Learning Complex Dexterous Manipulation with Deep Reinforcement Learning and Demonstrations

Aravind Rajeswaran[1,3*], Vikash Kumar[1,3*], Abhishek Gupta[2], John Schulman[1], Emanuel Todorov[3], Sergey Levine[2]

***Abstract*— Dexterous multi-fingered hands are extremely versatile and provide a generic way to perform multiple tasks in human-centric environments. However, effectively controlling them remains challenging due to their high dimensionality and large number of potential contacts. Deep reinforcement learning (DRL) provides a model-agnostic approach to control complex dynamical systems, but has not been shown to scale to high-dimensional dexterous manipulation, and deployment on physical systems remains challenging due to sample inefficiency. The success of DRL in robotics has thus far been limited to simpler manipulators and tasks. In this work, we show that model-free DRL with natural policy gradients can effectively scale up to complex manipulation tasks with a high-dimensional 24-DoF hand, and solve them from scratch in simulated experiments. Furthermore, a small number of human demonstrations used to guide exploration can significantly reduce sample complexity, and enable learning within the equivalent of a few hours of robot experience. We demonstrate successful policies for highly complex tasks: object relocation, in-hand manipulation, tool use, and interaction with human-centric environments.**

I. Introduction

Dexterous manipulation with multi-fingered hands is an important and challenging problem. Multi-fingered manipulators are extremely versatile and are capable of performing a large variety of contact-rich tasks. However, this versatility comes at the price of high dimensional observation and action spaces, complex and discontinuous contact patterns, and under-actuation during non-prehensile manipulation. This makes controlling such manipulators difficult, especially for traditional model-based approaches which rely on accurate models and state estimates. Due to the difficulties of dexterous manipulation, prior work on manipulation with multi-fingered hands has largely focused on simpler tasks, such as grasping [2] or rotating an object in the hand [40].

In this work, we aim to address these challenges by using model-free deep reinforcement learning (DRL) which offers a model-agnostic approach to solving complex control problems. DRL has previously been demonstrated on whole-arm manipulation tasks [10], and complex locomotion tasks [12], however it has not yet been shown to scale successfully to dexterous manipulation. Unlike locomotion, hand manipulation evolves in a compact work-space with constraints and discontinuities. Frequent contact switches, as the object maneuvers around in the hand workspace, aggressively changes the progress direction. As a result policies often get stuck in bad local minima in absence of consistent progress momentum. In this work, we aim to show that model-free DRL methods can indeed solve complex dexterous manipulation tasks, circumventing the need for explicit modeling.

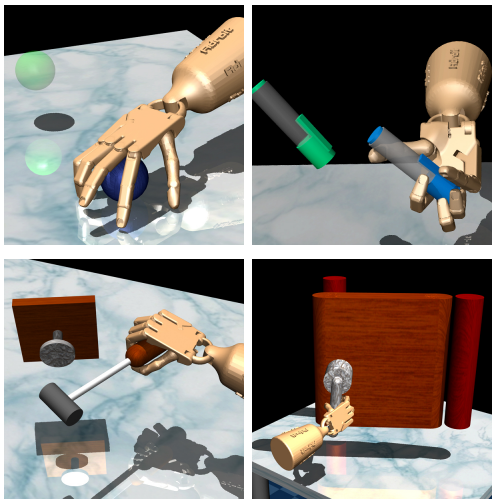


Fig. 1: We demonstrate that deep reinforcement learning can acquire a wide range of complex dexterous manipulation skills involving object relocation, in-hand manipulation (pen repositioning), tool use (hammering a nail), and interacting with human centric environments (opening a door). With the inclusion of human demonstrations, the training times can be reduced down to a few hours.

Traditionally, the main drawback of DRL methods has been their prohibitively high sample complexity. Furthermore, more complex tasks of the kind explored in this work often pose significant exploration and safety challenges, requiring considerable manual reward shaping as well as manual resetting of the system. To address these challenges, we incorporate demonstrations into DRL. This is done by combining a natural policy gradient method with demonstration-based pre-training and a demonstration-augmented surrogate objective. In cases where it is not feasible to obtain demonstrations, our training method still works and generates competent controllers, but slower.

With these components, we show that model-free DRL can learn to perform highly complex object-centric tasks using a five-fingered hand with a practical amount of experience – on the order of 5 hours of robot time, which is feasible to run on physical systems. We also show that when

* Equal contributions. [1] OpenAI, [2] University of California Berkeley, [3] University of Washington. This work was supported by the NSF.

Preprint submitted to 2018 IEEE International Conference on Robotics and Automation. Received September 15, 2017.

incorporating demonstrations, we can use much sparser rewards than what would be feasible when learning from scratch. These results suggest that our learning methods can be used in real-world learning of complex dexterous manipulation, although this remains to be demonstrated on physical systems.

II. Related Work

Hand manipulation is one of the most complex forms of motor control exhibited by humans. The difficulty of hand manipulation can be attributed to multiple factors including but not limited to – complexity of mechanical design, stringent sensing and spatial constraints, curse of dimensionality, complexity in synthesis of dexterous manipulation behaviors. Owing to these complexities most prior work has focused on simple tasks like reaching, grasping [2], rolling [40] etc with low degree of freedom manipulators.

There have been several recent advancements in designing highly dexterous manipulators. [44] presents an anthropomorphic robotic hand that replicates the important details of human hand bio-mechanics. [6], [11] present a soft flexible manipulator. [20] [43] present fast actuation driving an anthropomorphic hand past human speed with quick reflexes. As the capabilities of such manipulators improve, controlling them becomes increasingly harder.

Model-based methods such as [17] have applied model predictive control to synthesizing complex behaviors in real time via dimensionality augmentation. Contact-invariant optimization (CIO) [24] introduces auxiliary decision variables for directly specifying when and where contacts should occur, and optimized these variables jointly to synthesize hand manipulation behavior. Sampling [23] and motion capture [22] based approaches have also been able to synthesize complex manipulation trajectories but require detailed models and careful attention to details by an expert [4].

Model-free reinforcement learning methods [36] and versions with deep function approximators such as [33], [21] do not require a model of the dynamics, and instead optimize the policy directly. However, these methods have not yet been shown to scale effectively to dexterous manipulation, and are typically very sample inefficient and require well shaped rewards, making them impractical for real-world use. Our main goal here is to overcome these limitations.

Several prior works have considered methods for incorporating demonstrations to speed up and improve reinforcement learning. Methods such as dynamic movement primitives (DMP) [27] [38] [16] have been used to effectively combine demonstrations and reinforcement learning. However, these methods use a specific, often limited form of function representation. Our work uses deep neural networks for policy representation, which allow representation of complex dexterous manipulation policies. Similar to our method, demonstrations have been used to pre-train policies via behavior cloning [8], as well as to pre-train a Q-function by minimizing TD error [13]. Additionally demonstrations have been used to guide exploration through reward/policy shaping but these are often rule-based or work on discrete spaces making them difficult to apply to high dimensional dexterous manipulation [37] [5] [34]. Our method for combining RL with demonstrations uses a behavior cloning initialization along with a demo augmented surrogate objective described in Section IV-B.

Although model-free RL is typically too sample inefficient, some recent works have been able to reduce sample complexity through careful task set-up, and apply them on real world tasks. Rusu et. al [31] learn a policy in simulation and transfer it to the real world. Other work [9] shows that by pre-training visual features and an action decoder in simulation, model-free RL can learn simple tasks in the real world. In a different approach, [10] shows that with parallel training and data collection, model-free methods can be scaled to the real world for simple tasks. The work most closely related to ours in terms of motivation is [41], where demonstrations are incorporated into DDPG [21] by adding them to the replay buffer. This presents a natural and elegant way to combine demonstrations with an off-policy RL method. The method we propose in this work combines demonstrations with an on-policy policy gradient method. Off-policy methods, when successful, tend to be more sample efficient, but are generally more unstable [7]. On-policy methods on the other hands are more robust, and scale to high dimensional spaces [32]. Our results indicate that with the incorporation of demos, the sample complexity of on-policy methods can be dramatically reduced, while retaining their stability and robustness.

Perhaps most importantly, the main difference between this paper and prior work is the quality of the manipulation controllers and the complexity of the tasks we have been able to solve. This is not due to one particular algorithmic leap, but to a range of technical improvements combined with careful experimentation and advanced computing infrastructure. Another subtle but very desirable detail that is hard to quantify, but can be well appreciated in the accompanying video, is keen resemblance to *human-like* movements exhibited by our policies. The emergence of such behaviors are due to the use of demonstrations, without which RL finds successful yet erratic policies.

III. Problem Formulation

We model the control problem as a Markov decision process (MDP) in the episodic average reward setting, which is defined using the tuple: $\mathcal{M} = \{\mathcal{S}, \mathcal{A}, \mathcal{R}, \mathcal{T}, \rho_0, T\}$. $\mathcal{S} \subseteq \mathbb{R}^n$, $\mathcal{A} \subseteq \mathbb{R}^m$, and $\mathcal{R} : \mathcal{S} \times \mathcal{A} \to \mathbb{R}$ are (continuous) set of states, set of actions, and the reward function respectively. $\mathcal{T} : \mathcal{S} \times \mathcal{A} \to \mathcal{S}$ is the stochastic transition function; ρ_0 is the probability distribution over initial states; and T is the maximum episode length. We wish to solve for a stochastic policy of the form $\pi : \mathcal{S} \times \mathcal{A} \to \mathbb{R}$, which optimizes the average reward accumulated over the episode. Formally, the performance of a policy is evaluated according to:

$$\eta(\pi) = \frac{1}{T}\, \mathbb{E}_{\pi,\mathcal{M}} \left[\sum_{t=1}^{T} r_t \right]. \tag{1}$$

Preprint submitted to 2018 IEEE International Conference on Robotics and Automation. Received September 15, 2017.

In this finite horizon rollout setting, we define the value, Q, and advantage functions as follows:

$$V^{\pi}(s,t) = \mathbb{E}_{\pi,\mathcal{M}}\left[\sum_{t'=t}^{T} r_{t'}\right]$$

$$Q^{\pi}(s,a,t) = \mathbb{E}_{\mathcal{M}}\Big[\mathcal{R}(s,a)\Big] + \mathbb{E}_{s'\sim\mathcal{T}(s,a)}\Big[V^{\pi}(s',t+1)\Big]$$

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

We consider parameterized policies π_θ, and hence wish to optimize for the parameters (θ). Thus, we overload notation and use $\eta(\pi)$ and $\eta(\theta)$ interchangeably. In this work, we represent π_θ as a multivariate Gaussian with diagonal covariance. The mean of this distribution is represented by a feed-forward neural network, with a learnable parameter for the diagonal covariance matrix.

IV. Method

In this work, we use a combination of RL and imitation learning to solve complex dexterous manipulation problems. To reduce sample complexity and help with exploration, we collect a few expert demonstrations using the virtual reality system described in Section VI, and incorporate these into the RL process. We incorporate demonstrations in 2 ways - (1) providing better initialization for policy search, and (2) providing guidance for exploration by adding an auxiliary objective. We first outline the RL method and subsequently describe the incorporation of demonstrations.

A. RL algorithm: natural policy gradient

Policy gradient algorithms are a class of RL methods where the parameters of the policy are directly optimized typically using gradient based methods. Using the score function gradient estimator, the sample based estimate of the policy gradient can be derived to be: [42]:

$$\hat{g} = \frac{1}{NT}\sum_{i=1}^{N}\sum_{t=1}^{T}\nabla_\theta \log \pi_\theta(a_t^i|s_t^i)\hat{A}^{\pi}(s_t^i,a_t^i,t) \quad (2)$$

A straightforward gradient ascent using the above gradient estimate is the REINFORCE algorithm [42]. Gradient ascent with this direction is sub-optimal since it is not the steepest ascent direction in the metric of the parameter space [1]. Consequently, a local search procedure that follows the steepest path was proposed by Kakade [14], called the natural policy gradient (NPG), which has been expanded upon in subsequent works [3], [26], [25], [33]. Natural policy gradient is obtained by solving the following local optimization problem around iterate θ_k:

$$\begin{aligned} \underset{\theta}{\text{maximize}} \quad & g^T(\theta-\theta_k) \\ \text{subject to} \quad & (\theta-\theta_k)^T F_{\theta_k}(\theta-\theta_k) \le \delta, \end{aligned} \quad (3)$$

where F_{θ_k} is the Fisher Information Metric at the current iterate θ_k. We apply a normalized gradient ascent procedure, which has been shown to further stabilize the training process [25], [33], [29]. This results in the following update rule:

$$\theta_{k+1} = \theta_k + \sqrt{\frac{\delta}{g^T F_{\theta_k}^{-1} g}}\, F_{\theta_k}^{-1} g. \quad (4)$$

The version of natural policy gradient outlined above was chosen for simplicity and ease of implementation. The natural gradient performs covariant updates by rescaling the parameter updates according to curvature information present in the Fisher matrix, thus behaving almost like a second order optimization method. Furthermore, due to the normalized gradient procedure, the gradient information is insensitive to linear rescaling of the reward function, improving training stability.

B. Incorporating Demonstrations into Policy Gradients

We use demonstrations in the RL procedure to reduce sample complexity and alleviate the need for laborious task specific reward shaping. This is done in two main ways which are desribed below:

1) Pretraining with Behavior Cloning: A major challenge with RL methods is exploration, where multiple actions need to be tried to learn about it's consequences. If the policy is not initialized well, the learning process could be very slow since the algorithm would explore in state-action spaces that are not task relevant. To combat this, we use behavior cloning [28] to provide an informed policy initialization that efficiently guides exploration, and provide some gradient signal in the presence of sparse rewards. This idea has been explored in [8], and we show that this can dramatically reduce the sample complexity for dexterous manipulation tasks.

Behavior cloning corresponds to solving the following maximum-likelihood optimization problem:

$$\text{maximize}_\theta \sum_{(s,a^*)\in\rho_D} \ln \pi_\theta(a^*|s) \quad (5)$$

where ρ_D corresponds to the demonstrator (expert) trajectory distribution, with a^* denoting the action taken by the expert demonstrator. The optimizer of the above objective, called the behavior cloned policy, attempts to mimic the actions taken in the demonstrations at the states visited in those demonstrations.

In practice, behavior cloning does not guarantee that the cloned policy will be effective, due to the distributional shift between the demonstrated states and the policy's own states [30]. Indeed, we observed experimentally that the cloned policies themselves were usually not successful. However, initializing the weights for the mean of our gaussian policy using behavior cloning provides us with effective exploration for subsequent reinforcement learning. Furthermore, we initialize the stochasticity of the policy based on the standard deviation of actions in the demonstrations. This leads to exploration that is well informed by the demonstrations. In particular, dimensions which had low variance in the demonstrations will not be explored as much. In addition to guiding exploration, this also helps to reduce

variance, since action directions that are not task-relevant are not excited.

2) Demo Augmented Surrogate Function: Though behavior cloning provides a good initialization for RL, it does not optimally use the information present in the demonstration data. Different parts of the demonstration data are useful in different stages of learning, especially for tasks involving a sequence of behaviors. For example, the hammering task requires behaviors such as reaching, grasping, and hammering. Behavior cloning by itself cannot learn a policy that exhibits all these behaviors in the correct sequence with limited data. The result is that behavior cloning produces a policy that can often pick up the hammer but seldom swing it close to the nail. The demonstration data contains valuable information on how to hit the nail, but is lost when the data is used only for initialization. To capture all information present in the demonstration data, we add an auxiliary objective to the standard NPG update.

We consider a policy gradient that is a modification of (1) and uses an augmented loss as follows:

$$g_{aug} = \sum_{(s,a)\in\rho_\pi} \nabla_\theta \ln \pi_\theta(a|s) A^\pi(s,a) + \sum_{(s,a^*)\in\rho_D} \nabla_\theta \ln \pi_\theta(a^*|s) w(s,a^*) \tag{6}$$

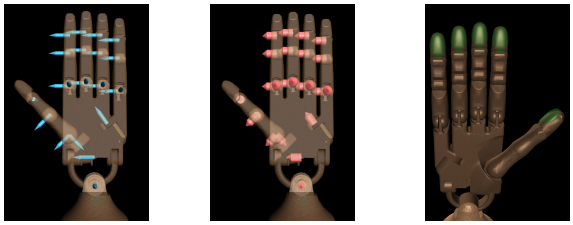
where $w(s,a)$ is a weighting function described below. If $w(s,a) = 0\ \forall(s,a)$ then we recover the policy gradient objective function. If $w(s,a) = c\ \forall(s,a)$ with sufficiently large c, it reduces to behavior cloning. We propose a mixture of the two to incorporate demonstrations into RL. This augmented gradient is then used in eq. (4) to perform a co-variant update. The analysis in [15] suggests that the policy gradient expression (2) is also valid for mixture trajectory distributions of the form $\rho = \alpha\rho_\pi + (1-\alpha)\rho_D$. Thus, a natural choice for the weighting function would be $w(s,a) = A^\pi(s,a)\ \forall(s,a) \in \rho_D$. However, it is not possible to compute the advantage for $(s,a) \in \rho_D$ without additional rollouts or assumptions [35]. Thus, we use the heuristic weighting scheme:

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

where λ_0 and λ_1 are hyperparameters, and k is the iteration counter. This is motivated by the premise that initially the actions suggested by the demonstrations are at least as good as the actions produced by the policy. However, towards the end when the policy is comparable in performance to the demonstrations, we do not wish to bias the gradient. Thus, we asymptotically decay the influence of the auxiliary objective.

V. Experimental setup

The real world presents a diversity of potential manipulation tasks. Solving individual tasks via custom task specific manipulators in a controlled setting has led to success in industrial automation. This is less feasible in an unstructured setting, such as the home, which provides a much broader diversity of tasks. A dexterous manipulator capable of handling task diversity and environmental complexity is needed. We use a simulated analogue of an anthropomorphic highly dexterous manipulator (ADROIT) [20]. Our experimental setup is modeled in MuJoCo [39], an efficient physics simulator which models contacts well - making it suitable for dexterous manipulation. The setup for collecting human demonstrations is through virtual reality, as described in Section VI.



(a) 24 Joints (blue arrows) (b) 24 Actuators (red arrows) (c) 5 Touch (green surface)

Fig. 2: 24 degree of freedom ADROIT hand

ADROIT Hand:

ADROIT is a 24 DoF anthropomorphic platform [20] designed for exploring and addressing challenges in dynamic and dexterous manipulation [19] [17]. The first, middle, and ring fingers have 4 DoF. Little finger and thumb have 5 DoF, while the wrist has 2 DoF. Each DoF is actuated using position control and is equipped with a joint angle sensor. The MuJoCo simulation of ADROIT quite accurately matches the kinematics, dynamics and sensing details of the physical hardware. In addition to dry friction in the joints, all hand-object contacts have planar friction. Object-fingertip contacts support torsion and rolling friction.

Tasks:

We consider four classes of tasks - object relocation, in hand manipulation, tool use and manipulating environmental props such as doors. These tasks are important since they represent of a large fraction of tasks which human hands perform in daily activities. They involve significant use of contacts and require application of accurate forces which are difficult to model.

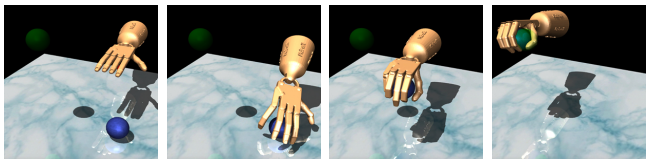


Fig. 3: Sequence of frames from a successful execution of the pickup task

1) Object relocation: Many manipulation tasks are variants of object relocation, where an object is picked up and moved to a target location. Such tasks are challenging because grasping the object itself requires establishing and maintaining a complex pattern of contacts, and the reward function is highly discontinuous.

In our version of this task, as shown in Figure 3, the goal is to use the ADROIT hand to pick up the blue ball and move it to the green target location. The positions of the ball and the target are randomized significantly, requiring the policy to generalize. The state $s_{relo} = [hand_{joints}; palm_{pos}, object_{pos}; object^{goal}_{pos}]$, success measure $\psi_{relo} = \mathbf{I}(||object_{pos} - object^{goal}_{pos}||_{l2} < 0.05)$, where $\mathbf{I}$ is the indicator function, and the reward

$$\begin{aligned} r_{relo} =& 10\mathbf{I}(||object_{pos} - object^{goal}_{pos}||_{l2} < 0.1) + \\ & 20\mathbf{I}(||object_{pos} - object^{goal}_{pos}||_{l2} < 0.05) \end{aligned} \tag{7}$$

Note that this reward is quite sparse, and could be applied to real world scenarios with minimal environment augmentation.

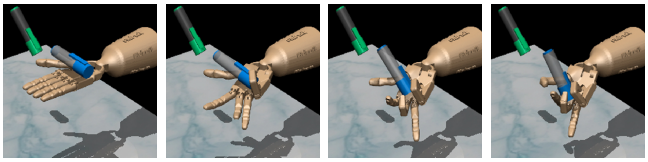


Fig. 4: Sequence of frames from a successful execution of the pen repositioning

2) In-hand Manipulation: Repositioning a pen: In hand-manipulation maneuvers like re-grasping, re-positioning, twirling objects, etc involves leveraging a high DoF manipulator's intrinsic dexterity to effectively navigate a difficult landscape filled with constraints and discontinuities imposed by joint limits and frequently changing contacts.

We consider the task of pen repositioning. As seen from Fig 4, the goal is to reposition the blue pen to a target position in-hand, visualized by the green pen. The base of the hand is fixed. The pen is highly underactuated and requires careful application of forces by the hand to reposition it. Most actions lead to catastrophic failure in terms of dropping the object. The goal configuration of the pen is significantly randomized across trials. The state $s_{pen} = [hand_{joints}; pen_{pos,rot}; pen^{goal}_{pos,rot}]$, success measure $\psi_{pen} = \mathbf{I}(||pen_{rot} - pen^{goal}_{rot}||_{cosine} > 0.95)$ and reward

$$\begin{aligned} r_{pen} =& 50(\mathbf{I}(||pen^{goal}_{pos} - pen_{pos}||_{l2} < 0.075) \otimes \\ & \mathbf{I}(||pen_{rot} - pen^{goal}_{rot}||_{cosine} > 0.95)) \end{aligned} \tag{8}$$

On this task, collection of human demonstrations is very challenging in the VR setup described in Section VI. Instead, to illustrate the effectiveness of our demonstration incorporation method, we used a computational neural network expert which has been trained with RL on a well shaped reward for many iterations. This expert serves to give demonstrations which are used to speed up training from scratch.

3) Tool Use: Hammer: Humans use tools such as hammers, levers, etc. to augment their capabilities. These tasks involve co-ordination between the fingers and the arm to apply the tool correctly. Unlike object relocation, these tasks are about trying to put the tool to use instead of just relocating it. This requires careful motor co-ordination to impart the required forces, and robust stabilization strategies capable of countering the destabilization responses.

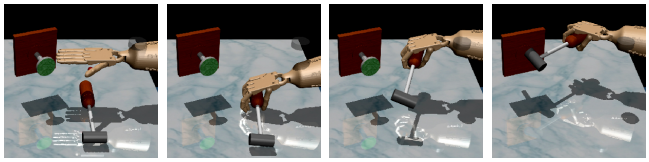


Fig. 5: Sequence of frames from a successful execution of the hammering nail task

We consider using a hammer to drive in a nail. As seen in Fig 5, the hand has to pick up the hammer from the ground, move it over to the nail and hammer in with a significant force to get the nail to move into the board. The nail has dry friction capable of absorbing up of 15N of force. There are more than one steps needed to perform this task, which require accurate grasping and positioning. The nail position is randomized. The state $s_{nail} = [hand_{joints,velocity}; palm_{pos}; hammer_{pos,rot}; nail^{goal}_{pos}; nail_{impactforce}]$, success measure $\psi_{nail} = \mathbf{I}(||nail_{pos} - nail^{goal}_{pos}||_{l2} < 0.01)$, and the reward

$$\begin{aligned} r_{nail} =& 75 * \mathbf{I}(||nail^{goal}_{pos} - nail_{pos}||_{l2} < 0.10) + \\ & 25 * \mathbf{I}(||nail^{goal}_{pos} - nail_{pos}||_{l2} < 0.02) - \\ & 10||nail^{goal}_{pos} - nail_{pos}||_{l2} \end{aligned} \tag{9}$$

Note that the reward function here depends only on the nail position relative to the final position in the board, and doesn't involve the position of the hammer or the hand.

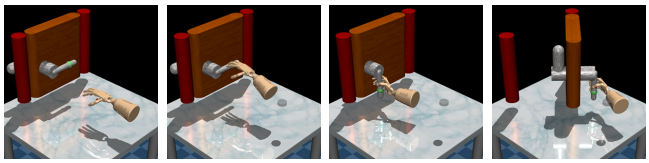


Fig. 6: Sequence of frames from a successful execution of the door opening task

4) Manipulating Environmental Props: Door Opening: Robots need to constantly interact with human-centric environments by modifying the environment itself - by opening drawers for fetching, moving furniture for cleaning, etc. The dynamics of these tasks are dominated by the inertial properties of the environment. The solution usually lies on a constrained manifold.

The specific task that we study is that of opening doors. As seen in Fig 6, the hand has to undo the latch before the door can be opened. The latch has a significant dry friction and a bias torque that forces the door to be closed. In addition to learning the semantics of opening the latch, learning agent needs to understand the solution manifold (the arc along with the door handle can move) once the the latch is undone. The door position is randomized significantly. The state $s_{door} = [hand_{joints}; palm_{pos}; door_{handle\ pos,latch,hinge}]$,

success measure $\psi_{door} = \mathbf{I}(door_{joint} > 1.4)$ and reward

$$r_{door} = 10\mathbf{I}(door_{pos} > 1.35) + 8\mathbf{I}(door_{pos} > 1.0) + 2\mathbf{I}(door_{pos} > 1.2) - 0.1||door_{pos} - 1.57||_{l2} \quad (10)$$

VI. Demonstrations

In order to incorporate demonstrations into our learning process as described in Section IV-B, we need a way to collect accurate demonstrations with the ADROIT hand. Standard methods like kinesthetic teaching are impractical with a complex systems such as ADROIT. We use the Mujoco HAPTIX system [18][1], which moves the process of demonstration data collection from the real world to virtual reality. This allows us to provide several high fidelity demonstrations for tasks involving large number of contacts and dynamic phenomena such as rolling, sliding, stick-slip, deformations and soft contacts. Since the demonstrations are provided in simulation, physically consistent details of the movements can be easily recorded. We gathered 25 successful demonstrations for all our tasks (with task randomization as outlined in Section V), with each demonstration consisting of state action trajectories needed to perform the task in simulation. To combat distribution drift, a small amount of noise (uniform random $[-0.1, 0.1]$ radians) is added to the actuators per timestep so that the policy can better capture relevant statistics about the data.

VII. Results

We aim to answer the following questions in our experimental evaluation

1) Can RL methods, and specifically the natural policy gradient, solve high dimensional dexterous manipulation tasks?
2) Can the use of demonstrations circumvent the manual reward shaping process, and enable RL to work with sparser reward signals?
3) Can sample complexity be reduced to a range that enables the possibility of real world execution?

In addition to answering these questions, we compare our method to several ablations - removing the demo augmented surrogate function, using NPG with and without reward shaping, and compared to a simplified version of DDPGfD. Our simplified DDPGfD implementation was based on an open source implementation of DDPG[2] modified to add demonstrations to the replay buffer. We found that this method was unable to make progress on any of the tasks considered in this work. This suggests that off-policy learning methods are more sensitive to hyperparameter choices, especially in tasks with high dimensional action spaces.

Our core results are summarized in Fig. 7 and Table I. The various shaped reward functions we use are outlined in

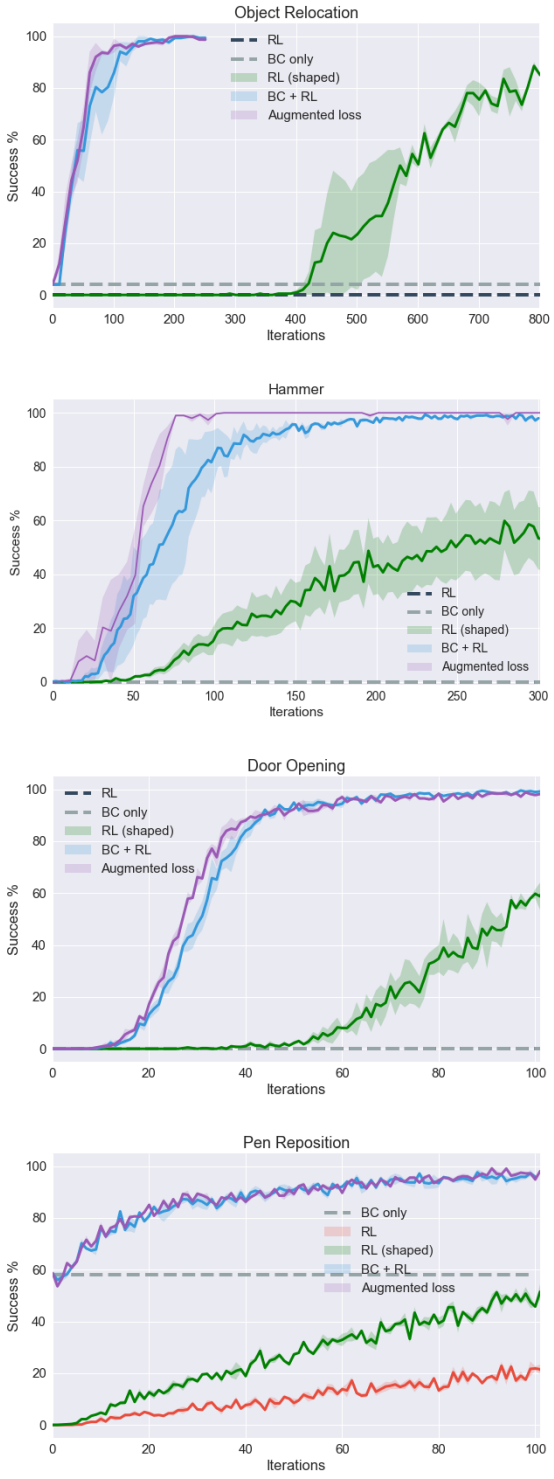


Fig. 7: Learning curves for the different hand manipulation tasks. We observe that use of demonstrations to initialize the policy consistently outperforms standard RL approaches, and also work with sparse rewards. Incorporation of the augmented loss consistently improves performance as well. Notably the sample complexity is reduced to practical scales. Unless noted otherwise, in these plots, all rewards are sparse as indicated in the equations (7)-(10)

[1]We used an upgraded version of the originally described system. The new set up uses the CyberGlove III system for recording the fingers, HTC vive tracker for tracking the base of the hand and HTC vive headset for stereoscopic visualization.

[2]OpenAI baselines

equations (11)-(14). The results qualitatively indicate that RL from scratch with sparse rewards is unable to make effective progress on the tasks we consider, except the pen repositioning task. Tasks requiring picking up or reaching objects present a different set of challenges than in-hand manipulation tasks. In particular, the former tasks are hard due to exploration. In-hand tasks on the other hand do not suffer from exploration challenges, but on the other hand, require expressive function approximators that can utilize the available reward signals.

In the object relocation task, we see that RL with sparse reward cannot make any progress at all due to the exploration challenges outlined before. With a shaped reward (eq. 11) that tries to circumvent the exploration challenges, we see that RL works but requires a large number of iterations (samples) as seen in Fig 7. On the other hand, initializing the RL process with the behavior cloned policy dramatically reduces the sample complexity. Use of the augmented loss improves the learning process even further. Along similar lines, RL with sparse reward is unable to solve the hammer or door tasks, which provides a similar challenge of grasping the hammer or reaching and turning the door handle. As indicated in Section IV, valuable information of how to perform sub-tasks that occur later in the sequence (e.g. opening the door, after reaching it) is lost when the demonstration data is used only for behavior cloning initialization. Consequently, the augmented loss increases the learning speed noticeably (Fig. 7). When provided with a suitably shaped reward as indicated in eqn (11) and eqn (12), we see that NPG is able to learn, however significantly slower than our method incorporating demonstrations. For the in-hand task, we see that behavior cloning performs surprisingly well achieving more than 50% success. We suspect that using a computational expert provides action targets that are easy to approximate compared to human expert demonstrations. The general theme that initializing the policy with demonstrations speeds up the learning process is also observed here, but since the improvements with behavior cloning initialization are dramatic, the additional benefit of the augmented loss seems minimal for this task. It is important to note that without significant reward shaping, RL is unable to make any progress on door opening, hammering or object relocation tasks and is very sample inefficient for pen repositioning. Our method is able to learn much faster although it is using sparse rewards.

The learning curves in Fig 7 and sample complexity results in Table I correspond to the same hyper-parameter setting for all the algorithm variants considered. RL from scratch often requires larger batch sizes, since initially, in addition to the gradients being small (low task success rate), they are also more noisy. The augmented loss requires smaller batch sizes since the demonstration data acts as additional samples and augmented the batch size. However, for the sake of objective comparisons, we have used the same hyperparameter settings for all the algorithms.

TABLE I: Sample and robot time complexity with the proposed algorithm (augmented loss + BC initialization) compared to just natural policy gradient with shaped (sh) and sparse rewards (sp). N is the number of RL iterations needed to achieve 95% success rate, Hours represent the robot hours needed to learn the task. Each iteration is 200 trajectories of length 2 seconds each.

Method	Ours		RL (sh)		RL(sp)	
Task	N	Hours	N	Hours	N	Hours
Obj 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

$$\begin{aligned} r^{shaped}_{relo} =& r_{relo} - 0.1||palm_{pos} - obj_{pos}||_{l2} + \\ & \mathbf{I}(obj_z > 0.04)(1.0 - 0.5||palm_{pos} - obj^{goal}_{pos}||_{l2} \\ & - 0.01||obj_{pos} - obj^{goal}_{pos}||_{l2}) \end{aligned} \tag{11}$$

$$\begin{aligned} r^{shaped}_{pen} =& r_{pen} - ||pen^{goal}_{pos} - pen_{pos}||_{l2} + \\ & ||pen_{rot} - pen^{goal}_{rot}||_{cosine} + \\ & 10\mathbf{I}(||pen_{rot} - pen^{goal}_{rot}||_{cosine} > 0.9) - \\ & 5\mathbf{I}(pen_z < 0.15) \end{aligned} \tag{12}$$

$$\begin{aligned} r^{shaped}_{nail} =& r_{nail} - 0.1||palm_{pos} - nail_{pos}||_{l2} - \\ & 0.1||hand_{jointvelocity}||_{l2} + \\ & 2\mathbf{I}(hammer_z > 0.04) \end{aligned} \tag{13}$$

$$r^{shaped}_{door} = r_{door} - ||palm_{pos} - handle_{pos}||_{l2} \tag{14}$$

VIII. Conclusion

In this work, we presented an application of RL methods, specifically policy gradients, to challenging dexterous manipulation tasks. We show that natural policy gradient methods are able to scale effectively to very high dimensional spaces and perform complex tasks when provided with well shaped reward functions. In order to take steps towards making these algorithms practical for real world application, we introduce a method to incorporate demonstrations into natural policy gradient - both through policy initialization with behavior cloning and through augmenting the objective with an auxiliary function. We find that incorporating a small number of demonstrations into RL in this way speeds up the learning process very significantly upto 30x over RL with shaped rewards, and allows NPG to work with much sparser rewards (where NPG doesn't work at all) in a significantly smaller number of samples. The estimated robot time that these dexterous manipulation tasks would take to learn with our method is on the order of 5 hours, which is practical to run on real systems. An interesting and meaningful effect of incorporating demonstrations is that the resulting behavior is much more "human-like" than the behavior obtained by simply running RL, which is very

desirable for robotic applications. Given the complexity of the tasks in our evaluation and the sample-efficiency of our demonstration-accelerated DRL method, we believe that our work provides a significant step toward practical real-world learning of complex dexterous manipulation. In future work, we hope to learn policies on real hardware systems, further reduce sample complexity by using novelty based exploration methods and learn policies from only raw visual inputs.

REFERENCES

[1] S. Amari. Natural gradient works efficiently in learning. *Neural Computation*, 10:251–276, 1998.

[2] H. B. Amor, O. Kroemer, U. Hillenbrand, G. Neumann, and J. Peters. Generalization of human grasping for multi-fingered robot hands. In *IROS 2012*.

[3] J. A. Bagnell and J. G. Schneider. Covariant policy search. In *IJCAI, 2003*, pages 1019–1024. Morgan Kaufmann, 2003.

[4] Y. Bai and C. K. Liu. Dexterous manipulation using both palm and fingers. In *Robotics and Automation (ICRA), 2014 IEEE International Conference on*, pages 1560–1565. IEEE, 2014.

[5] T. Brys, A. Harutyunyan, H. B. Suay, S. Chernova, M. E. Taylor, and A. Nowé. Reinforcement learning from demonstration through shaping. In *IJCAI*, pages 3352–3358, 2015.

[6] R. Deimel and O. Brock. A novel type of compliant and underactuated robotic hand for dexterous grasping. *I. J. Robotics Res.*, 35(1-3):161–185, 2016.

[7] Y. Duan, X. Chen, R. Houthooft, J. Schulman, and P. Abbeel. Benchmarking deep reinforcement learning for continuous control. In *ICML*, 2016.

[8] D. S. et al. Mastering the game of go with deep neural networks and tree search. *Nature*, 529, 2016.

[9] A. Ghadirzadeh, A. Maki, D. Kragic, and M. Björkman. Deep predictive policy training using reinforcement learning. *CoRR*, abs/1703.00727, 2017.

[10] S. Gu, E. Holly, T. P. Lillicrap, and S. Levine. Deep reinforcement learning for robotic manipulation with asynchronous off-policy updates. In *ICRA 2017*.

[11] A. Gupta, C. Eppner, S. Levine, and P. Abbeel. Learning dexterous manipulation for a soft robotic hand from human demonstrations. In *IROS 2016*.

[12] N. Heess, D. TB, S. Sriram, J. Lemmon, J. Merel, G. Wayne, Y. Tassa, T. Erez, Z. Wang, S. M. A. Eslami, M. A. Riedmiller, and D. Silver. Emergence of locomotion behaviours in rich environments. *CoRR*, abs/1707.02286, 2017.

[13] T. Hester, M. Vecerik, O. Pietquin, M. Lanctot, T. Schaul, B. Piot, A. Sendonaris, G. Dulac-Arnold, I. Osband, J. Agapiou, J. Z. Leibo, and A. Gruslys. Learning from demonstrations for real world reinforcement learning. *CoRR*, abs/1704.03732, 2017.

[14] S. Kakade. A natural policy gradient. In *NIPS*, 2001.

[15] S. Kakade and J. Langford. Approximately optimal approximate reinforcement learning. In *ICML 2002*.

[16] J. Kober and J. Peters. Learning motor primitives for robotics. In *ICRA*, pages 2112–2118. IEEE, 2009.

[17] V. Kumar, Y. Tassa, T. Erez, and E. Todorov. Real-time behaviour synthesis for dynamic hand-manipulation. In *ICRA*, pages 6808–6815. IEEE, 2014.

[18] V. Kumar and E. Todorov. Mujoco haptix: A virtual reality system for hand manipulation. In *Humanoids*. IEEE, 2015.

[19] V. Kumar, E. Todorov, and S. Levine. Optimal control with learned local models: Application to dexterous manipulation. In *ICRA 2016*.

[20] V. Kumar, Z. Xu, and E. Todorov. Fast, strong and compliant pneumatic actuation for dexterous tendon-driven hands. In *ICRA*, pages 1512–1519. IEEE, 2013.

[21] T. P. Lillicrap, J. J. Hunt, A. Pritzel, N. Heess, T. Erez, Y. Tassa, D. Silver, and D. Wierstra. Continuous control with deep reinforcement learning. *CoRR*, abs/1509.02971, 2015.

[22] C. K. Liu. Dextrous manipulation from a grasping pose. In *ACM Transactions on Graphics*, volume 28, page 59. ACM, 2009.

[23] A. T. Miller and P. K. Allen. Graspit! a versatile simulator for robotic grasping. *IEEE Robotics & Automation Magazine*, 2004.

[24] I. Mordatch, Z. Popović, and E. Todorov. Contact-invariant optimization for hand manipulation. In *ACM SIGGRAPH/Eurographics symposium on computer animation*, pages 137–144. Eurographics Association, 2012.

[25] J. Peters. Machine learning of motor skills for robotics. *PhD Dissertation, University of Southern California*, 2007.

[26] J. Peters and S. Schaal. Natural actor-critic. *Neurocomputing*, 71:1180–1190, 2007.

[27] J. Peters and S. Schaal. Reinforcement learning of motor skills with policy gradients. *Neural Networks*, 21(4):682–697, 2008.

[28] D. Pomerleau. ALVINN: an autonomous land vehicle in a neural network. In *NIPS 1988]*, pages 305–313, 1988.

[29] A. Rajeswaran, K. Lowrey, E. Todorov, and S. Kakade. Towards generalization and simplicity in continuous control. *ArXiv e-prints*, 2017.

[30] S. Ross, G. J. Gordon, and D. Bagnell. A reduction of imitation learning and structured prediction to no-regret online learning. In *AISTATS 2011*, volume 15 of *JMLR*, pages 627–635, 2011.

[31] A. A. Rusu, M. Vecerik, T. Rothörl, N. Heess, R. Pascanu, and R. Hadsell. Sim-to-real robot learning from pixels with progressive nets. *CoRR*, abs/1610.04286, 2016.

[32] J. Schulman, S. Levine, P. Abbeel, M. I. Jordan, and P. Moritz. Trust region policy optimization. In *Proceedings of the 32nd International Conference on Machine Learning, ICML 2015, Lille, France, 6-11 July 2015*, volume 37 of *JMLR Workshop and Conference Proceedings*, pages 1889–1897. JMLR.org, 2015.

[33] J. Schulman, S. Levine, P. Moritz, M. Jordan, and P. Abbeel. Trust region policy optimization. In *ICML*, 2015.

[34] K. Subramanian, C. L. I. Jr., and A. L. Thomaz. Exploration from demonstration for interactive reinforcement learning. In *International Conference on Autonomous Agents & Multiagent Systems, Singapore, May 9-13, 2016*, pages 447–456. ACM, 2016.

[35] W. Sun, A. Venkatraman, G. J. Gordon, B. Boots, and J. A. Bagnell. Deeply aggrevated: Differentiable imitation learning for sequential prediction. In *ICML 2017*.

[36] R. S. Sutton and A. G. Barto. *Reinforcement learning: An introduction*, volume 1.

[37] M. E. Taylor, H. B. Suay, and S. Chernova. Integrating reinforcement learning with human demonstrations of varying ability. In *AAMAS 2011*.

[38] E. Theodorou, J. Buchli, and S. Schaal. Reinforcement learning of motor skills in high dimensions: A path integral approach. In *ICRA 2010*.

[39] E. Todorov, T. Erez, and Y. Tassa. MuJoCo: A physics engine for model-based control. In *ICRA*, 2012.

[40] H. van Hoof, T. Hermans, G. Neumann, and J. Peters. Learning robot in-hand manipulation with tactile features. In *Humanoid Robots (Humanoids)*. IEEE, 2015.

[41] M. Vecerik, T. Hester, J. Scholz, F. Wang, O. Pietquin, B. Piot, N. Heess, T. Rothörl, T. Lampe, and M. A. Riedmiller. Leveraging demonstrations for deep reinforcement learning on robotics problems with sparse rewards. *CoRR*, abs/1707.08817, 2017.

[42] R. J. Williams. Simple statistical gradient-following algorithms for connectionist reinforcement learning. *Machine Learning*, 8(3):229–256, 1992.

[43] Z. Xu, V. Kumar, and E. Todorov. A low-cost and modular, 20-dof anthropomorphic robotic hand: design, actuation and modeling. In *Humanoid Robots 2013*.

[44] Z. Xu and E. Todorov. Design of a highly biomimetic anthropomorphic robotic hand towards artificial limb regeneration. In *ICRA 2016*.