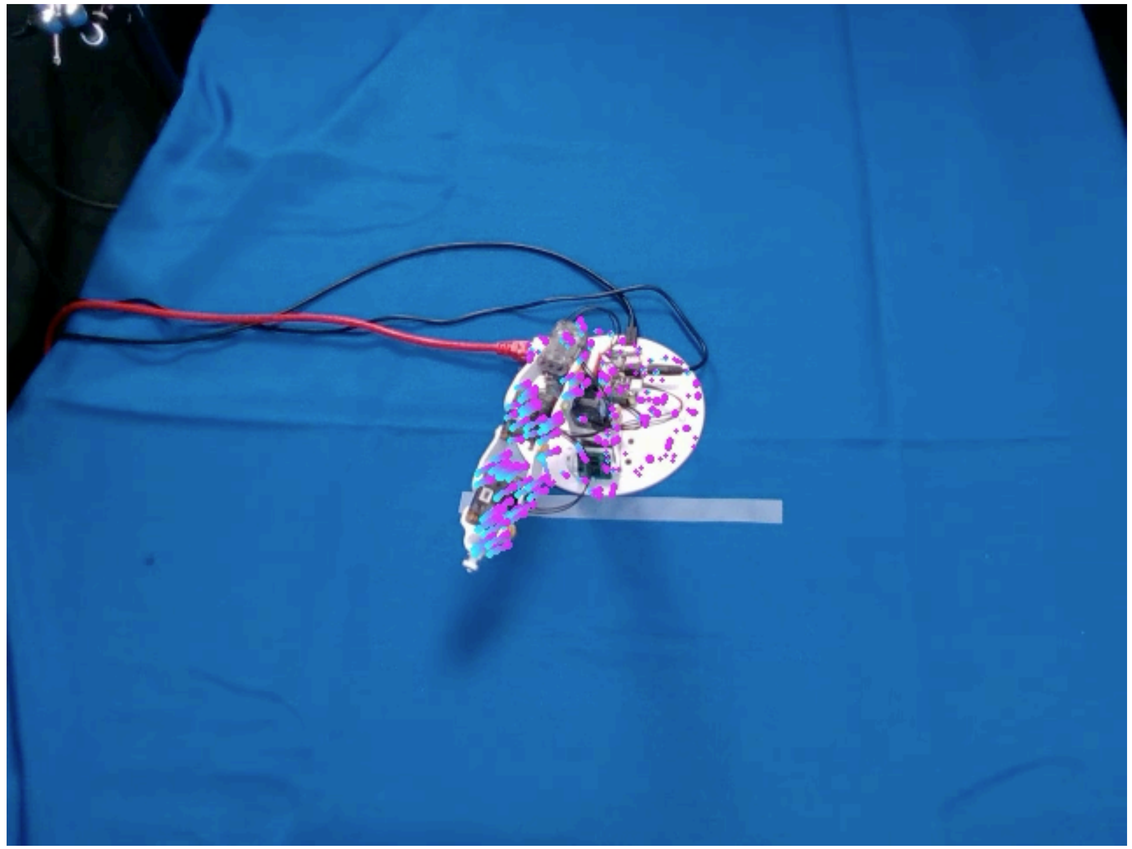
Execution



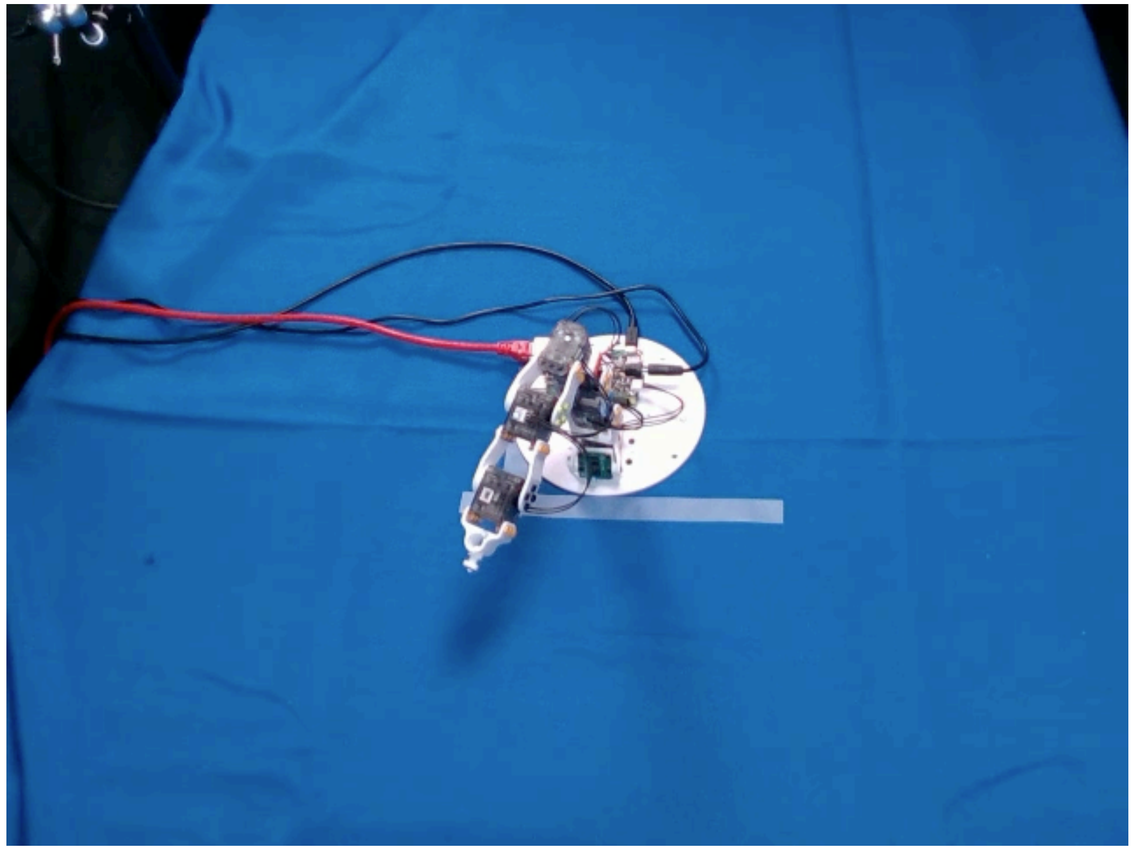
Manipulation results: Poppy Robot Arm

Task: draw the **letter “M”** in the air

Desired Motion



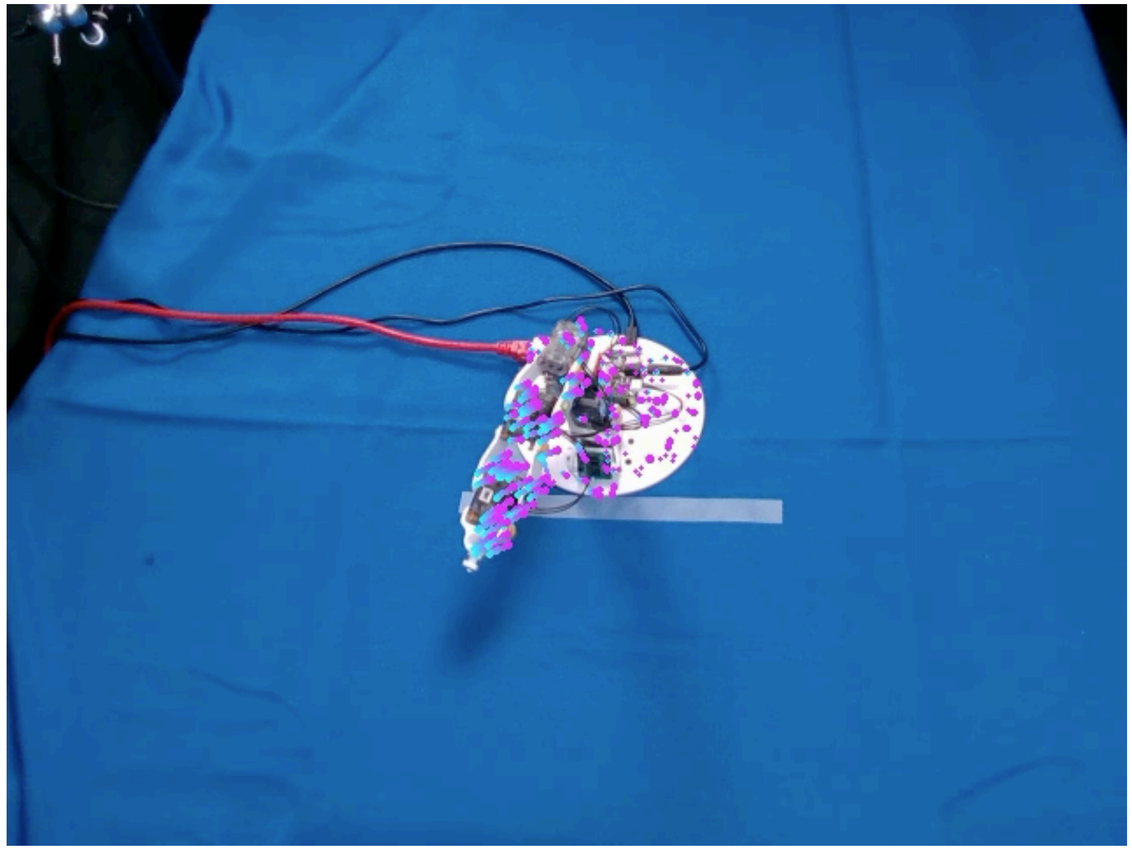
Achieved Trajectory



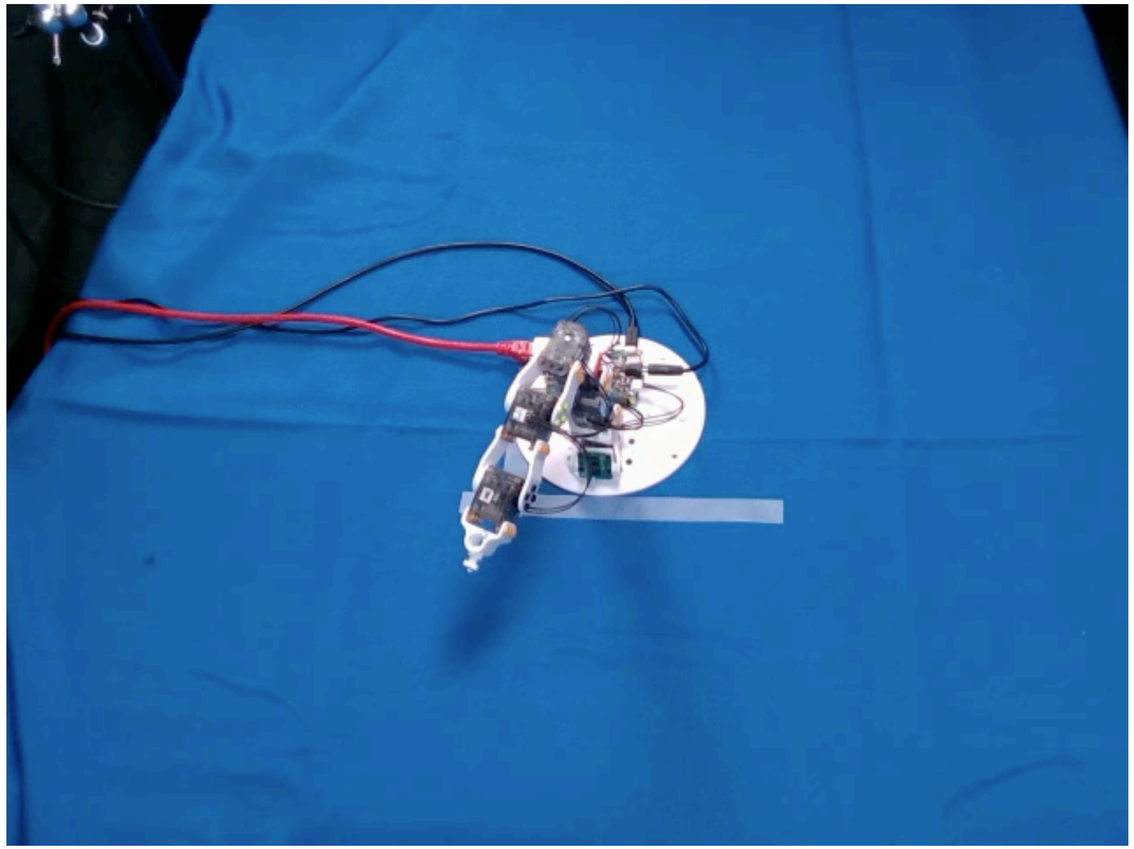
Manipulation results: Poppy Robot Arm

Task: draw the **letter “M”** in the air

Desired Motion



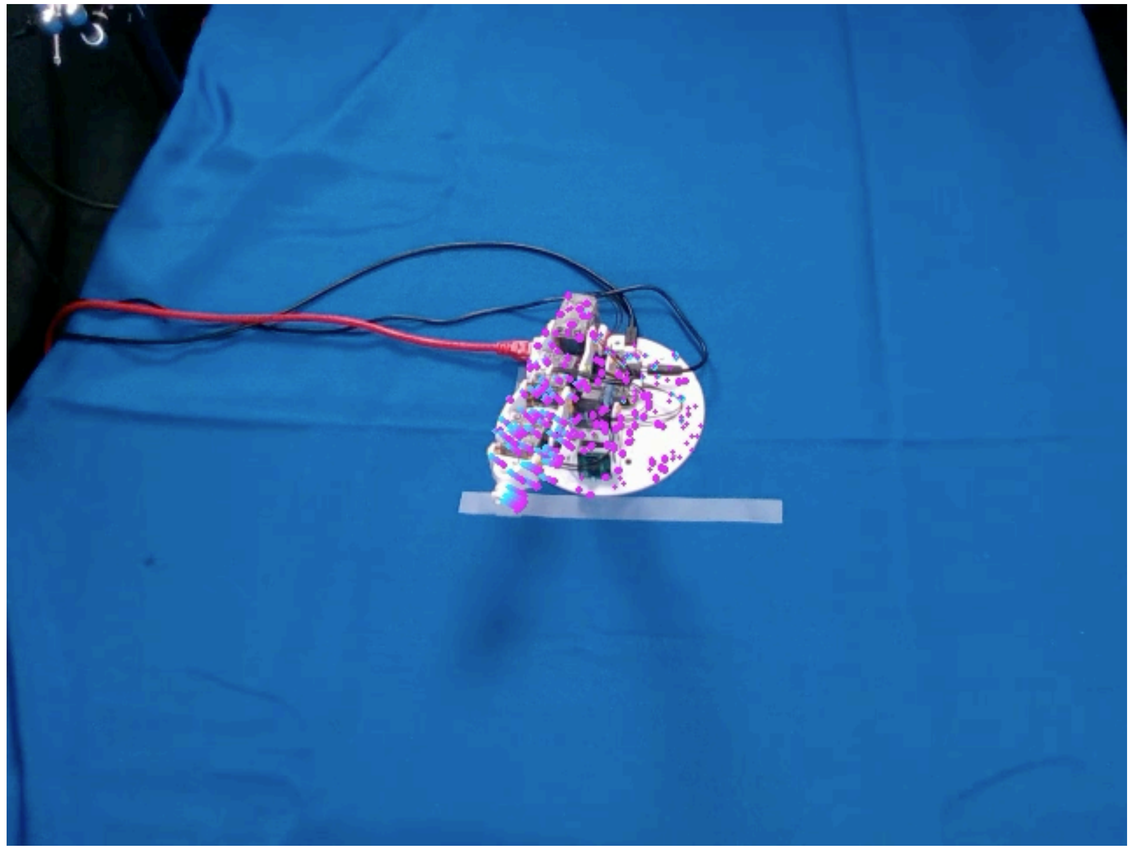
Achieved Trajectory



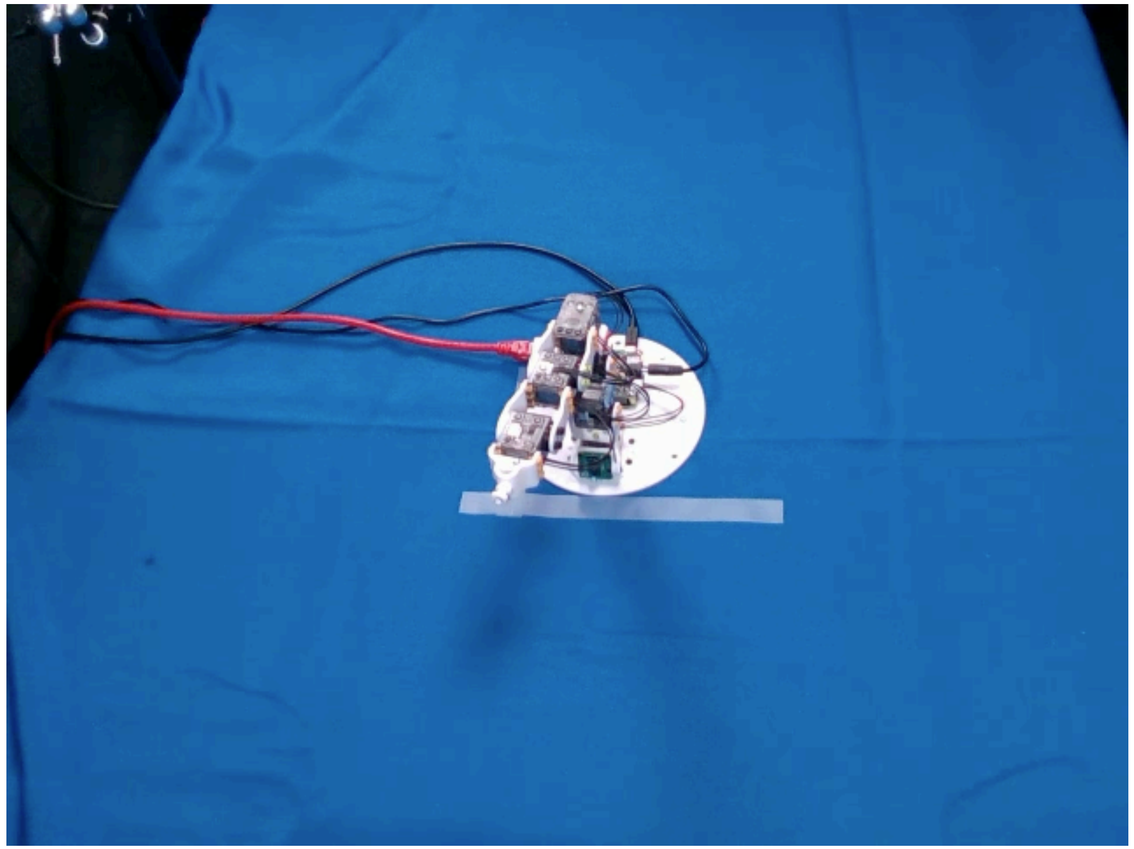
Manipulation results: Poppy Robot Arm

Task: draw the **letter “l”** in the air

Desired Motion



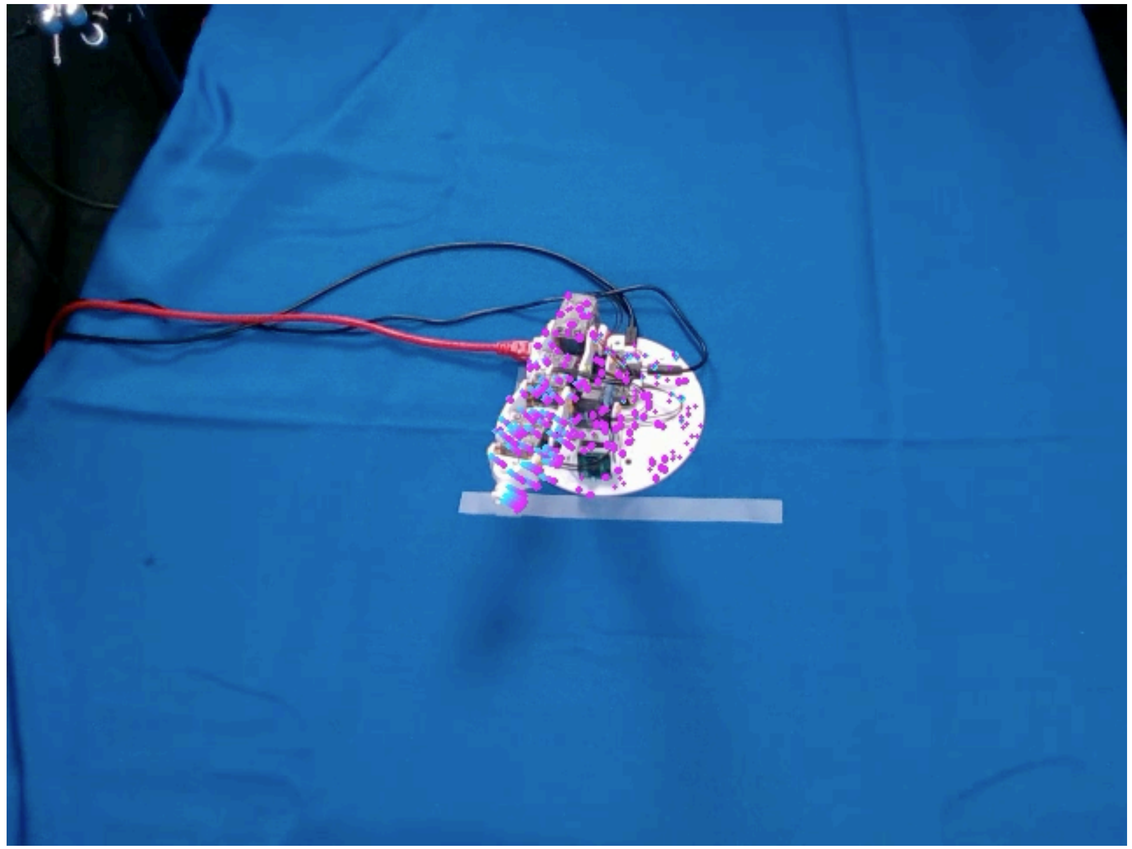
Achieved Trajectory



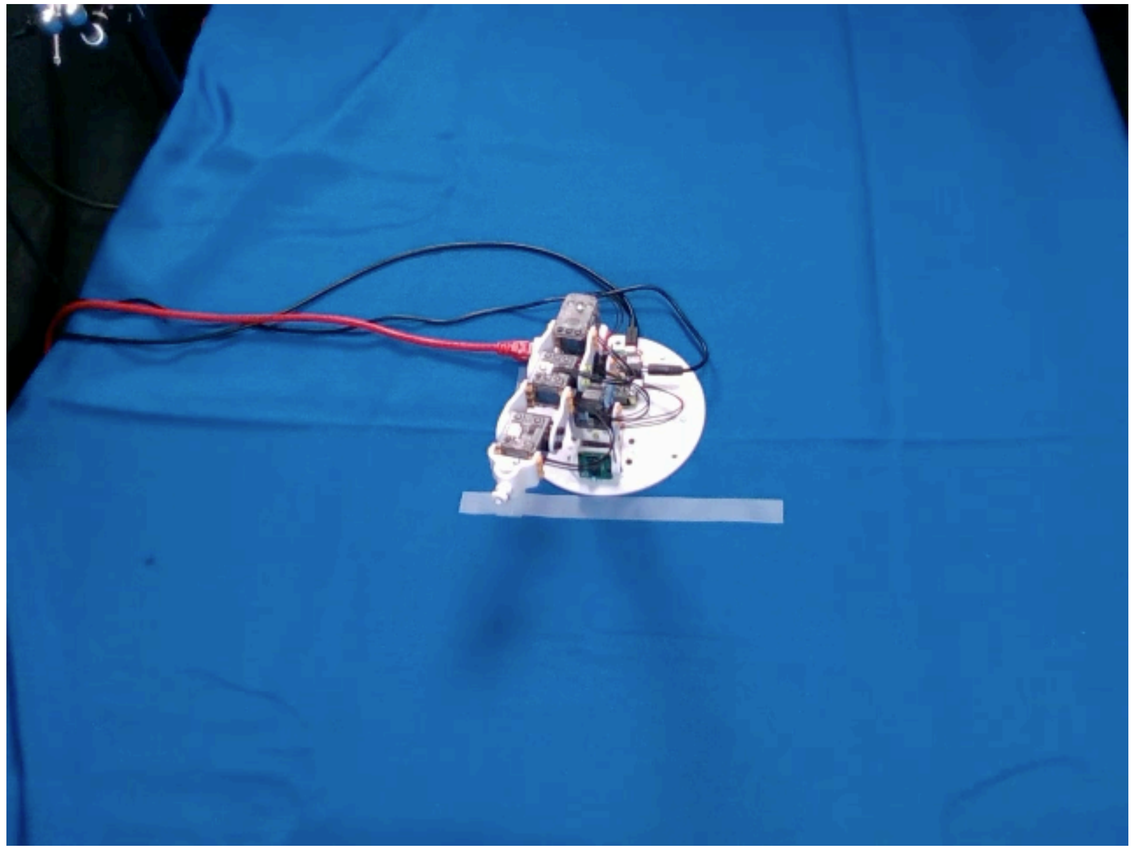
Manipulation results: Poppy Robot Arm

Task: draw the **letter “l”** in the air

Desired Motion



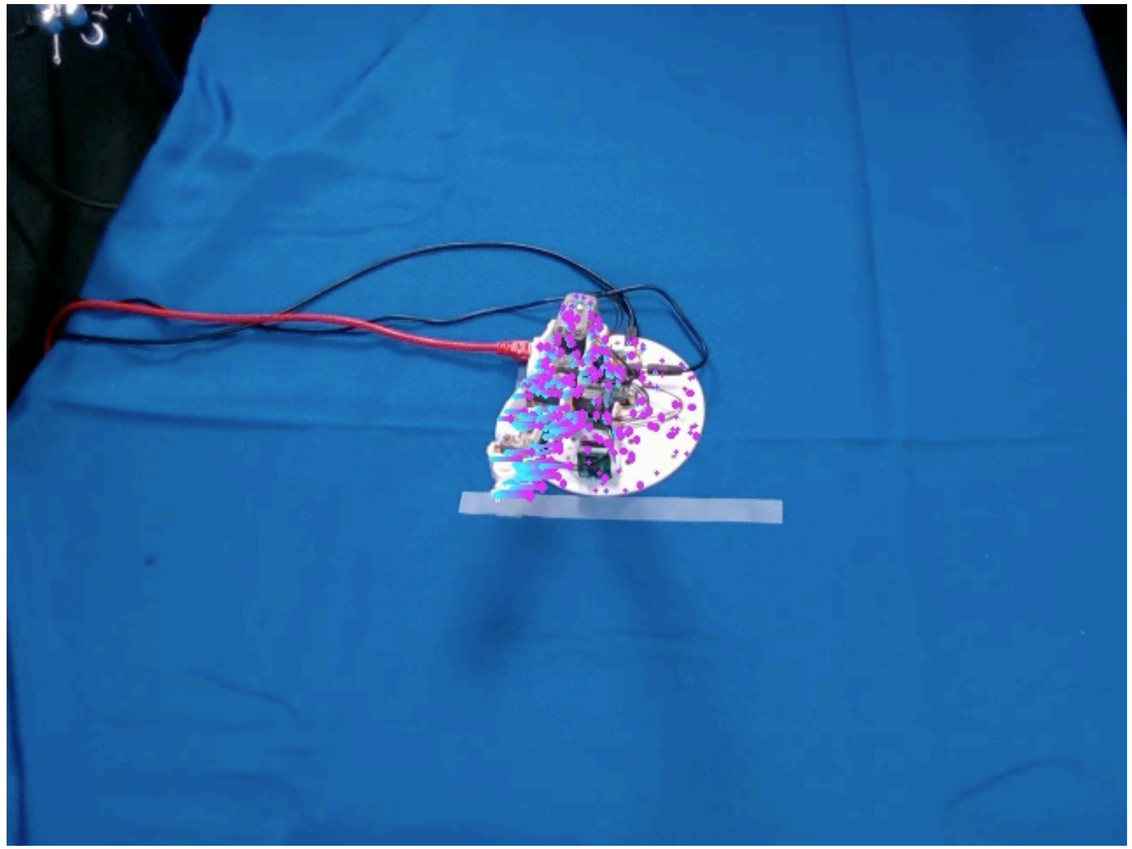
Achieved Trajectory



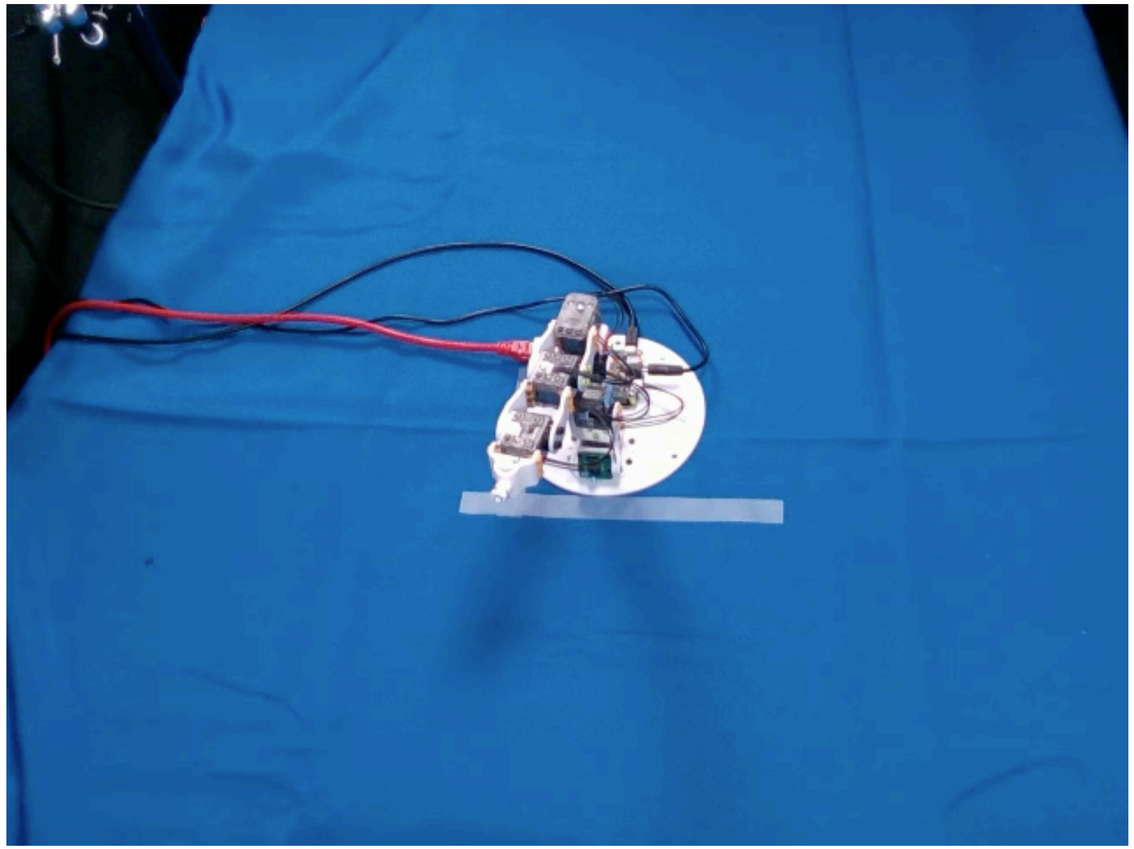
Manipulation results: Poppy Robot Arm

Task: draw the **letter “T”** in the air

Desired Motion



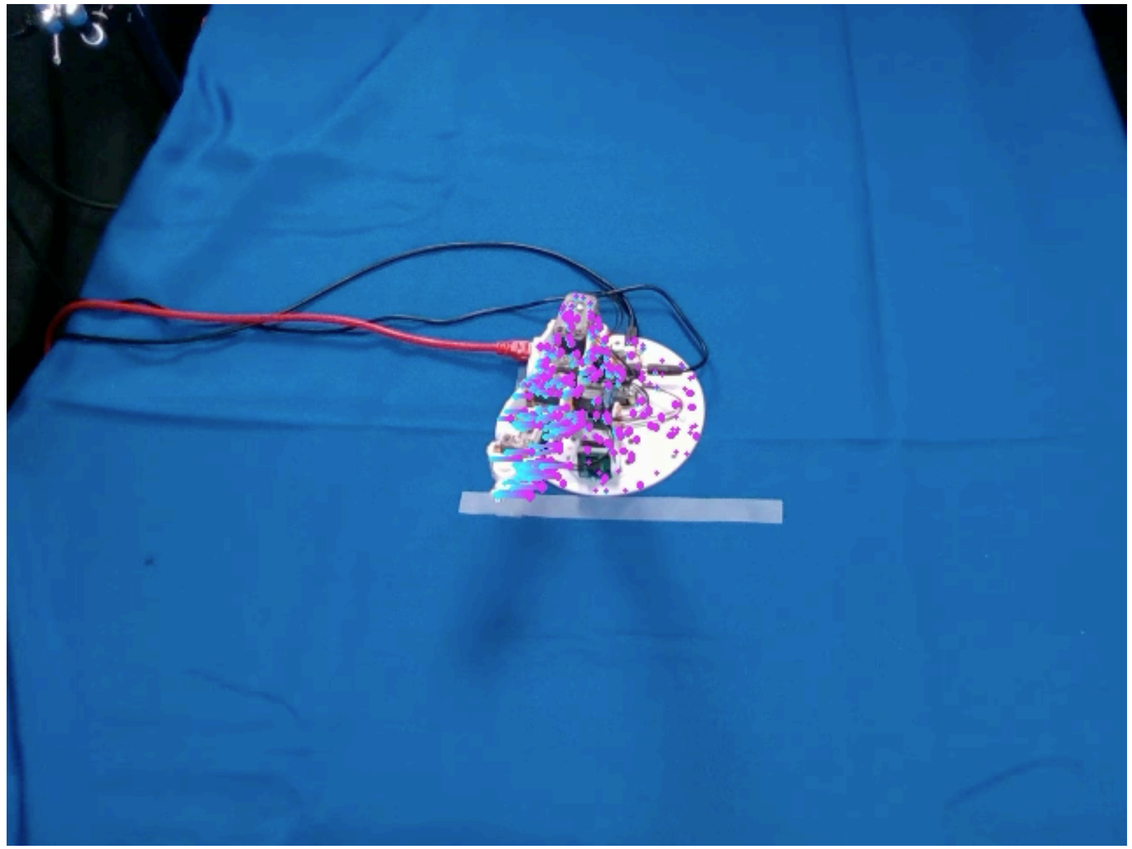
Achieved Trajectory



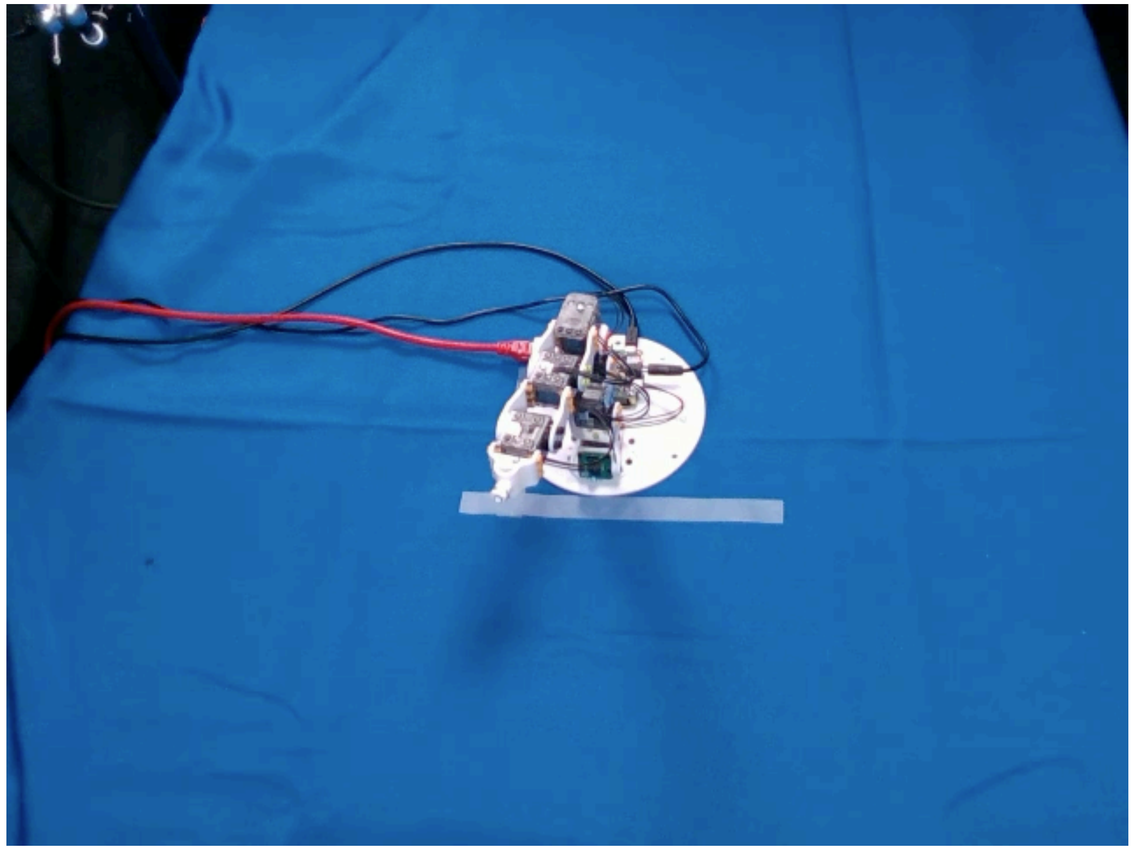
Manipulation results: Poppy Robot Arm

Task: draw the **letter “T”** in the air

Desired Motion

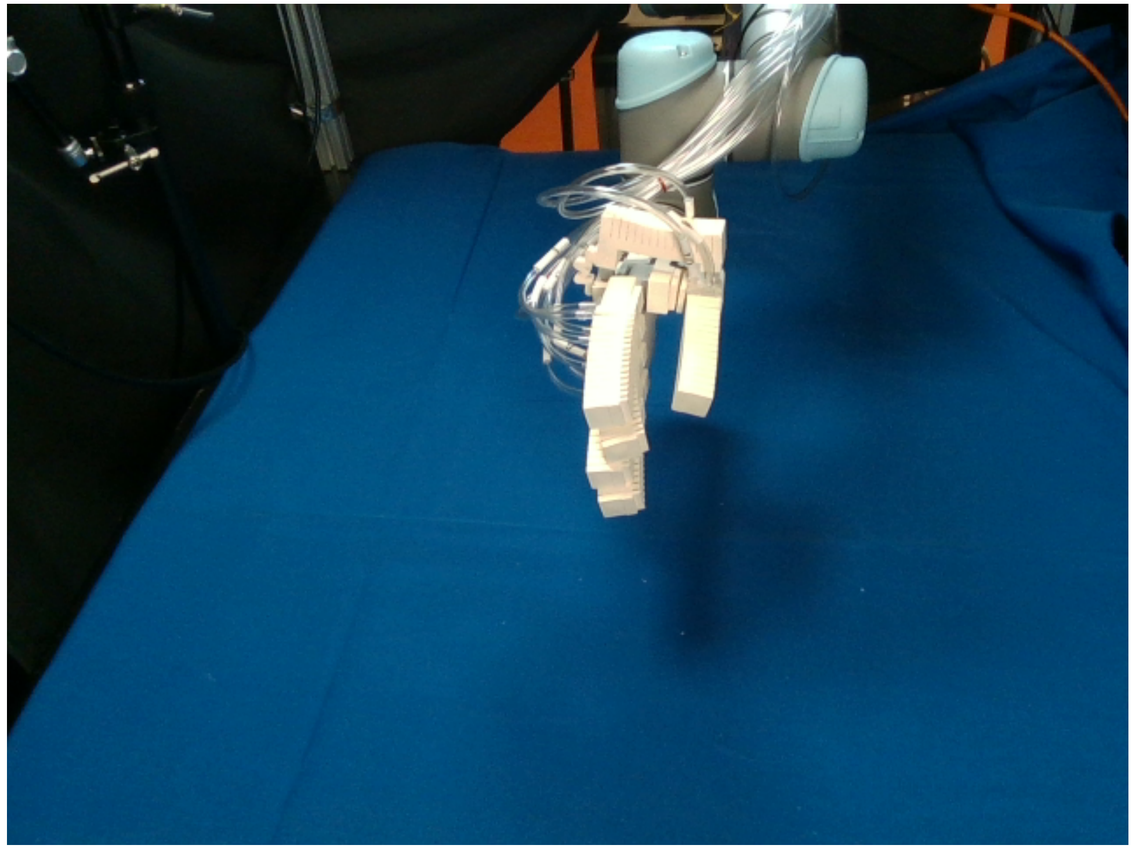


Achieved Trajectory



Control Robots we Couldn’t Before & New Representation for Interactive Video

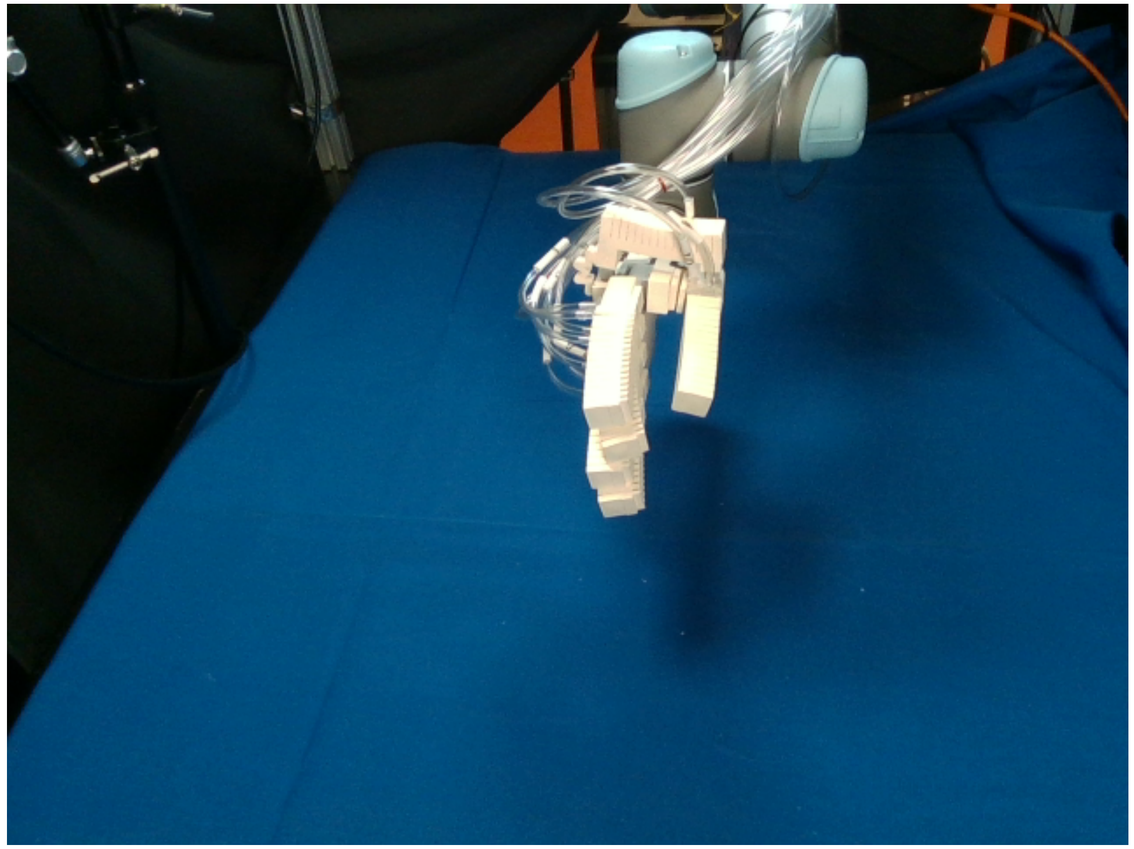
Input: Single Image



“Neural Jacobian Field” $\mathbf{J}(\mathbf{x}, \mathbf{I})$

Control Robots we Couldn’t Before & New Representation for Interactive Video

Input: Single Image



“Neural Jacobian Field” $\mathbf{J}(\mathbf{x}, \mathbf{I})$

Unifying 3D Representation and Control of Diverse Robots with a Single Camera

Sizhe Lester Li*, Annan Zhang, Boyuan Chen, Hanna Matusik, Chao Liu, Daniela Rus, and Vincent Sitzmann*

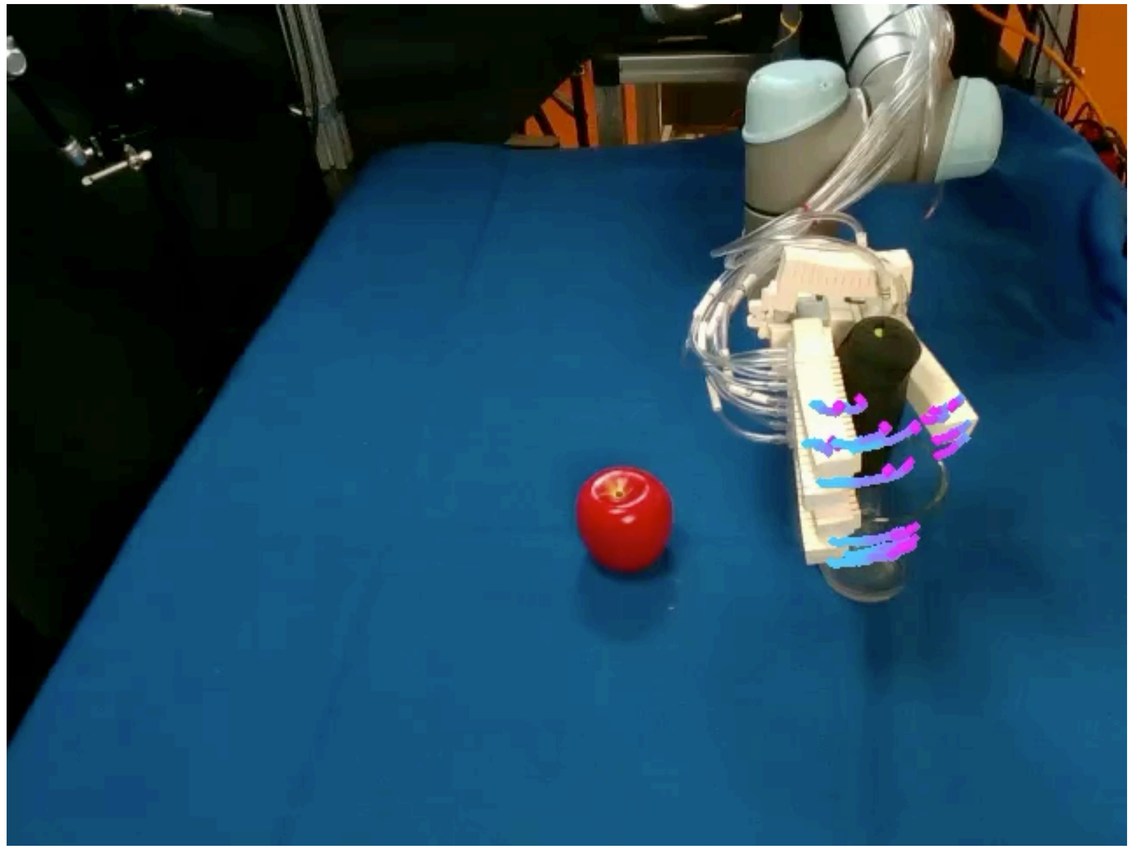
Mirroring the complex structures and diverse functions of natural organisms is a long-standing challenge in robotics [1, 2, 3, 4]. Modern fabrication techniques have dramatically expanded feasible hardware [5, 6, 7, 8], yet deploying these systems requires control software to translate desired motions into actuator commands. While conventional robots can easily be modeled as rigid links connected via joints, it remains an open challenge to model and control bio-inspired robots that are often multi-material or soft, lack sensing capabilities, and may change their material properties with use [9, 10, 11, 12]. Here, we introduce Neural Jacobian Fields, an architecture that autonomously learns to model and control robots from vision alone. Our approach makes no assumptions about the robot's materials, actuation, or sensing, requires only a single camera for control, and learns to control the robot without expert intervention by observing the execution of random commands. We demonstrate our method on a diverse set of robot manipulators, varying in actuation, materials, fabrication, and cost. Our approach achieves accurate closed-loop control and recovers the causal dynamic structure of each robot. By enabling robot control with a generic camera as the only sensor, we anticipate our work will dramatically broaden the design space of robotic systems and serve as a starting point for lowering the barrier to robotic automation. Additional materials can be found on the project website [1].

1 INTRODUCTION

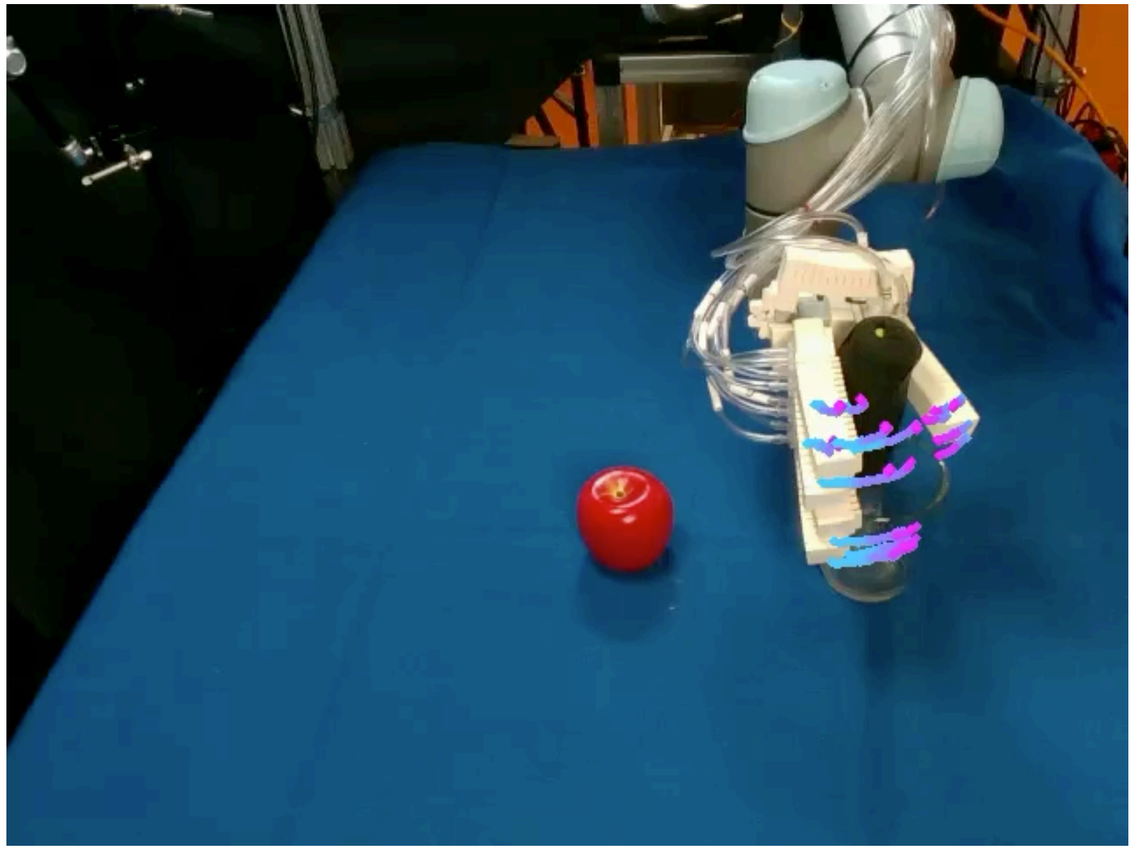
Modern manufacturing techniques promise a new generation of robotic systems inspired by the diverse mechanisms seen in nature. While conventional systems are precision engineered from rigid parts connected at discrete joints, bio-inspired robots generally combine soft, compliant materials and rigid parts, and often forego conventional motor-driven actuation for pneumatic and muscle-like actuators [9]. Recent work has demonstrated that such hybrid soft-rigid systems can already outperform conventional counterparts in certain

Later this module: How can we get the target trajectories? (The *planning* problem)

Motion Plan

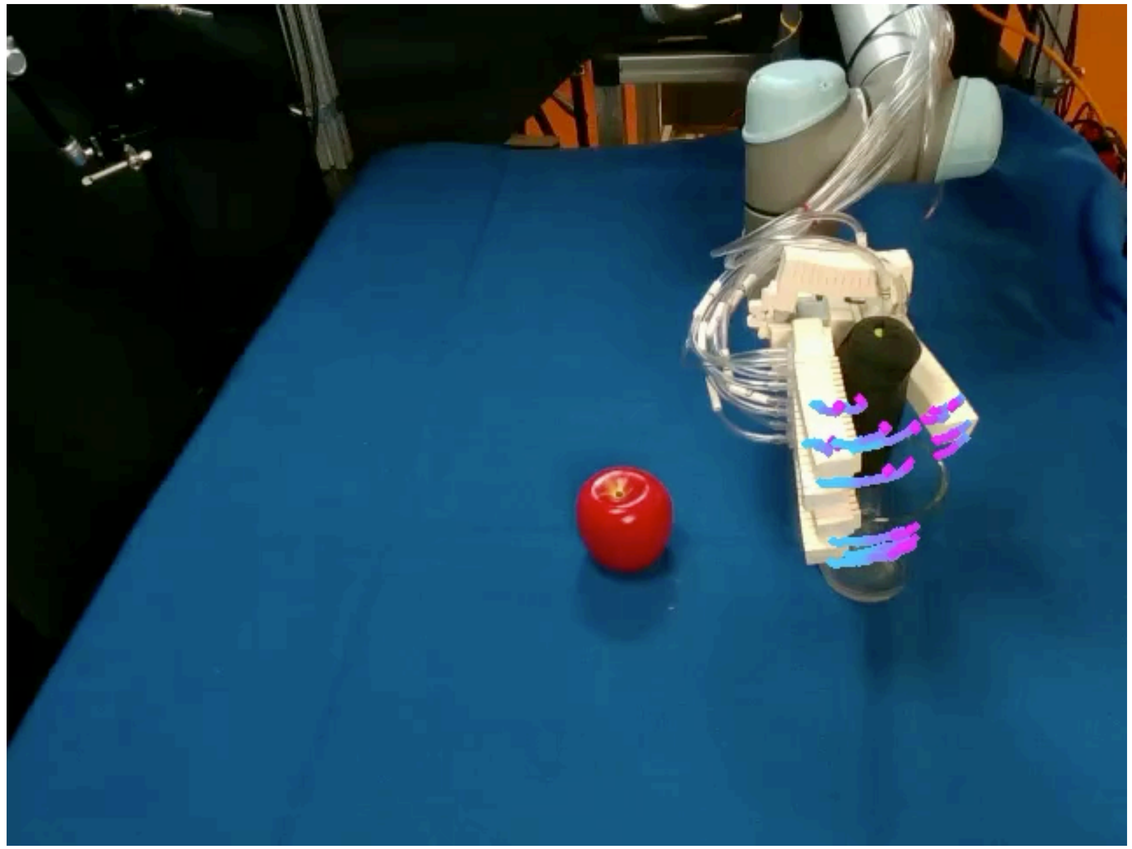


Execution



Later this module: How can we get the target trajectories? (The *planning* problem)

Motion Plan



Execution