Proceedings of the Joint Austrian Computer Vision and Robotics Workshop 2020 DOI: 10.3217/978-3-85125-752-6-11

Learning Manipulation Tasks from Vision-based Teleoperation

Matthias Hirschmanner, Bernhard Neuberger, Timothy Patten, Markus Vincze
Automation and Control Institute, TU Wien, Vienna, Austria
{hirschmanner,neuberger,patten,vincze}@acin.tuwien.ac.at

Ali Jamadi
Ferdowsi University of Mashhad, Mashhad, Iran
a.jamadi@mail.um.ac.ir

Abstract. *Learning from demonstration is an approach to directly teach robots new tasks without explicit programming. Prior methods typically collect demonstration data through kinesthetic teaching or teleoperation. This is challenging because the human must physically interact with the robot or use specialized hardware. This paper presents a teleoperation system based on tracking the human hand to alleviate the requirement of specific tools for robot control. The data recorded during the demonstration is used to train a deep imitation learning model that enables the robot to imitate the task. We conduct experiments with a KUKA LWR IV+ robotic arm for the task of pushing an object from a random start location to a goal location. Results show the successful completion of the task by the robot after only 100 collected demonstrations. In comparison to the baseline model, the introduction of regularization and data augmentation leads to a higher success rate.*

1. Introduction

Robot manipulation tasks in domestic services and industry are highly complex due to the various system components that are necessary to achieve the goal. As a result, it is difficult to directly program robust robot manipulation strategies. Reinforcement learning is an alternative approach that alleviates the requirement for human programming and instead enables a robot platform to learn from its own experience [4, 11, 15]. However, this approach suffers from substantial training time, with some work reporting training times in the order of months [15]. Learning from demonstration (LfD) is an attractive solution in which a human illustrates how to perform a task and the robot attempts to imitate [21, 2]. This requires no human programming and far fewer training examples compared to reinforcement learning methods.

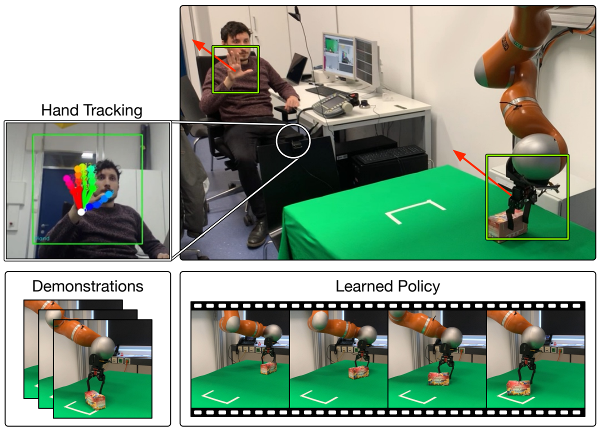

Figure 1. Teleoperating the robot arm using hand tracking from RGB images. The demonstrations are used to teach a policy to perform a task (e.g. push the box to the goal).

Demonstrations for learning are often collected through kinesthetic teaching [1] or teleoperation [27]. However, these methods are cumbersome because the human must either physically interact with the robot to generate example motions or control the robot system with specialized hardware that the operator may not have experience with. Despite the advances of teleoperation systems that enable novices to improve task performance after only a small number of attempts [8], the hardware is not always readily available. LfD can also leverage simulation [18] or by directly observing human activity [9, 13, 16, 24]. But these approaches demand additional solutions to transfer across domains.

To that end, we present an end-to-end system for LfD through vision-based teleoperation, which alleviates the necessity for virtual reality and teleoperation hardware while still directly controlling the robot platform to avoid the domain shift. We directly track the human hand using a webcam and use the estimated hand pose to control the end-effector of the robot. The demonstration data are used to train a neural network, based on the architecture of [27], to enable imitation by the robot system. We extend this work to include different regularization techniques during training and data augmentation to manage changes in brightness and imperfect demonstrations.

Our method is implemented for the KUKA LWR IV+ [3] robotic arm for the task of pushing objects. Experiments show that the robot is able to replicate the demonstrated task with as few as 100 recorded examples. In comparison to the baseline [27], our inclusion of regularization and data augmentation achieves a higher success rate.

In summary, we make the following contributions:

- A vision-based hand tracking system to teleoperate a robot arm to perform manipulation tasks.
- Training of a neural network with our generated teleoperated data that enables task imitation.
- Evaluation of the generalization of the imitation learning to unseen configurations.
- Improvements over the baseline by including regularization methods during the training.

The remainder of this paper is as follows. Section 2 reviews related work and Section 3 presents our approach. In Section 4 we present our experiments and results. Section 5 concludes the paper.

2. Related Work

A popular approach to program a robot to perform manipulation tasks is learning from demonstration [21, 2]. This involves recording example manipulation sequences and then to transfer the trajectories to the robot platform to perform the task itself. Trajectories are typically recorded using kinesthetic teaching [22, 19, 1], teleoperation [27, 10, 20] or generated in simulation [6, 18]. Given a set of demonstrations, these methods find an appropriate mapping in order to replicate the closest matching trajectory, often making adaptations due to the variation between the current and demonstrated scenarios. Some approaches represent the demonstrations as a set of primitives by encoding the trajectories and then generating robot motions through probabilistic methods, e.g., Gaussian mixtures [5], Gaussian processes [22] or dynamic movement primitives [19, 12]. This allows for a more efficient search for the most appropriate trajectory to replicate.

More recent works apply deep neural networks to learn visuomotor policies that map input images to robot trajectories through behavioral cloning [27, 20]. A network is trained on demonstrations to learn the image-to-action mapping such that a closed-loop controller commands the manipulator through sequences of states to complete the task. In this line of work, teleoperation is the preferred method to kinesthetic teaching because the human does not contaminate the training images.

Extensions have been made that generalize the models to multiple tasks, which allows few- or even one-shot learning of new tasks [6, 26]. These methods apply meta-learning to efficiently adapt a learned model, trained on many prior tasks, to a new task that is to be imitated. James *et al.* [10] take a different approach and use metric learning to create a task embedding. Imitating a new demonstration is achieved by training a control network to translate learned task embeddings into desired actions. Huang *et al.* [9] propose neural task graphs to learn the common structure of tasks and the conjugate relationship between observed states and actions.

Another direction of work is to learn by using only videos of humans performing tasks, e.g., [13, 16, 24]. However, human demonstrations do not provide sufficient supervision for learning. Therefore, other approaches explicitly learn the relationship between human and robot demonstrations in order to directly imitate human tasks in the online setting [26].

In this work, we build on the approaches for learning visuomotor policies through behavioral cloning. In particular, we adapt the methodology presented by Zhang *et al.* [27] by replacing the teleoperation hardware with a vision-based system. Our work is complementary to it as well as to the extension that incorporates human demonstrations [26] by using our teleoperation system as an alternative.

3. Approach

This section describes our approach for learning from demonstration, an overview is given in Figure 2. For teleoperation, a webcam is used to track the hand (Section 3.1) to generate positions that control the robot's end-effector (Section 3.2). During the trajectory, the RGB-D images from a ceiling mounted ASUS Xtion camera are recorded with the end-effector pose of the robot. Many demonstrations are shown to create a dataset that is used to train a deep imitation learning model (Section 3.3). The learned policy is then executed by the robot using only the live RGB-D images and end-effector poses.

3.1. Hand Tracking

The hand tracking method developed by Panteleris *et al.* [17] is used to estimate the 3D pose of the human demonstrator from RGB images in real-time. This approach consists of three steps: (1) Cropping the user hand in the image, (2) passing the cropped image to a 2D joint position estimator and (3) mapping the 2D joints on a 3D hand model to recover 3D positions of the joints.

For finding an initial bounding box of the hand, a deep neural network model [25] to detect hands in real-time is applied. Afterwards, the cropped image of the hand is passed through the hand key-point localization model of [7] to estimate the 2D location of the hand joints. It localizes the 21 key-points for the wrist, 5 fingertips and $5 \times 3 = 15$ finger joints. This specific model was selected because it matches or outperforms other state-of-the-art methods but with much lower computational requirements. In the end, the 2D locations of the joints are mapped to the 3D hand model via non-linear least-squares optimization. The 3D positions are then used as the initial step for the optimization of the next frame.

The 3D positions of the joints are also used to update the bounding box of the hand, which eliminates the need to use the hand detector model for each frame. However, failure of the hand tracking module based on the hand position and movement in the previous frames (e.g. due to sudden movements, occlusion, or failure in 2D localization), results in poor optimization. Therefore, to make the tracker more robust, the optimization score is checked to reset the optimizer's initial state and to use the hand detector to find a new bounding box for the hand if necessary.

3.2. Robot Control

For the teleoperation of the robot end-effector, the 3D hand position from the hand tracking system is compared with an initial hand position. If the difference between the current and the initial position for any Cartesian coordinate is above a certain threshold κ, a new end-effector position is calculated and commanded to the robot. This difference is then transformed from the camera frame to the robot base frame and denoted as $\mathbf{h}$. The transformation aligns the directions of the hand and end-effector movement to allow intuitive teleoperation.

The desired end-effector position $\mathbf{p}^*$ is calculated by adding the current end-effector position $\mathbf{p}$ and the value $\Delta\mathbf{p}$. This is calculated for each Cartesian coordinate with $p_i, h_i \in \mathbf{p}, \mathbf{h}$ according to:

$$\Delta p_i = \begin{cases} \alpha(\min\{h_{max}, h_i - \kappa\}) & \forall\ h_i > \kappa, \\ \alpha(\max\{h_{min}, -h_i - \kappa\}) & \forall\ h_i < -\kappa, \\ 0 & \text{otherwise}, \end{cases} \tag{1}$$

where h_{max} expressing the upper, h_{min} the lower limit and α as a parameter that indirectly allows the sensitivity to speed to be tuned.

As described in Section 3.3, $\Delta\mathbf{p}$ is directly learned. When executing the learned policy, $\Delta\mathbf{p}$ is used to calculate the desired end-effector position. For either the teleoperation or the task execution by the learned policy, the desired end-effector position is updated continuously and commanded to the robot. The orientation of the end-effector could be changed similarly but was not necessary for our specific task.

3.3. Deep Imitation Learning

We employed the algorithm presented by [27] and adapted it in several ways to work with our robotic setup involving imperfect demonstrations and changing environment conditions (e.g. brightness). The adapted network can be seen in Figure 2. The input $\mathbf{o}_t$ at each timestep t consists of the cropped and scaled color image $\mathbf{I}_t \in \mathbb{R}^{120\times160\times3}$, depth image $\mathbf{D}_t \in \mathbb{R}^{120\times160\times1}$, and the 5 most recent end-effector positions $\mathbf{p}_{t-4:t} \in \mathbb{R}^{15}$. After 3 convolutional layers, the data is passed through a spatial softmax layer introduced in [14]. During training, the output of this layer is used for auxiliary predictions of the current end-effector position and the end-effector position at the end of each demonstration with two fully connected layers per auxiliary prediction. The output of the network is the change of the end-effector position of the robot $\Delta\mathbf{p}$ in millimetres. Compared to [27], we omit one convolutional layer but use more units in our dense layers, which slightly reduces training time without deteriorating performance. Since we do not change the orientation of the end-effector for this simple task, we can simplify the output of the network at time t to be $\Delta\mathbf{p}_t = \pi_\theta(\mathbf{o}_t) \in \mathbb{R}^3$.

The input data is augmented by randomly changing the brightness during training and batch normalization is added after each layer to better cope with

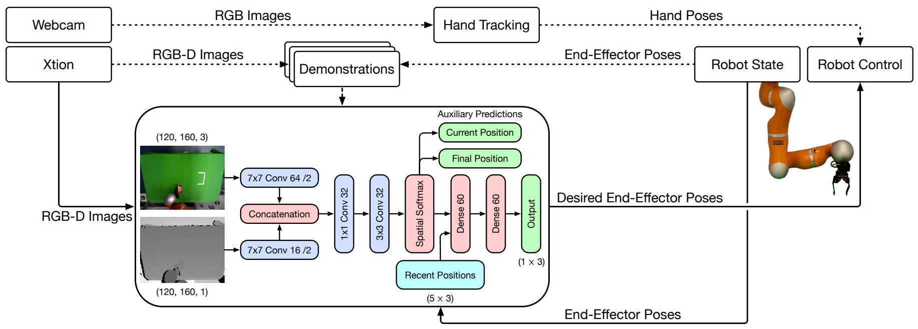

Figure 2. Overview of the system. The dashed lines show the procedure to collect demonstrations for training. The continuous lines show the information flow during policy execution.

the changing lighting conditions in the test environment. Additionally, we added dropout of the recent end-effector positions to avoid the robot following the same trajectory during most executions and not taking the object position into account.

The overall loss is defined as

$$\mathcal{L}(\theta) = \lambda_{l1}\mathcal{L}_{l1} + \lambda_{l2}\mathcal{L}_{l2} + \lambda_c\mathcal{L}_c + \lambda_s\mathcal{L}_s + \lambda_{aux}\Sigma_a\mathcal{L}^{(a)}_{aux}. \tag{2}$$

The first two terms are the $l1$ and $l2$ losses. $\mathcal{L}_c$ is the cosine loss and $\mathcal{L}_{aux}$ are the $l2$ losses of the auxiliary predictions. Compared to [27], we added the loss

$$\mathcal{L}_s = \exp\left(-||\pi_\theta(\mathbf{o}_t)||^2\right) \tag{3}$$

that penalizes very slow speeds. The weights were chosen as $\lambda_{l1} = 1.0$, $\lambda_{l2} = 0.01$, $\lambda_c = 0.05$, $\lambda_s = 0.1$, and $\lambda_{aux} = 0.01$.

4. Experiments

This section presents the experimental results. We first describe the setup and procedure for collecting demonstration data. We analyze the performance of our method with respect to the network design.

4.1. Experimental Setup

All experiments are conducted with a KUKA LWR IV+ [3] robotic arm using the provided control unit. The arm has 7 degrees of freedom and is controlled with position commands for the joints. The arm is mounted on the ceiling with a small table standing underneath it on which the target object (box) rests. The goal region is marked with tape. An ASUS Xtion RGB-D camera is mounted to the ceiling to capture the scene from above. For hand tracking, a separate webcam is used and faces the operator.

The algorithms for the hand tracking and the task execution run on a remote PC connected to the KUKA control unit via Ethernet. The communication between the remote PC and the control unit is enabled through the kuka-lwr-ros package[1] using the fast research interface (FRI) [23].

For data collection, the teleoperator directly faces the robot and the webcam. For each demonstration, the box is positioned randomly on the table. The teleoperator moves the box to the goal position using our control scheme. We collected 98 demonstrations with an average length of 42.8 s with a rate of 10 Hz for our evaluation. That is significantly above the average demonstration time per task of [27], which is between 3.7 s and 11.6 s and necessitates our changes to the architecture to deal with these imperfect demonstrations.

4.2. Results

For the evaluation, the workspace of the robot on the table is divided into a grid of 9 different positions with 20 cm intervals. Per position, the learned policy is executed for 4 different rotations of the box $(-45°, 0°, 45°, 90°)$. We measure both if the box is pushed towards the goal (*started push*) as well as if at least part of it is pushed into the goal (*success*). If the robot starts to push the box, but loses it, we restart the policy manually and keep the box in the same position when the end-effector stops or leaves the workspace. This could be automated with a simple heuristic. If the task can be achieved in a consecutive trial, we still count it as a success.

As shown in Table 1, our learned policy started to push the box in the right direction in 86.1 % of the cases and reached the goal in 58.3 % of the overall attempts. A reason for most failure cases is the grid nature of our workspace separation, which inherently tests the robot on the edges of its workspace where it is much more difficult to perform the task.

We conducted an ablation study to evaluate our changes to the original architecture of [27]. We re-

[1]https://github.com/epfl-lasa/kuka-lwr-ros

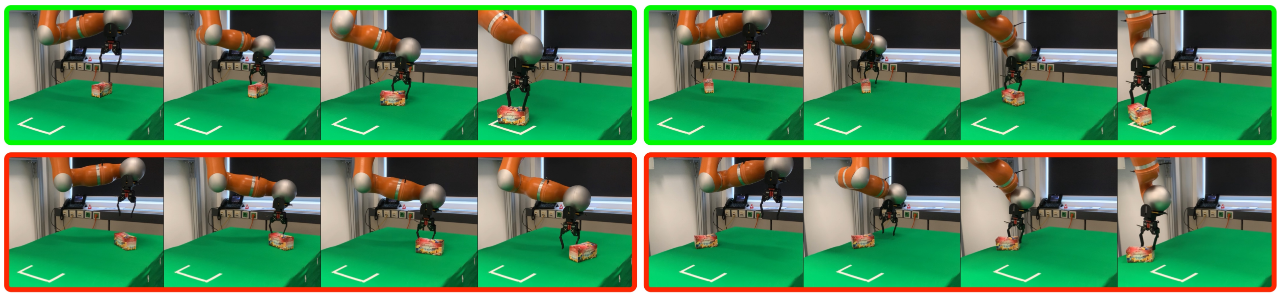
Figure 3. Examples of the learned policy. First row shows successful trials. Bottom row shows failures.

Table 1. Ablation study

	Started Push	Success
Vanilla Policy	50.0 %	27.8 %
No Dropout	66.7 %	27.8 %
No Brightness Aug.	75.0 %	41.7 %
Our Policy	86.1 %	58.3 %

implemented the original model and adapted it to our robotic platform and task. To test the effect of individual changes, we applied our policy once without dropout and once without data augmentation. The vanilla policy and our policy without dropout only achieve a success rate of 27.8 %, which were almost exclusively the trials when the box was located in a middle position and only required a straight push.

The purpose of applying dropout to the end-effector pose input of the network is to put more emphasize on the input images. With the added dropout, the success rate rises to 41.7 %. Brightness augmentation alone did not improve the overall success rate over the vanilla policy. However, the combination of dropout and brightness augmentation achieved a success rate of 58.3 %. We introduced the data augmentation due to changing lighting conditions in the test environment during the demonstrations. For the evaluation we kept the lighting conditions the same.

Qualitative results are presented in Figure 3. The first row shows sequences of successful trials in which the box is pushed to the goal. The second row shows examples of failures. In one case, the robot end-effector slides past the box and the policy loses the target. In the second case, the box is pushed to a location that is not the goal.

5. Conclusion

This paper presented an approach for learning from demonstration using a vision-based solution for robot teleoperation. A hand tracking method was employed to generate commands that control the robot's end-effector as the human operator completes a manipulation task. The set of demonstrations were used to train a deep imitation learning network that learns a policy, enabling the robot to imitate the task. Experiments showed that the introduction of regularization and data augmentation increased the success rate over the baseline method.

For future work, we plan to combine the LfD approach with reinforcement learning in simulation. By starting from the learned policy in simulation, the training time of reinforcement learning approaches can be greatly reduced. Additionally, combining real data with synthetic data collected in simulation mitigates the problem of domain adaptation of pure reinforcement learning methods. Another avenue is to use more high-level knowledge of the scene (e.g. object pose) to make the approach less susceptible to environment changes.

Acknowledgments

This research is partially supported by the Vienna Science and Technology Fund (WWTF), project RALLI (ICT15-045), the Austrian Science Foundation (FWF), project InDex (I3969-N30), and Festo AG & Co. KG.

References

[1] B. Akgun, M. Cakmak, K. Jiang, and A. L. Thomaz. Keyframe-based learning from demonstration. *The Int. Journal of Social Robotics*, 4(4):343–355, 2012.

[2] 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.

[3] R. Bischoff, J. Kurth, G. Schreiber, R. Koeppe, A. Albu-Schäffer, A. Beyer, O. Eiberger, S. Haddadin, A. Stemmer, G. Grunwald, et al. The KUKA-DLR lightweight robot arm-a new reference platform for robotics research and manufacturing. In *Proc. of Int. Symposium on Robotics and German Conference on Robotics*, 2010.

[4] A. Boularias, J. A. Bagnell, and A. Stentz. Learning to manipulate unknown objects in clutter by reinforcement. In *Proc. of AAAI Conf. on Artificial Intelligence*, pages 1336–1342, 2015.

[5] S. Calinon, P. Evrard, E. Gribovskaya, A. Billard, and A. Kheddar. Learning collaborative manipulation tasks by demonstration using a haptic interface. In *Proc. of Int. Conf. on Advanced Robotics*, pages 1–6, 2009.

[6] Y. Duan, M. Andrychowicz, B. Stadie, O. Jonathan Ho, J. Schneider, I. Sutskever, P. Abbeel, and W. Zaremba. One-shot imitation learning. In *Advances in Neural Information Processing Systems 30*, pages 1087–1098, 2017.

[7] F. Gouidis, P. Panteleris, I. Oikonomidis, and A. Argyros. Accurate hand keypoint localization on mobile devices. In *Proc. of IEEE Int. Conf. on Machine Vision Applications*, 2019.

[8] M. Hirschmanner, C. Tsiourti, T. Patten, and M. Vincze. Virtual reality teleoperation of a humanoid robot using markerless human upper body pose imitation. In *Proc. of IEEE-RAS Int. Conf. on Humanoid Robots*, 2019.

[9] D. Huang, S. Nair, D. Xu, Y. Zhu, A. Garg, L. Fei-Fei, S. Savarese, and J. C. Niebles. Neural task graphs: Generalizing to unseen tasks from a single video demonstration. In *Proc. of IEEE/CVF Conf. on Computer Vision and Pattern Recognition*, pages 8557–8566, 2019.

[10] S. James, M. Bloesch, and A. J. Davison. Task-embedded control networks for few-shot imitation learning. In *Proc. of Conf. on Robot Learning*, pages 783–795, 2018.

[11] D. Kalashnikov, A. Irpan, P. Pastor, J. Ibarz, A. Herzog, E. Jang, D. Quillen, E. Holly, M. Kalakrishnan, V. Vanhoucke, and S. Levine. Scalable deep reinforcement learning for vision-based robotic manipulation. In *Proc. of Conf. on Robot Learning*, pages 651–673, 2018.

[12] G. Konidaris, S. Kuindersma, R. Grupen, and A. Barto. Robot learning from demonstration by constructing skill trees. *The Int. Journal of Robotics Research*, 31(3):360–375, 2012.

[13] V. Krüger, D. L. Herzog, S. Baby, A. Ude, and D. Kragic. Learning actions from observations. *IEEE Robotics Automation Magazine*, 17(2):30–43, 2010.

[14] S. Levine, C. Finn, T. Darrell, and P. Abbeel. End-to-end training of deep visuomotor policies. *J. Mach. Learn. Res.*, 17(1):13341373, Jan. 2016.

[15] S. Levine, P. Pastor, A. Krizhevsky, J. Ibarz, and D. Quillen. Learning hand-eye coordination for robotic grasping with deep learning and large-scale data collection. *The Int. Journal of Robotics Research*, 47(4-5):421–436, 2018.

[16] Y. Liu, A. Gupta, P. Abbeel, and S. Levine. Imitation from observation: Learning to imitate behaviors from raw video via context translation. In *Proc. of IEEE Int. Conf. on Robotics and Automation*, pages 1118–1125, 2018.

[17] P. Panteleris, I. Oikonomidis, and A. Argyros. Using a single RGB frame for real time 3D hand pose estimation in the wild. In *Proc. of IEEE Winter Conf. on Applications of Computer Vision*, pages 436–445, 2018.

[18] A. Pashevich, R. Strudel, I. Kalevatykh, I. Laptev, and C. Schmid. Learning to augment synthetic images for Sim2Real policy transfer. In *Proc. of IEEE/RSJ Int. Conf. on Intelligent Robots and Systems*, pages 2651–2657, 2019.

[19] P. Pastor, L. Righetti, M. Kalakrishnan, and S. Schaal. Online movement adaptation based on previous sensor experiences. In *Proc. of IEEE/RSJ Int. Conf. on Intelligent Robots and Systems*, pages 365–371, 2011.

[20] R. Rahmatizadeh, P. Abolghasemi, L. Blni, and S. Levine. Vision-based multi-task manipulation for inexpensive robots using end-to-end learning from demonstration. In *Proc. of IEEE Int. Conf. on Robotics and Automation*, pages 3758–3765, 2018.

[21] S. Schaal. Is imitation learning the route to humanoid robots? *Trends in cognitive sciences*, 3(6):233–242, 1999.

[22] M. Schneider and W. Ertel. Robot learning by demonstration with local Gaussian process regression. In *Proc. of IEEE/RSJ Int. Conf. on Intelligent Robots and Systems*, pages 255–260, 2010.

[23] G. Schreiber, A. Stemmer, and R. Bischoff. The fast research interface for the KUKA lightweight robot. In *IEEE ICRA 2010 Workshop on Innovative Robot Control Architectures for Demanding (Research) Applications – How to Modify and Enhance Commercial Controllers*, 2010.

[24] P. Sermanet, C. Lynch, Y. Chebotar, J. Hsu, E. Jang, S. Schaal, and S. Levine. Time-contrastive networks: Self-supervised learning from video. In *Proc. of IEEE Int. Conf. on Robotics and Automation*, pages 1134–1141, 2018.

[25] D. Victor. Handtrack: A library for prototyping real-time hand tracking interfaces using convolutional neural networks. *GitHub repository*, 2017.

[26] T. Yu, C. Finn, S. Dasari, A. Xie, T. Zhang, P. Abbeel, and S. Levine. One-shot imitation from observing humans via domain-adaptive meta-learning. In *Proc. of Robotics: Science and Systems*, 2018.

[27] 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 *Proc. of IEEE Int. Conf. on Robotics and Automation*, pages 5628–5635, 2018.