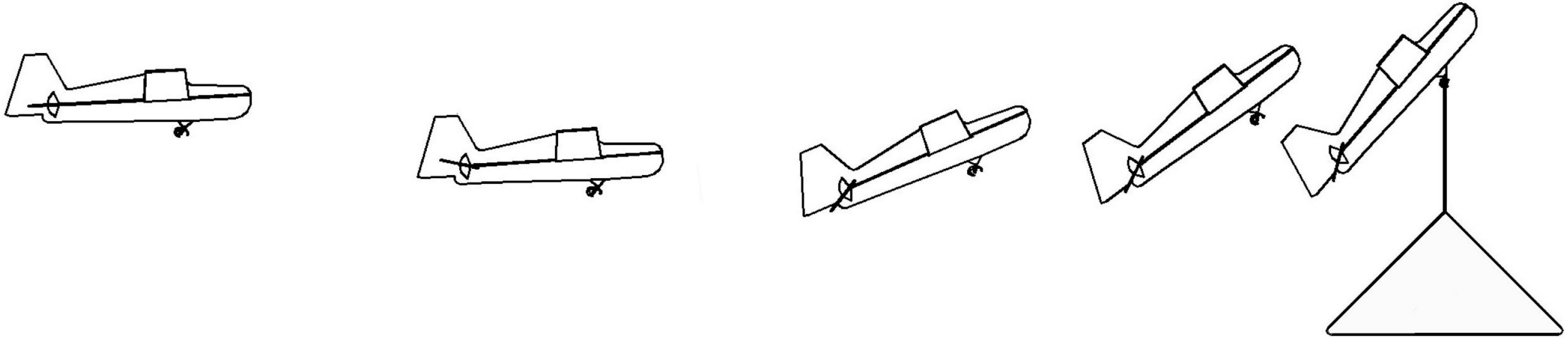
Robust Control with Perception in the Loop: **Toward "Category-Level" Manipulation**

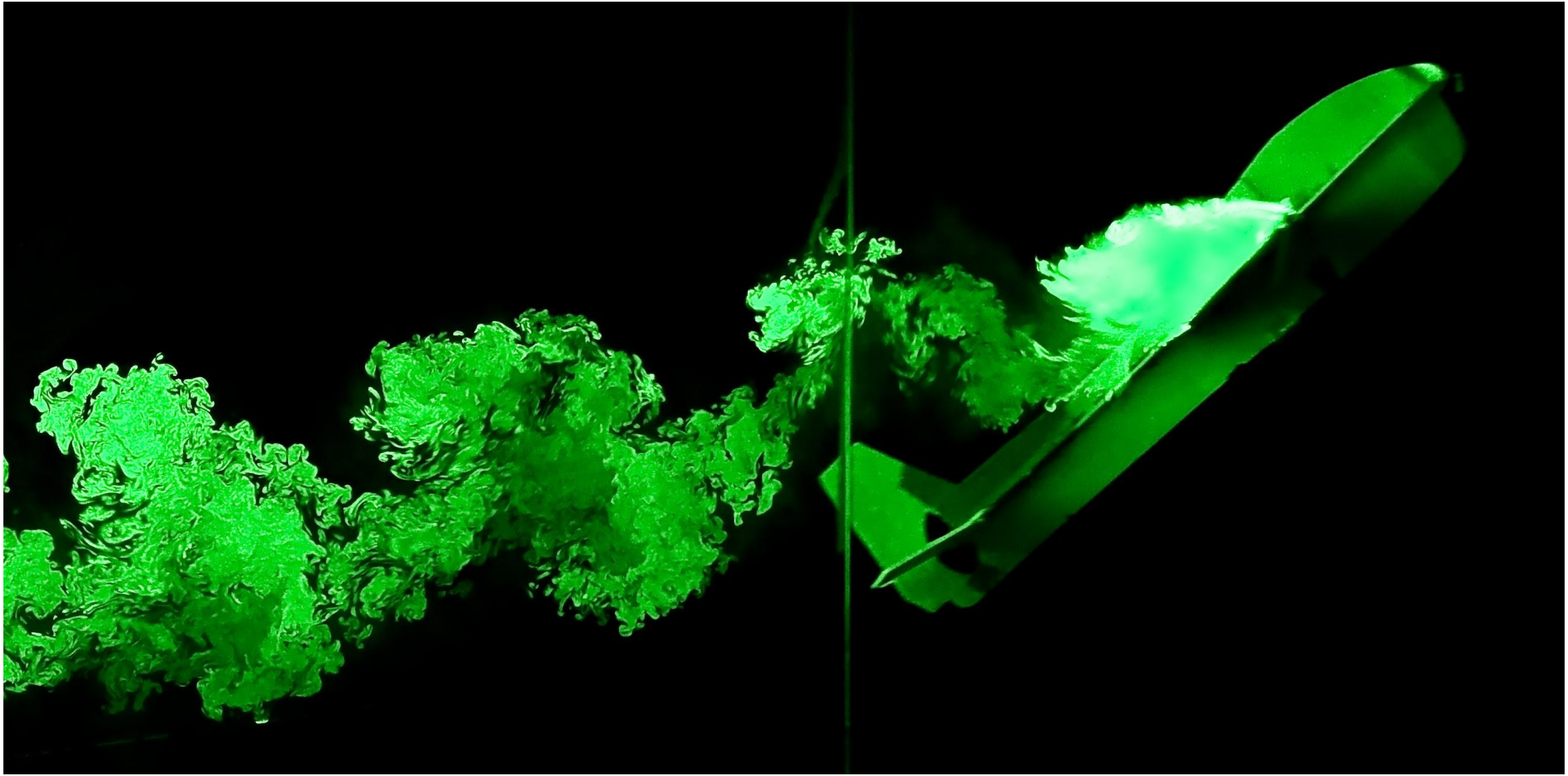
Russ Tedrake
Nov 2019

Massachusetts Institute of Technology

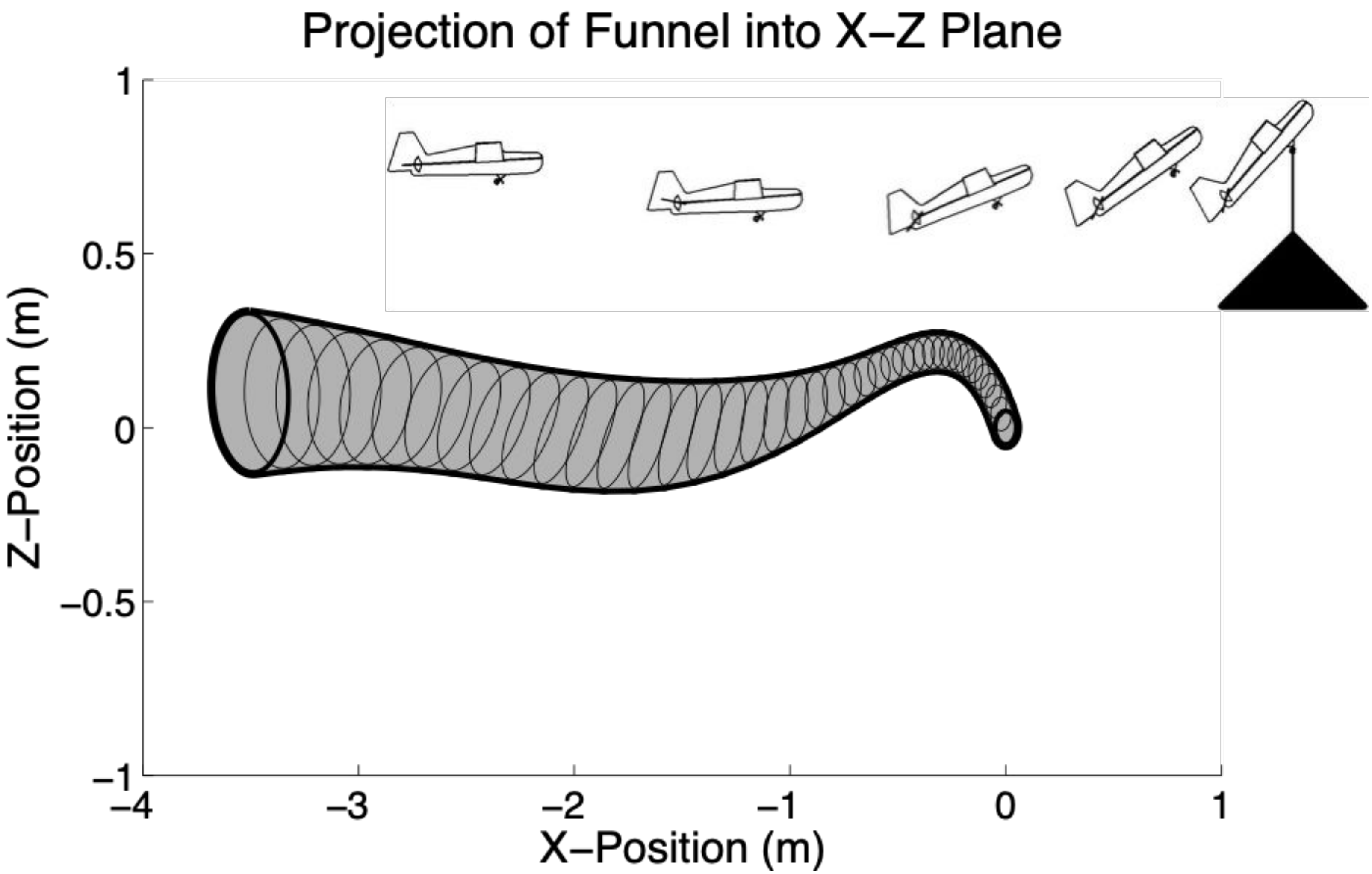
TOYOTA RESEARCH INSTITUTE

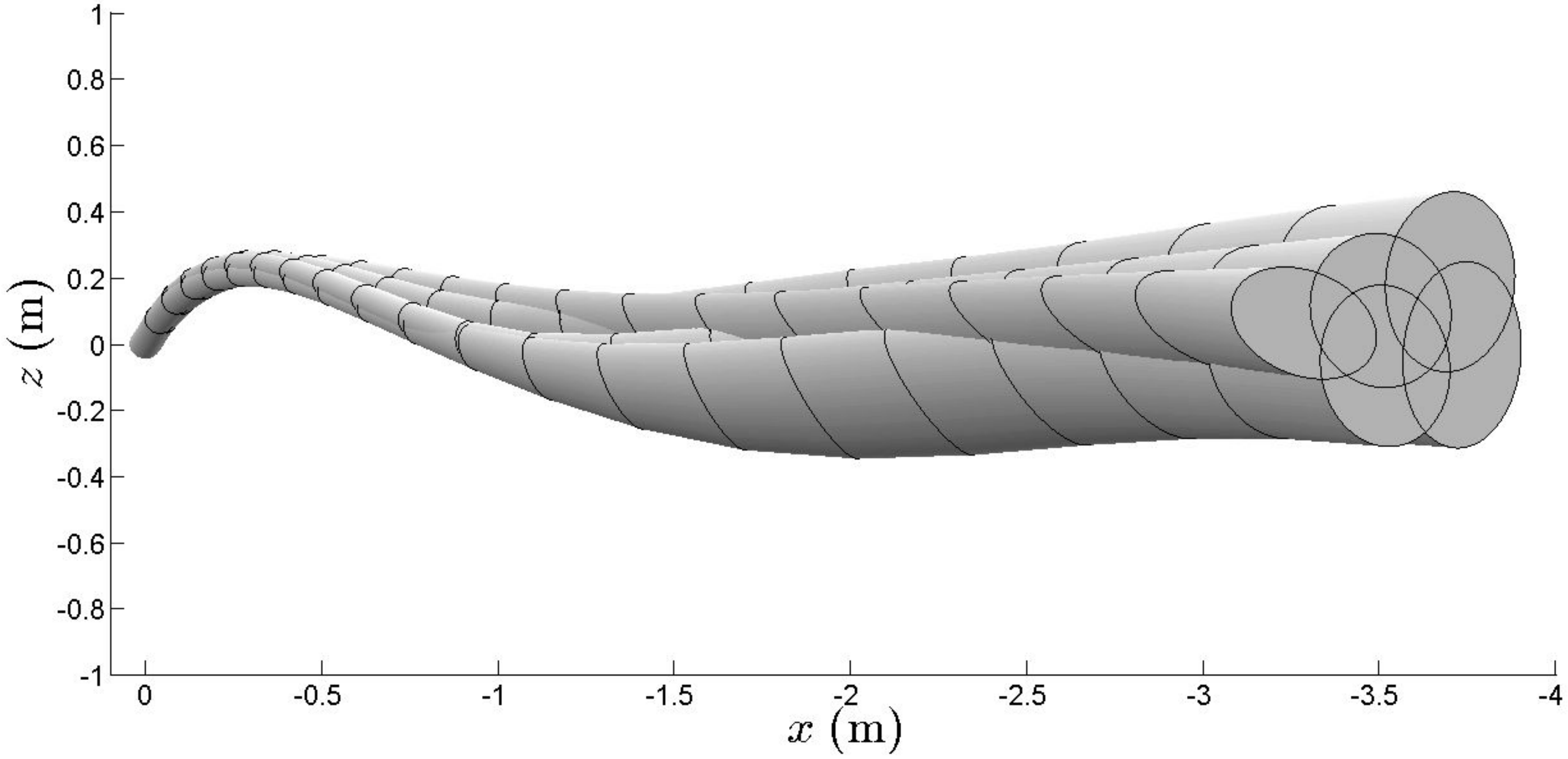
Can we make a control system for a fixed-wing airplane to land on a perch like a bird?



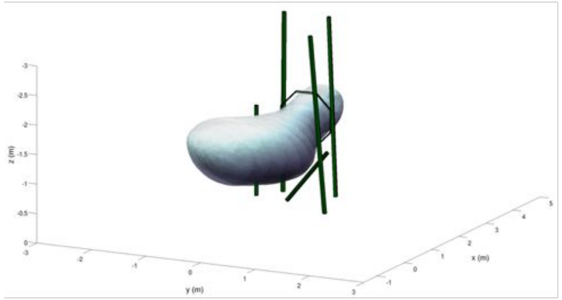


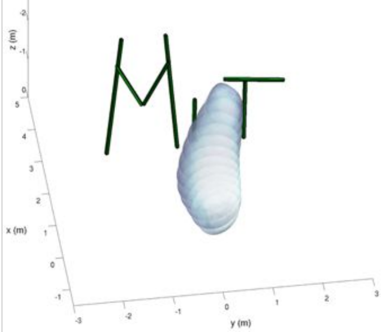
MIT Robot Locomotion Group


Projection of Funnel into X−Z Plane
Z−Position (m)
X−Position (m)
1
0.5
0
−0.5
−1
−4
−3
−2
−1
0
1



1
0.8
0.6
0.4
0.2
0
-0.2
-0.4
-0.6
-0.8
-1
z (m)
0
-0.5
-1
-1.5
-2
-2.5
-3
-3.5
-4
x (m)

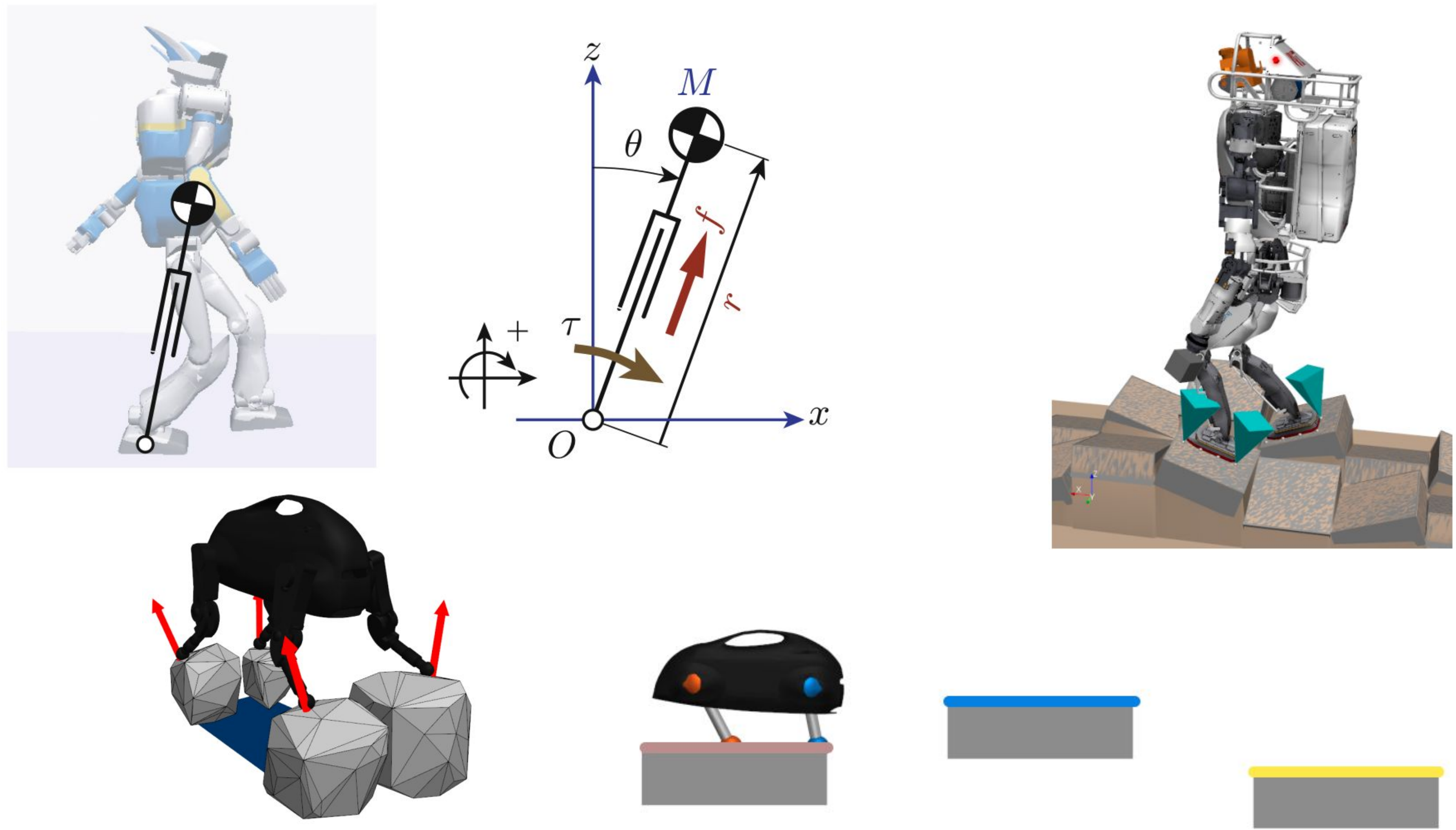




ONR MURI: Provable-safe high-speed flight through forests

w/ Ani Majumdar


DANGER
HIGH VOLTAGE



z
M
θ
f
r
τ
$+$
x
O

My new favorite challenge:

Dexterous Manipulation


10

Many challenges for RL/Control

- Synthesis (which RL algorithm?)
- How do we **specify the task**?
    - Need evaluation function using real-world sensors
    - Over what set of environments?
- How do we **represent the policy**?
    - What is the "state space"?
    - Need "Output feedback"
- Can we meaningfully quantify **distributional robustness**?

Let’s narrow the scope (a bit):

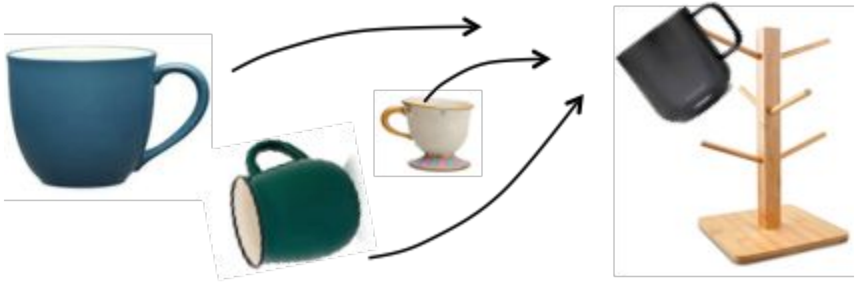
“Category-Level” Manipulation

Problem Statement

Manipulate potentially unknown rigid objects from a **category** (e.g. mugs, shoes) into desired **target configurations**

SE(3) pose is difficult to generalize across category

So how do we even **specify the task**?
What's the cost function?
Images of mugs on the rack?



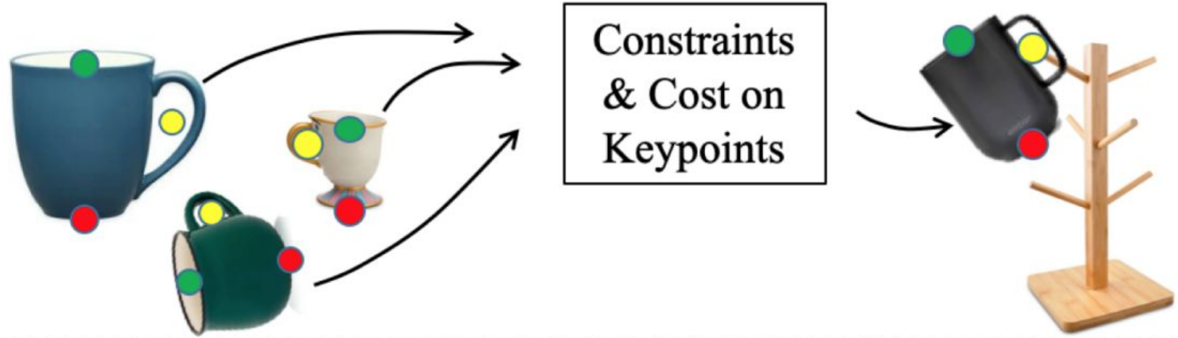
3D Keypoints provide rich, class-general semantics

IEEE TRANSACTIONS ON PATTERN ANALYSIS AND MACHINE INTELLIGENCE, VOL. XXX, NO. XXX, AUGUST YYYY

OpenPose: Realtime Multi-Person 2D Pose Estimation using Part Affinity Fields

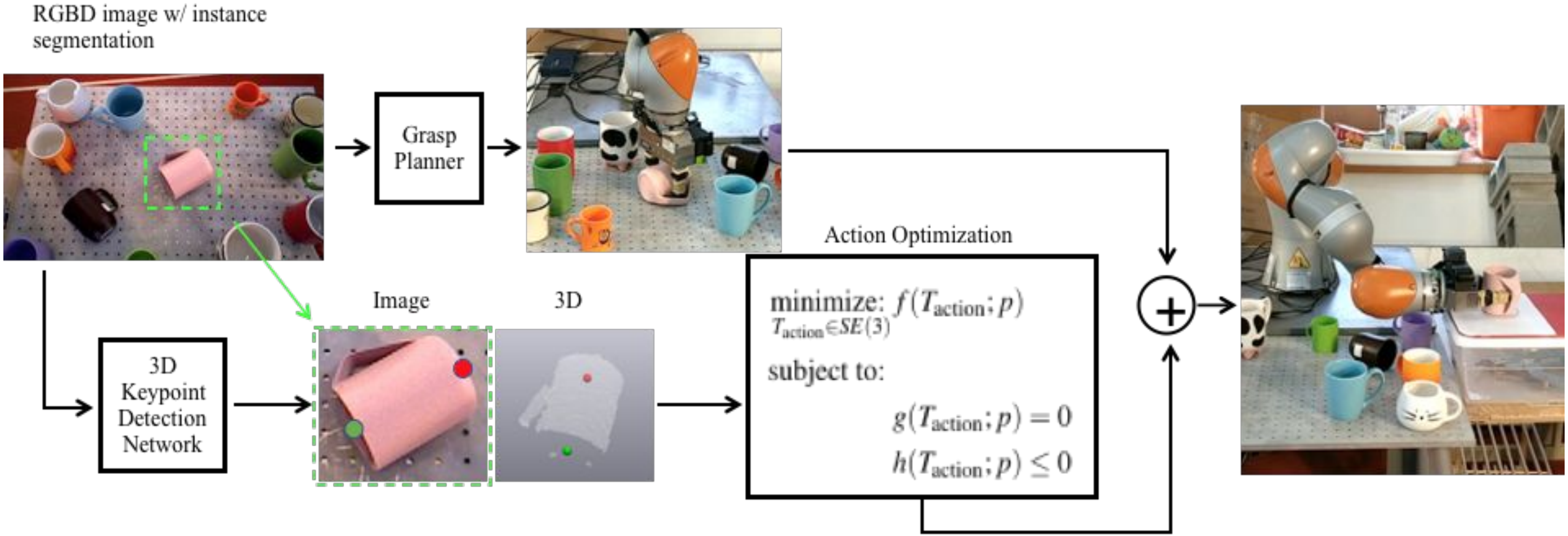
Zhe Cao, *Student Member, IEEE,* Gines Hidalgo, *Student Member, IEEE,* Tomas Simon, Shih-En Wei, and Yaser Sheikh

… and robust performance in practice



kPAM pipeline

No template model or pose appears in this pipeline.

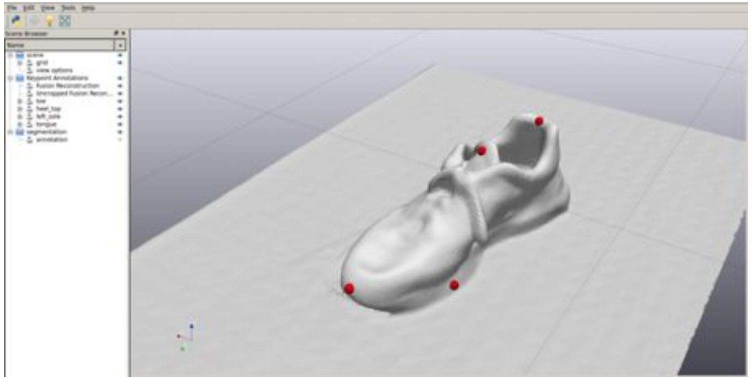


Keypoint Training Data

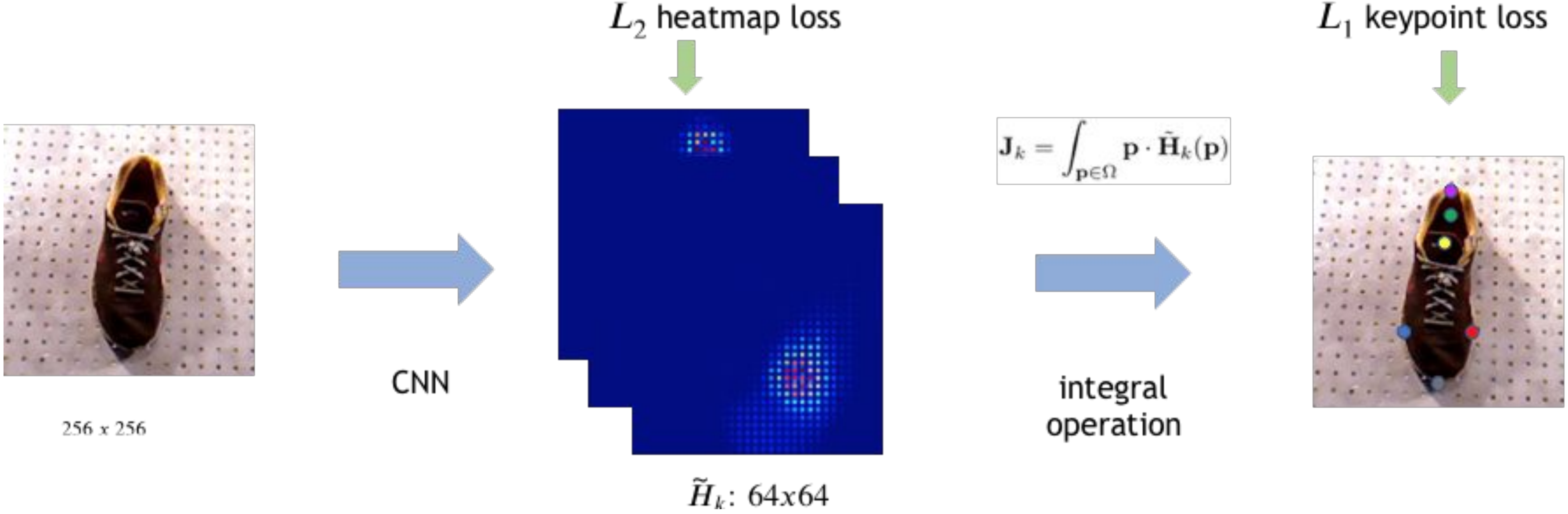
Dense Reconstruction helps overcome partial observability

Keypoint network

Architecture based on:
Sun, Xiao, et al. "Integral human pose regression." Proceedings of the European Conference on Computer Vision (ECCV). 2018.



Screenshot from our custom annotation tool



kPAM results

Object Type	# train objects	# scenes	# images
Shoe	10	43	39,403
Mug	21	74	70,094

(c) Training dataset statistics

# test objects	# Trials	Placed on shelf	Heel Error (cm)	Toe Error (cm)
20	100	98%	1.09 ± (1.29)	4.34 ± (3.05)

(d) Shoes on Rack

Initial Orientation	# test objects	# Trials	Placed upright on shelf	< 3cm error	< 5cm error
Upright	40	80	100%	97.5%	100%
Horizontal	19	38	97.3%	89.4%	94.7%

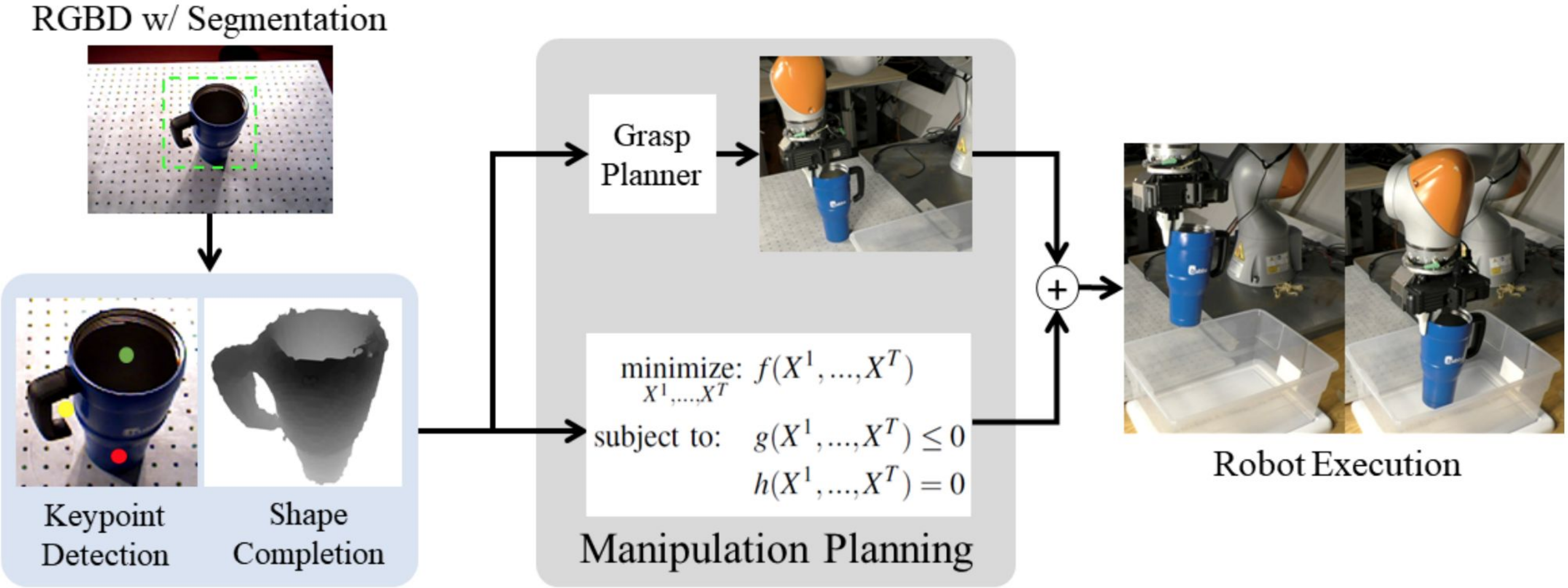
(e) Mugs on Shelf

Mug Size	# test objects	# Trials	Success Rate
Regular	25	100	100%
Small	5	20	50%

(f) Mugs on Rack

kPAM-SC: now with shape completion

Motion planning step can now include non-collision constraints on the object



Q: How do we **specify a diversity of tasks**?

Proposal: For many geometric tasks, simple costs and constraints on **semantically-labeled keypoints**.

IEEE TRANSACTIONS ON PATTERN ANALYSIS AND MACHINE INTELLIGENCE, VOL. XXX, NO. XXX, AUGUST YYYY

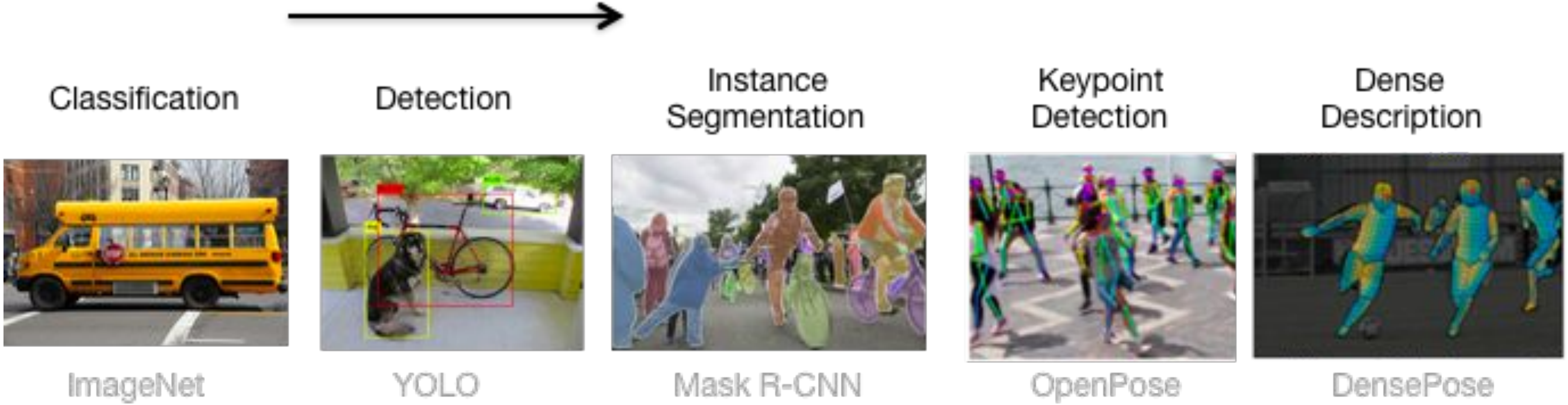
OpenPose: Realtime Multi-Person 2D Pose Estimation using Part Affinity Fields

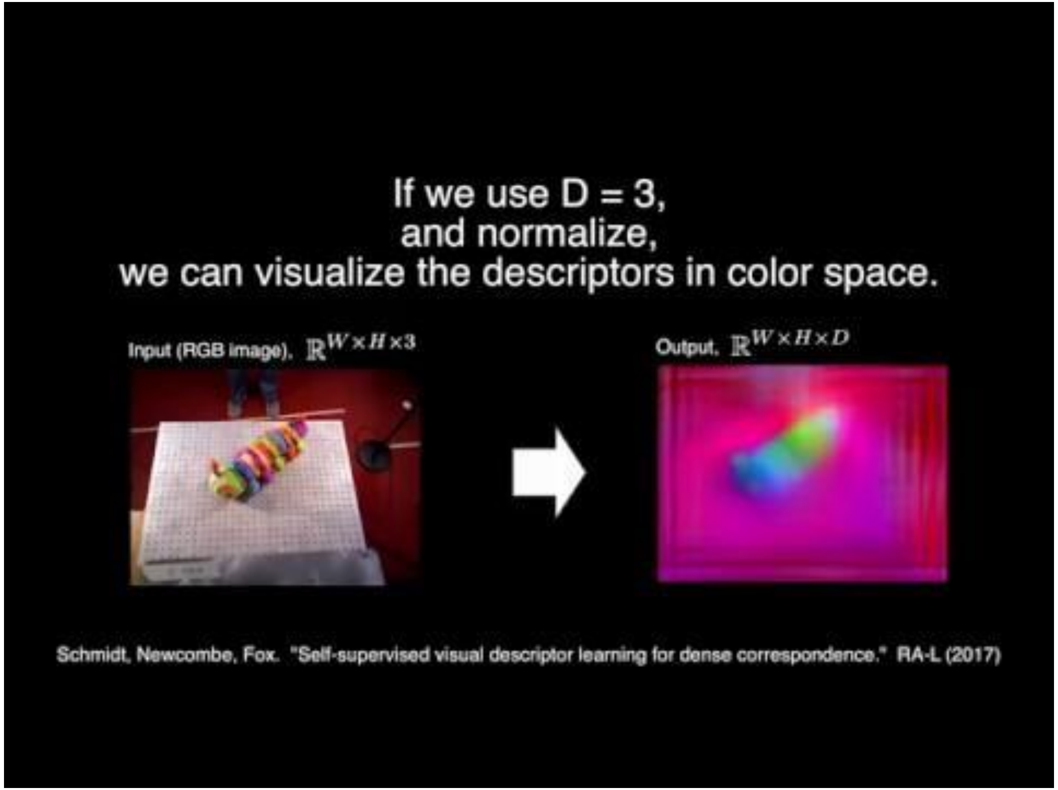
Zhe Cao, *Student Member, IEEE,* Gines Hidalgo, *Student Member, IEEE,* Tomas Simon, Shih-En Wei, and Yaser Sheikh

How do we **represent the policy**?

Dense Object Nets in Visuomotor Policy Learning

Leveraging advances in deep perception



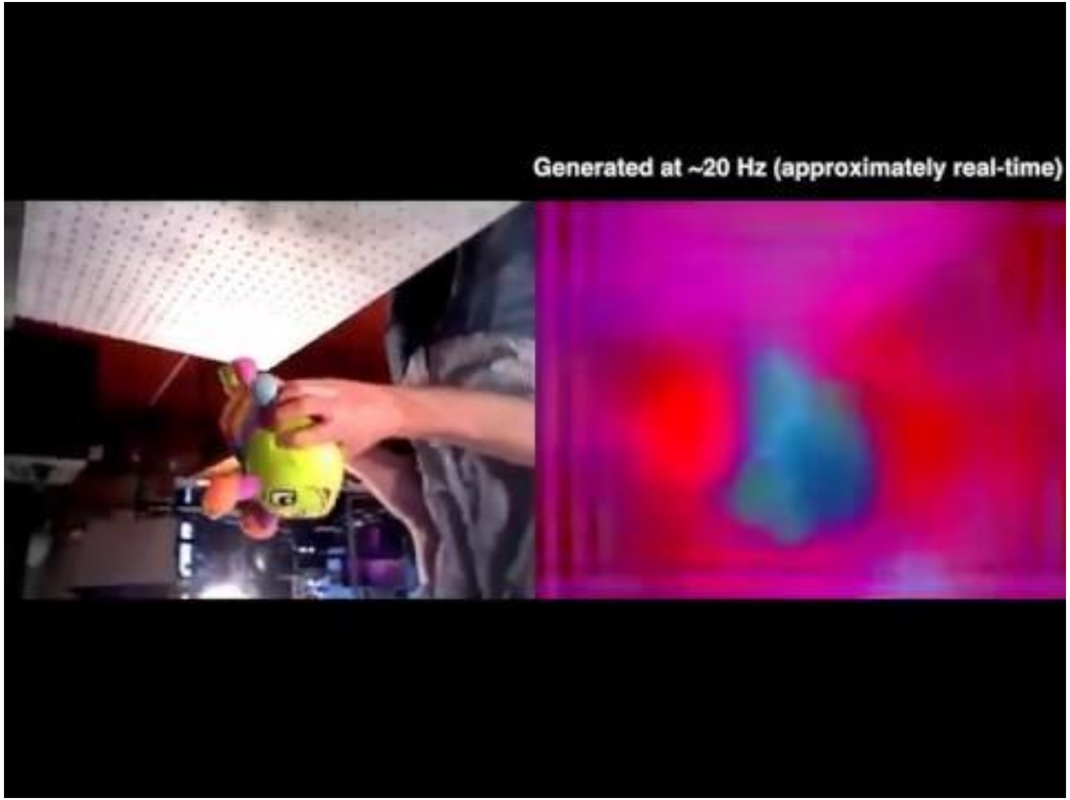


Dense Object Nets: Learning Dense Visual Object Descriptors By and For Robotic Manipulation

Peter R. Florence*, Lucas Manuelli*, Russ Tedrake

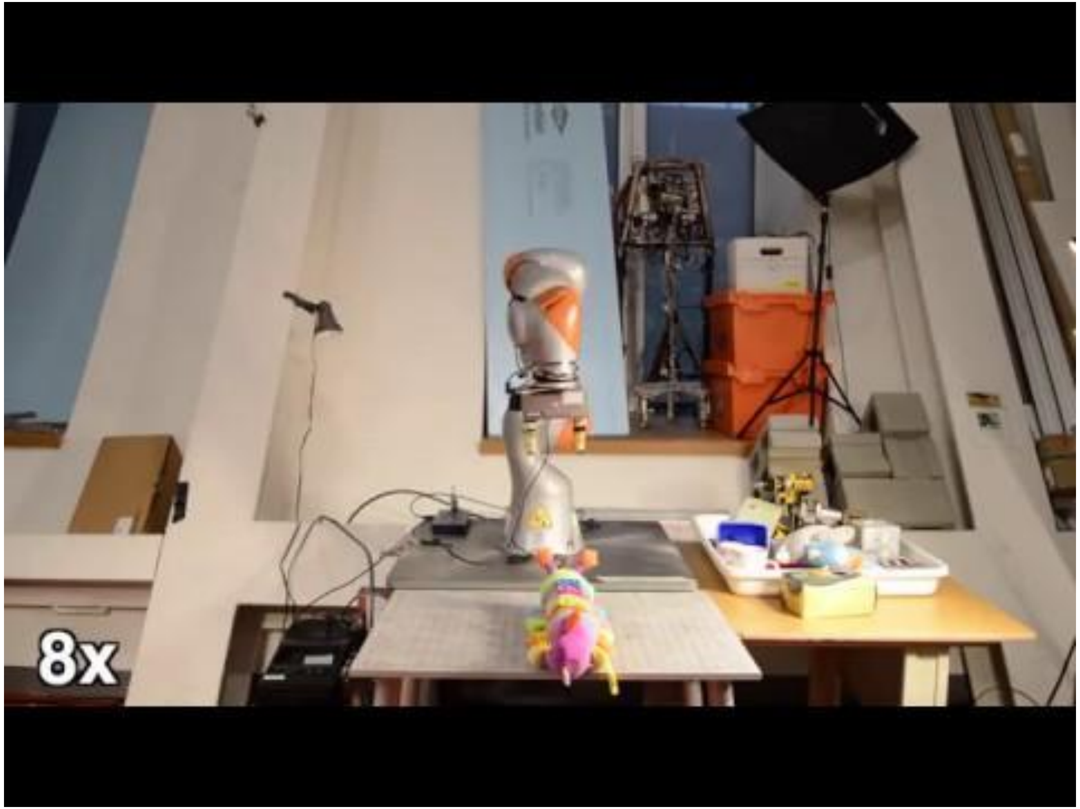
CoRL 2018

Dense Object Nets



Dense Object Nets

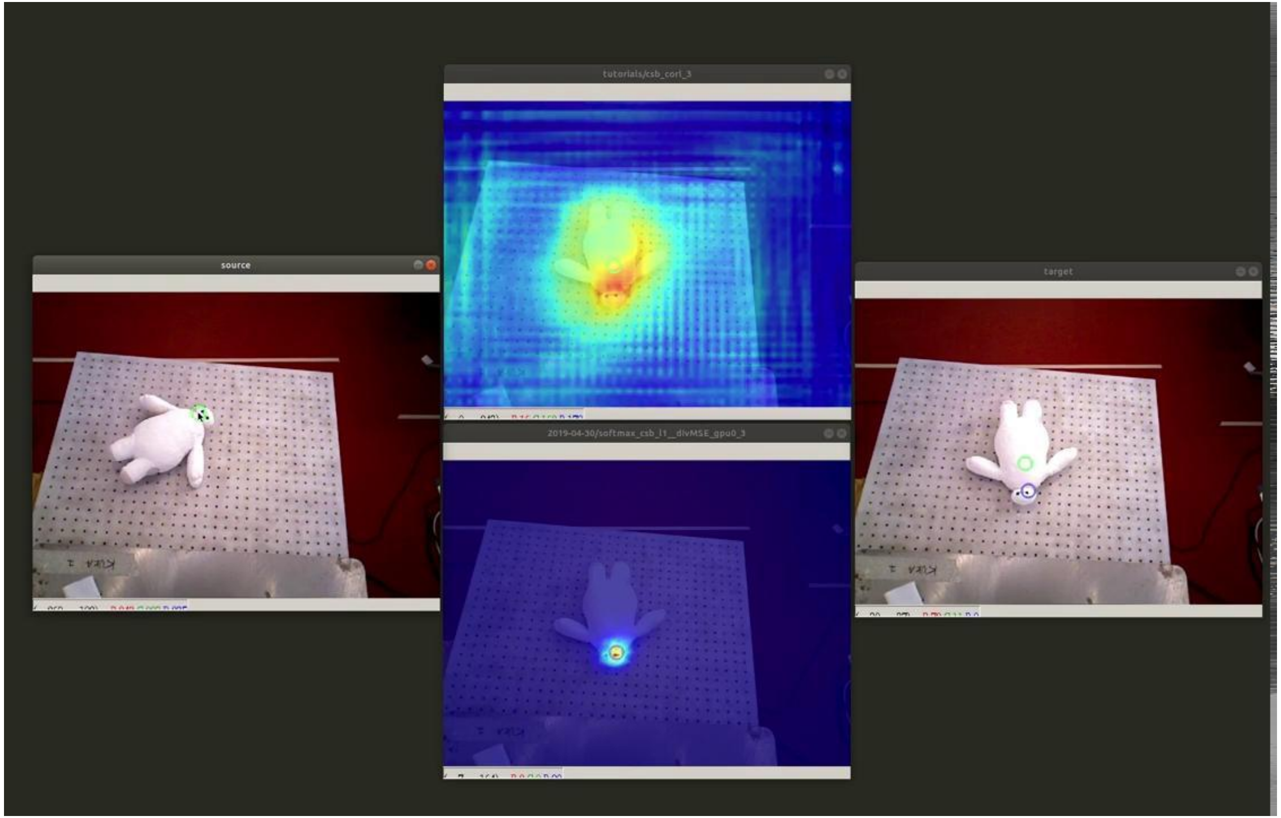
In initial paper, our tasks were very simple. Just “grab here”.



Do these representations facilitate more complicated control?

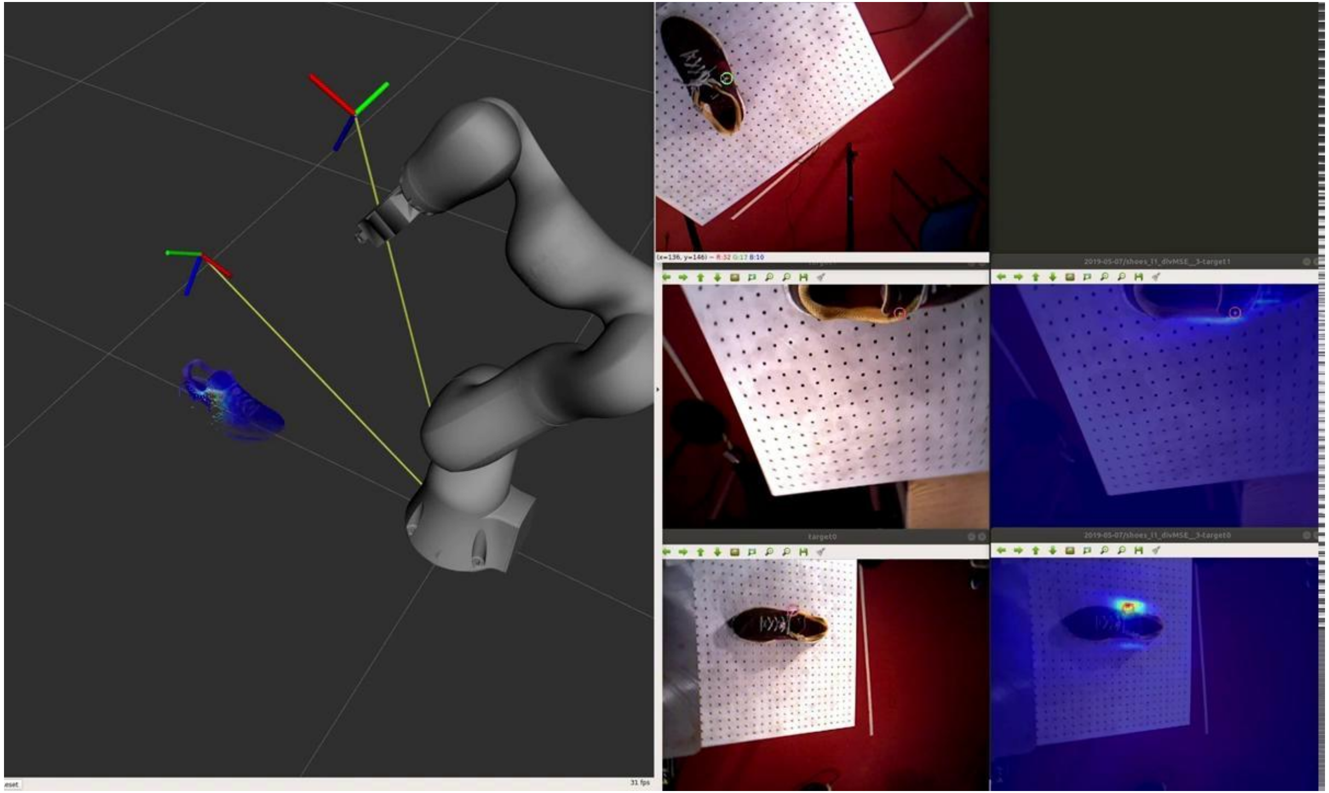
(Dramatically) improved dense descriptor training

in 2D

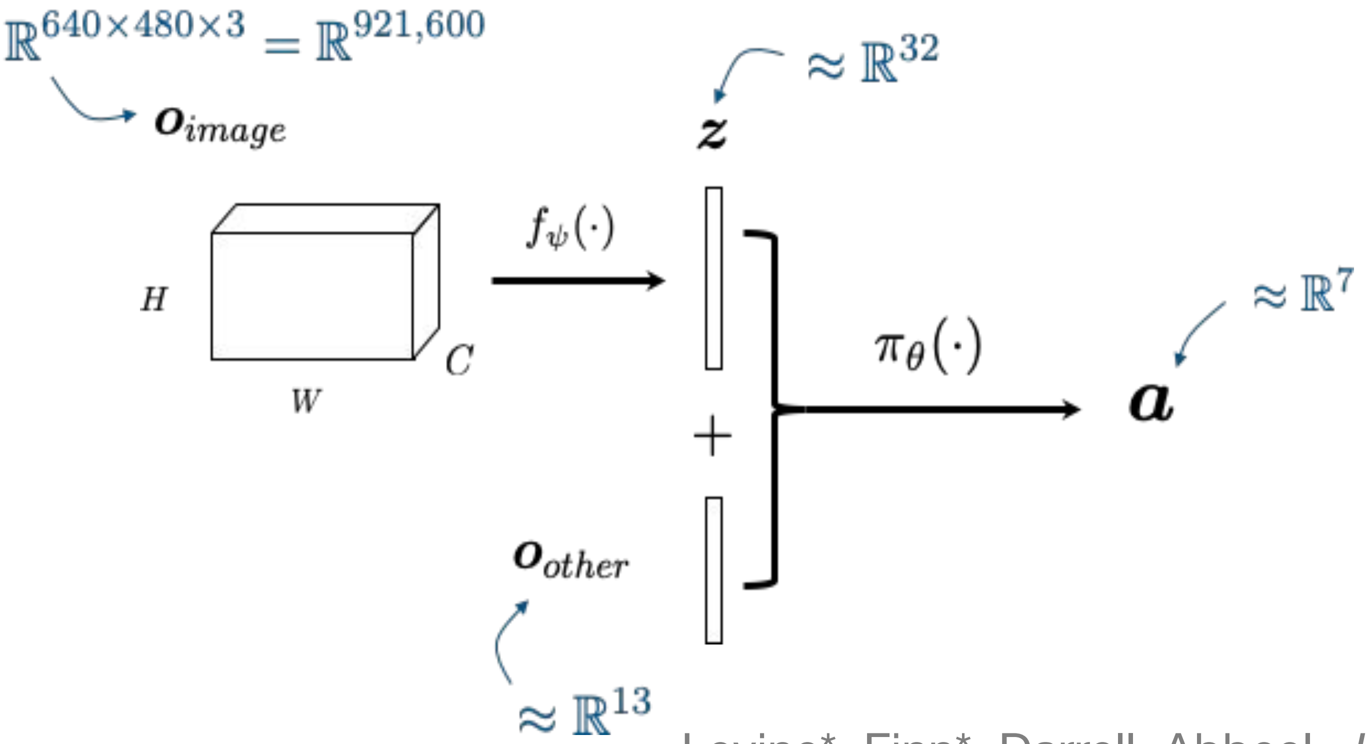


(Dramatically) improved dense descriptor training

And now 3D



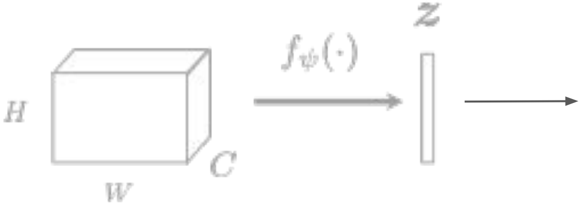
Visuomotor policies



Levine*, Finn*, Darrell, Abbeel, *JMLR* 2016

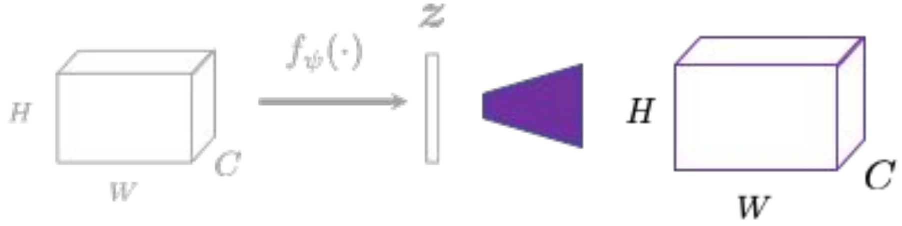
Primary existing methods for training visual portion

1. Pose-based auxiliary loss

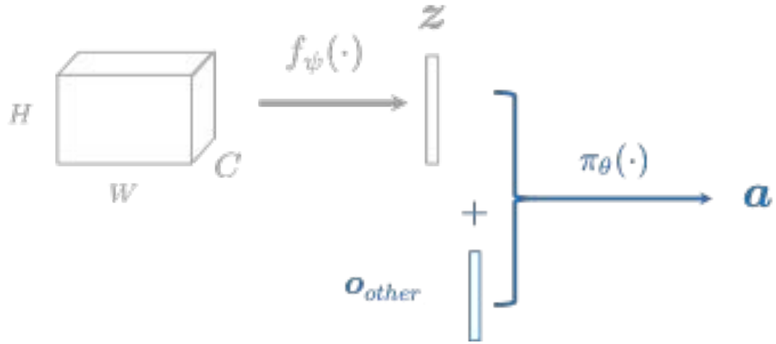


Estimate object/hand pose (but hard for class-general or deformable)

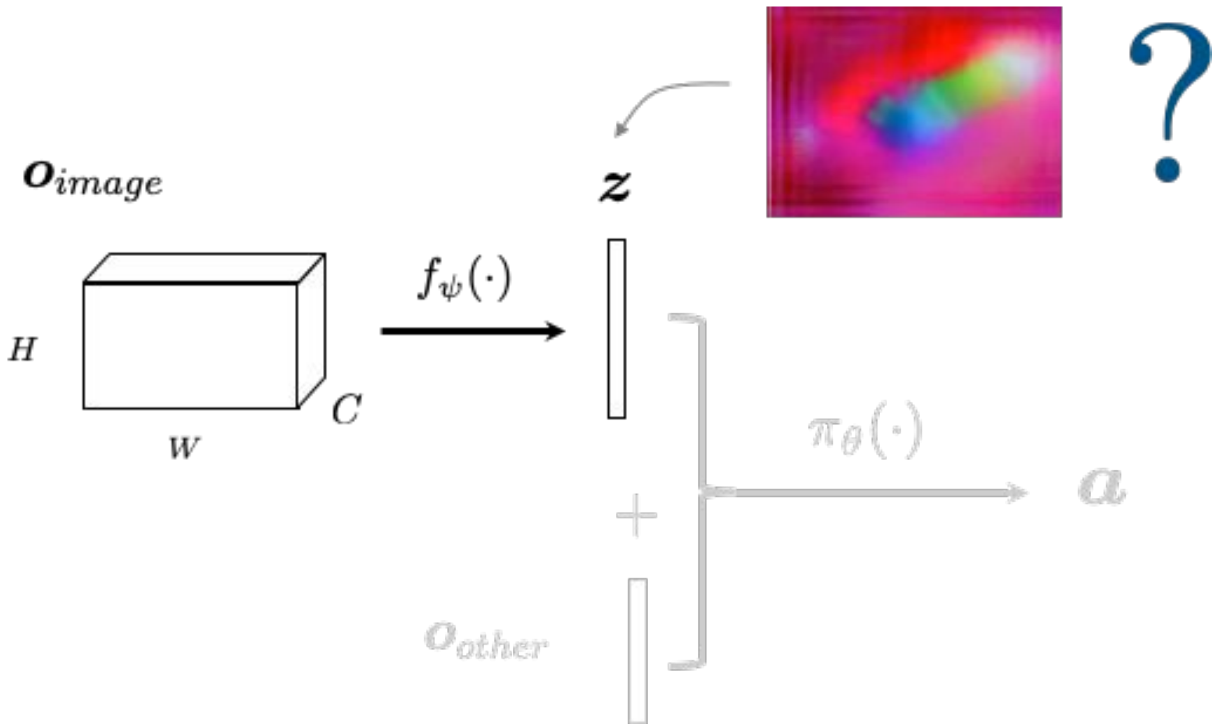
2. Auto-encoding



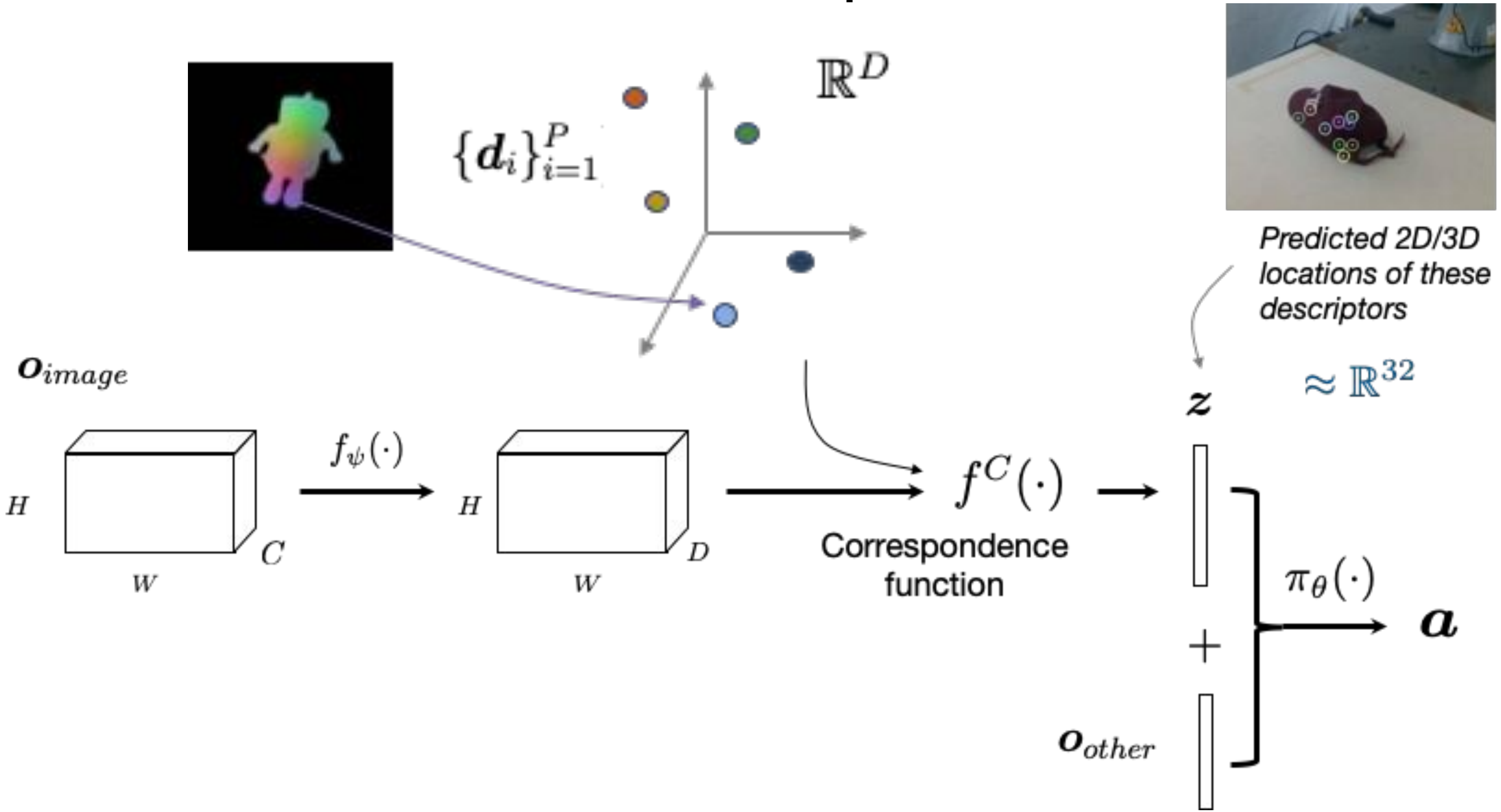
3. End-to-end



Idea: What if we use dense correspondences?



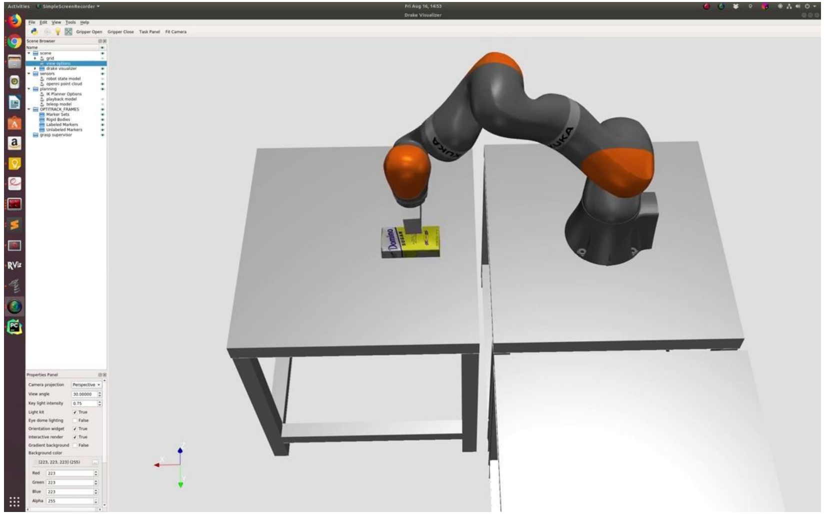
Idea: Use a small set of descriptors



Evaluations are based on imitation learning

w/ standard “behavior cloning” objective

from hand-coded policies in simulation

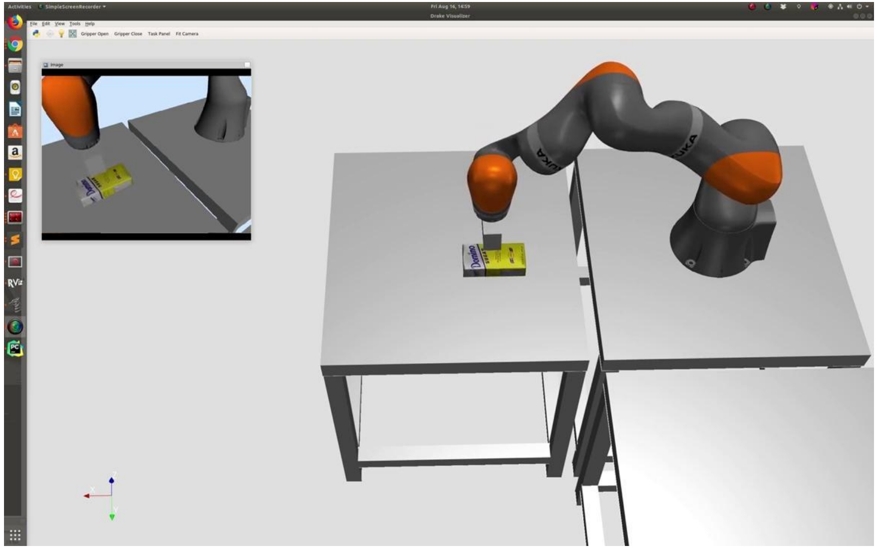


and tele-op on real robot

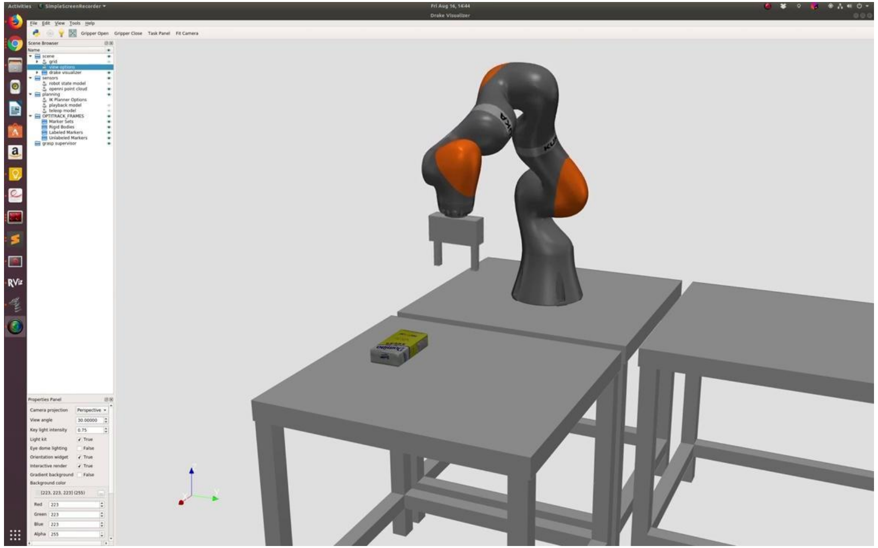
\+ some novel(?) heuristics for data augmentation

Simulation experiments

Policy is a small LSTM network (~ 100 LSTMs)



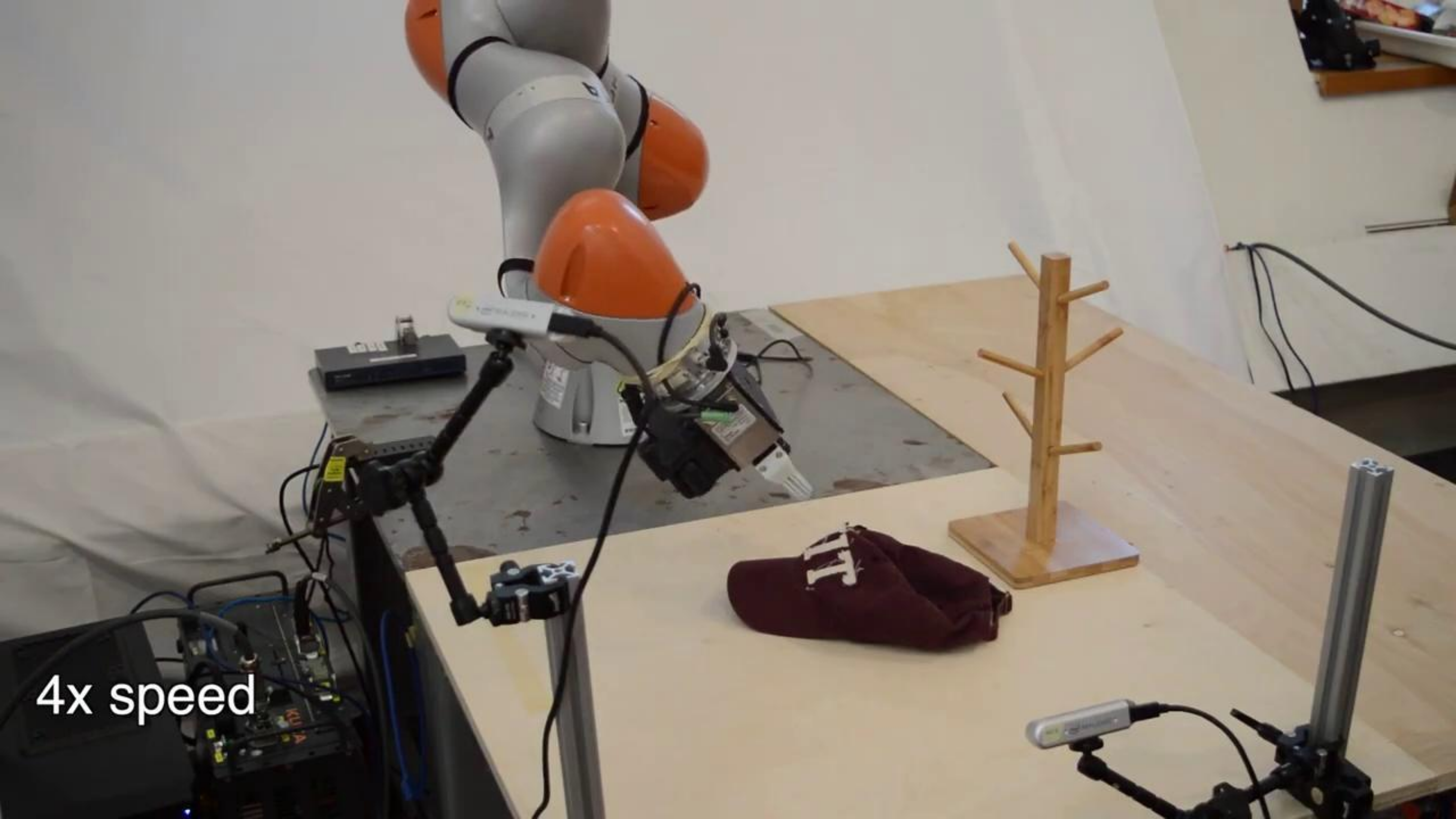
“push box”



“flip box”


4x speed


4x speed

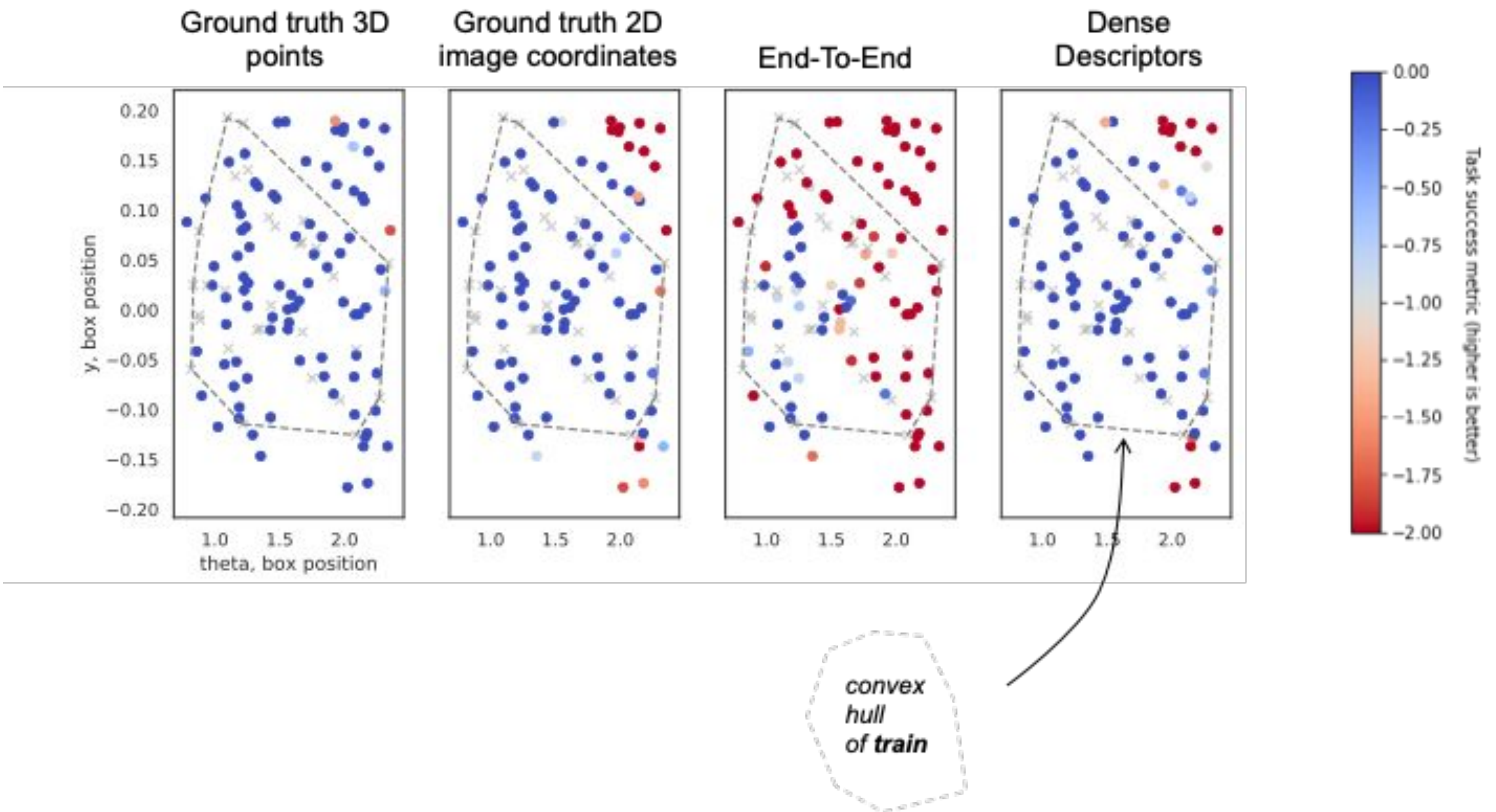

4x speed

Method / Task	Reach T only	Reach T + R	Push box	Push plate
Ground truth 3D points	100.0	100.0	100.0	90.5
Ground truth 2D image coord.	94.1	95.6	100.0	92.0
RGB policy input				
Autoencoder, frozen	8.1	61.1	31.0	53.0
Autoencoder w/ mask, frozen	16.3	10.0	73.0	67.0
Autoencoder, then End-to-End	40.7	38.9	–	16.0
End-to-End	43.0	32.2	**100.0**	5.5
End-to-End (34-layer ResNet)	–	3.3	–	–
DD 2D image coord. (ours)	**94.1**	**97.8**	**100.0**	**87.0**
RGBD policy input				
DD 3D coord. (ours)	**100.0**	**100.0**	–	**98.0**

TABLE I: Summary of simulation results (success rate, as %). DD = Dense Descriptor. See Appendix for task success criteria and additional details.

Task	Success criterion	Trained with manual disturbances	*Without disturbances* # attempts	# success	%	*With disturbances* # attempts	# success	%	*Demonstration data* # total	time (min.)
Push sugar box	box is < 3 cm from finish line	yes	6	6	100.0	70	68	97.1	51	13.9
Flip shoe, single instance	shoe is upright	no	43	42	97.7	40	35	87.5	50	6.5
Flip shoe, class-general										
previously seen shoes (14)	shoe is upright	no	43	38	88.4	–	–	–	146	17.5
novel shoes (12)	shoe is upright	no	22	17	77.3	–	–	–	146	17.5
Pick-then-hang hat on rack	hat is on the rack	yes	50	42	84.0	41	28	68.3	52	11.5
Push-then-grab plate	plate is off the table	yes	22	21	95.5	27	22	81.5	94	27.4
Total			186			178				

TABLE III: Summary of task attempts and success rates for hardware validation experiments. Autonomous re-tries are counted as successes.


Ground truth 3D points
Ground truth 2D image coordinates
End-To-End
Dense Descriptors
y, box position
theta, box position
Task success metric (higher is better)
convex hull of **train**

Can we meaningfully quantify **distributional robustness**?

How should we represent distributions at the category level?

Requirements authoring

In controls (polytopic/ellipsoidal, etc)

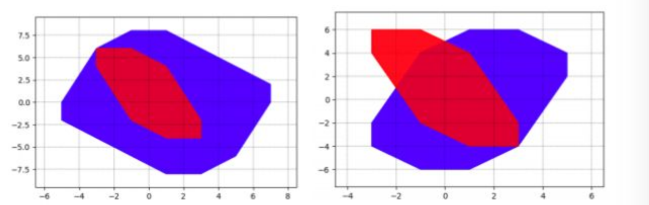
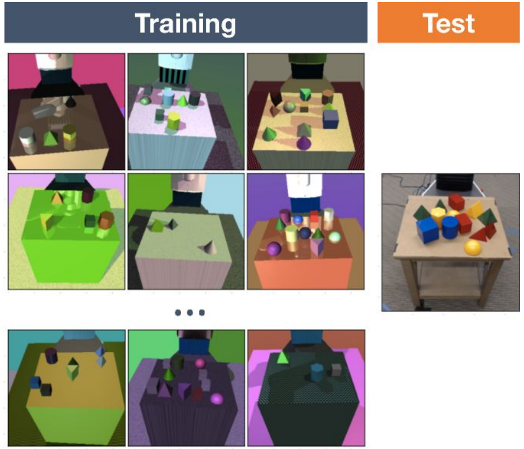


Fig. 1. Example 1: Zonotope Containment Problem: [left] $\mathbb{Z}_l \subseteq \mathbb{Z}_r$, [Right] $\mathbb{Z}_l \nsubseteq \mathbb{Z}_r^*$, where the last column of G_r is dropped.

Domain randomization in reinforcement learning



27 Feb 2018 | 16:48 GMT

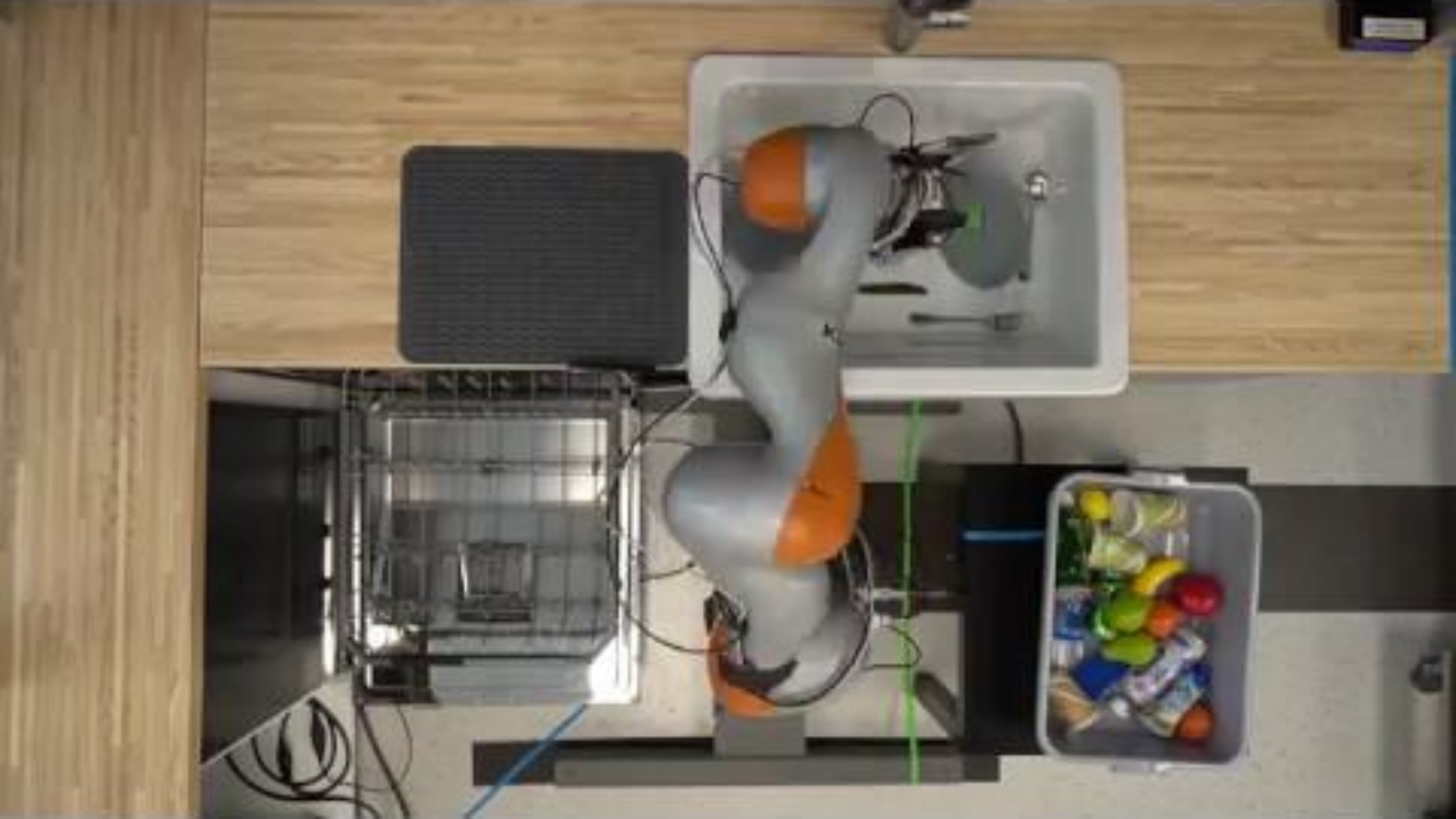
Creating Driving Tests for Self-Driving Cars

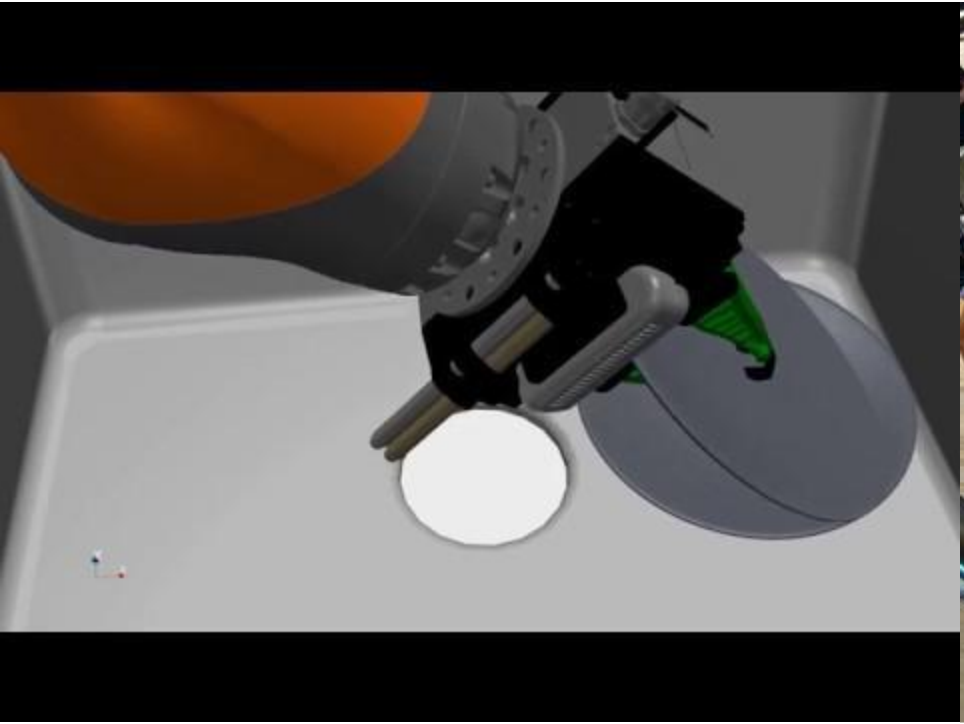
Volvo-backed Zenuity wants to prove that autonomous vehicles can drive more safely than humans

By Erik Coelingh and Jonas Nilsson

Developing autonomous systems in the real world.

TOYOTA
RESEARCH INSTITUTE






More advanced falsification via nonlinear black-box optimization and rare event simulation.

Scalable End-to-End Autonomous Vehicle Testing via Rare-event Simulation

NIPS 2018

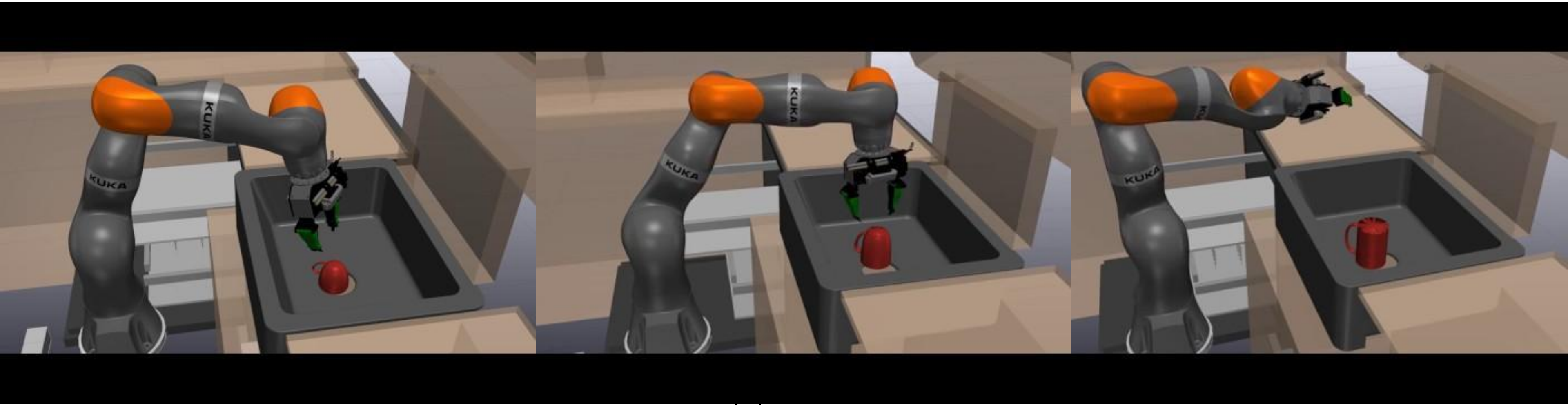
Search Algorithm	$\gamma = 0.14$	$\gamma = 0.16$	$\gamma = 0.18$	$\gamma = 0.40$	$\gamma = 0.42$
Naive	(2.0±2.0)e-5	(22.0± 6.6)e-5	(82.0±12.8)e-5	(334.4±8.0)e-4	(389.7±8.6)e-4
Cross-entropy	(3.2±2.6)e-6	(25.8 ± 4.5)e-5	(84.6± 9.3)e-5	(334.5 ± 8.0)e-4	(386.4 ± 8.6)e-4

Table 1: Estimate of rare-event probability p_γ (non-vision ego policy), with standard deviations

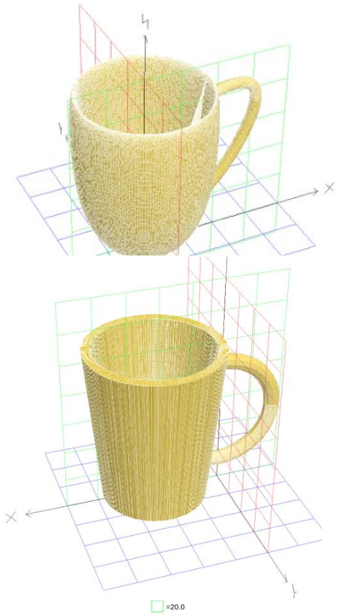
Search Algorithm	$\gamma = 0.26$	$\gamma = 0.28$	$\gamma = 0.30$	$\gamma = 0.50$	$\gamma = 0.52$
Naive	(8.0±4.0)e-3	(8.0±4.0)e-3	(12.0±4.9)e-3	(13.8±1.5)e-2	(15.6±1.6)e-2
Cross-entropy	(2.7±2.1)e-3	(5.4±2.7)e-3	(6.4±2.7)e-3	(7.6±1.0)e-2	(8.1±1.0)e-2

Table 2: Estimate of rare-event probability p_γ (vision-based ego policy), with standard deviations

Procedural dishes



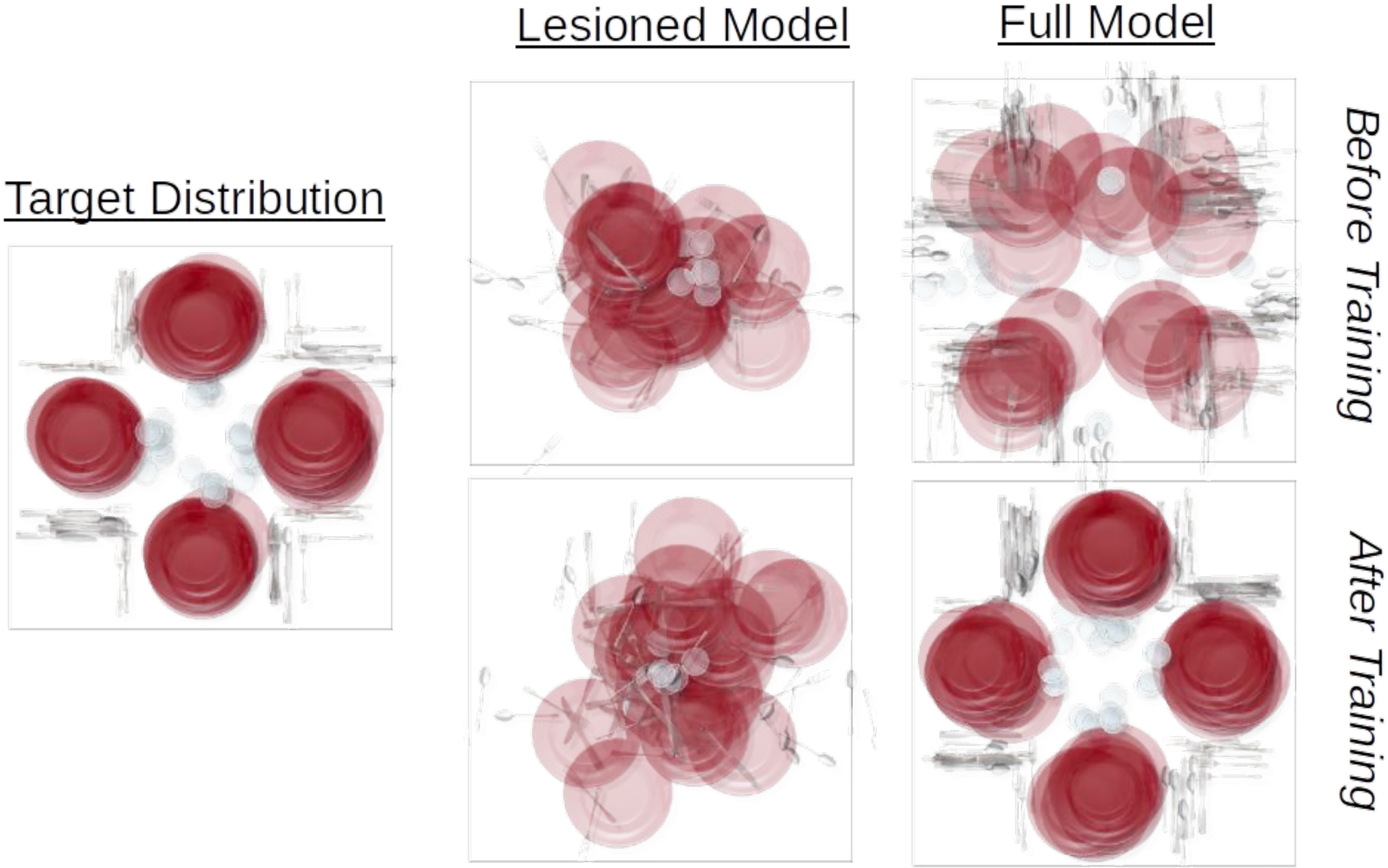
Procedural dishes



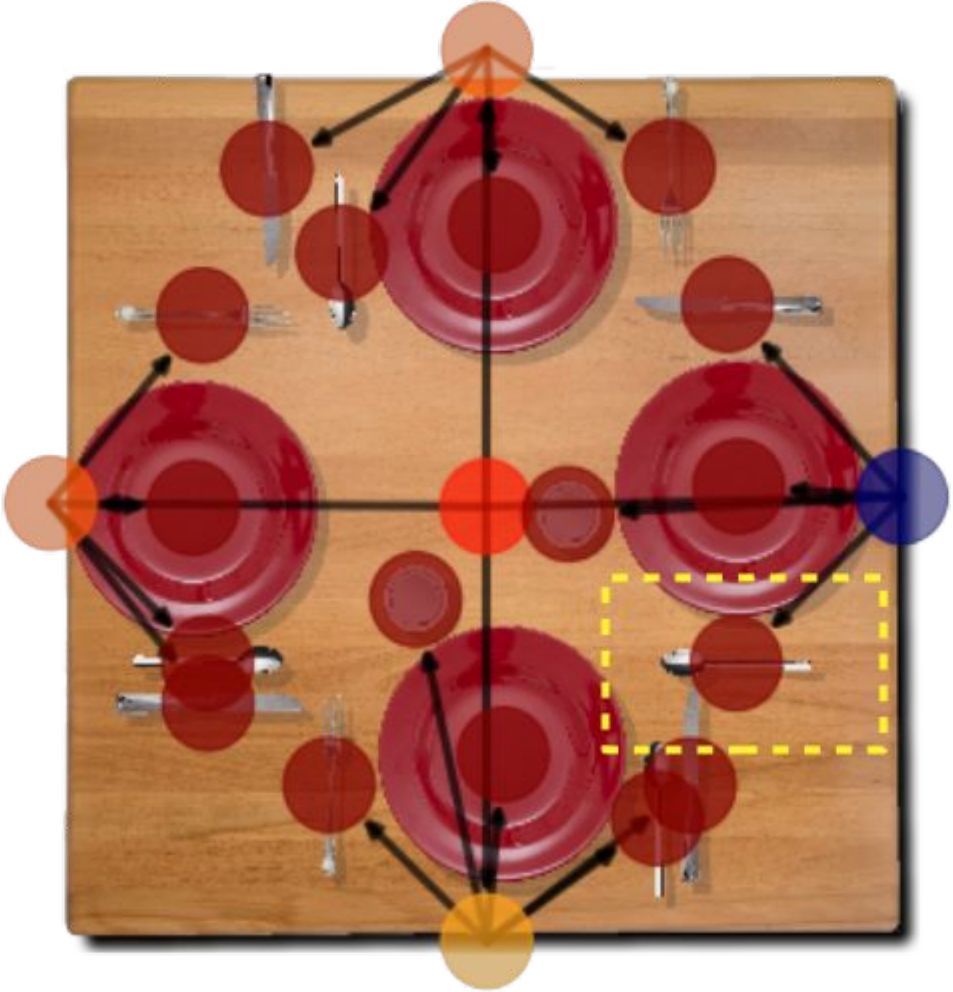
TOYOTA
RESEARCH INSTITUTE

Generative Modeling of Environments with Scene Grammars and Variational Inference

Gregory Izatt and Russ Tedrake
$\{gizatt, russt\}$@csail.mit.edu


Lesioned Model
Full Model
Target Distribution
Before Training
After Training

Outlier detection



To achieve robustness, do I need to simulate the diversity of the world?

Can we simulate everything in the kitchen?

Napkins? Ketchup? Soba noodles?

How accurate do our models have to be?

How do I provide test coverage for every possible kitchen?

Hypothesis: Only need a sufficiently rich sandbox to deploy

+ continual improvement (fleet learning)

Summary

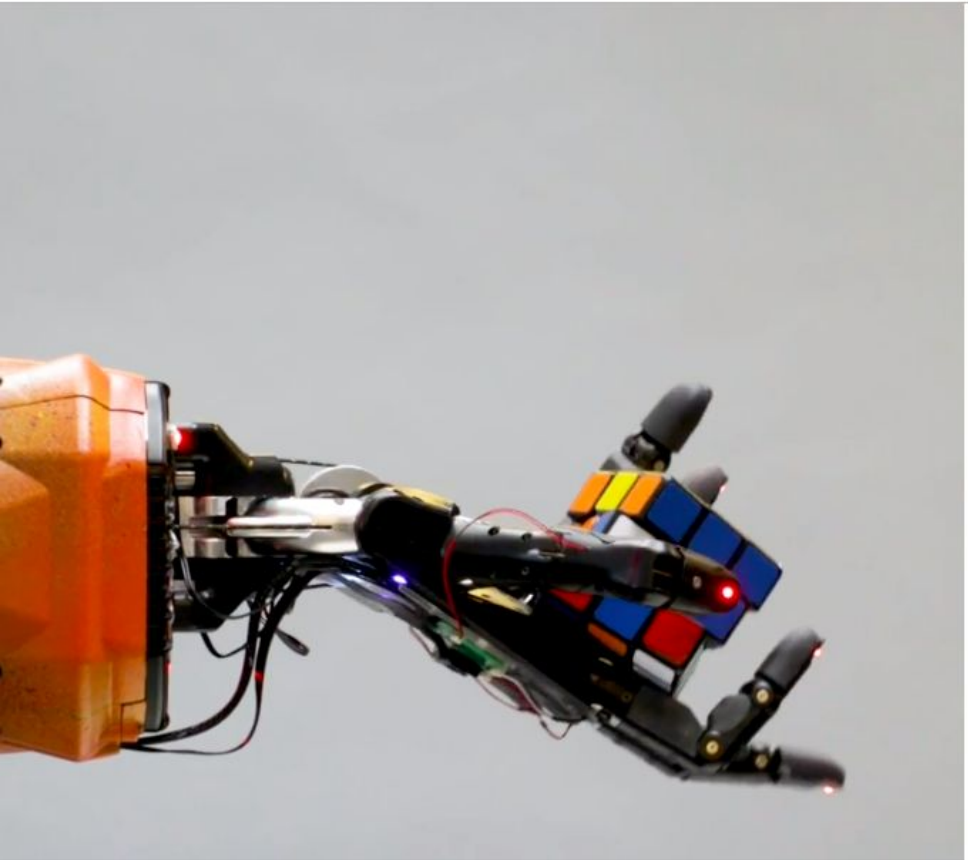
Optimization brought us today's "**modern control**"...

..with strong results for relatively simple forms uncertainty.

Real world uncertainty and "domain randomization" in RL is much richer.
"Black-box" optimization in RL still works.

Dealing with perception and "open-worlds" may cause the next major shift in controls research; we need the maturity of control to help address fundamental problems in robustness and sample-complexity.

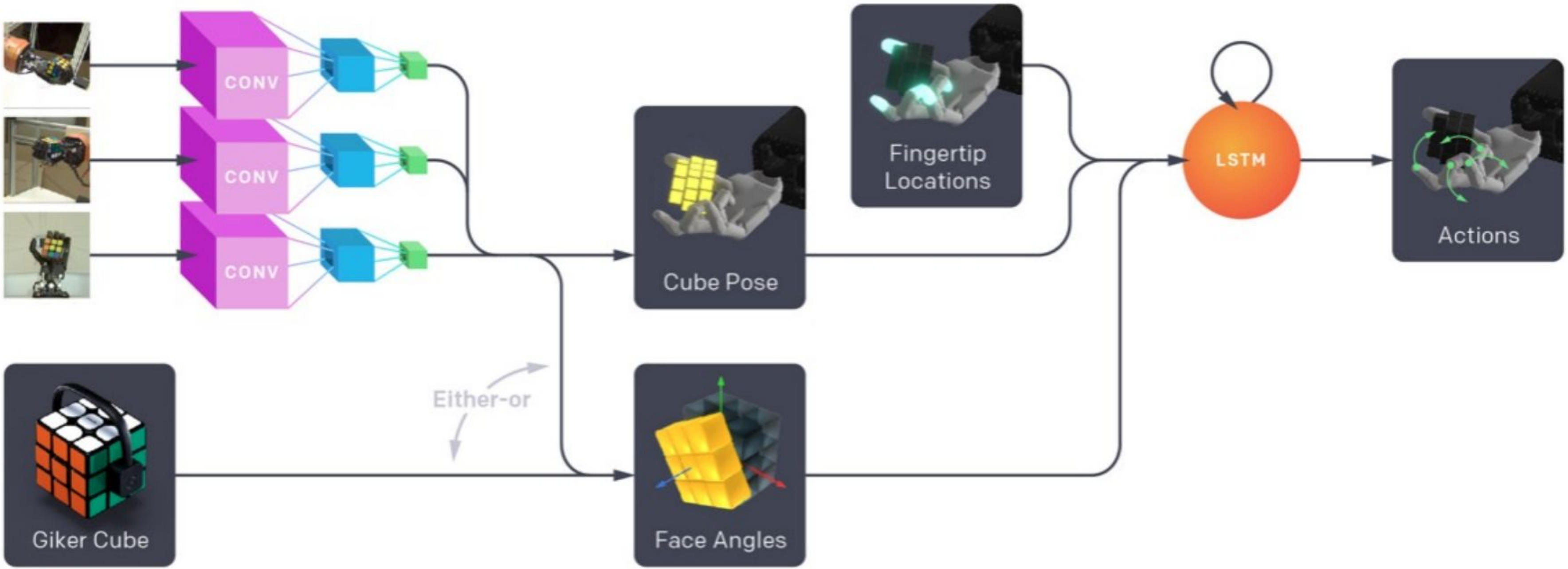
There is so much that we don’t know how to do yet!



The New York Times

If a Robotic Hand Solves a Rubik's Cube, Does It Prove Something?

A five-fingered feat could show important progress in A.I. research. It is also a stunt.



“For the Rubik’s cube task, we use 8 × 8 = 64 NVIDIA V100 GPUs and 8 × 115 = 920 worker machines with 32 CPU cores each. … The cumulative amount of experience ... is roughly **13 thousand years**.”

Solving Rubik's Cube with a Robot Hand by OpenAI, arXiv:1910.07113

Table 6: Performance of different policies on the Rubik's cube for a fixed fair scramble goal sequence. We evaluate each policy on the real robot (N=10 trials) and report the mean $\pm$ standard error and median number of successes (meaning the total number of successful rotations and flips). We also report two success rates for applying half of a fair scramble ("half") and the other one for fully applying it ("full"). For ADR policies, we report the entropy in nats per dimension (npd). For "Manual DR", we obtain an upper bound on its ADR entropy by running ADR with the policy fixed and report the entropy once the distribution stops changing (marked with an "*").

Policy	Sensing		ADR Entropy	Successes (Real)		Success Rate	
	Pose	Face Angles		Mean	Median	Half	Full
Manual DR	Vision	Giiker	-0.569^{*} npd	1.8 ± 0.4	2.0	0 %	0 %
ADR	Vision	Giiker	-0.084 npd	3.8 ± 1.0	3.0	0 %	0 %
ADR (XL)	Vision	Giiker	0.467 npd	17.8 ± 4.2	12.5	30 %	10 %
ADR (XXL)	Vision	Giiker	**0.479 npd**	$\mathbf{26.8 \pm 4.9}$	**22.0**	**60 %**	**20 %**
ADR (XXL)	Vision	Vision	**0.479 npd**	12.8 ± 3.4	10.5	20 %	0 %