HiFaive: Learning Human-Inspired Dexterous Manipulation with the Faive Robotic Hand

Erik Bauer[1], Elvis Nava[1,2,3], and Robert K. Katzschmann[1,2]

***Abstract*— From the beginnings of human evolution, dexterous manipulation skills have been crucial in our everyday lives. Our hands, combined with our cognitive system, give us vast abilities to interact with our surroundings. We strive to give robots the same abilities to empower them as helpful companions that can alleviate our everyday work. Dexterous manipulation is a problem which classical model-based methods have been unable to conquer. We propose an end-to-end learning-based approach leveraging transformer neural networks that learn from human examples obtained by teleoperating the robot. Using the Action Chunking with Transformers framework (ACT), our approach uses a highly integrated learning pipeline that predicts chunks of robot actions based on visual information and the current robot state. We demonstrate successful policy deployments for a simple pick-and-place scenario in real-world experiments using a biomimetic, tendon-driven robotic hand mounted on a robotic arm, using 130 episodes of expert demonstrations collected on the robotic system. These promising first results show the potential of behavior cloning for learning dexterous human-like manipulation as a highly-integrated end-to-end approach for both perception and control. Building up on such learning-based frameworks, we can finally take a step towards the long elusive goal of bringing manipulation skills to embodied intelligence.**

I. Introduction

Human hands have few counterparts in nature with a similarly powerful combination of dexterity and versatility. Development of dexterous manipulation in humanoids is estimated to have emerged as early as 2 million years ago [1]. Human hands have been a subject of intensive study and have been a fascination particularly for the field of humanoid robotics. The first human-like robotic hands (*Belgrade hands*) date back to the 1960s. However, with increased progress in industrial automation, very simple end-effectors became a favoured alternative over dexterous hand designs. Most robots today are deployed in factory floors, where they operate under very controlled conditions, performing specific repetitive tasks.

We can largely attribute this development to the lack of one key element: intelligence. For human-level manipulation, much more is needed than just the hand itself - which is, ultimately, not much more than an outstanding biological end-effector. Instead, what makes dexterous manipulation such a difficult problem lies in the cognitive area. Manipulation is a task that requires tight coordination of nearly all human senses. Particularly, visual information is of high importance for humans to find objects with their hands.

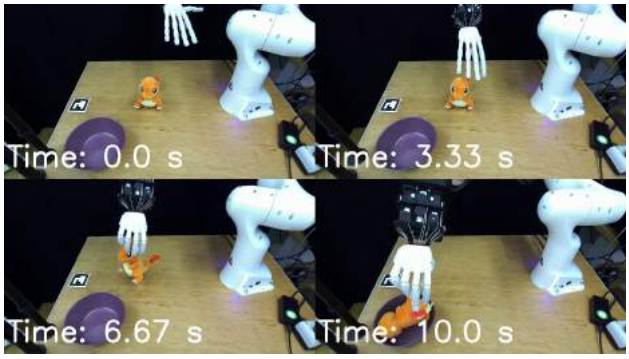

Fig. 1. A successful grasp of a plush toy. The robot is controlled by an *ACT* (Action Chunking with Transformers) [7] model, which predicts future robot states based on visual input from 2 RGB cameras and the current robot state.

Giving such cognitive skills to robots has been a challenging task that has only begun to be solved with the advent of machine learning. The first learning-based approaches enabled manipulation through visual servoing. The main idea of visual servoing is to use either geometry-based or learning-based methods to predict grasping poses for the manipulator based on some visual input [2], [3], [4], [5], [6]. While visual servoing approaches have been shown to work well for simple pick-and-place tasks, their deeper understanding of the world is inherently limited by their ability to only learn grasps, while most other control and planning tasks (for example, what to do with the grasped object) are still being done by relatively classical methods that require explicit programming.

Let us consider the future use case of household assistant robots. Such robots would be employed in a vast amount of scenarios doing a wide variety of tasks. It is infeasible to manually plan for every occasion with classical planning tools. This naturally introduces the idea that instead of just learning *grasps*, we learn *behaviors*. Instead of predicting a single grasping state with a learned model and leaving the rest to classical predefined planning mechanisms, behavior cloning models predict sequences of robot states to successfully complete the task the model was trained on [8], [9], [10], [11], [12], [13], [14], [7]. The key advantage compared to previous visual servoing methods is that with behavior cloning, we can give robots a deeper understanding of their tasks and are able to replace classical task-level planners.

We introduce HiFaive: Learning **H**uman-**i**nspired dexter-

[1]Soft Robotics Lab, ETH Zurich, Switzerland
[2]ETH AI Center, ETH Zurich, Switzerland
[3]Institute of Neuroinformatics, ETH Zurich & University of Zurich, Switzerland
{erbauer, enava, rkk}@ethz.ch

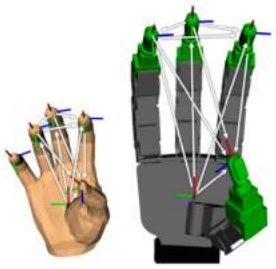

Fig. 2. Visualization of the keyvectors for an Allegro hand shown by Sivakumar et al. [19]. The keyvectors connect the wrist to all fingertips and all fingertips to each other. For the *Faive* hand, we can utilize all 15 keyvectors from the human hand for full retargeting.

ous manipulation with the **Faive** Robotic Hand. The Faive hand was first introduced by Toshimitsu et al. [15], to which we refer for more details on the hand itself. This work presents a full view on deploying a behavior cloning framework on a robot setup, going from system architecture and teleoperation for data collection to training and deploying behavior cloning policies (as shown in Figure 1). Particularly, using a conditional variational autoencoder transformer architecture (ACT) introduced by Zhao et al. [7], we show promising results for manipulating objects with our biomimetic, five-fingered robotic hand (the *Faive* hand) on a robotic arm.

II. Teleoperation and Data Collection

Our main source of training data are expert teleoperations conducted with the robot. An episode of training data consists of the full successful execution of a task where the robot is controlled by a human operator. Biomimetic robotic hands such as ours require sophisticated methods for controlling them. Our system, in total, has 17 controllable degrees of freedom (11 for the hand + 6 for the arm). In this section, we present our solution for real-time teleoperation at 20 Hz, which enables the operator to control both the *Faive* hand and the robotic arm it is mounted on with high precision while being able to perform a wide variety of highly complex movements.

A. Optimization-based hand retargeting

The *Faive* hand has 16 degrees of freedom which can be controlled in its joint space by 11 joint angles. An intuitive way of controlling robotic hands is with our own human hands through the use of hand motion capture. For hand motion capture, we use *FrankMocap* [16] or the *DepthAI* [17] hand tracker for the OAK-D camera to generate hand state estimates (in MANO [18] format) in the form of 21 3D locations of the hand joints. Having captured the human hand state, we need to map from the human hand state to the robot hand state. To map from human hand state $\theta_{human} \in \mathbb{R}^{21\times 3}$ to robot hand state $\theta_{robot} \in \mathbb{R}^{11}$, we use optimization-based retargeting using keyvectors similar to Sivakumar et al. [19].

We define keyvectors K_i^{human}, K_i^{robot} for human and robot hand as vectors from each fingertip to all other fingertips and from the palm to all fingertips (Figure 2). Since both hands have five fingers, this amounts to 15 keyvectors that capture the *relative positions* of the fingers with respect to each other and the palm. Then, we implicitly define the map

$$f : \theta_{human} \mapsto \theta_{robot}$$

through an energy function such that

$$\theta_{robot} = \text{argmin}_{\theta_{robot}} \sum_{i=0}^{15} \left|\left| K_i^{human} - s_i K_i^{robot} \right|\right|_2^2$$

where both the human and robot keyvectors are constructed with the current respective hand states and s_i are scaling factors to match the length of the keyvectors.

We can solve for θ_{robot} by numerical optimization using gradient-descent-like algorithms. We qualitatively found that using *RMSprop* [20] with a high stepsize LR $= 2.5$ for two iterations yielded the best results for precise real-time control at 20 Hz.

B. Arm control

For controlling the 6D pose of the end-effector of our *Franka Emika Panda* robotic arm (position and rotation), we make use of two different methods. One is based upon hand pose estimation by solving the PnP problem with the estimated 3D hand joints and their reprojected pixel coordinates in the image space (we use an *OpenCV* [21] solver with RANSAC). The alternative option is based upon using a *SpaceMouse*, a digital input device for single-handed operation that allows controlling 6 degrees of freedom.

Both of these methods have their advantages: controlling the robot pose with your own human hand pose is intuitive and allows for an easy entrypoint for untrained users. However, it lacks the very high precision of joystick-based controllers such as the *SpaceMouse*. In contrast, using a *SpaceMouse* limits users to more experienced operators, but enables much higher precision, particularly for controlling the rotation of the end-effector.

With both methods, we send twists to the high-level arm controller while low-level control relies on *Ruckig* [22] trajectory generation. To collect our dataset for training, we relied entirely on *SpaceMouse* operation.

III. System Architecture

Our system architecture leverages *ROS Noetic* [23] as middleware to give us the needed flexibility in operating modes (human teleoperation, replaying recordings, deploying learned policies). Furthermore, *ROS* makes data collection significantly easier through its *rosbag* interface. For robot perception, we use two statically mounted cameras: one *OAK-D Pro* for a top view and one *ZED 2i* camera for a front view. We only use the RGB information from both cameras. Another *OAK-D Pro* is used for hand motion capture of a human operator. The architecture is shown in Figure 3.

For collecting data, we use the *rosbag* interface to tap into *ROS* message streams. Then, the raw data is converted to compressed HDF5 files for easy use for training behavior cloning models. We collect a 17-dimensional robot state (11

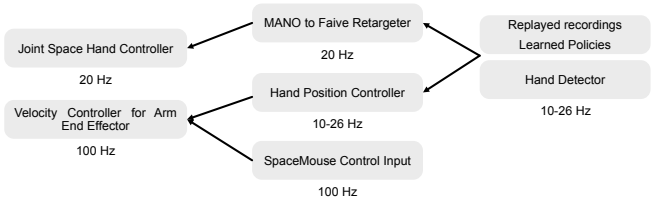

Fig. 3. System architecture for either real-time teleoperation at 20 Hz, replaying recorded data or deploying learned policies. To communicate in between processes, *ROS* is used with simple publisher-subscriber connections. For controlling the arm end-effector, either the human hand pose estimation or the *SpaceMouse* can be used.

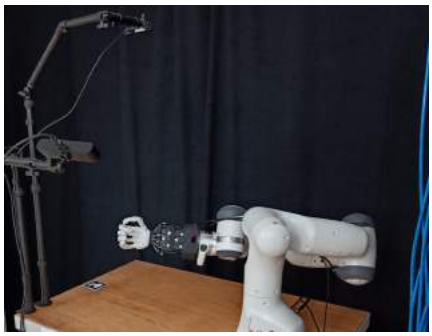

Fig. 4. View of the robot workspace. Two cameras are mounted on arms to enable visual perception. There is an ArUco marker [24] placed on the table to verify a constant position of the cameras.

hand joint states + 6 arm pose dimensions) as well as RGB images from front and top views.

IV. Behavior Cloning

Most recently, there has been a strong research interest in behavior cloning for robotic manipulation, resulting in a variety of different ideas for model architectures. Most models share the fact that they heavily rely on transformer networks [25], which have proven themselves to be highly capable for sequence-to-sequence prediction tasks such as predicting future robot states. In the scope of this work, we adapt the *ACT* (Action Chunking with Transformers, Figure 5) model architecture. To the best of our knowledge, we are the first to adapt this framework for a dexterous robotic hand.

The main differentiating feature of *ACT* is that instead of only predicting single robot states as most other behavior cloning frameworks do, it predicts chunks of k actions. In closed loop operation, after each model query, temporal ensembling is employed to combine the current k predictions with the $k-1$ overlapping state predictions from the last timestep.

V. Results

In the following, we present the results achieved with the previously described architecture. Within the scope of this work, it is our aim to show a system-level approach for enabling behavior cloning research, whereas we reserve a stronger focus on the behavior cloning architectures used (e.g. *ACT*) for future research.

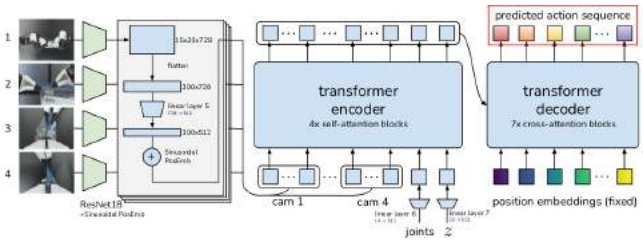

Fig. 5. The *ACT* architecture as shown by Zhao et al. [7]. It uses a conditional variational autoencoder with transformers to learn from multiple camera inputs and the robot state to predict chunks of k future robot states.

A. Data collection and training

We collected data for two pick-and-place tasks with different difficulty levels and trained two single-task models using the collected data.

1) Plush toy pick-and-place: We collected 130 episodes of expert teleoperations, picking the plush toy and placing it in a plastic bowl. In this task, the robot's initial pose was randomized, however, the plush toy and the bowl remained at an approximately constant position.

2) Rubik's cube pick-and-place: We collected 214 episodes of expert teleoperations, picking the Rubik's cube and placing it in a plastic bowl. In this task, the robot's initial pose, the Rubik's cube initial pose and the bowl position was randomized, significantly increasing the difficulty compared to the non-randomized task.

3) Training: For each task, we trained the *ACT* model architecture using only the collected examples from that task as opposed to using all training data as in multi-task approaches. Model hyperparameters were mostly taken over from the original *ACT* implementation (i.e. LR $= 1 \cdot 10^{-5}$,, training for 8000 epochs, prediction chunk size of $k = 100$). A full ablation study concerned with the adaption of *ACT* on our robot setup remains for future work.

B. Manipulation Results

Deploying the model, we were able to observe successful results for the first task (plush toy pick-and-place) as shown in Figure 6. The model managed to grasp the target object from different initial positions and successfully transport it to the bowl. If the task was not completed successfully, the typical failure mode was that the model attempted to grasp the object, however, failed to successfully pick the object. Throughout all attempts, the model consistently moved towards the target object in grasping attempts.

For the second task with higher difficulty (Rubik's cube pick-and-place), despite the model showing some understanding of the task, results were largely unsuccessful. The typical failure mode consisted of the model moving the robot towards the target object, however, failing to grasp it due to lacking precision and then retrying unsuccessful grasping attempts. Depending on the initial position, the model would also sometimes fail to move precisely towards the target object. While the manipulation was not successful, we can still see that to some degree, the model learned to

Fig. 6. A successful policy execution: the robot setup picks up the plush toy and places it into the bowl. The model can successfully fulfill this task from different initial robot states.

correlate visual features with future actions. Likely, more training examples are needed for successful manipulation, given the increased complexity with completely randomized initial, target object and goal states.

VI. Conclusion

In conclusion, in this work we have presented a highly integrated, end-to-end approach for learning human-inspired dexterous manipulation with a robotic hand. Presenting teleoperation methods and our system architecture, we provide a system-level overview of our robotic system and its operating modes. We motivate our choice of learning framework and show a successful working example of our entire pipeline, integrating both perception and control into one learning-based control unit. Finally, in real-world experiments, we have demonstrated successful policy deployment for a simple pick-and-place task and show present data-constrained limitations for more executing more complex tasks. We lay out our future research directions with a stronger focus on investigating current behavior cloning models and introducing modifications such as novel input modalities like force sensors for training.

References

[1] F. A. Karakostis, D. F. B. Haeufle, I. Anastopoulou, K. Moraitis, G. Hotz, V. Tourloukis, and K. Harvati, "Biomechanics of the human thumb and the evolution of dexterity," *Current Biology*, vol. 31, pp. 1317 – 1325.e8, 2021. [Online]. Available: https://api.semanticscholar.org/CorpusID:231723181

[2] Y. Jiang, S. Moseson, and A. Saxena, "Efficient grasping from rgbd images: Learning using a new rectangle representation," *2011 IEEE International Conference on Robotics and Automation*, pp. 3304–3311, 2011.

[3] B. S. Zapata-Impata, "Using geometry to detect grasping points on 3d unknown point cloud," in *ICINCO*, 2017.

[4] B. S. Zapata-Impata, P. Gil, J. Pomares, and F. Torres, "Fast geometry-based computation of grasping points on three-dimensional point clouds," *International Journal of Advanced Robotic Systems*, vol. 16, 2019.

[5] A. Mousavian, C. Eppner, and D. Fox, "6-dof graspnet: Variational grasp generation for object manipulation," *2019 IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 2901–2910, 2019.

[6] S. Levine, C. Finn, T. Darrell, and P. Abbeel, "End-to-end training of deep visuomotor policies," *ArXiv*, vol. abs/1504.00702, 2016.

[7] T. Zhao, V. Kumar, S. Levine, and C. Finn, "Learning fine-grained bimanual manipulation with low-cost hardware," 2023.

[8] C. Finn, T. Yu, T. Zhang, P. Abbeel, and S. Levine, "One-shot visual imitation learning via meta-learning," *ArXiv*, vol. abs/1709.04905, 2017.

[9] S. James, M. Bloesch, and A. J. Davison, "Task-embedded control networks for few-shot imitation learning," *ArXiv*, vol. abs/1810.03237, 2018.

[10] S. Nair, A. Rajeswaran, V. Kumar, C. Finn, and A. Gupta, "R3m: A universal visual representation for robot manipulation," 2022.

[11] A. Brohan, N. Brown, J. Carbajal, Y. Chebotar, J. Dabis, C. Finn, K. Gopalakrishnan, K. Hausman, A. Herzog, J. Hsu, J. Ibarz, B. Ichter, A. Irpan, T. Jackson, S. Jesmonth, N. J. Joshi, R. Julian, D. Kalashnikov, Y. Kuang, I. Leal, K.-H. Lee, S. Levine, Y. Lu, U. Malla, D. Manjunath, I. Mordatch, O. Nachum, C. Parada, J. Peralta, E. Perez, K. Pertsch, J. Quiambao, K. Rao, M. Ryoo, G. Salazar, P. Sanketi, K. Sayed, J. Singh, S. Sontakke, A. Stone, C. Tan, H. Tran, V. Vanhoucke, S. Vega, Q. Vuong, F. Xia, T. Xiao, P. Xu, S. Xu, T. Yu, and B. Zitkovich, "Rt-1: Robotics transformer for real-world control at scale," 2022.

[12] A. Brohan, N. Brown, J. Carbajal, Y. Chebotar, K. Choromanski, T. Ding, D. Driess, C. Finn, P. R. Florence, C. Fu, M. G. Arenas, K. Gopalakrishnan, K. Han, K. Hausman, A. Herzog, J. Hsu, B. Ichter, A. Irpan, N. J. Joshi, R. C. Julian, D. Kalashnikov, Y. Kuang, I. Leal, S. Levine, H. Michalewski, I. Mordatch, K. Pertsch, K. Rao, K. Reymann, M. S. Ryoo, G. Salazar, P. R. Sanketi, P. Sermanet, J. Singh, A. Singh, R. Soricut, H. Tran, V. Vanhoucke, Q. H. Vuong, A. Wahid, S. Welker, P. Wohlhart, T. Xiao, T. Yu, and B. Zitkovich, "Rt-2: Vision-language-action models transfer web knowledge to robotic control," *ArXiv*, vol. abs/2307.15818, 2023. [Online]. Available: https://api.semanticscholar.org/CorpusID:260293142

[13] K. Shaw, S. Bahl, and D. Pathak, "Videodex: Learning dexterity from internet videos," 2022.

[14] C. Wang, L. Fan, J. Sun, R. Zhang, L. Fei-Fei, D. Xu, Y. Zhu, and A. Anandkumar, "Mimicplay: Long-horizon imitation learning by watching human play," 2023.

[15] Y. Toshimitsu, B. Forrai, B. G. Cangan, U. Steger, M. Knecht, S. Weirich, and R. K. Katzschmann, "Getting the ball rolling: Learning a dexterous policy for a biomimetic tendon-driven hand with rolling contact joints," *ArXiv*, vol. abs/2308.02453, 2023. [Online]. Available: https://api.semanticscholar.org/CorpusID:260611411

[16] Y. Rong, T. Shiratori, and H. Joo, "Frankmocap: Fast monocular 3d hand and body motion capture by regression and integration," *CoRR*, vol. abs/2008.08324, 2020. [Online]. Available: https://arxiv.org/abs/2008.08324

[17] geaxgx, "DepthAI Hand Tracker," 4 2023. [Online]. Available: https://github.com/geaxgx/depthai_hand_tracker

[18] J. Romero, D. Tzionas, and M. J. Black, "Embodied hands," *ACM Transactions on Graphics (TOG)*, vol. 36, pp. 1 – 17, 2017. [Online]. Available: https://api.semanticscholar.org/CorpusID:245838515

[19] A. Sivakumar, K. Shaw, and D. Pathak, "Robotic telekinesis: Learning a robotic hand imitator by watching humans on youtube," *ArXiv*, vol. abs/2202.10448, 2022.

[20] T. Tieleman, G. Hinton *et al.*, "Lecture 6.5-rmsprop: Divide the gradient by a running average of its recent magnitude," *COURSERA: Neural networks for machine learning*, vol. 4, no. 2, pp. 26–31, 2012.

[21] G. Bradski, "The OpenCV Library," *Dr. Dobb's Journal of Software Tools*, 2000.

[22] L. Berscheid and T. Kröger, "Jerk-limited real-time trajectory generation with arbitrary target states," *ArXiv*, vol. abs/2105.04830, 2021. [Online]. Available: https://api.semanticscholar.org/CorpusID:234357718

[23] M. Quigley, B. Gerkey, K. Conley, J. Faust, T. Foote, J. Leibs, E. Berger, R. Wheeler, and A. Ng, "Ros: an open-source robot operating system," in *Proc. of the IEEE Intl. Conf. on Robotics and Automation (ICRA) Workshop on Open Source Robotics*, Kobe, Japan, May 2009.

[24] S. Garrido-Jurado, R. Muñoz-Salinas, F. J. Madrid-Cuevas, and M. J. Marín-Jiménez, "Automatic generation and detection of highly reliable fiducial markers under occlusion," *Pattern Recognit.*, vol. 47, pp. 2280–2292, 2014.

[25] A. Vaswani, N. M. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, and I. Polosukhin, "Attention is all you need," *ArXiv*, vol. abs/1706.03762, 2017.