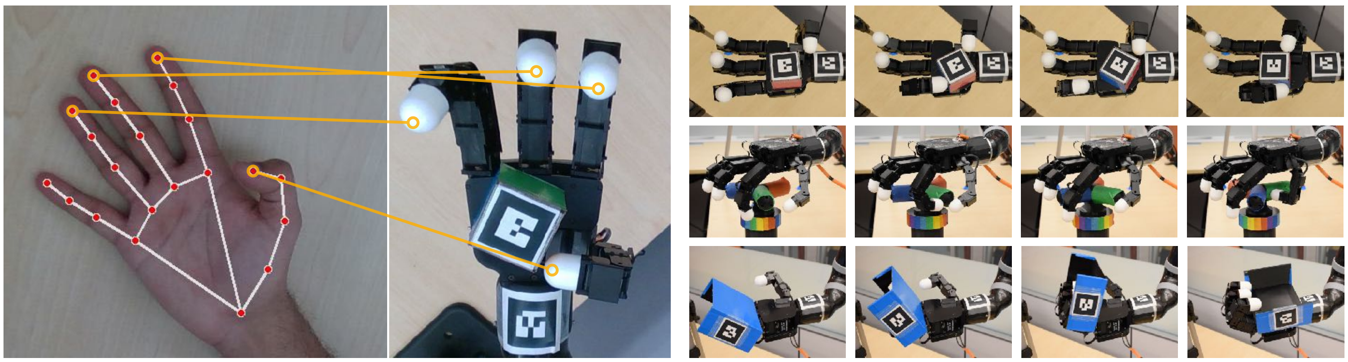
Dexterous Imitation Made Easy: A Learning-Based Framework for Efficient Dexterous Manipulation

Sridhar Pandian Arunachalam†, Sneha Silwal†, Ben Evans, and Lerrel Pinto
New York University



(a) Teleoperation through a single RGB camera.

(b) Learned policies for dexterous manipulation.

Fig. 1: The framework for dexterous manipulation consists of two phases. (a) Demonstrations are collected using a real-time hand tracker on a single visual stream of a human operator's hand. The estimated fingertip 2D pixel coordinates are retargeted to 3D coordinates in the robot frame. (b) Given these demonstrations, dexterous manipulation policies are learned on both a real Allegro Hand, using nearest neighbor-based imitation, and on a simulated Allegro Hand using RL augmented with demonstrations.

***Abstract*— Optimizing behaviors for dexterous manipulation has been a longstanding challenge in robotics, with a variety of methods from model-based control to model-free reinforcement learning having been previously explored in literature. Perhaps one of the most powerful techniques to learn complex manipulation strategies is imitation learning. However, collecting and learning from demonstrations in dexterous manipulation is quite challenging. The complex, high-dimensional action-space involved with multi-finger control often leads to poor sample efficiency of learning-based methods. In this work, we propose 'Dexterous Imitation Made Easy' (DIME) a new imitation learning framework for dexterous manipulation. DIME only requires a single RGB camera to observe a human operator and teleoperate our robotic hand. Once demonstrations are collected, DIME employs standard imitation learning methods to train dexterous manipulation policies. On both simulation and real robot benchmarks we demonstrate that DIME can be used to solve complex, in-hand manipulation tasks such as 'flipping', 'spinning', and 'rotating' objects with the Allegro hand. Our framework along with pre-collected demonstrations will be made publicly available at: `https://nyu-robot-learning.github.io/dime`.**

I. Introduction

The ability to dexterously manipulate objects with multi-fingered hands has been crucial to the development of general-purpose manipulation in humans [1, 2, 3]. However, multi-finger control in robots requires complex contact-rich interactions, achieved through high-dimensional actions. Due to this, most of our robots today often only employ primitive end-effectors [4], which significantly limits their dexterity. To address this gap in dexterity, early works focused on developing physics-informed controllers that required precise modeling of the object-hand interaction [5, 6, 7]. Since such an ability to model the world may not be present in real-world scenarios, more recently, learning-based approaches have shown promise for general-purpose dexterity.

Over the last few years, we have witnessed several impressive results in the use of large-scale reinforcement learning (RL) for dexterous manipulation. For instance, through extensive simulator modeling, domain randomization, and millions of samples of training, behaviors such as cube rotation and Rubik's cube manipulation were demonstrated on the Shadow Hand [8, 9]. However, such model-free RL techniques often require manual reward design along with several weeks of training on industry-scale compute. This begs the question – can we learn dexterous behaviors in a sample-efficient manner?

Perhaps the most sample-efficient way to learn robotic skills is imitation learning [10, 11]. Here, given a handful of demonstrations recorded by a human operator, the robot is tasked to imitate that behavior. So why not use imitation learning for dexterous manipulation? The key challenge is that obtaining demonstrations for high-dimensional systems is quite challenging – kinesthetic teaching [12] requires significant conditioning on the robot; custom-built cyber gloves [13] are expensive; visual piloting [14] requires registration and calibration of a multi-camera system.

In this work, we present Dexterous Imitation Made Easy (DIME), a new robotic system for both collecting and learning from visual demonstrations. Given visual inputs

† denotes equal contribution. Correspondence to {`sridhar`, `ssilwal`}`@nyu.edu`.

from a single RGB camera, DIME enables human operators to teleoperate multi-fingered robotic hands that can produce high-quality demonstrations for dexterous manipulation (See Fig. 1(a)). Unlike prior frameworks, collecting demonstrations with DIME requires minimal calibration and human training. This is achieved by using off-the-shelf hand pose detectors [15] to obtain fingertip positions that are then retargeted and fed to fingertip controllers on our robotic hand. Across our experiments, collecting a demonstration takes around 100 seconds on average.

Given these demonstrations, DIME is then tasked with learning policies to solve desired dexterous tasks. For this, we investigate two broad settings – imitation in simulation and imitation on robot hardware. In simulation, we show that standard model-free RL combined with imitation learning [16] is able to train dexterous manipulation policies in about 2 days of simulated training time. While on the real robot, we demonstrate how non-parametric, nearest-neighbor learning [17, 18] can achieve high performance on tasks immediately (See Fig. 1(a)). This shows that the demonstrations obtained from DIME are versatile across imitation learning paradigms in simulation and on real robots.

In summary, this paper presents DIME, a framework that enables efficient dexterous manipulation through imitation. Concisely, our three primary contributions are: (a) We have developed an easy-to-use teleoperation framework for dexterous manipulation that can be used with untrained human operators. (b) We demonstrate the demonstrations obtained from DIME are compatible with state-of-the-art imitation learning algorithms. (c) We empirically study the interaction of imitation learning techniques with DIME and successfully solve dexterous manipulation tasks such as 'flipping', 'turning', and 'rotating' objects. To the best of our knowledge, DIME is the first work to successfully train dexterous manipulation policies using inexpensive demonstrations.

II. Related Work

Our framework builds on top of several important works in collecting demonstrations, hand tracking, imitation learning, and reinforcement learning. In this section, we describe prior research that is relevant to our work.

A. Obtaining Robot Demonstrations

Many methods have been proposed to collect demonstrations, which can accelerate the learning of complex robotic tasks. Kinesthetic training involves a human physically guiding the robot to complete a task [12, 19, 20, 21]. Although this is a powerful technique for providing demonstrations to robotic arms, they are difficult to use for multi-fingered hands due to the significantly larger action space. Virtual reality-based teleoperation [22], or the use of assistive tools [23, 24] have been successful for manipulation tasks. However, again they have not been shown to be useful for challenging dexterous tasks. Perhaps the most commonly used method to obtain demonstrations for dexterous hands is the use of a CyberGlove [13]. Here, a custom-made glove is used to precisely measure a human operator's hand movements, which is then transferred onto a real robot. However, such a system is expensive and requires calibration before being run. More recently the DexPilot system [14] has shown how dexterous demonstrations can be produced without a Cyberglove by instead using a rig of RGBD camera that estimate the operator's hand pose. Our framework DIME is inspired by this work and builds on top of it by alleviating the need for multiple cameras and the associated challenges of registration and calibration.

B. Vision-Based Hand Tracking

There has been a recent push in the computer vision community to detect and estimate the pose of human hands using image [25, 15, 26] and depth-based [27, 28]. Obtaining hand estimates from a single camera provides a significant advantage over expensive gloves [13] or precisely-calibrated camera rigs [14]. We however note that single-camera approaches can run into unobservability due to occlusions. In our experiments, we find that our human operators avoid occlusion regions and can hence still solve dexterous tasks. Most similar to our methodology of collecting demonstrations, recent work [29] has shown how the FrankMocap [30] pose estimator can be used to teleoperate the Allegro hand. DIME builds on top of this work by collecting more dexterous demonstrations and training behavior policies using them. Instead of FrankMocap, DIME uses the MediaPipe hand detector [15], a real-time hand tracking pipeline that produces a hand skeleton from a single RGB camera. This detector is trained on a variety of hand poses across synthetic, wild, and in-house datasets to detect 21 keypoints on the hand in a 2.5D coordinate space. Coupled with GPU acceleration, this allows for more accurate and real-time pose estimates compared to other hand detectors.

C. Imitation Learning

Imitation learning is a technique to learn a policy from expert demonstrations. Behavior Cloning (BC), encourages a model to mimic expert actions by learning over a dataset in a supervised learning fashion and has been applied to a wide range of problems [31, 32, 33]. It is well-known that the performance of models suffers when the test data distribution shifts away from the training domain, even in the linear case [34]. DAgger [35] attempts to overcome this problem by allowing query access to an expert policy during training, something not necessarily feasible for all problems. Another approach to imitation is Inverse Reinforcement Learning (IRL), where the underlying reward function is explicitly inferred [36, 37]. Following this, a policy is trained to maximize this reward function. In the context of dexterous manipulation, such IRL approaches often require extensive online training, which is not always possible for real-robotic applications. Instead, for DIME, we use a non-parametric, nearest-neighbor approach [18] to map observations to actions due to its simplicity, bounded action space, and empirical success on manipulation tasks.

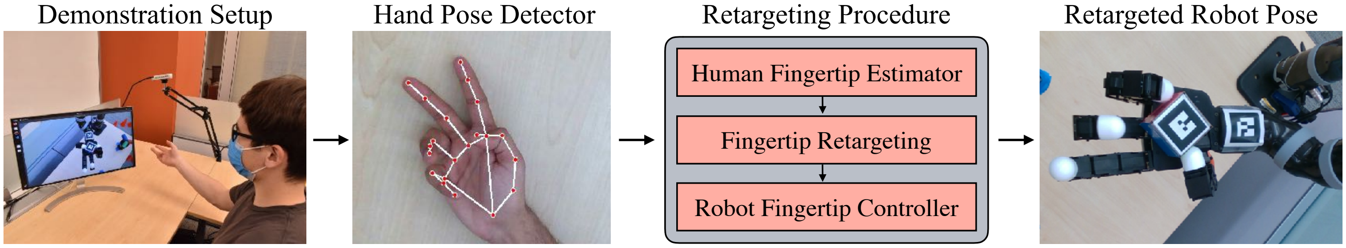


Fig. 2: Overview of the teleoperation framework in DIME. Given RGB streams of a human operator's hand, a hand pose detector followed by a retargeting procedure is used to control the fingertips of the robot's hand. Visual feedback of the teleoperated actions is then provided back to the operator for real-time teleoperation.

D. Finetuning Imitation with Reinforcement Learning

While there has been some success in using pure RL for learning policies for dexterous tasks, they are limited to simple behaviors [38], by the use of perfect models [39], or require years of simulated time for a single task [40]. Incorporating external data alongside reinforcement learning algorithms has been proposed for off-task data [41, 42] and with expert task-specific data by augmenting off-policy replay buffers and incorporating additional demonstration-based loss terms [43, 44]. Most relevant to our work is the DAPG algorithm [16], which has demonstrated that RL can be used to finetune behavior cloning imitation policies on a simulated Adroit hand. Similarly, we have found DAPG to be a versatile method that is able to solve imitation learning problems on a simulated Allegro hand.

III. Imitation for Dexterity

A. Overview

Our system, 'Dexterous Imitation Made Easy' (DIME) consists of two phases – teleoperation and imitation. In the teleoperation phase, human operators control a four-fingered Allegro hand through a visual RGB stream of their hand. This is achieved by estimating the human's hand pose, determining their fingertip locations, and retargeting them to the robot's fingertips. The robot then uses an inverse-kinematics based controller to reach desired fingertip positions. During this process, the human operator is shown real-time visual feedback of their actions. This feedback allows the operator to correct for errors and produce high-quality demonstrations for dexterous manipulation tasks.

Once a desired number of demonstrations are obtained, we begin the imitation phase. Here we investigate two settings, learning in simulation and learning in real. In the simulation setting, demonstrations are collected by teleoperating a simulated model of the Allegro hand. While in the real-robot setting, demonstrations are collected directly on the real Allegro hand. Each of these settings affords us the ability to use different learning algorithms. Simulation learning allows for sample-complex policy gradient approaches that can correct for noise in the demonstrations, while real-robot learning allows for sample-efficient non-parametric approaches. We do not run policy gradient-based learning on our real robot due to the sample complexity of such methods along with preventing accidental damage to the robot. Nevertheless, we believe that the policy gradient experiments in simulation show that DIME can be used to accelerate model-free RL as well. Details of both phases of DIME and their respective components are presented next.

B. Hand Pose Mapping

To map human hand positions to the robot, we use a single RGB camera and the MediaPipe hand detector to extract 2.5D landmarks of an operator's hand [15]. Rather than inferring and mapping the rotational state for each joint on the human hand to the robot, we map directly from 2.5D space to 3D. Because the model does not output absolute depth estimates, we choose to ignore the depth and treat the fingertips as if they were on a plane above the palm. We then map directly from human hand fingertip locations in 2D to robot fingertip locations in 3D space, keeping the robot fingertips at a fixed height above the palm. We map the index, middle, and ring fingers along the y plane. Since the Allegro Hand only has four fingers, we use the pinky finger for finer control. We define a quadrilateral bound with which the human thumb can be detected and re-mapped to control the xy movement of the robot thumb.

Our mapping is easy to calibrate and only requires moving the hand to a sequence of reference positions at the extremes of the finger positions. We take the positions of both the human and robot hands at the extrema and linearly interpolate to produce desired target 3D positions for intermediate locations in space. The desired 3D positions are then fed to a robot controller which is described in the next section.

C. Allegro Hand Controller

Given desired fingertip locations in 3D space, we compute the required joint angles using a model of the robot and an inverse kinematics solver. For every joint, we compute the torque required to compensate for gravity and the control torque from a PD controller to reach the desired joint angle. This control loop is run at 300Hz, while desired positions are streamed at 30Hz. We use ROS [45] to facilitate communication between the hand detector and robot. This framework is hence compatible with remote teleoperation once the remote machine is registered on our ROS network.

D. Demonstration Collection

To collect demonstrations, we run the hand pose detection and mapping in real-time on a single desktop computer.

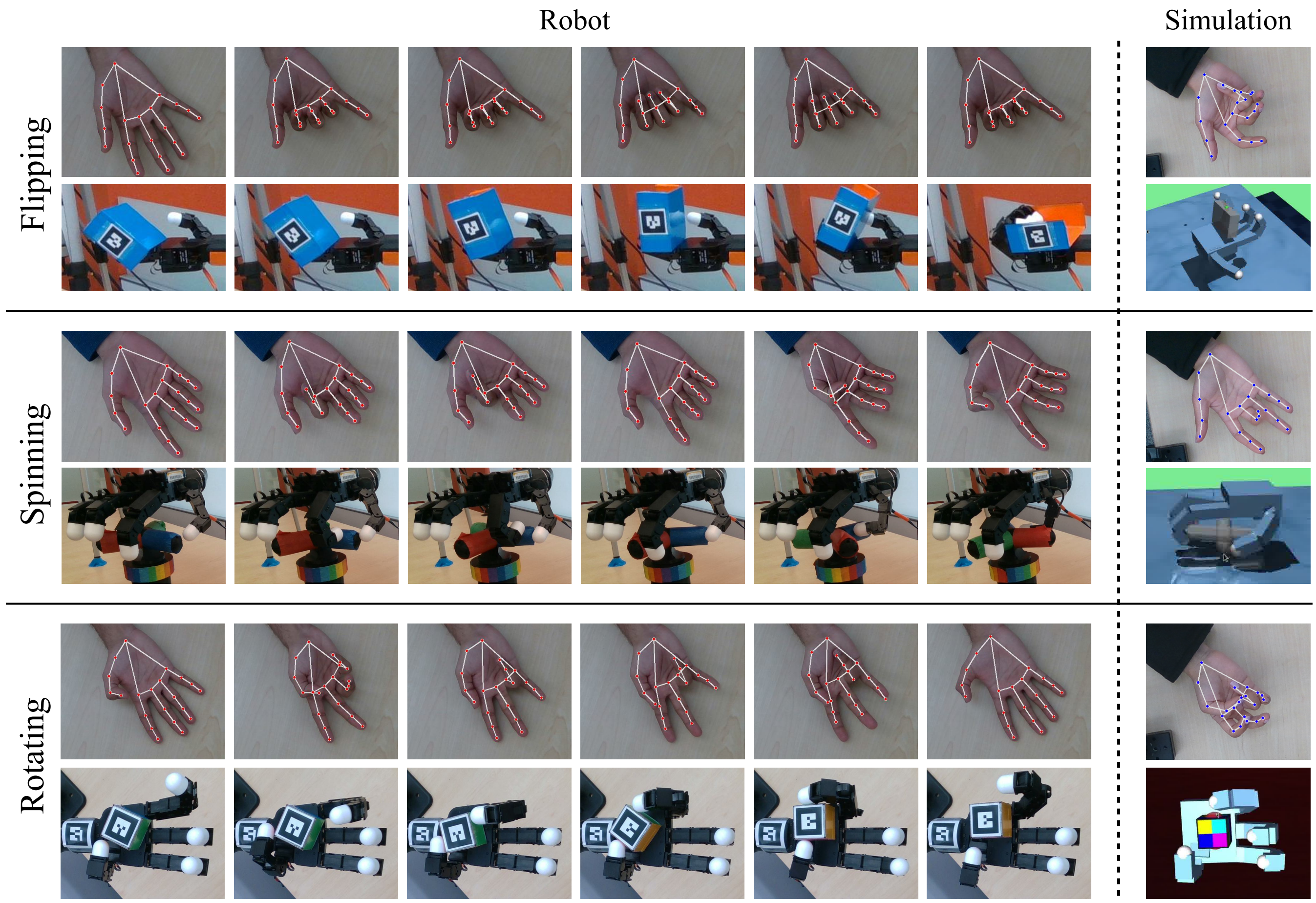


Fig. 3: The demonstration collection process for the three tasks. For each task, the operator's hand in the upper row is depicted followed by the corresponding state of the robot's hand in the lower row. The rightmost column visualizes the operator's actions during demonstration collection along with the simulated MuJoCo environments for each task.

We record an image of the Allegro Hand, the state of the object, and the calculated target fingertip locations at 5 Hz. To estimate the state of objects we use ROS's AR tracking library. The robotic system can hence be described by the 16 joint angles of the Allegro Hand along with the 3D position and rotation of the object. For state-based imitation, we use an observation space consisting of the 3D positions of all 4 fingertips and the 3D position of the object with respect to the wrist joint. For image-based imitation, we use the RGB images of the Allegro Hand. The action space is the 3D position of the 4 desired fingertip locations, also relative to the wrist. We exclude transitions with changes less than 2 centimeters for imitation learning on the robot. This is done to account for the human operator pausing intermittently while collecting demonstrations. To study the usefulness of collected demonstrations, DIME integrates demonstration collection for both simulated hand control and real hand control. Examples of demonstrations can be seen in Fig. 3.

Given a set of demonstrations collected through DIME, our framework for dexterous imitation studies two types of imitation learning algorithms – RL finetuning for our simulated hand and non-parametric learning for our real hand. In the following sections we describe the algorithms.

E. Reinforcement Learning in Simulation

To imitate from demonstrations collected in simulation, we use Demonstration-Augmented Policy Gradients (DAPG) [16], a powerful imitation based algorithm for dexterous manipulation. DAPG incorporates demonstrations into the reinforcement learning procedure, which has been shown to greatly accelerate learning on a suite of difficult dexterous manipulation tasks with a 24 degree-of-freedom Adroit hand. They do so by augmenting the standard policy gradient with a weighted behavior cloning gradient term,

$$\begin{aligned} g_{aug} = \sum_{(s,a)\in\rho_\pi} & [\nabla_\theta \log \pi_\theta(a|s) A^\pi(s,a)] + \\ & \sum_{(s,a)\in\rho_D} [\nabla_\theta \log \pi_\theta(a|s) w(s,a)] \end{aligned} \tag{1}$$

where ρ_π, ρ_D are the state-action distributions generated by the policy and demonstrations, respectively, and $w(s,a)$ is an exponential weighting function that decays to zero over the course of training. The dynamic weighting term allows the policy to quickly learn actions useful to the task and the resulting policy is able to solve tasks where standard RL fails with greater robustness to variations in mass and geometry than standard RL methods. We choose to use this

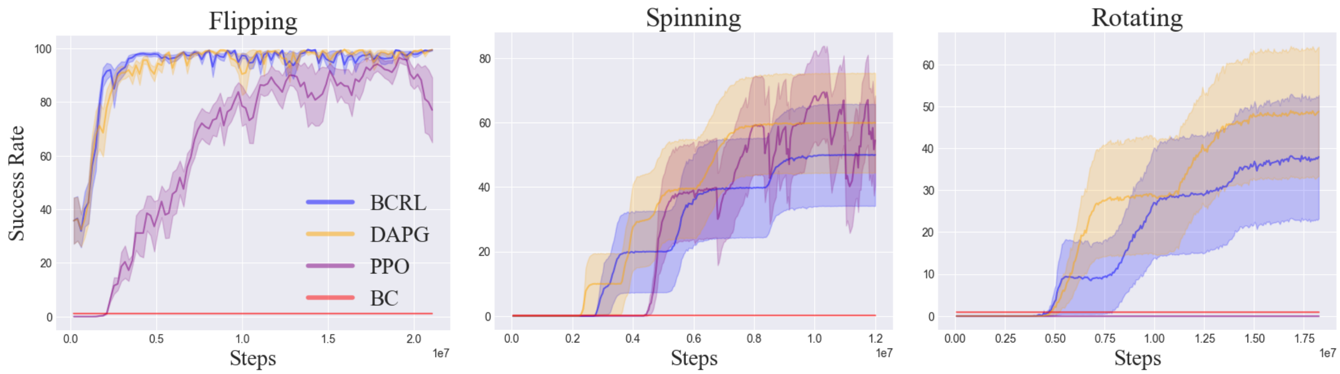


Fig. 4: Success curves on simulated control for various RL and behavior cloning approaches with the shaded region indicating ± 1 standard error measured across 10 seeds. For all tasks behavior cloning finetuned with RL (BCRL & DAPG) achieve high success rates.

method due to its simplicity and accelerated learning over prior policy-gradient approaches [43, 46, 47].

F. Imitation on the Robot

Although DAPG can quickly learn dexterous skills in simulation, it would still require online training on the order of several days for real-robot training. Hence to study the usefulness of DIME for real robots, we turn to more sample-efficient non-parametric imitation methods. The simplest method in this class of algorithms is k-nearest neighbors, which takes the current input, finds the k closest inputs in the training dataset, and averages the outputs to produce a prediction [48]. Locally weighted regression is a similar non-parametric method that weights the outputs by a similarity metric in the input space and has been used successfully for state-based [17] (INN) and image-based [18] (VINN) robot learning tasks. For state-based nearest neighbors, we use the 15 dimensional observation space containing the 3D fingertip positions for each of the 4 fingers and the 3D position of the object. For image-based nearest neighbors, we use 2048 dimensional feature representations of the input image obtained from an encoder trained using BYOL [49] on the images from the demonstrations [18]. We find that using nearest-neighbor based imitation to select actions provides a strong baseline for dexterous manipulation tasks using a small number of demonstrations and minimal hyperparameter tuning.

IV. Experimental Evaluations

In this section, we experimentally evaluate DIME on dexterous manipulation problems, both in simulation and on our real Allegro Hand. Our experiments seek to answer two central questions:

- Can DIME be used to collect high-quality demonstrations for dexterity?
- Can the produced demonstrations be used to train dexterous behaviors?

A. Dexterous Manipulation Tasks

We look to tackle three dexterous manipulation tasks that reflect the challenges present in developing robot dexterity.

1) Flipping: Given a rectangular object placed on the fingers of the Allegro Hand, the hand is tasked to flip the object to the center of the palm. Solving this task requires precise coordination between the fingers since uneven movements result in the object falling off the hand. Performing this task is counted as a success when the object flips and lands within 2 cm of the palm's center within 1 minute.

2) Spinning: Given a three-pronged knob attached to a table, the Allegro hand is tasked to continuously spin the knob in place. This task is based on the ROBEL benchmark [50], and does not include the rotation of the knob in the observation space on hardware, but does in simulation. Performing this task is counted as a success when 120 degrees of rotation is achieved in 1 minute.

3) Rotating: Given a cube placed in the center of the Allegro hand, the hand is tasked to continuously rotate the cube in the plane. Solving this task requires the robot to make multi-fingered contacts to both rotate and correct for deviations of the cube from the center of the hand. Performing this task is counted as a success when 90 degrees of rotation is achieved in 1 minute.

In simulation, for each task, a MuJoCo [51] environment is created with a similar success metric as with the real robot environment. However, unlike the real environment, a dense reward function is created for each environment that is linear with the distance between the object and its target pose. This reward function is required for optimization using RL. To ensure reproducibility, our simulated environments and accompanying demonstrations will be publicly released.

B. Demonstration Collection

For each of our tasks, we first collect 30 demonstrations of each of them using DIME. Representative examples of these demonstrations can be seen in Fig. 3. For simulation tasks, we collect 10 demonstrations for each task. On average we notice that each demonstration for *flipping*, *spinning*, and *rotating* requires 30, 120, and 150 seconds to teleoperate respectively. This indicates that the *rotating* task is the hardest among our set of tasks for human operators. We notice that since we use a single camera, there are often

Fig. 5: Robot runs for both state-based (INN) and image-based (VINN) non-parametric nearest neighbors along with state-based parametric behavior cloning (BC) are visualized across the three tasks. INN performs the best across all three tasks, while VINN is able to solve the *flipping* and *rotating* tasks. BC is unable to solve any task and suffers from distributional mismatch [35].

occlusions during teleoperation that can cause inaccuracies in the MediaPipe hand detector [15]. To address this, human operators automatically adjust their hands through continuous visual feedback from the teleoperated robot hand.

C. Imitation Learning Algorithms

Given demonstrations collected in simulation and real, we study the use of the following imitation learning algorithms.

1) Behavior Cloning (BC & VBC): Here, a parametric neural network model is used to predict actions given states using supervised training [52, 24]. We experiment with both vision-based (VBC) and state-based (BC) observations for real-robot training.

2) Nearest Neighbors (INN & VINN): Here, a non-parametric model is used to match given observations with examples in the demonstration [17, 18] as described in Section III-F. The actions corresponding to the best match are then applied to the robot. We experiment with both vision-based (VINN) and state-based (INN) observations for real-robot training.

3) Proximal Policy Optimization (PPO): Here, a model-free RL optimizer is used to train policies in simulation [47]. Policies trained with PPO are initialized randomly.

4) Behavior Cloning with RL finetuning (BCRL): Here, model-free RL is used on top of a behavior-cloned policy in simulation. This can be viewed as PPO initialized from BC-trained policies.

5) Demonstration Augmented Policy Gradient (DAPG): Here, a policy gradient approach is used to train policies using both a cloning loss similar to BC and an RL loss similar to PPO. More details on this algorithm are presented in Section III-E.

The RL-based algorithms, PPO, BCRL, and DAPG are run only in simulation since running them on the real robot would require significant training time and could raise safety issues. For these experiments, we made use of the MJRL codebase [16]. The hyperparameters for each of these algorithms are selected through a hyperparameter search. Implementations, along with tuned hyperparameters of all imitation algorithms will be publicly released to ensure reproducibility. Visualization of trained policies across all tasks can be found on our project website: https://nyu-robot-learning.github.io/dime

D. Imitation Learning in Simulation

The policies trained with BCRL and DAPG produced similar results to one another, whereas PPO methods had more erratic movements. While this erratic behavior afforded success in the environments with easier tasks (*flipping* and *spinning*), policies from PPO were not able to successfully rotate the cube placed on the palm. The policies that used demonstrations were also qualitatively better than the tele-operated demonstrations, which would often take longer to record and have more abrupt starts and stops. Both BCRL and DAPG policies were significantly smoother. An example of this is that the policy learned for *rotating* would be able to pick up and rotate the cube between two fingers, whereas our demonstrations would keep the block on the palm and push with the thumb and last finger in opposing directions.

Surprisingly, the PPO policies were able to solve the *spinning* task as well as the DAPG policies. Upon visualizing the policies, we noticed that the PPO policies were successful because extreme, random movements of the middle and last finger were enough to spin the handle. Whereas the policies learned with demonstrations were more ‘human-like’ and smoother. We further notice that pure behavior cloning (BC) fails on all tasks. Since the number of demonstrations used is relatively small [24], BC policies are unable to remain in the support of demonstration data and fail [35]. In contrast, RL finetuning on top of the BC policies (BCRL & PPO) allows the robot to account for states it hasn’t seen in the demonstrations.

E. Imitation Learning on Real Allegro Hand

Quantitative results of our real robot experiments are presented in Table I. For each algorithm and each task, we run the robot for ten trials. Success is determined by the metrics discussed in Section IV-A. Qualitative visualization of the runs is depicted in Fig. 5. We notice that non-parametric nearest neighbors (INN) outperforms parametric behavior cloning approaches across all tasks. This result is in line with recent work in non-parametric imitation [18] being superior to behavior cloning in domains with a limited number of demonstrations (~ 30). We note that on the Rotation task, which is quite difficult to solve even by human operators through teleoperation, INN achieves a perfect success rate on 90-degree rotation. Performance degrades mildly with larger rotation angles when the cube moves outside the manipulable region on the palm.

To further emphasize the usefulness of DIME, we run a visual imitation learning algorithm VINN [18]. The input visual information is the RGB robot images shown in Fig. 5. Here, visual representations are first optimized independently for each task using the self-supervised BYOL algorithm [49]. Following this, similar to INN, nearest-neighbor matching is done to select actions. Although the performance of visual imitation is lower on average than state-based imitation, we notice strong performances on the *flipping* and *rotation* task.

TABLE I: Success rates on our real Allegro hand using DIME.

Method Used	Flipping	Turning	Rotation	
			90°	180°
INN (State Based)	80%	60%	100%	80%
Behavior Cloning (State Based)	0%	0%	0%	0%
VINN (Image Based)	90%	0%	70%	50%
Behavior Cloning (Image Based)	0%	0%	0%	0%

However, on the *spinning* task, VINN is unable to learn good representations due to the visual complexity of the scene. We believe this is due to the inability of BYOL to learn effective representations for cluttered scenes with a limited number of demonstrations.

V. Limitations and Discussion

We have presented DIME, a framework to collect and learn from inexpensive demonstrations. Through experimental evaluations, we have shown that DIME can solve several dexterous manipulation tasks with high success rates. However, we believe that this is just the first step towards training dexterous robots from inexpensive demonstrations. There are still two key limitations of this framework. First, our demonstration collection pipeline requires the specification of a z-plane. This is because of the inherent depth ambiguity with RGB cameras. We believe this could be resolved by utilizing depth information from stereo or depth cameras. Second, for real-robot experiments, we notice that some tasks such as *spinning* do not have high success rates. As evidenced by our RL experiments in simulation, we believe that developing new sample-efficient RL finetuning methods can significantly alleviate this challenge. To encourage future work on dexterous manipulation we will publicly release our custom-built hand controllers, demonstration collection pipeline, demonstration data, and learning algorithms.

References

[1] H. Moravec, *Mind children: The future of robot and human intelligence*. Harvard University Press, 1988.

[2] L. Biagiotti, F. Lotti, C. Melchiorri, and G. Vassura, “How far is the human hand,” 2004.

[3] A. M. Dollar and R. D. Howe, “The sdm hand as a prosthetic terminal device: a feasibility study,” in *ICRR*. IEEE, 2007.

[4] K. Tai, A.-R. El-Sayed, M. Shahriari, M. Biglarbegian, and S. Mahmud, “State of the art robotic grippers and applications,” *Robotics*, 2016.

[5] M. R. Dogar and S. S. Srinivasa, “Push-grasping with dexterous hands: Mechanics and a method,” in *IROS*, 2010.

[6] Y. Bai and C. K. Liu, “Dexterous manipulation using both palm and fingers,” in *ICRA*, 2014.

[7] S. Andrews and P. G. Kry, “Goal directed multi-finger manipulation: Control policies and analysis,” *Computers & Graphics*, 2013.

[8] OpenAI, M. Andrychowicz, B. Baker, M. Chociej, R. Józefowicz, B. McGrew, J. Pachocki, A. Petron, M. Plappert, G. Powell, A. Ray, J. Schneider, S. Sidor, J. Tobin, P. Welinder, L. Weng, and W. Zaremba, “Learning dexterous in-hand manipulation,” *arXiv*, 2018.

[9] OpenAI, I. Akkaya, M. Andrychowicz, M. Chociej, M. Litwin, B. McGrew, A. Petron, A. Paino, M. Plappert, G. Powell, R. Ribas, J. Schneider, N. Tezak, J. Tworek, P. Welinder,

L. Weng, Q. Yuan, W. Zaremba, and L. Zhang, "Solving rubik's cube with a robot hand," *arXiv*, 2019.

[10] P. Bakker and Y. Kuniyoshi, "Robot see, robot do: An overview of robot imitation," in *AISB96 Workshop on Learning in Robots and Animals*, 1996.

[11] S. Schaal, "Is imitation learning the route to humanoid robots?" *Trends in cognitive sciences*, 1999.

[12] B. Akgun, M. Cakmak, K. Jiang, and A. L. Thomaz, "Keyframe-based learning from demonstration," *International Journal of Social Robotics*, vol. 4, no. 4, pp. 343–355, 2012.

[13] M. Caeiro-Rodríguez, I. Otero-González, F. A. Mikic-Fonte, and M. Llamas-Nistal, "A systematic review of commercial smart gloves: Current status and applications," *Sensors*, 2021.

[14] A. Handa, K. Van Wyk, W. Yang, J. Liang, Y.-W. Chao, Q. Wan, S. Birchfield, N. Ratliff, and D. Fox, "Dexpilot: Vision based teleoperation of dexterous robotic hand-arm system," *arXiv preprint arXiv:1910.03135*, 2019.

[15] F. Zhang, V. Bazarevsky, A. Vakunov, A. Tkachenka, G. Sung, C.-L. Chang, and M. Grundmann, "Mediapipe hands: On-device real-time hand tracking," 2020.

[16] A. Rajeswaran, V. Kumar, A. Gupta, G. Vezzani, J. Schulman, E. Todorov, and S. Levine, "Learning complex dexterous manipulation with deep reinforcement learning and demonstrations," *RSS*, 2018.

[17] S. Schaal and C. Atkeson, "Robot juggling: implementation of memory-based learning," *IEEE CSM*, 1994.

[18] J. Pari, N. M. Shafiullah, S. P. Arunachalam, and L. Pinto, "The surprising effectiveness of representation learning for visual imitation," 2021.

[19] K. Muelling, A. Boularias, B. Mohler, B. Schölkopf, and J. Peters, "Learning strategies in table tennis using inverse reinforcement learning," *Biological cybernetics*, vol. 108, no. 5, pp. 603–619, 2014.

[20] I. Lenz, R. A. Knepper, and A. Saxena, "Deepmpc: Learning deep latent features for model predictive control." in *Robotics: Science and Systems*. Rome, Italy, 2015.

[21] P. Sharma, L. Mohan, L. Pinto, and A. Gupta, "Multiple interactions made easy (mime): Large scale demonstrations data for imitation," *CoRL*, 2018.

[22] T. Zhang, Z. McCarthy, O. Jow, D. Lee, X. Chen, K. Goldberg, and P. Abbeel, "Deep imitation learning for complex manipulation tasks from virtual reality teleoperation," in *ICRA*, 2018.

[23] S. Song, A. Zeng, J. Lee, and T. Funkhouser, "Grasping in the wild: Learning 6dof closed-loop grasping from low-cost demonstrations," *RA-L*, 2020.

[24] S. Young, D. Gandhi, S. Tulsiani, A. Gupta, P. Abbeel, and L. Pinto, "Visual imitation made easy," 2020.

[25] R. Y. Wang and J. Popović, "Real-time hand-tracking with a color glove," *ACM Trans. Graph.*, vol. 28, no. 3, jul 2009.

[26] L. Guan Ming, J. Prayook, and A. Wei Tech, "Mobilehand: Real-time 3d hand shape and pose estimation from color image," in *ICONIP*, 2020.

[27] T. Schmidt, R. A. Newcombe, and D. Fox, "Dart: Dense articulated real-time tracking," in *RSS*, 2014.

[28] G. Moon, J. Y. Chang, and K. M. Lee, "V2v-posenet: Voxel-to-voxel prediction network for accurate 3d hand and human pose estimation from a single depth map," *CoRR*, vol. abs/1711.07399, 2017.

[29] A. Sivakumar, K. Shaw, and D. Pathak, "Robotic telekinesis: Learning a robotic hand imitator by watching humans on youtube," 2022.

[30] Y. Rong, T. Shiratori, and H. Joo, "Frankmocap: A monocular 3d whole-body pose estimation system via regression and integration," in *ICCV Workshops*, 2021.

[31] D. A. Pomerleau, "Alvinn: An autonomous land vehicle in a neural network," in *NeurIPS*, D. Touretzky, Ed., vol. 1. Morgan-Kaufmann, 1988.

[32] P. Florence, C. Lynch, A. Zeng, O. Ramirez, A. Wahid, L. Downs, A. Wong, J. Lee, I. Mordatch, and J. Tompson, "Implicit behavioral cloning," 2021.

[33] M. Bojarski, D. Del Testa, D. Dworakowski, B. Firner, B. Flepp, P. Goyal, L. D. Jackel, M. Monfort, U. Muller, J. Zhang, *et al.*, "End to end learning for self-driving cars," *arXiv:1604.07316*, 2016.

[34] A. Venkatraman, B. Boots, M. Hebert, and J. A. Bagnell, "Data as demonstrator with applications to system identification."

[35] S. Ross, G. J. Gordon, and J. A. Bagnell, "A reduction of imitation learning and structured prediction to no-regret online learning," 2011.

[36] P. Abbeel and A. Y. Ng, "Apprenticeship learning via inverse reinforcement learning," in *ICML*, 2004, p. 1.

[37] U. Syed and R. E. Schapire, "A game-theoretic approach to apprenticeship learning," in *NeruIPS*, J. Platt, D. Koller, Y. Singer, and S. Roweis, Eds., 2007.

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

[39] K. Lowrey, A. Rajeswaran, S. Kakade, E. Todorov, and I. Mordatch, "Plan online, learn offline: Efficient learning and exploration via model-based control," 2019.

[40] OpenAI, M. Andrychowicz, B. Baker, M. Chociej, R. Jozefowicz, B. McGrew, J. Pachocki, A. Petron, M. Plappert, G. Powell, A. Ray, J. Schneider, S. Sidor, J. Tobin, P. Welinder, L. Weng, and W. Zaremba, "Learning dexterous in-hand manipulation," 2019.

[41] C. Lynch, M. Khansari, T. Xiao, V. Kumar, J. Tompson, S. Levine, and P. Sermanet, "Learning latent plans from play," 2019.

[42] D. Yarats, D. Brandfonbrener, H. Liu, M. Laskin, P. Abbeel, A. Lazaric, and L. Pinto, "Don't change the algorithm, change the data: Exploratory data for offline reinforcement learning," 2022.

[43] M. Vecerík, 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*, vol. abs/1707.08817, 2017.

[44] A. Nair, D. Chen, P. Agrawal, P. Isola, P. Abbeel, J. Malik, and S. Levine, "Combining self-supervised learning and imitation for vision-based rope manipulation," in *ICRA*, 2017.

[45] Stanford Artificial Intelligence Laboratory et al., "Robotic operating system." [Online]. Available: https://www.ros.org

[46] S. M. Kakade, "A natural policy gradient," in *NeurIPS*, T. Dietterich, S. Becker, and Z. Ghahramani, Eds., 2001.

[47] J. Schulman, F. Wolski, P. Dhariwal, A. Radford, and O. Klimov, "Proximal policy optimization algorithms," 2017.

[48] E. Fix and J. L. Hodges, "Discriminatory analysis. nonparametric discrimination: Consistency properties," *International Statistical Review*, vol. 57, no. 3, pp. 238–247, 1989.

[49] J.-B. Grill, F. Strub, F. Altché, C. Tallec, P. Richemond, E. Buchatskaya, C. Doersch, B. Avila Pires, Z. Guo, M. Gheshlaghi Azar, *et al.*, "Bootstrap your own latent-a new approach to self-supervised learning," *NeurIPS*, 2020.

[50] M. Ahn, H. Zhu, K. Hartikainen, H. Ponte, A. Gupta, S. Levine, and V. Kumar, "Robel: Robotics benchmarks for learning with low-cost robots," in *CoRL*, 2020.

[51] E. Todorov, T. Erez, and Y. Tassa, "Mujoco: A physics engine for model-based control," in *IROS*, 2012.

[52] D. A. Pomerleau, "Alvinn: An autonomous land vehicle in a neural network," in *NIPS*, 1989.