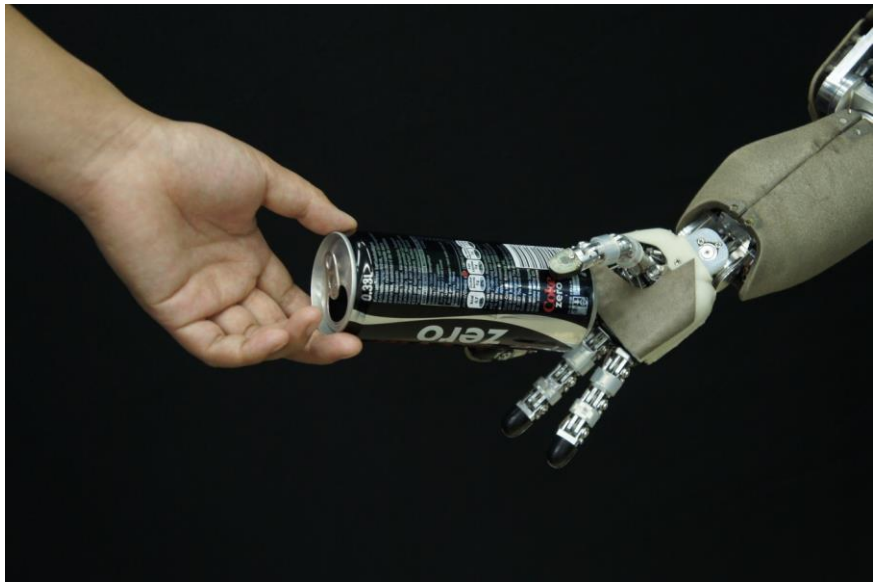
Learning Dexterous In-Hand Manipulation (18.08)

Seungjae Ryan Lee

endtoend.ai

Human vs Robot

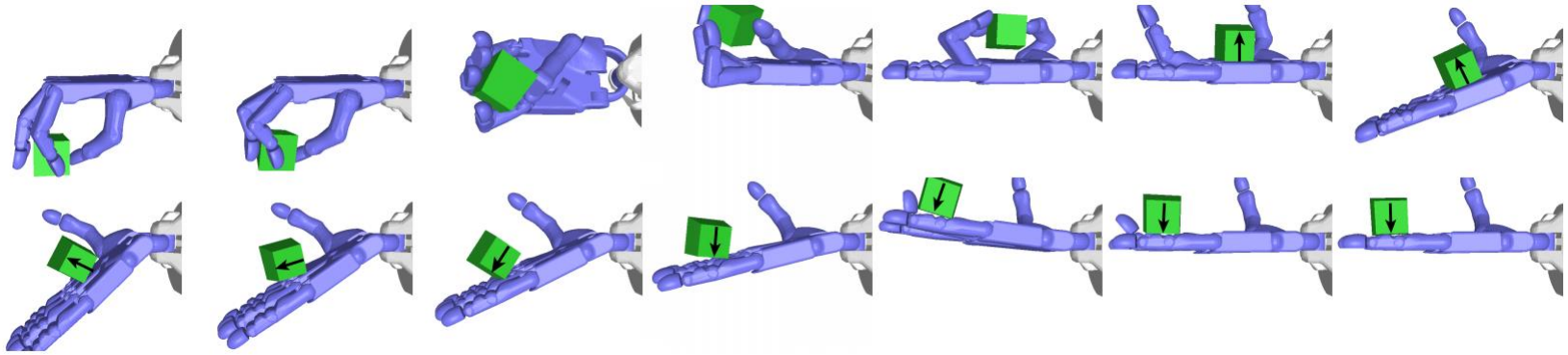
- Humans perform various *dexterous manipulation* tasks with hands
- Robots are designed for *specific tasks in constrained settings*
  - Unable to utilize complex end-effectors



http://lasa.epfl.ch/research_new/grasping/

Robot with Hand

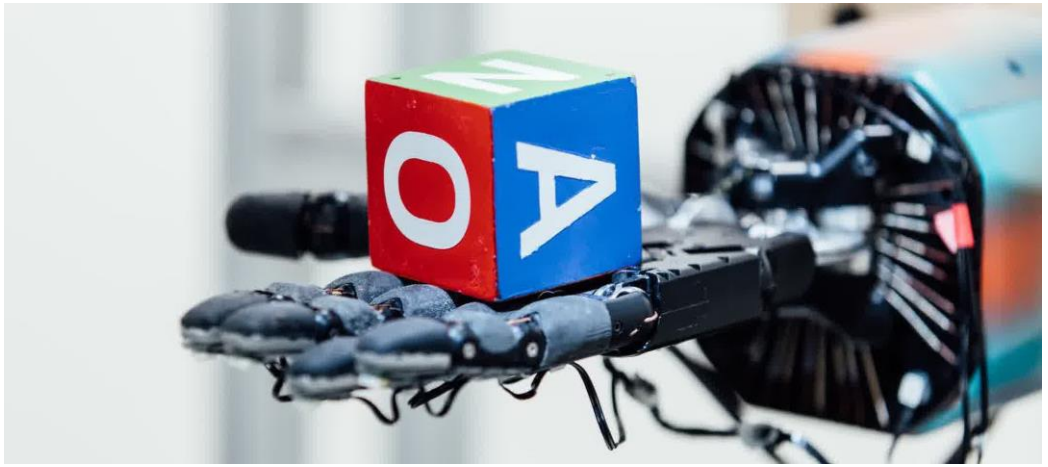
- Human hand has been an inspiration for research
- Limited results due to complexity of system
  - Simulation-only manipulation (no attempt on transferring)
  - Physical-only manipulation (slow and costly to run)



https://www.cc.gatech.edu/~ybai30/hand/paper_hand.pdf

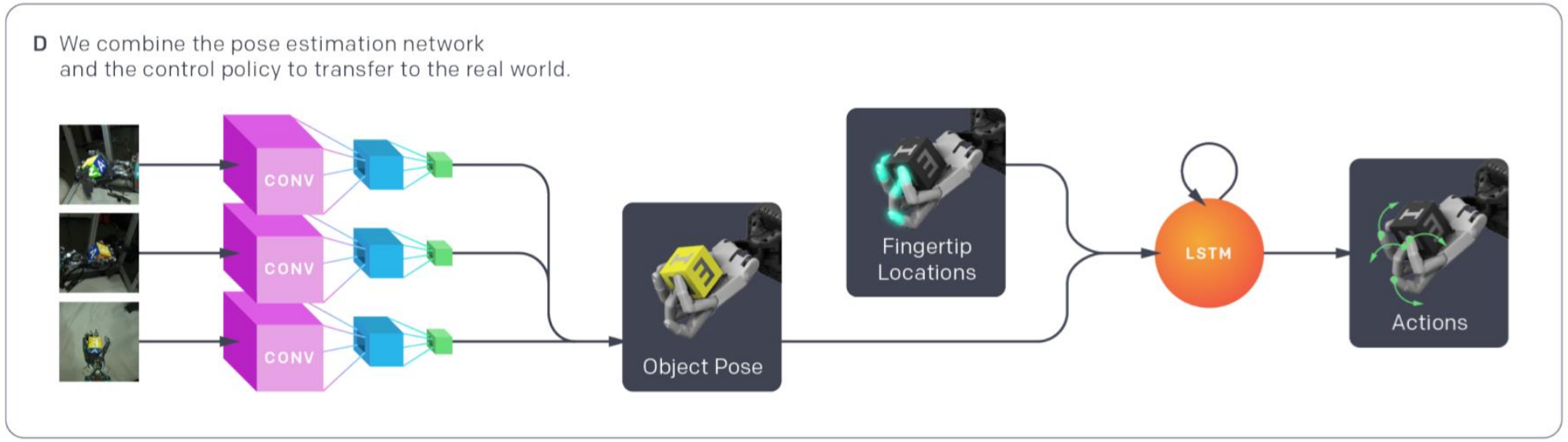
The Task: In-Hand Object Reorientation

- Object placed on robot hand
- Redirect the object to desired target configuration in-hand
- New goal provided if current goal achieved
- Continues until the object is dropped or after 50 successes



Dactyl: Approach

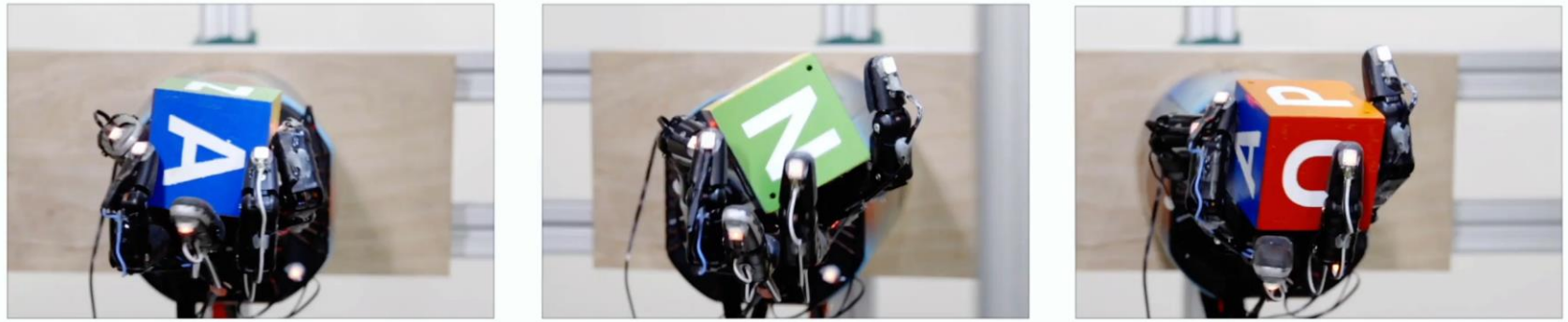
- *Transfer Learning* from simulator to real environment
  - Use extensive randomization and effects
  - Use memory augmented control policies to learn how to *adapt*
  - Use distributed RL



endtoend.ai

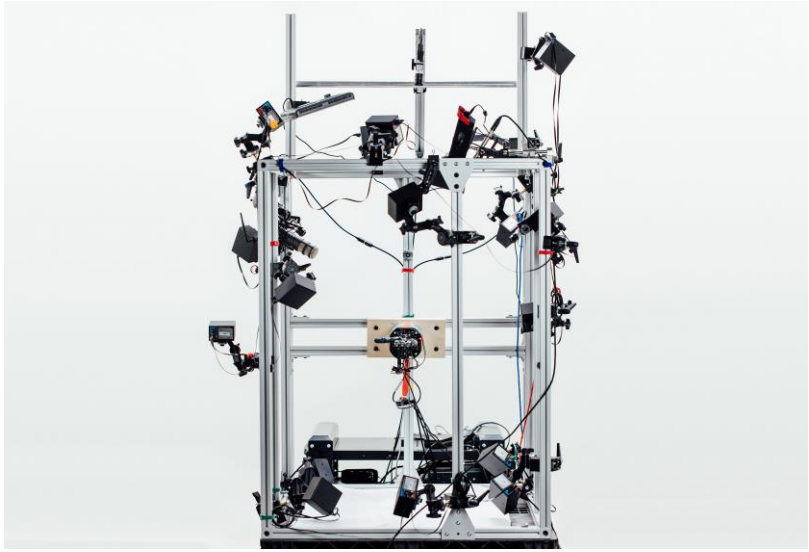
Dactyl: Results

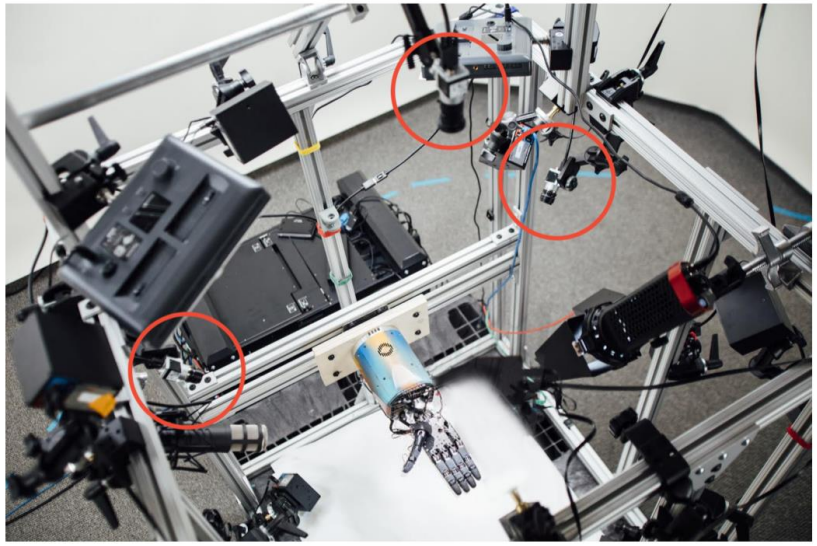
- Shows unprecedented levels of dexterity
- Discovers human-like grasp types without guidance



The Real World: Shadow Dexterous Hand

- 24 Degrees of Freedom (DOF)
- Hand: Cartesian position of 5 fingertips
- Object: Cartesian position and 3 RGB cameras (to handle real-world)





The Simulation: MuJoCo and Unity

- Simulate the physics with MuJoCo
- Render images with Unity
- Rough approximation of physical setup
  - Direct torque vs. Tendon-based actuation
  - Rigid body vs. Deformable body contact model
  - Creates a “reality gap”

Actions and Rewards

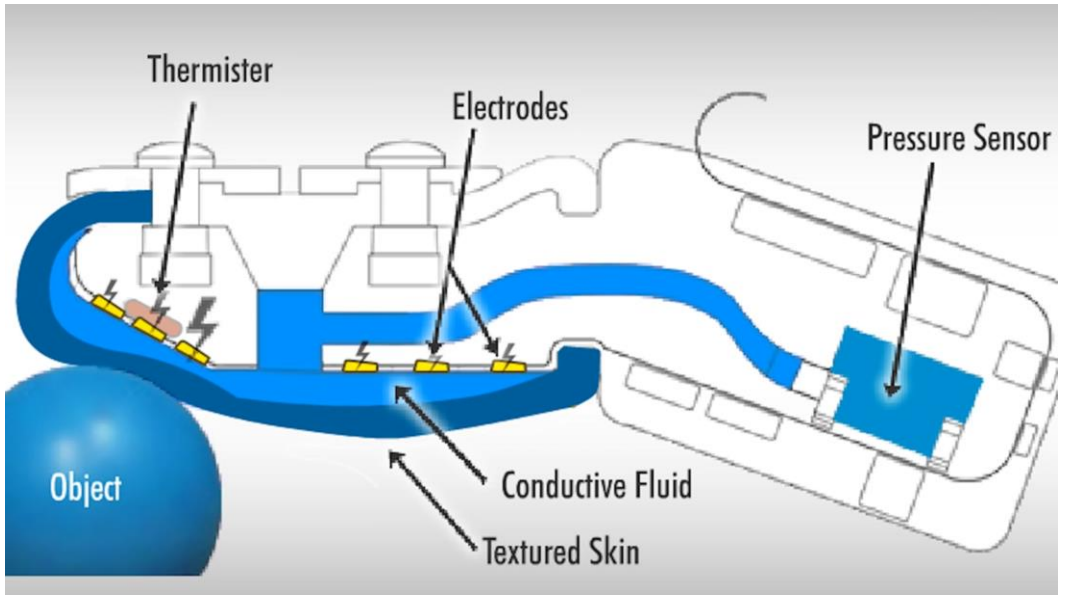
- Actions: relative joint angles
  - Discretized action into 11 bins (better performance than continuous)
  - Smoothed with exponential moving average to prevent robot breakage

- Reward: $r_t = d_t - d_{t+1}$
  - d_t, d_{t+1} are angles between desired and goal object orientation before and after transition
  - +5 when goal is achieved
  - -20 when object is dropped

Fostering Transferability: Observations

- Avoid adding some sensors as observations
  - Subject to state-dependent noise that **cannot be modelled**
  - Can be noisy and hard to calibrate

- Ex) Fingertip tactile sensor
  - Atmospheric pressure
  - Temperature
  - Shape of contact
  - Intersection geometry



https://www.youtube.com/watch?v=W_O-u9PNUMU

endtoend.ai

Fostering Transferability: Randomization

- Robust: Less likely to overfit to the simulated environment
- More likely to transfer successfully to the physical robot

- Types of randomizations:
  - Observation noise
  - Physics randomizations
  - Unmodeled effects
  - Visual appearance randomizations

Randomization: Observation Noise

- Correlated noise sampled each episode
- Uncorrelated noise sampled each timestep

Table 7: Standard deviation of observation noise.

Measurement	**Correlated noise**	**Uncorrelated noise**
fingertips positions	1mm	2mm
object position	5mm	1mm
object orientation	0.1rad	0.1rad
fingertip marker positions	3mm	
hand base marker position	1mm	

Randomization: Physics

- Parameters centered around values found during model calibration
- Sampled each episode

Table 1: Ranges of physics parameter randomizations.

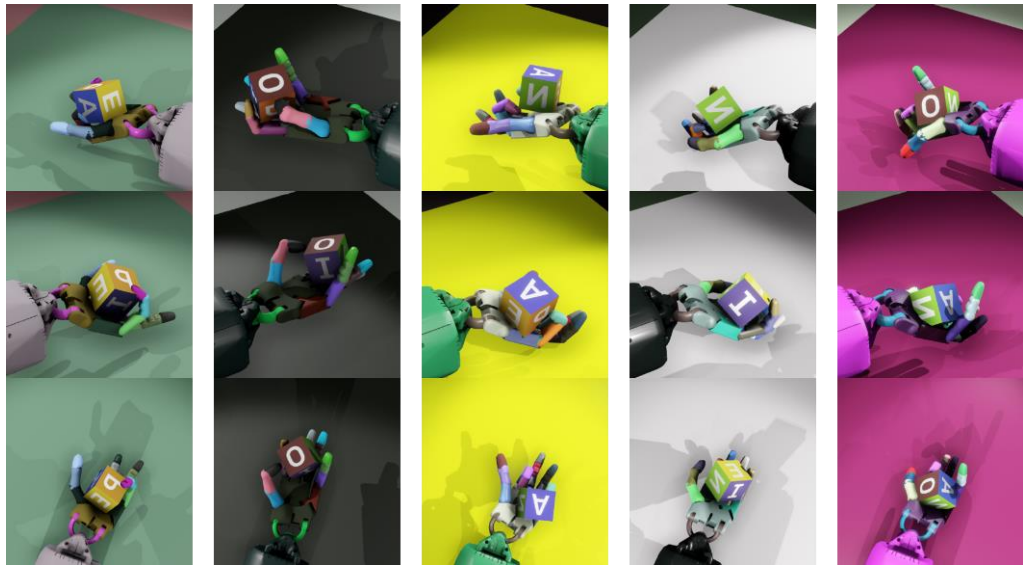
Parameter	**Scaling factor range**	**Additive term range**
object dimensions	uniform($[0.95, 1.05]$)	
object and robot link masses	uniform($[0.5, 1.5]$)	
surface friction coefficients	uniform($[0.7, 1.3]$)	
robot joint damping coefficients	loguniform($[0.3, 3.0]$)	
actuator force gains (P term)	loguniform($[0.75, 1.5]$)	
joint limits		$\mathcal{N}(0, 0.15)$ rad
gravity vector (each coordinate)		$\mathcal{N}(0, 0.4)$ m/s^2

Randomization: Unmodeled effects

- Physical robot experiences effects not modeled in simulation
- Accommodated by simulating such effects
  - Simulate *motor backlash*
  - Simulate action delay and noise
  - Simulate motion capture tracking errors
- Sometimes apply random forces on object

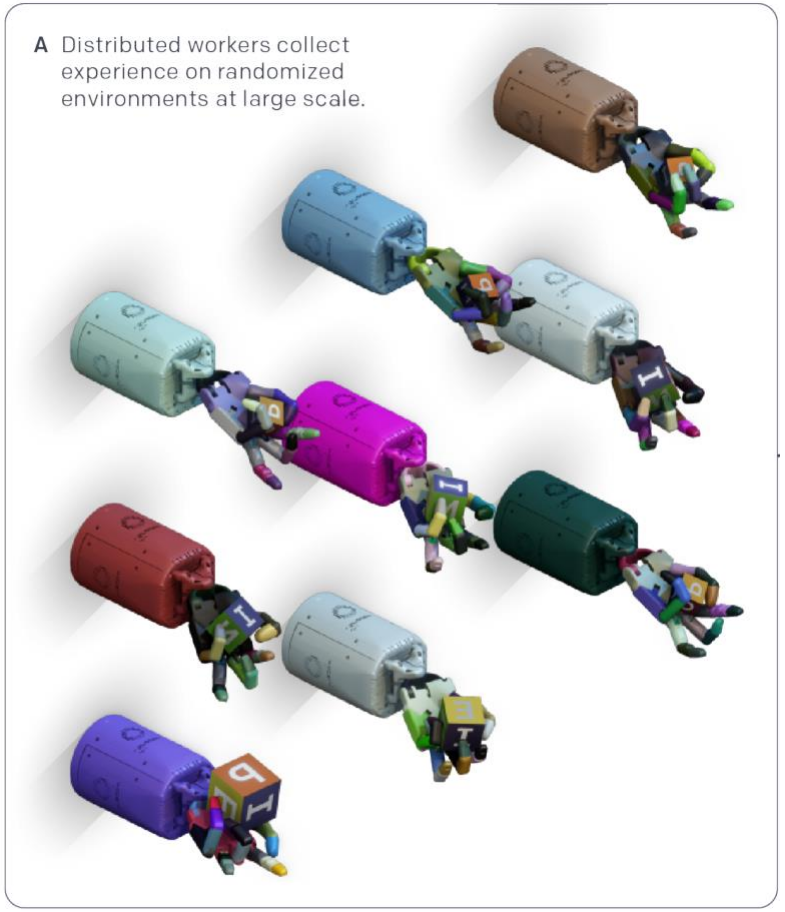
Randomization: Visual Appearance

- Randomize various visual aspects
  - camera position / intrinsics
  - Lighting conditions
  - Pose of hand and object
  - Materials and textures



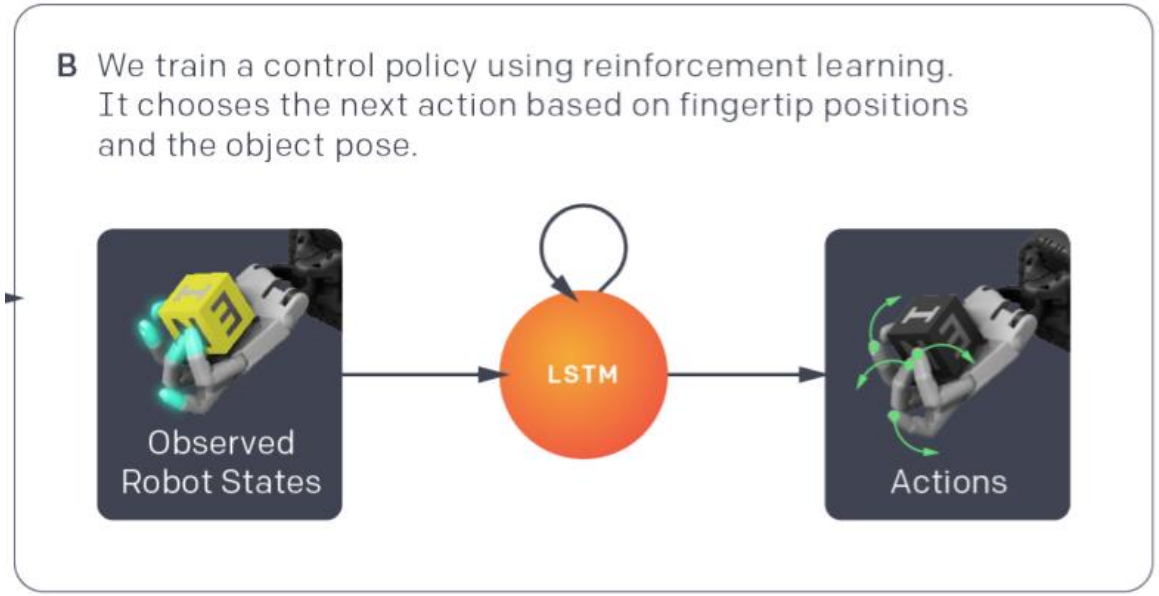
Step A: Collect Data

- Use just *simulations* to generate data
- Randomized environments



Step B: Train Control Policy for Simulation

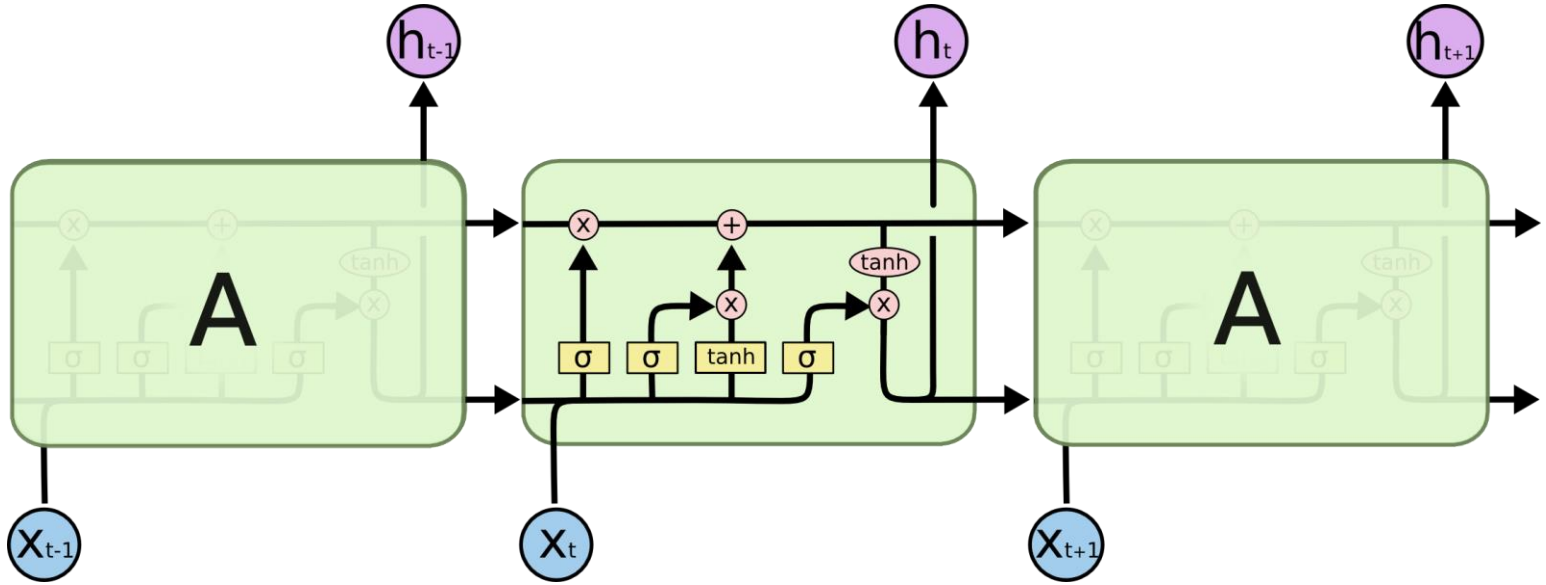
- Map observed states and rewards to actions
  - Fingertip locations: available in both simulation and physical robot
  - Object pose: available only in simulation



endtoend.ai

Policy Architecture: LSTM

- Randomizations persist across episode
- Agent can identify the environment's property and adapt accordingly
- Use Long Short-term Memory (LSTM)

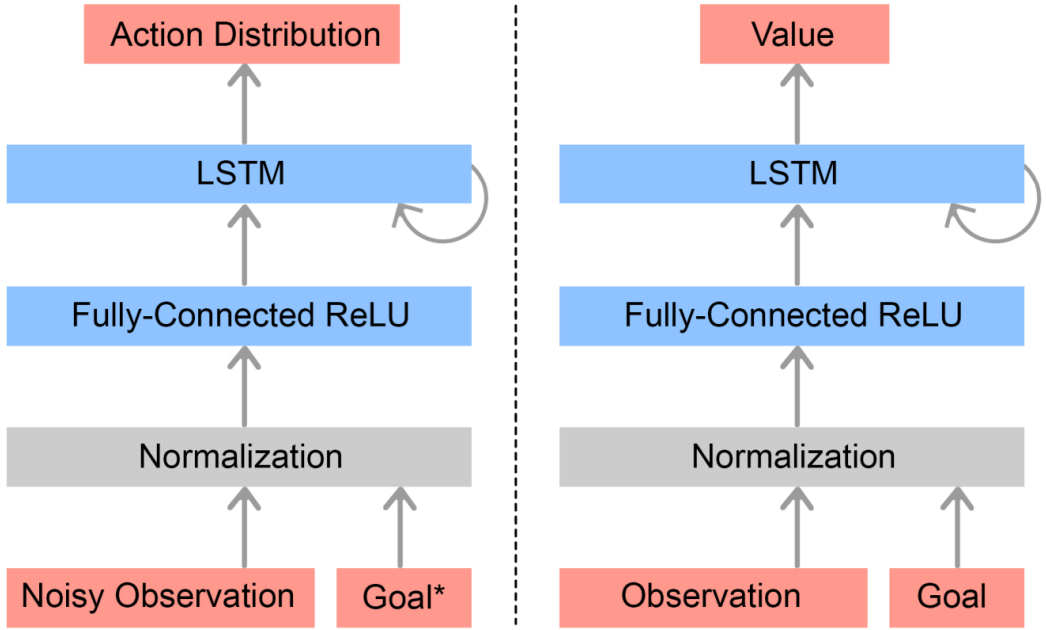


http://colah.github.io/posts/2015-08-Understanding-LSTMs/

endtoend.ai

Policy Architecture: PPO

- Trained with Proximal Policy Optimization (PPO)
  - Policy network (actor) and value network (critic) has same architecture



Policy Architecture: Asymmetric AC

- Value network has access to more info not available in real robots
- Simplifies learning good value estimates

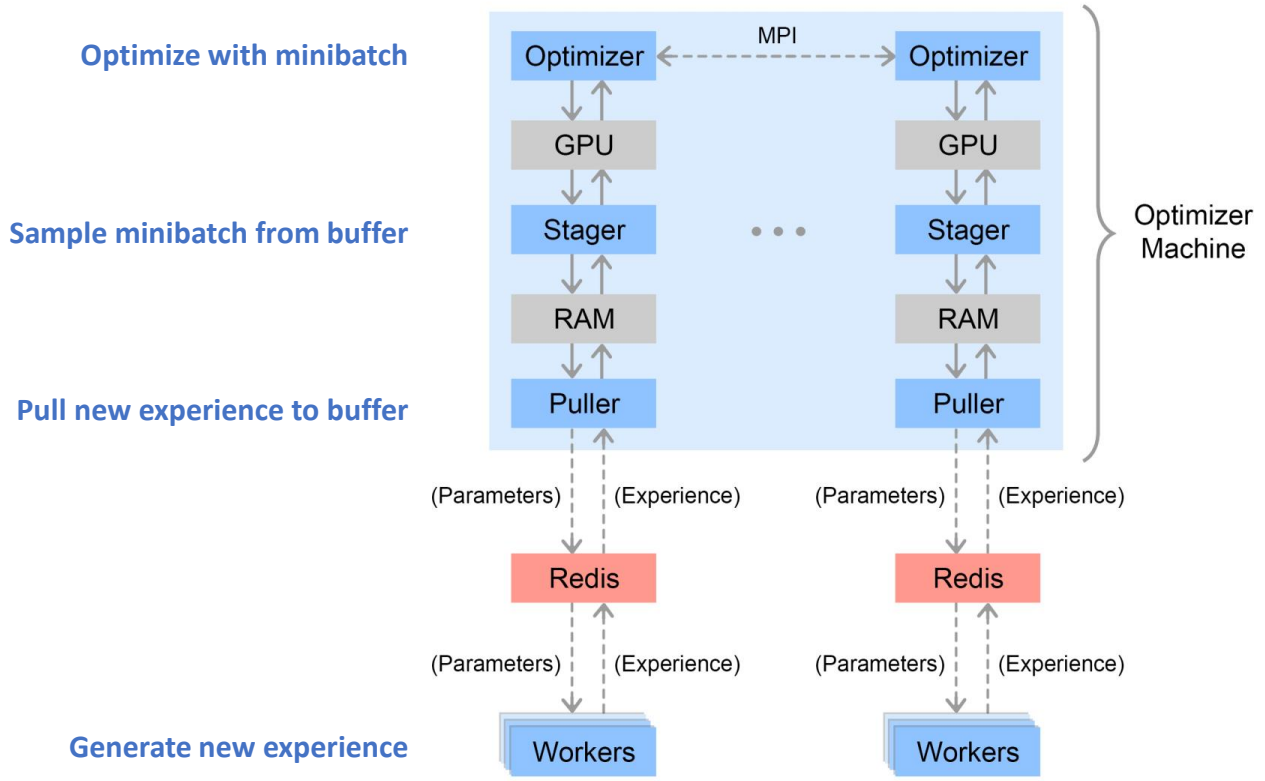
Table 2: Observations of the policy and value networks, respectively.

Input	Dimensionality	Policy network	Value network
fingertip positions	15D	✓	✓
object position	3D	✓	✓
object orientation	4D (quaternion)	×[4]	✓
target orientation	4D (quaternion)	×	✓
relative target orientation	4D (quaternion)	✓	✓
hand joints angles	24D	×	✓
hand joints velocities	24D	×	✓
object velocity	3D	×	✓
object angular velocity	4D (quaternion)	×	✓

https://arxiv.org/abs/1710.06542

Distributed Training with Rapid

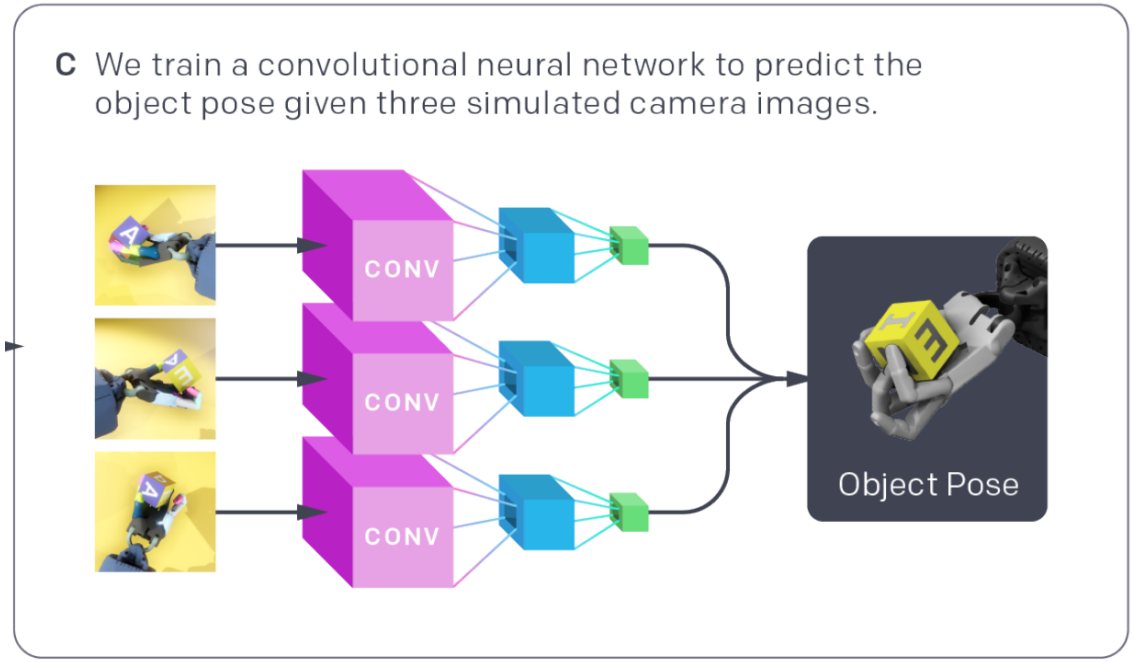
- Same distributed implementation as OpenAI Five



endtoend.ai

Step C: Train Pose Estimator

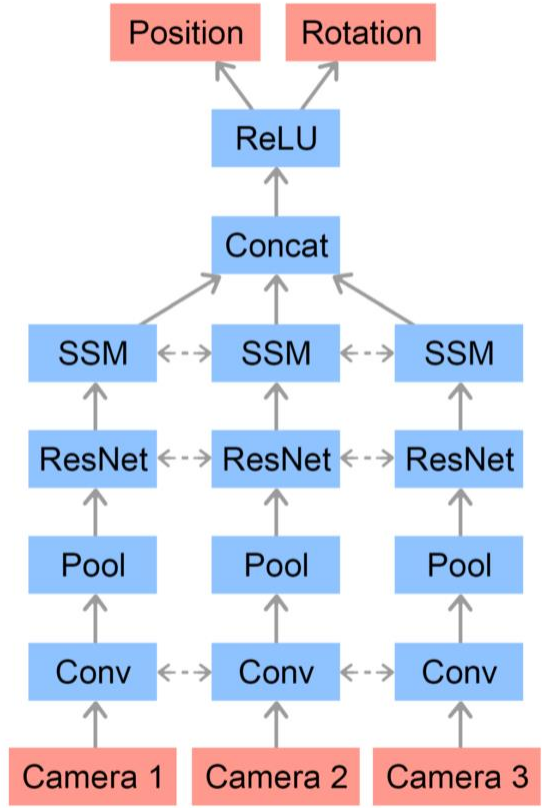
- Estimate the object's pose from vision
  - Object's pose is not available in real-world



endtoend.ai

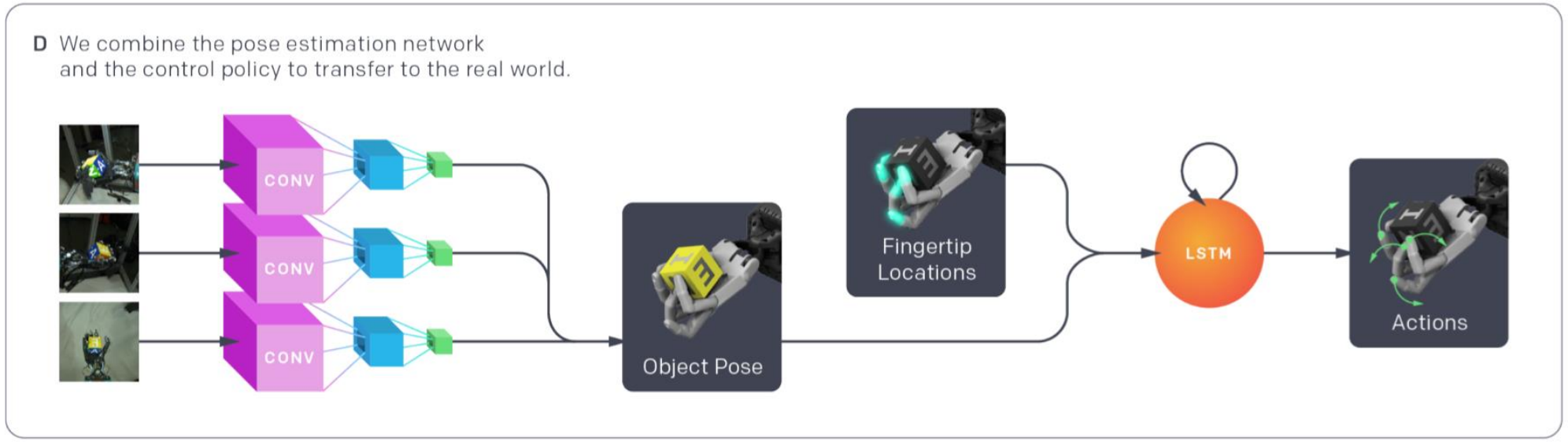
State Estimation from Vision

- Gather 1M states from simulator with trained policy
  - 3 RGB cameras with different viewpoints
- Predict both position and orientation of object
- Use minibatch gradient descent to minimize MSE
  - Shared CONV, RESNET, SSM layers between cameras
  - Use data augmentation



Step D: Combine and Transfer

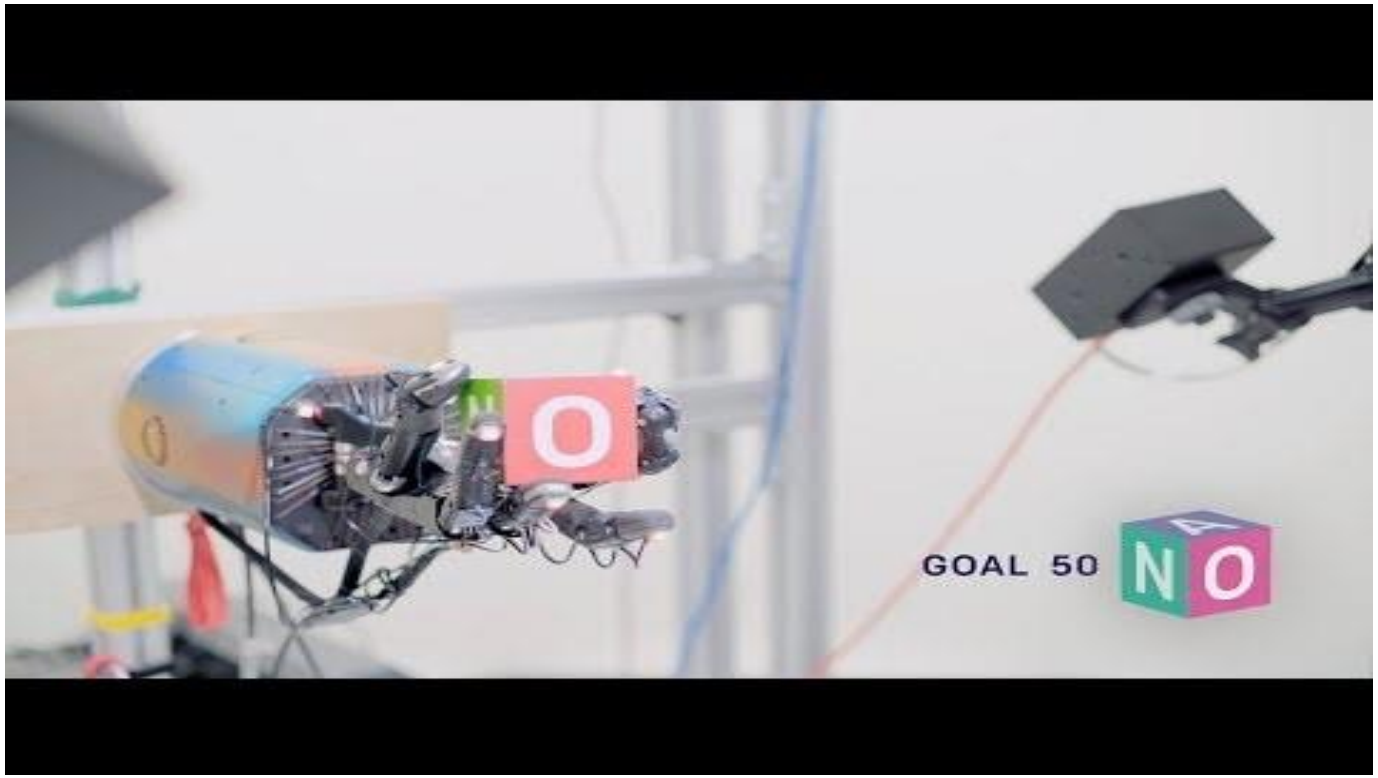
- Combine B and C to create an agent usable in real world



endtoend.ai

Qualitative Results

- Naturally learn humanlike grasps dexterous manipulation strategies



Quantitative Results

- # of *consecutive* successful rotations up to 50 until object is dropped
- Challenging to obtain due to *robot breakage* during experiments

Simulated task	**Mean**	**Median**	**Individual trials (sorted)**
Block (state)	43.4 ± 13.8	50	-
Block (state, locked wrist)	44.2 ± 13.4	50	-
Block (vision)	30.0 ± 10.3	33	-
Octagonal prism (state)	29.0 ± 19.7	30	-
Physical task			
Block (state)	18.8 ± 17.1	13	50, 41, 29, 27, 14, 12, 6, 4, 4, 1
Block (state, locked wrist)	26.4 ± 13.4	28.5	50, 43, 32, 29, 29, 28, 19, 13, 12, 9
Block (vision)	15.2 ± 14.3	11.5	46, 28, 26, 15, 13, 10, 8, 3, 2, 1
Octagonal prism (state)	7.8 ± 7.8	5	27, 15, 8, 8, 5, 5, 4, 3, 2, 1

Quantitative Results: Reality Gap

- # of *consecutive* successful rotations up to 50 until object is dropped

Simulated task	Mean	Median	Individual trials (sorted)
Block (state)	43.4 ± 13.8	50	-
Block (state, locked wrist)	44.2 ± 13.4	50	-
Block (vision)	30.0 ± 10.3	33	-
Octagonal prism (state)	29.0 ± 19.7	30	-
Physical task			
Block (state)	18.8 ± 17.1	13	50, 41, 29, 27, 14, 12, 6, 4, 4, 1
Block (state, locked wrist)	26.4 ± 13.4	28.5	50, 43, 32, 29, 29, 28, 19, 13, 12, 9
Block (vision)	15.2 ± 14.3	11.5	46, 28, 26, 15, 13, 10, 8, 3, 2, 1
Octagonal prism (state)	7.8 ± 7.8	5	27, 15, 8, 8, 5, 5, 4, 3, 2, 1

Quantitative Results: Vision

- Vision model is comparable to state model in real world

Simulated task	Mean	Median	Individual trials (sorted)
Block (state)	43.4 ± 13.8	50	-
Block (state, locked wrist)	44.2 ± 13.4	50	-
Block (vision)	30.0 ± 10.3	33	-
Octagonal prism (state)	29.0 ± 19.7	30	-
Physical task			
Block (state)	18.8 ± 17.1	13	50, 41, 29, 27, 14, 12, 6, 4, 4, 1
Block (state, locked wrist)	26.4 ± 13.4	28.5	50, 43, 32, 29, 29, 28, 19, 13, 12, 9
Block (vision)	15.2 ± 14.3	11.5	46, 28, 26, 15, 13, 10, 8, 3, 2, 1
Octagonal prism (state)	7.8 ± 7.8	5	27, 15, 8, 8, 5, 5, 4, 3, 2, 1

Quantitative Results: Different Object

- Learn successful policies that transfer, but larger reality gap
- Failed on spheres due to rolling behaviors

Simulated task	Mean	Median	Individual trials (sorted)
Block (state)	43.4 ± 13.8	50	-
Block (state, locked wrist)	44.2 ± 13.4	50	-
Block (vision)	30.0 ± 10.3	33	-
Octagonal prism (state)	29.0 ± 19.7	30	-
Physical task			
Block (state)	18.8 ± 17.1	13	50, 41, 29, 27, 14, 12, 6, 4, 4, 1
Block (state, locked wrist)	26.4 ± 13.4	28.5	50, 43, 32, 29, 29, 28, 19, 13, 12, 9
Block (vision)	15.2 ± 14.3	11.5	46, 28, 26, 15, 13, 10, 8, 3, 2, 1
Octagonal prism (state)	7.8 ± 7.8	5	27, 15, 8, 8, 5, 5, 4, 3, 2, 1

Ablation of Randomizations

- Randomizations are vital to real-world performance

Training environment	Mean	Median	Individual trials (sorted)
All randomizations (state)	18.8 ± 17.1	13	50, 41, 29, 27, 14, 12, 6, 4, 4, 1
No randomizations (state)	1.1 ± 1.9	0	6, 2, 2, 1, 0, 0, 0, 0, 0, 0
No observation noise (state)	15.1 ± 14.5	8.5	45, 35, 23, 11, 9, 8, 7, 6, 6, 1
No physics randomizations (state)	3.5 ± 2.5	2	7, 7, 7, 3, 2, 2, 2, 2, 2, 1
No unmodeled effects (state)	3.5 ± 4.8	2	16, 7, 3, 3, 2, 2, 1, 1, 0, 0
All randomizations (vision)	15.2 ± 14.3	11.5	46, 28, 26, 15, 13, 10, 8, 3, 2, 1
No observation noise (vision)	5.9 ± 6.6	3.5	20, 12, 11, 6, 5, 2, 2, 1, 0, 0

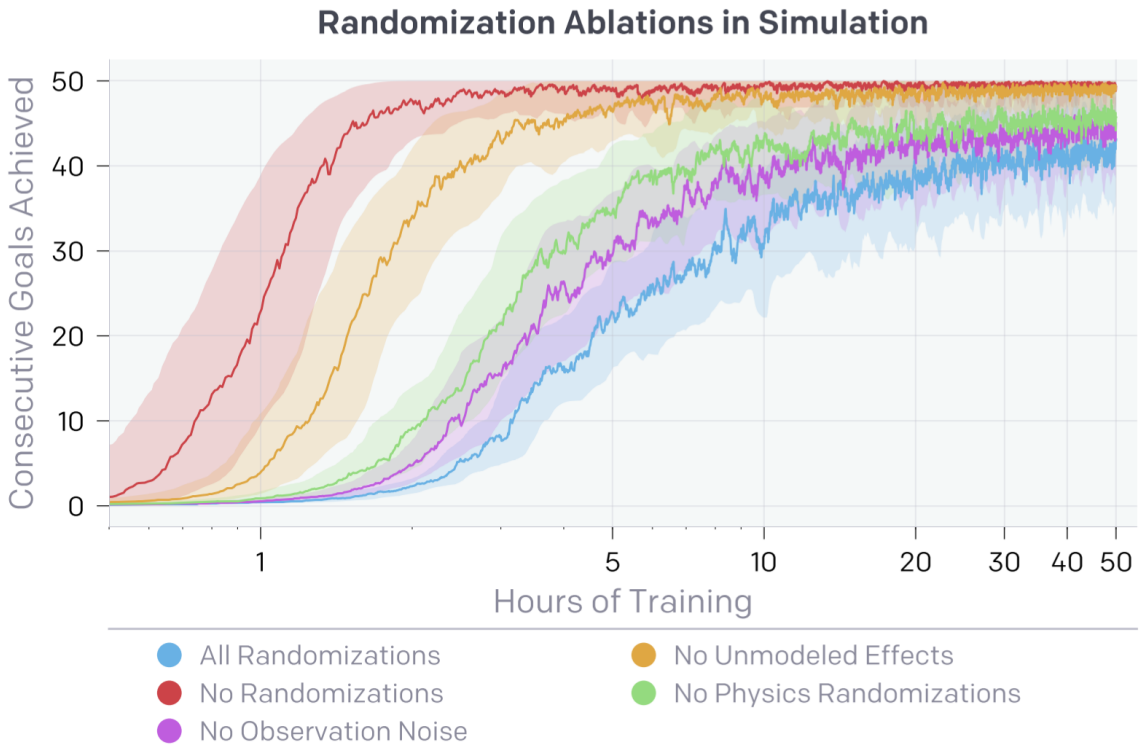
Ablation of Randomizations: Observation

- Little performance drop for observation noise
  - Due to accurate motion capture system
  - Vital when pose estimates are much more noisy

Training environment	Mean	Median	Individual trials (sorted)
All randomizations (state)	18.8 ± 17.1	13	50, 41, 29, 27, 14, 12, 6, 4, 4, 1
No randomizations (state)	1.1 ± 1.9	0	6, 2, 2, 1, 0, 0, 0, 0, 0, 0
No observation noise (state)	15.1 ± 14.5	8.5	45, 35, 23, 11, 9, 8, 7, 6, 6, 1
No physics randomizations (state)	3.5 ± 2.5	2	7, 7, 7, 3, 2, 2, 2, 2, 2, 1
No unmodeled effects (state)	3.5 ± 4.8	2	16, 7, 3, 3, 2, 2, 1, 1, 0, 0
All randomizations (vision)	15.2 ± 14.3	11.5	46, 28, 26, 15, 13, 10, 8, 3, 2, 1
No observation noise (vision)	5.9 ± 6.6	3.5	20, 12, 11, 6, 5, 2, 2, 1, 0, 0

Ablation of Randomizations: Limitations

- Takes ≈ 33 times more time (100 years of experience vs 3 years)

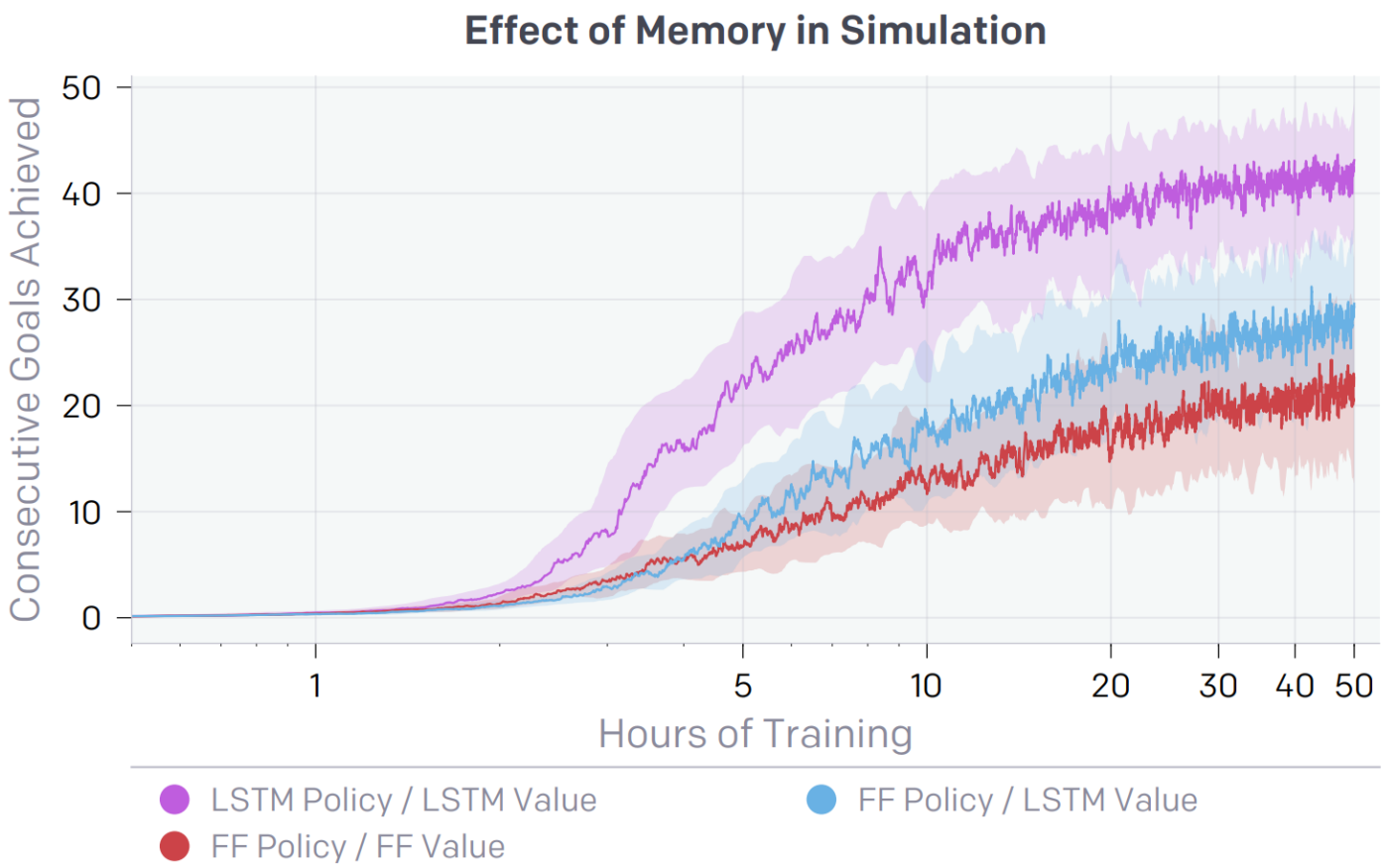


Effect of Memory in Policies

- LSTM achieves better performance than feedforward (FF) network
  - Was able to predict environment randomizations
  - Ex) 80% accuracy in predicting if the block's size was bigger or smaller than the average

Network architecture	Mean	Median	Individual trials (sorted)
LSTM policy / LSTM value (state)	18.8 ± 17.1	13	50, 41, 29, 27, 14, 12, 6, 4, 4, 1
FF policy / LSTM value (state)	4.7 ± 4.1	3.5	15, 7, 6, 5, 4, 3, 3, 2, 2, 0
FF policy / FF value (state)	4.6 ± 4.3	3	15, 8, 6, 5, 3, 3, 2, 2, 2, 0

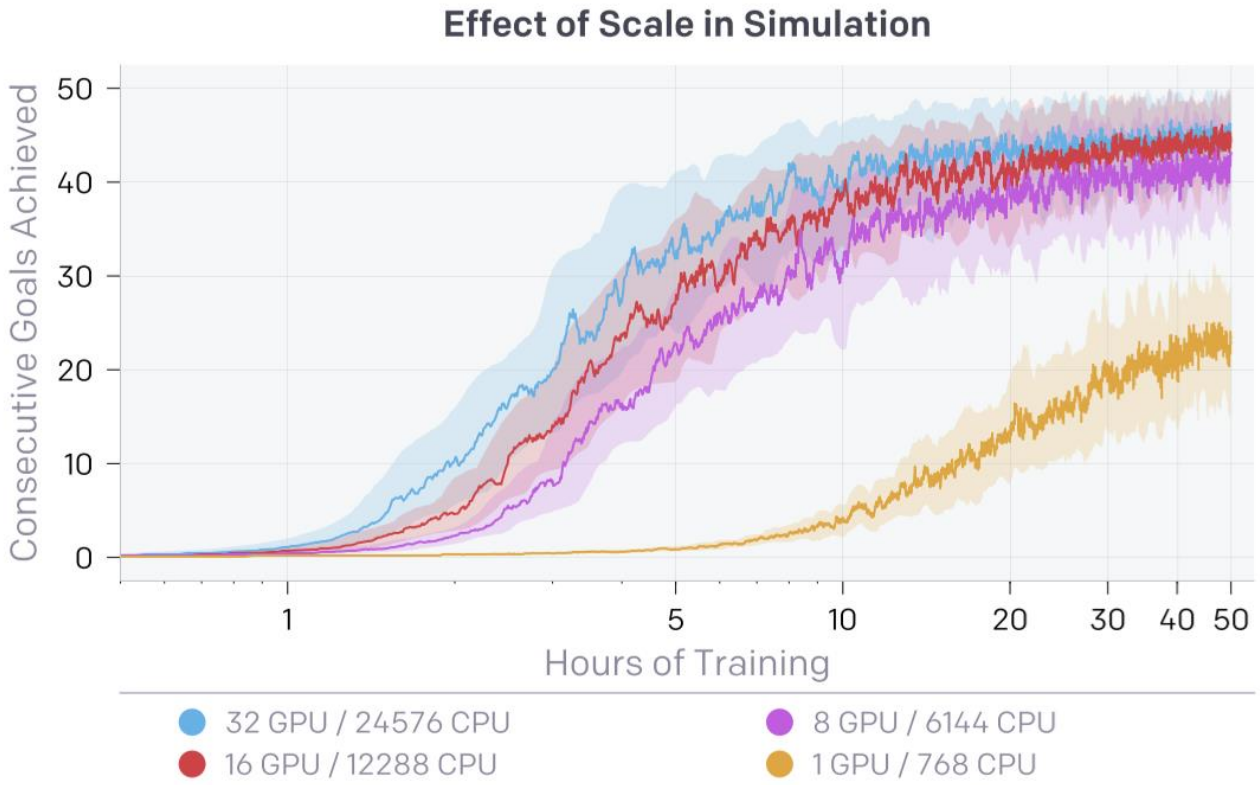
Effect of Memory in Policies



endtoend.ai

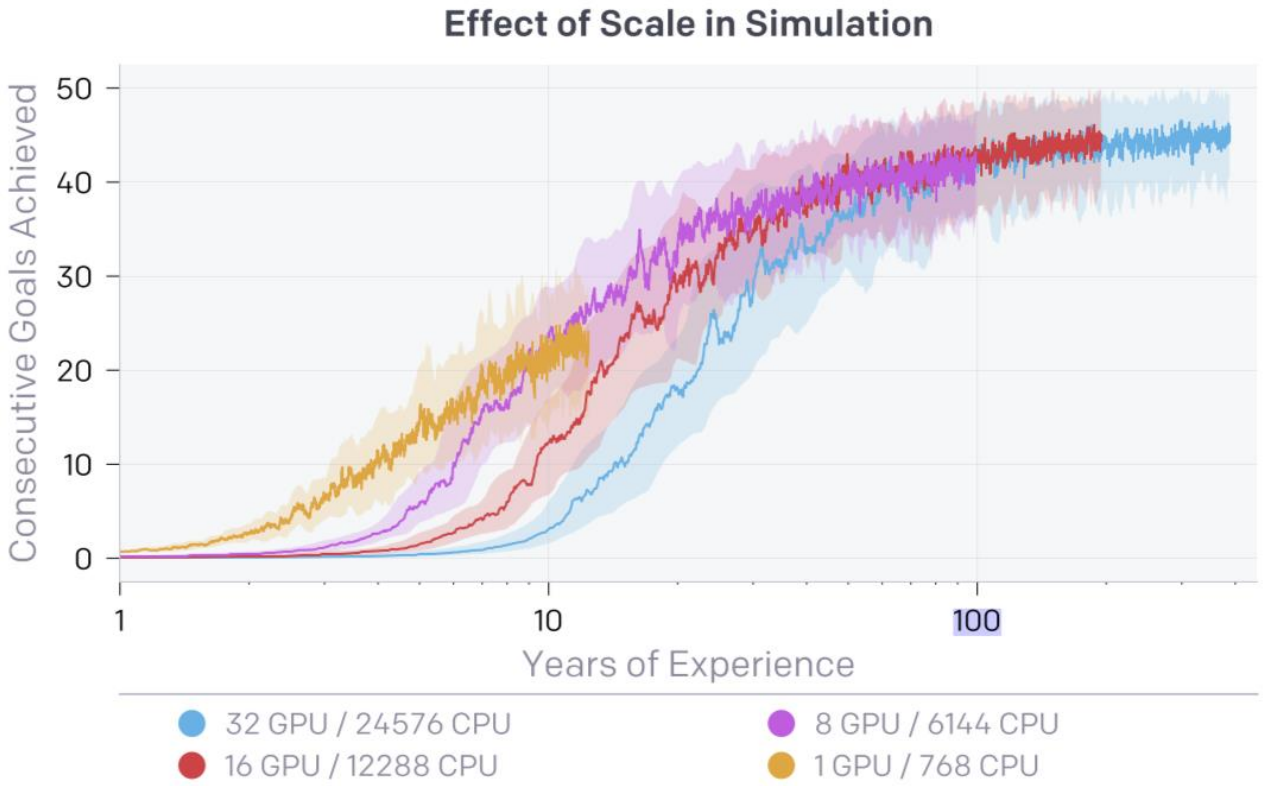
Sample Complexity and Scale

- Diminishing return with further scaling



Sample Complexity and Scale

- Little affect on how fast the agent learns w.r.t years of experience



Vision Performance

- Low error on synthetic data
  - Successful Unity-to-MuJoCo transfer
- Higher error on real data
  - Reality gap, noises, occlusions, etc…
  - Still performs reasonably well

Test set	**Rotation error**	**Position error**
Rendered images (Unity)	$2.71° \pm 1.62$	$3.12\text{mm} \pm 1.52$
Rendered images (MuJoCo)	$3.23° \pm 2.91$	$3.71\text{mm} \pm 4.07$
Real images	$5.01° \pm 2.47$	$9.27\text{mm} \pm 4.02$

What Surprised Us

1. Tactile sensing was not necessary

2. Randomization for one object generalized to similar objects
   - System for a block worked well for an octagonal prism
   - Had trouble with sphere due to its rolling behavior

3. System engineering is as important as good algorithm
   - System without timing bug performed consistently better

What Didn’t Work

1. Faster reaction time ≠ Better Performance
   - No noticeable difference between 40ms and 80ms

2. Using real data for training vision model did not work
   - Latency and measurement error
   - Hassle to collect due to configuration changes

Thank you!

Original Paper: https://arxiv.org/abs/1808.00177

Paper Recommendations:

- Using Simulation and Domain Adaptation to Improve Efficiency of Deep Robotic Grasping
- Asymmetric Actor Critic for Image-Based Robot Learning
- Generalizing from Simulation

You can find more content in www.endtoend.ai/slides

endtoend.ai