

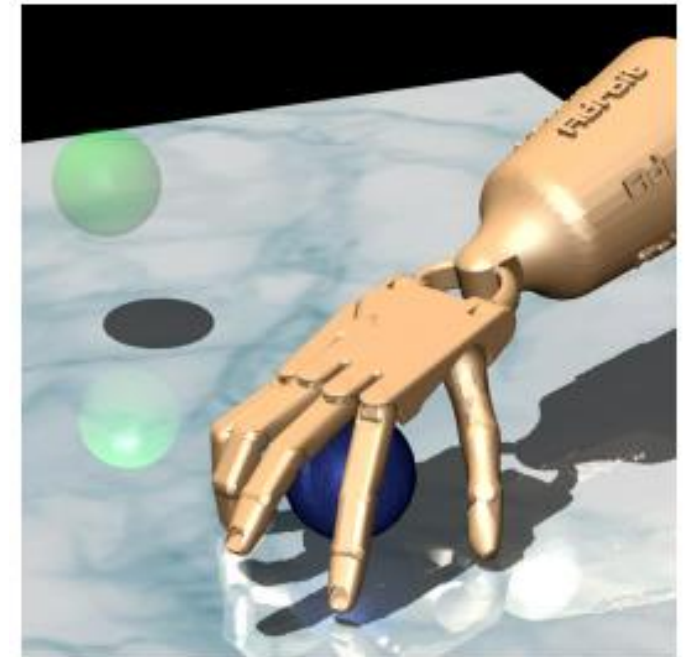
Learning Complex Dexterous Manipulation with Deep Reinforcement Learning and Demonstrations

Aravind Rajeswaran , Vikash Kumar , Abhishek Gupta, Giulia
Vezzani , John Schulman , Emanuel Todorov , Sergey Levine

Jason Rebello
UTIAS

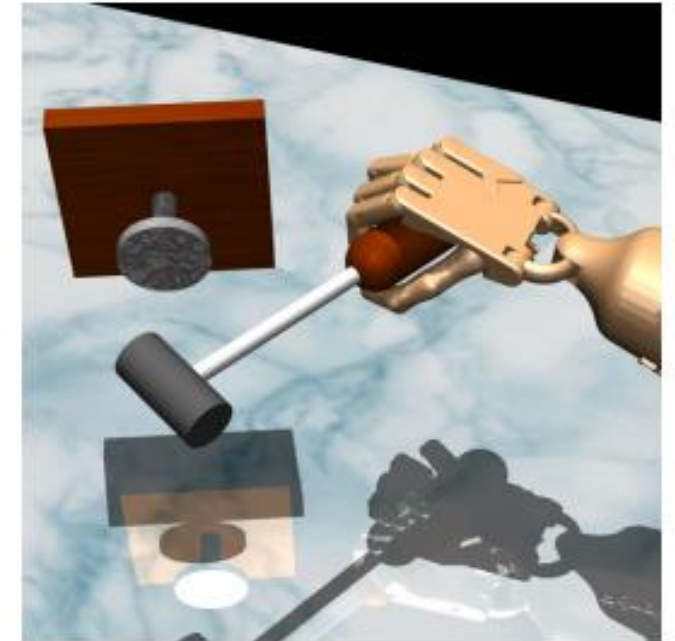
- Motivation
- Contributions
- Methodology
- Experiments
- Future Work

- Dextrous multi-fingered hands are ***extremely versatile***
- ***Control*** is challenging due to high dimensionality, complex contact patterns
- Previous methods require ***reward shaping***
- DRL limited to ***simpler manipulators and simple tasks***
- Lack of physical systems due to ***sample inefficiency***



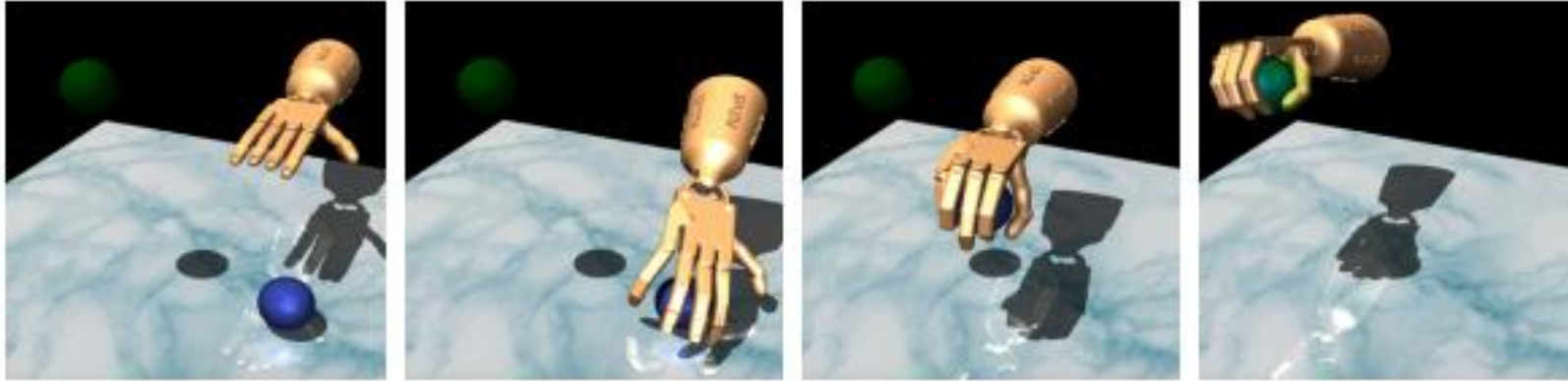
Object relocation task

- Manipulation with **24-DOF** hand
- **Model Free** DRL
- Used in **complex tasks with variety of tools**
- Small number of **human demonstrations** reduces sample complexity
- **Reduces learning time**
- **Robust** and natural movements



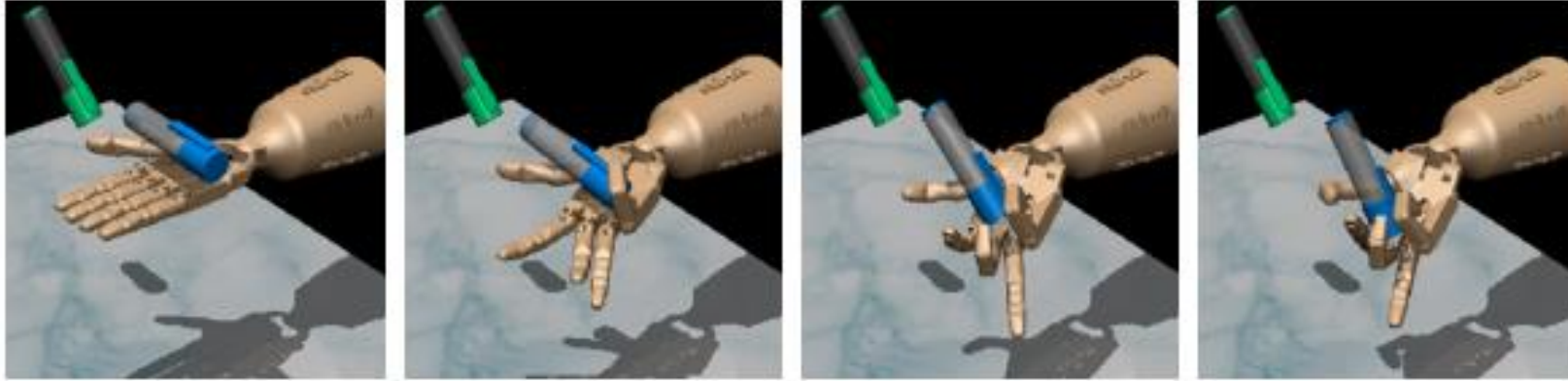
Tool use task

Object Relocation



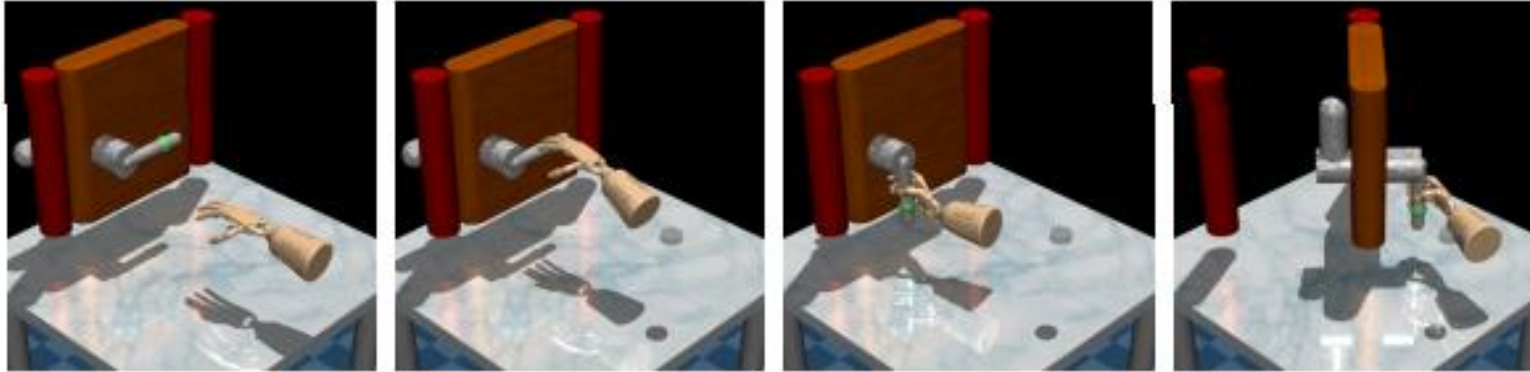
- Move Blue ball to green position
- Task complete when ball is epsilon ball away from target
- Positions of ball and target are randomized
- Main challenge is exploration (reach object, grab and move to target location)

In-hand Manipulation



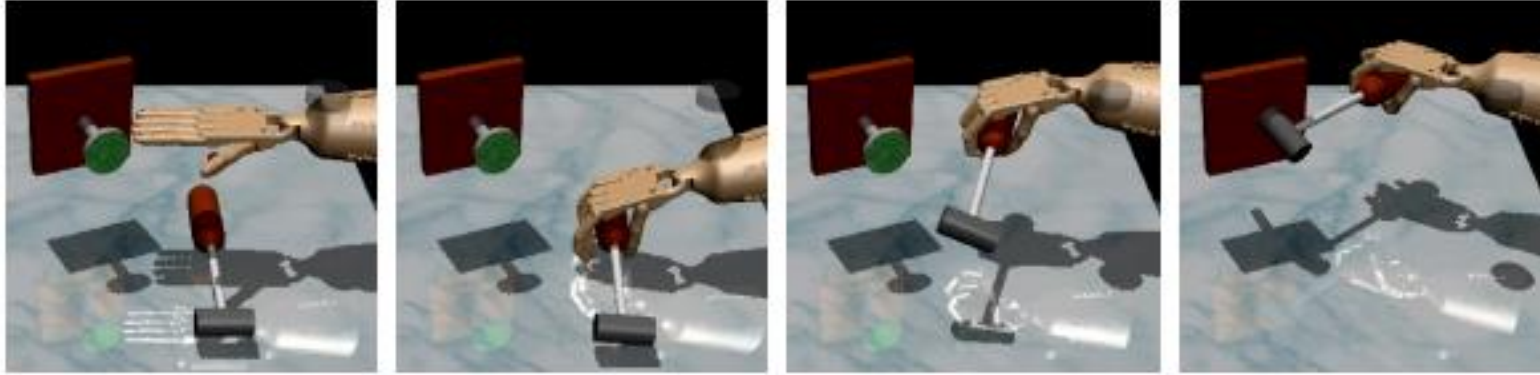
- Reposition blue pen to match orientation of green target
- Task complete when orientation is achieved
- Base of hand is fixed
- Large number of contacts with complex solutions
- Used a well shaped reward for training an expert

Door Opening



- Undo latch and swing door open
- Task complete when door touches door stopper
- No information of latch explicitly provided
- A lot of hidden sub-tasks
- Position of door is randomized

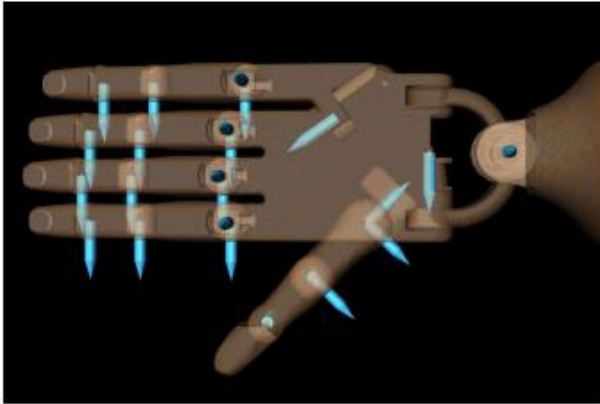
Tool Use



- Pickup and hammer nail
- Task complete when entire nail is inside the board
- Use tool instead of just relocation
- Multiple steps in task

Experimental Setup

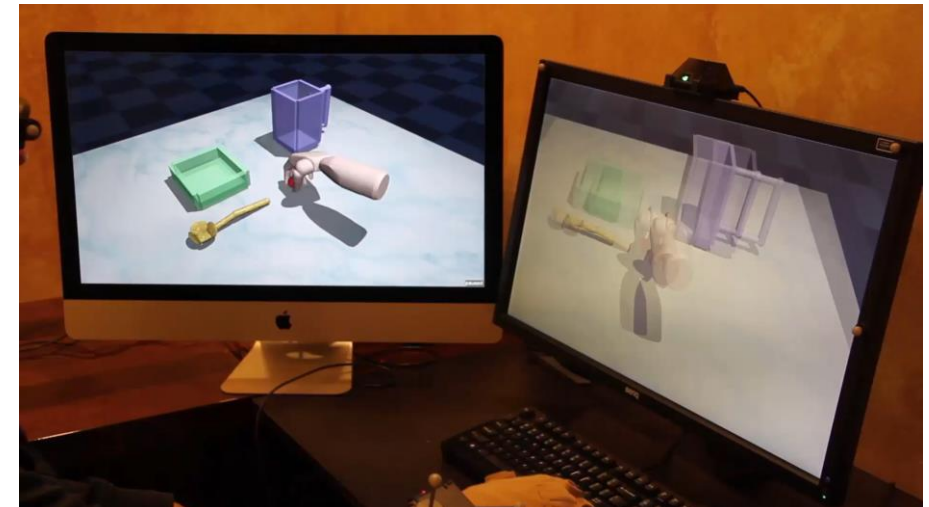
ADROIT hand



HTC headset



HAPTIX Simulator



- 24-DOF hand
- First, middle, ring – 4 DOF each
- Little finger, thumb – 5 DOF each
- Wrist – 2 DOF
- Actuated with position control and has joint angle sensor
- MuJoCo physics simulation with friction
- 25 demonstrations for each task



CyberGlove 3

MDP definition:

$$\mathcal{M} = \{ \mathcal{S}, \mathcal{A}, \mathcal{R}, \mathcal{T}, \rho_0, \gamma \}$$

States

Rewards

Initial Probability distribution

Actions

Transition dynamics

Discount Factor

Value function:

$$V^\pi(s) = \mathbb{E}_{\pi, \mathcal{M}} \left[\sum_{t=0}^{\infty} \gamma^t r_t \mid s_0 = s \right]$$


Q function:

$$Q^\pi(s, a) = \underbrace{\mathbb{E}_{\mathcal{M}} [\mathcal{R}(s, a)]}_{\text{Reward for taking action } a \text{ in state } s} + \underbrace{\mathbb{E}_{s' \sim \mathcal{T}(s, a)} [V^\pi(s')]}_{\text{Expected reward in state } s'}$$

Advantage function:

$$A^\pi(s, a) = Q^\pi(s, a) - V^\pi(s)$$

- Directly optimize parameters of policy to maximize objective

Vanilla Policy Gradient: Sub-optimal 

$$g = \frac{1}{NT} \sum_{i=1}^N \sum_{t=1}^T \nabla_{\theta} \log \pi_{\theta}(a_t^i | s_t^i) \hat{A}^{\pi}(s_t^i, a_t^i, t)$$

Fisher Information Matrix:

$$F_{\theta} = \frac{1}{NT} \sum_{i=1}^N \nabla_{\theta} \log \pi_{\theta}(a_t^i | s_t^i) \nabla_{\theta} \log \pi_{\theta}(a_t^i | s_t^i)^T$$

- Fisher information matrix measures the curvature (sensitivity) of policy relative to model parameters
- Fisher information matrix is related to the Hessian matrix

- Limit policy change based on parameter change
- Fisher information matrix maps between parameter space and policy space
- Generally use learning rate in optimization
- Poor step size leads to poor initialization
- Use Fisher information matrix to perform update

Gradient ascent update:

$$\theta_{k+1} = \theta_k + \sqrt{\frac{\delta}{g^T F_{\theta_k}^{-1} g}} F_{\theta_k}^{-1} g$$

Steepest Ascent direction

Normalized step-size

- Challenges with using NPG
 - RL requires careful reward shaping
 - Impractical number of samples to learn (approx. 100 hours)
 - Unnatural movement
 - Not as robust to environmental variations
- Solution
 - Combine RL with demonstrations
 - Guide exploration and decrease sample complexity
 - Robust and natural looking behaviour
 - Demonstration Augmented Policy Gradient (DAPG)

- Exploration in PG achieved with stochastic action distribution
- Poor initialization leads to slow exploration
- Behavioral Cloning (BC) guides exploration
- Reduces sample complexity

$$\underset{\theta}{\text{maximize}} \quad \sum_{(s,a) \in \rho_D} \ln \pi_{\theta}(a|s)$$

- Mimic actions taken in demonstrations
- Does not guarantee effectiveness of policy due to distributional shift

- BC does not make optimal use of demonstrations
- Cannot learn subtasks (reaching, grasping, hammering)
- BC policy (only grasping)
- Capturing all data

$$g_{aug} = \sum_{(s,a) \in \rho_{\pi}} \overbrace{\nabla_{\theta} \ln \pi_{\theta}(a|s) A^{\pi}(s,a)}^{\text{Policy gradient}} + \sum_{(s,a) \in \rho_D} \overbrace{\nabla_{\theta} \ln \pi_{\theta}(a|s) w(s,a)}^{\text{Behavioral cloning}}$$

\swarrow Dataset from policy \swarrow Dataset from demonstrations \nwarrow Weighting function

$$w(s,a) = \lambda_0 \lambda_1^k \max_{(s',a') \in \rho_{\pi}} A^{\pi}(s',a') \quad \forall (s,a) \in \rho_D$$

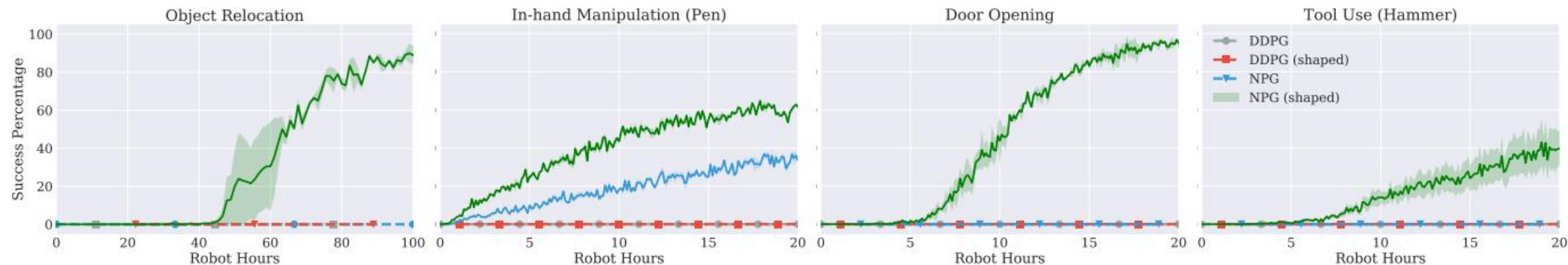
\nwarrow hyperparameters \swarrow iteration

Reinforcement learning from scratch

- Can RL cope with high dimensional manipulation tasks ?
- Is it robust to variations in environment ?
- Are movements safe and can they be used on real hardware ?

- Compare NPG vs DDPG (Deep Deterministic Policy Gradient)
- DDPG is a policy gradient actor-critic algorithm that is off-policy
- Stochastic policy for exploration, estimates deterministic policy
- Score based on percentage of successful trajectories (100 samples)
- Sparse Reward vs Reward shaping

Reinforcement learning from scratch



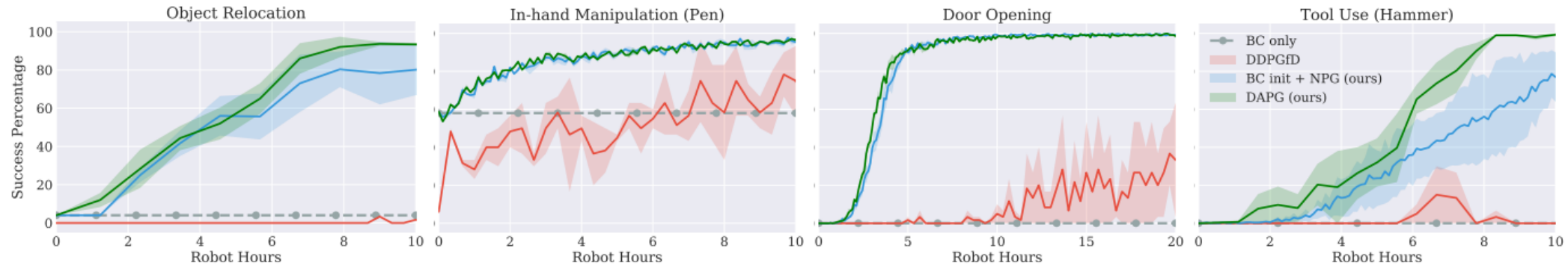
- NPG learns with reward shaping, DDPG fails to learn
- DDPG is sample efficient but sensitive to hyper-parameters
- Resulting policies have unnatural behaviors
- Poor sample efficiency, cant use on hardware
- Cannot generalize to unseen environment (weight and ball size change)

Reinforcement learning with demonstrations

- Does incorporating demonstrations reduce learning time?
- Comparison of DAPG vs DDPGfD (.. from Demonstrations)?
- Does it result in human like behaviour ?

- DDPGfD better version of DDPG (demonstrations in replay buffer, prioritized experience replay, n-step returns, regularization)
- Only use sparse rewards

Reinforcement learning with demonstrations



RL iterations to achieve 90% success

- DAPG outperforms DDPGfD
- DAPG requires few robot hours
- Can be used on real hardware
- Robust and human behavior
- Generalizes to unseen environment

Method	DAPG (sp)		RL (sh)		RL (sp)	
Task	N	Hours	N	Hours	N	Hours
Relocation	52	5.77	880	98	∞	∞
Hammer	55	6.1	448	50	∞	∞
Door	42	4.67	146	16.2	∞	∞
Pen	30	3.33	864	96	2900	322

NPG

- Tests on real hardware
- Reduce sample complexity using novelty based exploration methods
- Learn policies from raw visual inputs and tactile sensing

Learning Complex Dexterous Manipulation with Deep Reinforcement Learning & Demonstrations



Aravind Rajeswaran*, Vikash Kumar *, Abhishek Gupta, John Schulman,
Emanuel Todorov, Sergey Levine
OPENAI, UC BERKELEY, UW SEATTLE

*Thank you &
Questions ?*