Multiple Interactions Made Easy (MIME) : Large Scale Demonstrations Data for Imitation

Pratyusha Sharma* **Lekha Mohan*** **Lerrel Pinto** **Abhinav Gupta**

The Robotics Institute
Carnegie Mellon University

Abstract: In recent years, we have seen an emergence of data-driven approaches in robotics. However, most existing efforts and datasets are either in simulation or focus on a single task in isolation such as grasping, pushing or poking. In order to make progress and capture the space of manipulation, we would need to collect a large-scale dataset of diverse tasks such as pouring, opening bottles, stacking objects etc. But how does one collect such a dataset? In this paper, we present the largest available robotic-demonstration dataset (MIME) that contains 8260 human-robot demonstrations over 20 different robotic tasks[2]. These tasks range from the simple task of pushing objects to the difficult task of stacking household objects. Our dataset consists of videos of human demonstrations and kinesthetic trajectories of robot demonstrations. We also propose to use this dataset for the task of mapping 3rd person video features to robot trajectories. Furthermore, we present two different approaches using this dataset and evaluate the predicted robot trajectories against ground-truth trajectories. We hope our dataset inspires research in multiple areas including visual imitation, trajectory prediction and multi-task robotic learning.

Keywords: Learning from Demonstration, Kinesthetic data

1 Introduction

One of the biggest success stories in recent AI research is the emergence of data-driven approaches. And as we can expect a key ingredient in these data-driven approaches is **DATA** itself. In computer vision, for example, the emergence of ImageNet [1] was a key moment for data driven methods like ConvNets. In recent years, data-driven approaches have started to gain momentum in the field of robotics as well. For example, Pinto and Gupta [2], displayed how data collection can be scaled to 50K examples for tasks such as grasping and how deep learning approaches improve with increasing amounts of data. Since then, data-driven algorithms have been scaled up in terms of number of datapoints [3] and shown to be useful for other task such as poking [4].

Most of the existing robotics datasets focus on a single task in isolation such as grasping Pinto and Gupta [2], Levine et al. [3], pushing [5, 6], poking [4] or knot-tying [7]. This is not surprising; even the early computer vision datasets initially focused on single tasks such as faces [8] and cars [9]. But the real success of computer vision came from building datasets than span across hundreds and thousands of categories. What the diversity of data allowed was to learn a generic visual representation that could then be transferred for variety of tasks. Inspired from this observation, we argue that it is critical for data-driven manipulation algorithms to be given diverse data of manipulation tasks with hundreds of different objects. But how do you design and collect a large-scale dataset of diverse robotic manipulations?

Unlike existing large-scale datasets which focus on simple tasks, self-supervision via random exploration is unlikely to succeed for complex manipulation tasks like stacking objects. Therefore, learning complex-manipulation requires supervised learning (demonstrations) rather than self-

*Equal contribution. Correspondence: lekhawm@gmail.com, {pratyuss, lerrelp, abhinavg}@cs.cmu.edu
[2]Website: https://sites.google.com/view/mimedataset

2nd Conference on Robot Learning (CoRL 2018), Zürich, Switzerland.

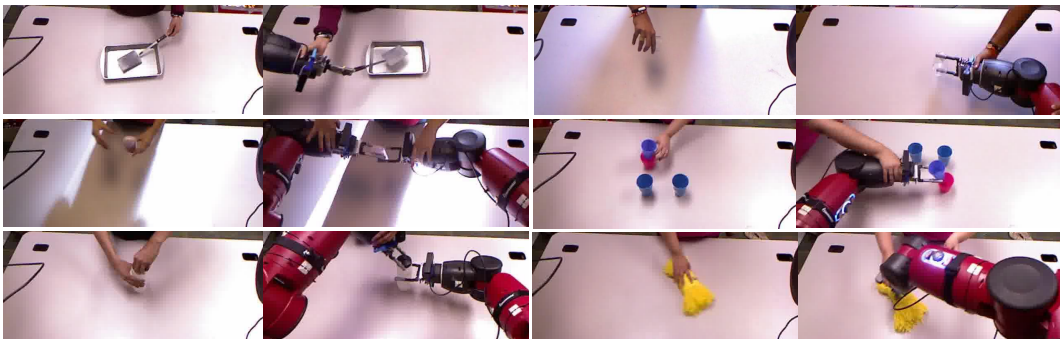

Figure 1: Example datapoints from six diferent task-categories in our MIME Dataset. On the left we show human demonstration and on the right we show how kinesthetic trajectories are being collected.(Clockwise from top-left the tasks are stirring, pouring, stacking, wiping, opening a bottle and passing.)

supervision. But to truly scale the abilities of our agents, we will need to scale the amount of demonstration data available. In this work, we take this first step towards creating large-scale demonstration data: specifically, we collect the largest available robotic demonstration dataset (MIME) that contains **8260** *human-robot* demonstrations with over **20** different robotic tasks. These tasks range from the simple task of pushing objects to the difficult task of stacking household objects. Collecting a dataset of this scale involves several challenging questions: (a) What type of data do we collect?; (b) How do we collect this data?; and (c) How do we ensure that the collected data is meaningful?

One of the key design decisions in creating MIME is the mode of getting expert demonstrations. Although several forms of Learning from Demonstration (LfD) data collection strategies are present in literature, we select two ways of capturing demonstrations: (a) Kinesthetic Demonstrations [10]: a kinesthetic method for collecting demonstrations since it allows a human demonstrator to both express their desired motion as well as stay in the constrains of the robot; (b) Visual Demonstration [11]: Instead of only recording kinesthetic data, we also record a visual demonstration of how humans perform the same tasks. Hence for every kinesthetic robot demonstration, we also collect a corresponding video of the same human demonstrator performing the task with their hands.

The next challenge is the process of physically collecting these diverse demonstrations. For small scale datasets, often a single expert demonstrator collects all the data. However to really scale demonstration data, we need to get demonstrations from multiple human demonstrators. This comes with both advantages and disadvanges. The advantages are the kinesthetic data not being biased by a single demonstrator's peculiarities and the potential ability to parallelize data collection. The disadvantage however is that human demonstrators who haven't worked with the robot before find it hard to give kinesthetic demonstrations. We solve this challenge by carefully training the demonstrators and follow it by crosschecking the data by other human demonstrators.

While we foresee multiple ways of using this dataset (e.g, learning action representation to initialize RL), in this paper, we introduce and focus on the task of learning a mapping from visual demonstrations to robotic trajectories. Specifically, given the video of human demonstration, the goal is to predict the joint angles to achieve the same goal. However, there might be multiple possible robot trajectories that can lead to same goal state; therefore during test we collect multiple demonstrations for the same task and use the min-distance from the set of test trajectories as the evaluation metric. These experiments further validate the utility of MIME.

2 Related Work

2.1 Data Driven Robotics

Inspired from the successes in large scale computer vision datasets [1, 12], natural language datasets [13] and reinforcement learning frameworks [14, 15], the last few years has seen a growing interest in large scale robotics [2, 3]. The KITTI driving dataset [16] collected images from a car that inspired research in several tasks like 3D tracking, visual odometry and optical flow. Similarly

the RGBD SLAM dataset [17] has spurred several papers that have improved the state of the art in SLAM. We hope that MIME will similarly accelerate research in Learning from Demonstrations.

In the realm of learning from manipulator data, Lenz et al. [18] presented one of the first attempts to collect grasping data using expert annotations. Pinto and Gupta [2] took the next leap in large scale data collection where the robots collect data themselves without human supervision. This has been followed by attempts to scale even further using multiple robots [3], multiple tasks [19], adversarial learning [20] and curriculum learning [21]. Following grasping, several researchers have looked at the task of pushing objects[4, 6, 5]. Attempts for large scale data collection has also been pursued in rope manipulation [7], surgical robotics [22] and opening doors [23]. However all of these tasks look at single and often simple tasks like grasping or pushing objects. In MIME, we collect data for around 20 different tasks, which will accelerate robot learning not just for the individual tasks but for imitation learning in general.

Another direction in scaling up data is in robotic simulators. Simulators offer large scale data that can be collected much faster than in the real world. GraspIt! [24] developed an interface to evaluate robotic grasps in a simulator. DexNet [25, 26] took this further by using cloud computing and synthetic object models to learn grasp models. In driving, several simulators like CARLA [27] and AirSim [28] promise easier self driving research. For indoor navigation simulators like AI2-THOR [29], SUNCG [30] and Matterport3D [31] have emerged. Simulators can also be interleaved with reinforcement learning for faster learning. Several works [32, 33] show promise in this direction. However, transferring policies from simulators to the real world is often challenging due to the reality-gap. This prompts us to create a real-world dataset that would allow for better transfer in the real world.

2.2 Learning From Demonstrations (LfD)

Learning from demonstrations encapsulates the field of learning robotic strategies or policies from human or expert demonstrations. An in-depth survey of LfD can be found in Argall et al. [10], Kober et al. [34]. Compared to self-supervized robot learning, LfD methods allows for learning more complex policies. Intuitively, an expert demonstration vastly cuts down the exploration space and can provide strong guidance during policy learning [34]. This has enabled autonomous helicopter aerobatics [35], table-tennis playing [36] and drone flying [37]. However these methods often focus on learning from a handful of expert demonstrations for a single task.

Scaling expert demonstrations has been receiving interest recently with Zhang et al. [38] presenting an approach that using tele-operation to collect demonstrations. Here hundreds of demonstrations are collected using a Virtual Reality interface for 15 tasks. However, this data isn't public and used in a task specific manner. For general purpose demonstrations, kinesthetically moving the robot ensures that the robot is in good configuration spaces. This allows for easier whole arm manipulation rather than only end-effector manipulation. Imitation learning using data from multiple tasks has also shown promise [39]. We believe that our dataset MIME can be used to further research in this area with more diverse objects and larger complexity of tasks.

Learning from demonstrations with multiple tasks is also connected to multi-task learning in the domain of computer vision. Large scale datasets like ImageNet [1] allowed for single models to simultaneously classify for multiple categories. These pre-trained classification models [40] can be then used to speed up learning for other visual tasks like detection [41] and action classification [11]. However the available large scale visual datasets do not contain physical actions. This makes it hard to transfer learned information to new tasks. Since MIME contains rich visual information for 20 different tasks, we believe it will accelerate the progress in multi-task learning from demonstrations.

3 The MIME Dataset

We now describe our methodology to collect the MIME demonstration dataset. There are several challenges in this effort: First, what is the right vocabulary of tasks? Second, how do we scale up collection of kinesthetic trajectories? Finally, how do we correct the errors made by humans in demonstration collection?

Figure 2: Multiple views of RGBD data from a robot demonstration.

3.1 Vocabulary of Tasks

The first question that we need to tackle is the vocabulary of tasks for which we collect both the kinesthetic trajectories and the video of human demonstration. The key consideration in selection of tasks is: (a) tasks should be easy enough to be performed by our Baxter robot; (b) should not require haptic feedback for successful performance; (c) diverse enough for us to learn the embedding of tasks. A complete list of the tasks can be seen in Table 1. Finally to increase the diversity of the data, we collect demonstrations over a variety of objects which can be seen in Fig. 3.

Table 1: Task-wise data splits

Task	train	val	test	Task	train	val	test
Pour	231	29	30	Close book	253	32	32
Stir	406	51	51	Pick (single hand)	524	65	66
Pass	388	48	49	Pick (both hands)	199	25	25
Stack	411	51	52	Poke	433	54	55
Place objects in box	293	37	37	Pull (two hands)	372	46	47
Open bottles	378	47	48	Push (two hands)	315	39	40
Push	304	25	26	Toy car trajectories	334	42	42
Rotate	335	42	42	Roll	340	43	43
Wipe	233	29	30	Drop objects	372	46	47
Press buttons	252	31	32	Pull (single hand)	328	41	41

3.2 Robot Setup

To collect the demonstration data, we use a Baxter robot in gravity compensated kinesthetic mode. The Baxter is a dual arm manipulator with 7DoF arms equipped with two-fingered parallel grippers. Furthermore, the robot is equipped with a Kinect mounted on the robot's head, and two SoftKinetic DS325 cameras, each mounted on the robot's wrist. The head camera acts like an external camera observing the task on the table, while the wrist cameras act as eye in robot cameras that move as the arm moves. During every robot demonstration, all the RGBD images as shown in Fig[2], the robot joint angles and the gripper positions are synchronized and stored.

3.3 Data Collection Procedure

To collect a large scale demonstration dataset, we need mutiple demonstrators to collect data. This brings about a unique challenge: new users often find it difficult to kinesthetically operate the robot. Hence we first train every human demonstrator with lessons to safely handle the robot. Once a demonstrator is comfortable in operating the robot, a specific task is assigned to the participant.

To facilitate smooth data collection, the participants use specific push buttons on the robot, which in turn displays the instructions on steps to follow. The main features of this setup are to home the robot, record data, visualize successive steps in the demo and display hardware or software errors so the setup can be checked before proceeding. Demonstrations performed by the participants are reviewed by other participants, who visually verify the collected trajectories using a visualizer we made. Using certain keyboard keys the reviewer can accept/reject the demo. If rejected, the trajectories are checked and redone by the demonstrator.

For improved robustness and to avoid bias, every participant gives multiple demonstrations for the same task using a variety of objects. After completing demonstrations for a particular task, the participant is assigned a new task. This ensures that each task has multiple demonstrators collecting data for it. The data collection pipeline was iteratively improved based on participant feedback.

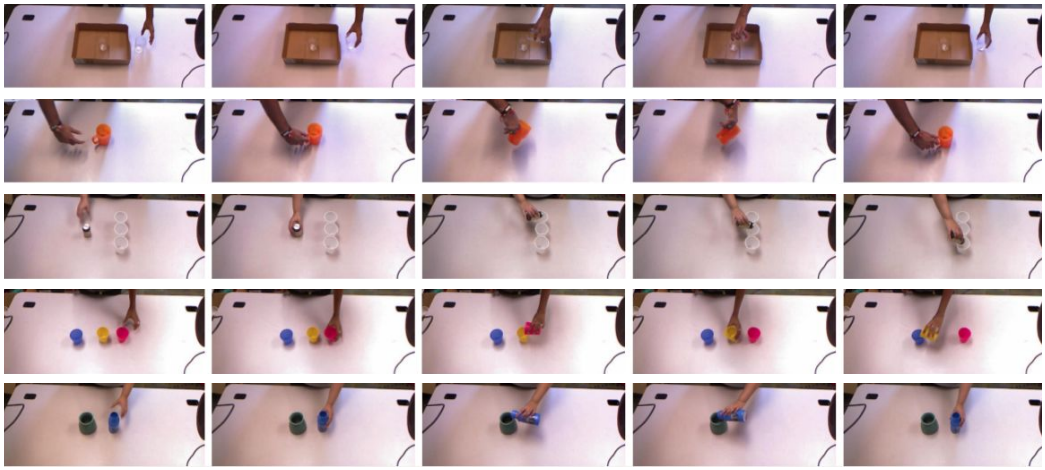

Figure 3: Diversity within a task. For example, in the task of pouring there is diversity in objects (shape, opaque/transparent), arrangement of objects, and how many objects are being poured into.

Note every data point consists of a human demonstration and a corresponding robot demonstration. We then use a verification stage to check the quality of collected data. Specifically, all the collected data was reviewed by other participants and incorrect demonstrations were removed from the dataset. An interesting observation is that the erroneous trajectories were mostly collected by newer participants. After a few trials, the participants developed the ability to accurately manoeuvre the robot, resulting in less errors and faster trajectories. At the end of the data collection process, we have 8260 demonstrations for 20 discrete tasks (Table 1).

4 Experiments

We demonstrate the quality of MIME by performing a suite of tasks. These tasks will also highlight the importance of the various components of our dataset. These tasks are: Task recognition, and Behaviour Cloning.

To evaluate the algorithms we split the dataset into a train set, validation set, and test set as seen in Table 1. The distribution of the different task categories is the same over the three splits, with 80% of the data is in the training set, 10% in the validation set and 10% in the test set. However for each of the test-set demonstrations we collect multiple test trajectories. This captures the multi-modal nature of our problem and allows us to compute the error by estimating the distance between predicted trajectories and a set of ground-truth trajectories.

4.1 Task Recognition

We first evaluate on task recognition, where the goal is to classify trajectories based on joint information alone. This is done to demonstrate the discriminative signal in the joint angle data.Since joint trajectories are sequential in nature with trajectory $\tau \equiv (s_0, s_1, ..s_{T-1})$, we employ two methods to evaluate on this task, dynamic time warping and a long short term memory architecture (LSTM) [42] as our model. While DTW analytically computes the distance between the time series data it can be slow to use it over large datasets. Hence, we also run a learning based method which captures the knowledge of the data in its weights.

Dynamic Time Warping(DTW): DTW is an algorithm that is often used to look at the proximity of different time series data. We employed DTW between each of the trajectories in the test data with all the trajectories in the training data. We then classified each of the test trajectories into the class of the trajectory of the training data with which it's distance was the least. In this experiment, we again highlight the importance of large-scale data but for the non-learning approach (DTW). Here we see that using just 10% of data yields an accuracy of 58.9%, while using the full 100%, we get a significantly higher accuracy of 79.7%.

Learning Methods: We use a single layer LSTM cell followed by a linear layer. At every timestep, the observed state along with the learned hidden state is used to update the prediction of task class. The final prediction, when the last observed state is fed in, is used as the predicted task label. Note that s_t represents the vector of joint angles at time t and T is the length of our trajectory. Our LSTM is unrolled for T steps and fed s_t as input at timestep t. The output at every step is a 20 dimensional vector representing the probability of the predicted task. A cross entropy loss is computed at every timestep and cumulated to a final loss. This final loss is minimized using the Adam Optimizer [43].

On the held-out test set, we compare the predicted label at the end feeding in a trajectory with the true label. This LSTM model achieves an accuracy of 61.2%. Furthermore visualizing the confusion matrix (Fig. 4) depicts an interesting trend of task correlation. In most cases, an incorrectly classified datapoint is often from a similar task that has similar or overlapping trajectories. For example picking with a single hand is confused with stacking and placing in a box is confused with picking.

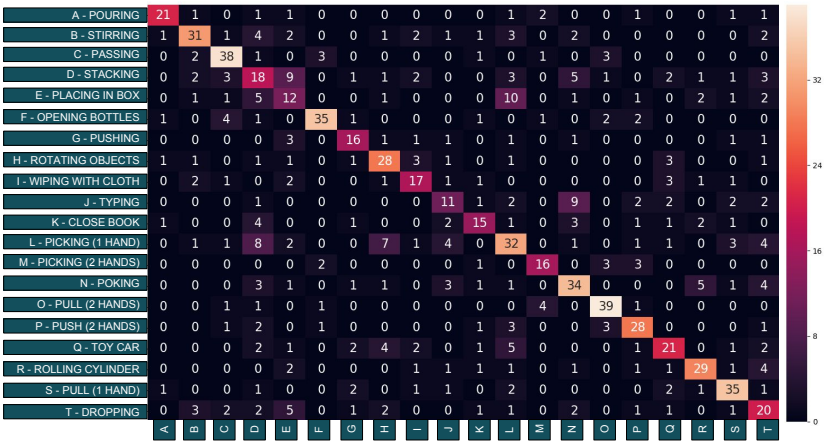

Figure 4: Confusion matrix of task classification from joint angles.

4.2 Behavioural Cloning

One of the challenging tasks we plan to handle using this dataset is behaviour cloning. Specifically, in this task: given the third-person video of a human doing a demonstration and the corresponding robot trajectory for the same task, we want to learn a mapping between visual features to robot trajectories. We will describe the data, a baseline formulation and evaluation metric in the next section.

Data Preparation: Since the trajectory sequences of different tasks are of different lengths, we sub-sample the robot joint angle trajectories and the human demonstration video frames to a fixed constant length of 50.

Extracting visual features from demonstration: Next, we need to extract visual features for the input video. One possibility is to use i3D [44] or non-local neural network from Wang et al. [45]. However, in both these cases, the temporal information is lost in the final features. Therefore, instead for each video we extract 50 sub-clips; each sub-clip is 2-frame clip. We extract features for each sub-clip using a pre-trained model from Wang et al. [45]. This leads to a temporally ordered set of 50 sub-clip features.

State Vector: Apart from using the demonstration features, our imitation policy needs to observe the current object location and state to predict the action plan. To capture the current state, we use a single head-camera image from the robot demonstration instead of the whole video. This image is passed through a pre-trained VGG network [46] to obtain image features.

Output Space: One possibility is to use the current state vector and video features to regress to joint angles directly. However, we note that for each demonstration there are multiple ways to imitate it and hence the multimodal nature of output. Therefore, instead of direct regression, which might regress to the mean of the multimodal trajectories possible, we plan to use a classification-based approach.

Specifically, first we use the set of joint angles s_t and cluster them using k-means. This allows vector quantization of the output space and hence the model predicts the output cluster center at each time instant. But one issue with the above is that if we use one vector-quantization across all possible joint angles, fine-grained manipulation motion tend to be clustered in one group leading to the averaging effect. Therefore, instead, we follow a variable sparsity approach for clustering. Since any trajectory involves first approaching the object and then interacting with it, we cluster the first half of the trajectories ($t \leq 25$) sparsely while densely clustering the second half ($t > 25$). This helps us better capture the nuances of manipulating the object by finer clustering of the prediction space near the objects. In total we obtain 600 clusters with 150 in the sparse region and 450 in the dense region.

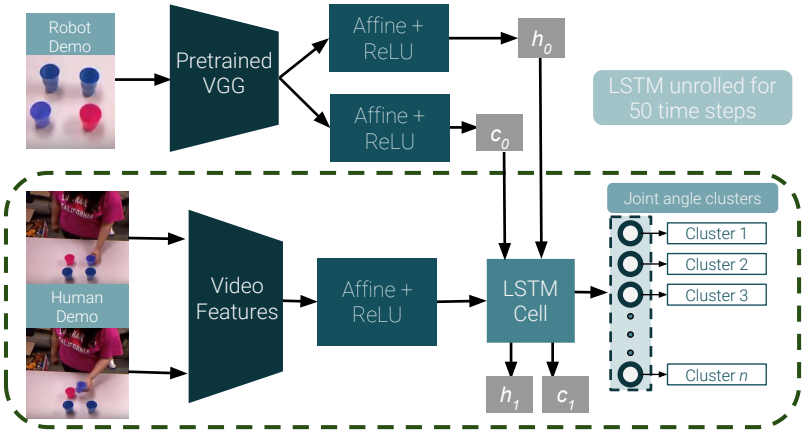

Figure 5: Behavioural Cloning - Model Architecture.

Model Architecture: Now that we have explained our input and output space, we explain our final model. Our model is visualized in Fig. 5. The goal of this model is to predict the joint angles s_t given the original configuration of objects and a human video feature at every time-step. Hence this LSTM model runs across the length of the human video and gives us the predicted trajectory corresponding to the task in the human video. The LSTM's initial hidden state and cell state is set as the VGG feature of the robot demonstration image. The human demonstration video features are then used as input into the LSTM sequentially for 50 episodes. The loss for this network is the cross entropy loss between the predicted joint cluster-number and the ground truth joint cluster-number.

Evaluation: Next, we describe our evaluation metric for comparing predicted trajectories with the ground-truth trajectories for test videos. We use the Mean Squared Error (MSE) between the predicted trajectory and the true trajectory. We argue that though the mean square error might not be the best metric to evaluate the performance of the trajectory it turns out that using the MSE to evaluate the performance of a trajectory does help us evaluate how close a predicted trajectory is to one possible ground-truth. Individual classwise MSE errors are summarized in Table 2. It can be seen that tasks that are multi-modal in nature, like placing in box, incur a larger loss.

Table 2: Task-wise MSE on held out test set

Task	MSE	Task	MSE
Pouring	0.111	Close book	0.83487
Stirring	0.1061	Single hand picking	0.1262
Passing	0.1396	Both hand picking	0.1172
Stacking	0.106	Poking	0.1179
Placing in a box	0.1403	Two hand pull	0.1066
Opening bottles	0.1245	Two hand push	0.1049
Pushing	0.1325	Toy car trajectories	0.1206
Rotating objects	0.1024	Rolling cylinders	0.1236
Wiping with cloth	0.1435	Dropping	0.1211
Typing on keyboard	0.1149	One hand pull	0.1506

Multimodality in the trajectory predicted: A manipulation task can be solved by multiple different trajectories. There could be differences in the trajectory followed to reach the object, differences in location of grasping, or differences due to where the object was left after manipulation. Hence, given a state of the environment, there exist several possible trajectories that can achieve the task. To handle this we took two steps. The steps being classification instead of regression (which was discussed earlier in this section) and evaluation against multiple ground truths. In the following paragraph we discuss the second step.

If there are multiple possible trajectories and an approach selects one of them but the GT is the some other; then the approach gets penalized even though it was correct. To handle this, we collected multiple ground truth trajectories (while more is better we collected 2 in this paper) for all the tasks in the test set. The MSE was then calculated as the minimum of the MSE between the predicted trajectory and the each of the ground truth trajectories. Using this multiple GT trajectories, our MSE fell from 0.1296 to 0.1076.

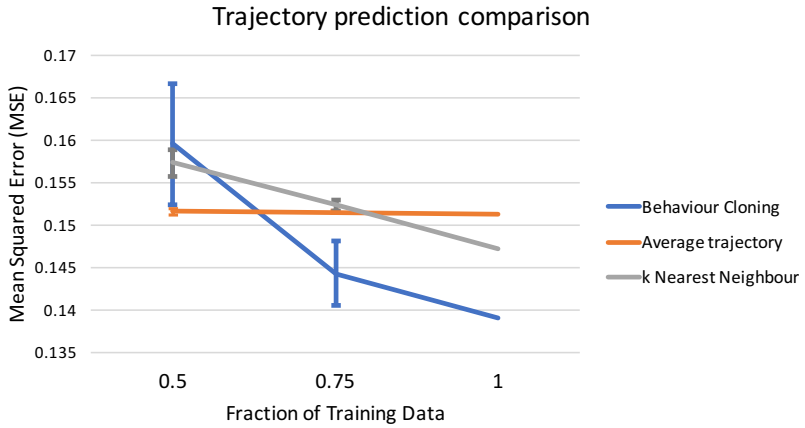

Figure 6: Evaluation of behavior cloning, average trajectory and k-NN approach. The error bars show ± 1 standard deviation when training on random splits with a fixed fraction of training data.

4.3 Comparing baselines

We also implemented some other simple baselines to demonstrate how hard the dataset is. Specifically, we tried two baseline methods.

Average trajectory: In this experiment, we used the average trajectory for each task and use it to compare with GT trajectory.

k-nearest neighbours: As a second baseline we use Nearest-Neighbor baseline. Specifically, we compute the video features of the query human demonstration and then use it to retrieve the k-NN robot trajectories. The MSE loss was computed between the predicted trajectory against the multi-modal ground truth trajectories. For our experiments we use k=11 as it yielded the best performance.

Fig. 6 illustrates the variation of MSE with respect to the amount of data used. It can be seen that the behaviour cloning provides increasing performance with increasing amount of data used to train it. The method also does better than the two baselines while having a steeper improvement in performance with increase in data.

5 Conclusion

In this paper, we have presented one of the largest human and robot demonstration dataset to date. Our dataset consist of 8260 human-object interactions and 8260 robot trajectories for the same task on 20 diverse tasks. We demonstrate the use of our dataset for the task of visual imitation: mapping 3rd-person video features to robot trajectories.

References

[1] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei. Imagenet: A large-scale hierarchical image database. In *CVPR*, 2009.

[2] L. Pinto and A. Gupta. Supersizing self-supervision: Learning to grasp from 50k tries and 700 robot hours. In *ICRA*, 2016.

[3] S. Levine, P. Pastor, A. Krizhevsky, and D. Quillen. Learning hand-eye coordination for robotic grasping with deep learning and large-scale data collection. *CoRR*, abs/1603.02199, 2016.

[4] P. Agrawal, A. V. Nair, P. Abbeel, J. Malik, and S. Levine. Learning to poke by poking: Experiential learning of intuitive physics. In *NIPS*, 2016.

[5] C. Finn, I. Goodfellow, and S. Levine. Unsupervised learning for physical interaction through video prediction. In *NIPS*, 2016.

[6] L. Pinto and A. Gupta. Learning to push by grasping: Using multiple tasks for effective learning. *ICRA*, 2017.

[7] 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. *ICRA*, 2017.

[8] M. Turk and A. Pentland. Eigenfaces for recognition. *J. Cognitive Neuroscience*, 3(1):71–86, Jan. 1991. ISSN 0898-929X. doi:10.1162/jocn.1991.3.1.71.

[9] S. Agarwal and D. Roth. Learning a sparse representation for object detection. In *ECCV*, 2002.

[10] B. D. Argall, S. Chernova, M. Veloso, and B. Browning. A survey of robot learning from demonstration. *Robotics and autonomous systems*, 57(5):469–483, 2009.

[11] K. Simonyan and A. Zisserman. Two-stream convolutional networks for action recognition in videos. In *NIPS*, 2014.

[12] M. Everingham, L. Van Gool, C. K. I. Williams, J. Winn, and A. Zisserman. The pascal visual object classes (voc) challenge. *IJCV*, 2010.

[13] M. Marcus, G. Kim, M. A. Marcinkiewicz, R. MacIntyre, A. Bies, M. Ferguson, K. Katz, and B. Schasberger. The penn treebank: annotating predicate argument structure. In *Proceedings of the workshop on Human Language Technology*, 1994.

[14] G. Brockman, V. Cheung, L. Pettersson, J. Schneider, J. Schulman, J. Tang, and W. Zaremba. Openai gym. *CoRR*, abs/1606.01540, 2016.

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

[16] A. Geiger, P. Lenz, C. Stiller, and R. Urtasun. Vision meets robotics: The kitti dataset. *IJRR*, 2013.

[17] J. Sturm, N. Engelhard, F. Endres, W. Burgard, and D. Cremers. A benchmark for the evaluation of rgb-d slam systems. In *IROS*, 2012.

[18] I. Lenz, H. Lee, and A. Saxena. Deep learning for detecting robotic grasps. *IJRR*, 2015.

[19] L. Pinto, D. Gandhi, Y. Han, Y.-L. Park, and A. Gupta. The curious robot: Learning visual representations via physical interactions. In *ECCV*, 2016.

[20] L. Pinto, J. Davidson, and A. Gupta. Supervision via competition: Robot adversaries for learning tasks. In *ICRA*. IEEE, 2017.

[21] A. Murali, L. Pinto, D. Gandhi, and A. Gupta. Cassl: Curriculum accelerated self-supervised learning. *ICRA*, 2018.

[22] Y. Gao, S. S. Vedula, C. E. Reiley, N. Ahmidi, B. Varadarajan, H. C. Lin, L. Tao, L. Zappella, B. Béjar, D. D. Yuh, et al. Jhu-isi gesture and skill assessment working set (jigsaws): A surgical activity dataset for human motion modeling. In *MICCAI Workshop: M2CAI*, volume 3, page 3, 2014.

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

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

[25] J. Mahler, F. T. Pokorny, B. Hou, M. Roderick, M. Laskey, M. Aubry, K. Kohlhoff, T. Kröger, J. Kuffner, and K. Goldberg. Dex-net 1.0: A cloud-based network of 3d objects for robust grasp planning using a multi-armed bandit model with correlated rewards. In *ICRA*, 2016.

[26] J. Mahler, M. Matl, X. Liu, A. Li, D. Gealy, and K. Goldberg. Dex-net 3.0: Computing robust robot suction grasp targets in point clouds using a new analytic model and deep learning. *arXiv preprint arXiv:1709.06670*, 2017.

[27] A. Dosovitskiy, G. Ros, F. Codevilla, A. López, and V. Koltun. Carla: An open urban driving simulator. *arXiv preprint arXiv:1711.03938*, 2017.

[28] S. Shah, D. Dey, C. Lovett, and A. Kapoor. Airsim: High-fidelity visual and physical simulation for autonomous vehicles. In *Field and Service Robotics*, 2018.

[29] Y. Zhu, R. Mottaghi, E. Kolve, J. J. Lim, A. Gupta, L. Fei-Fei, and A. Farhadi. Target-driven visual navigation in indoor scenes using deep reinforcement learning. In *ICRA*, 2017.

[30] S. Song, F. Yu, A. Zeng, A. X. Chang, M. Savva, and T. Funkhouser. Semantic scene completion from a single depth image. In *CVPR*, 2017.

[31] A. Chang, A. Dai, T. Funkhouser, M. Halber, M. Nießner, M. Savva, S. Song, A. Zeng, and Y. Zhang. Matterport3d: Learning from rgb-d data in indoor environments. *arXiv preprint arXiv:1709.06158*, 2017.

[32] F. Sadeghi and S. Levine. Cad2rl: Real single-image flight without a single real image. *arXiv preprint arXiv:1611.04201*, 2016.

[33] L. Pinto, M. Andrychowicz, P. Welinder, W. Zaremba, and P. Abbeel. Asymmetric actor critic for image-based robot learning. *RSS*, 2018.

[34] J. Kober, J. A. Bagnell, and J. Peters. Reinforcement learning in robotics: A survey. *IJRR*, 2013.

[35] P. Abbeel, A. Coates, and A. Y. Ng. Autonomous helicopter aerobatics through apprenticeship learning. *IJRR*, 2010.

[36] K. Mülling, J. Kober, O. Kroemer, and J. Peters. Learning to select and generalize striking movements in robot table tennis. *IJRR*, 2013.

[37] S. Ross, N. Melik-Barkhudarov, K. S. Shankar, A. Wendel, D. Dey, J. A. Bagnell, and M. Hebert. Learning monocular reactive uav control in cluttered natural environments. In *ICRA*, 2013.

[38] T. Zhang, Z. McCarthy, O. Jow, D. Lee, K. Goldberg, and P. Abbeel. Deep imitation learning for complex manipulation tasks from virtual reality teleoperation. *arXiv preprint arXiv:1710.04615*, 2017.

[39] C. Finn, T. Yu, T. Zhang, P. Abbeel, and S. Levine. One-shot visual imitation learning via meta-learning. *arXiv preprint arXiv:1709.04905*, 2017.

[40] A. Krizhevsky, I. Sutskever, and G. E. Hinton. Imagenet classification with deep convolutional neural networks. In *NIPS*, 2012.

[41] R. Girshick, J. Donahue, T. Darrell, and J. Malik. Rich feature hierarchies for accurate object detection and semantic segmentation. In *CVPR*, 2014.

[42] S. Hochreiter and J. Schmidhuber. Long short-term memory. *Neural Comput.*, 1997.

[43] D. P. Kingma and J. Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

[44] C. Feichtenhofer, A. Pinz, and R. P. Wildes. Spatiotemporal residual networks for video action recognition. *CoRR*, abs/1611.02155, 2016. URL http://arxiv.org/abs/1611.02155.

[45] X. Wang, R. Girshick, A. Gupta, and K. He. Non-local neural networks. In *CVPR*, 2018.

[46] K. Simonyan and A. Zisserman. Very deep convolutional networks for large-scale image recognition. *CoRR*, abs/1409.1556, 2014. URL http://arxiv.org/abs/1409.1556.