Autonomous Robots (2025) 49:5
https://doi.org/10.1007/s10514-024-10188-y

View: visual imitation learning with waypoints

Ananth Jonnavittula[1] · **Sagar Parekh**[1] · **Dylan P. Losey**[1]

Received: 26 April 2024 / Accepted: 25 December 2024 / Published online: 18 January 2025

Abstract

Robots can use visual imitation learning (VIL) to learn manipulation tasks from video demonstrations. However, translating visual observations into actionable robot policies is challenging due to the high-dimensional nature of video data. This challenge is further exacerbated by the morphological differences between humans and robots, especially when the video demonstrations feature humans performing tasks. To address these problems we introduce **V**isual **I**mitation l**E**arning with **W**aypoints (VIEW), an algorithm that significantly enhances the sample efficiency of human-to-robot VIL. VIEW achieves this efficiency using a multi-pronged approach: extracting a condensed prior trajectory that captures the demonstrator's intent, employing an agent-agnostic reward function for feedback on the robot's actions, and utilizing an exploration algorithm that efficiently samples around waypoints in the extracted trajectory. VIEW also segments the human trajectory into grasp and task phases to further accelerate learning efficiency. Through comprehensive simulations and real-world experiments, VIEW demonstrates improved performance compared to current state-of-the-art VIL methods. VIEW enables robots to learn manipulation tasks involving multiple objects from arbitrarily long video demonstrations. Additionally, it can learn standard manipulation tasks such as pushing or moving objects from a single video demonstration in under 30 min, with fewer than 20 real-world rollouts. Code and videos here: https://collab.me.vt.edu/view/

Keywords Visual imitation learning · Deep learning · Few-shot learning

1 Introduction

Imagine teaching a person to pick up a cup placed on a table. The quickest method is often to physically demonstrate this task. Through physical demonstration, the observer can discern which object to pick up and how to manipulate that object. Humans efficiently learn everyday tasks in this way, including moving items, making tea, or stirring the contents of a pan.

Teaching robots these same tasks, however, proves to be more cumbersome. Typically, robots employ either imitation learning (IL) or reinforcement learning (RL) methods. IL generally requires many demonstrations from humans to obtain effective policies (Fang et al., 2019; Hussein et al., 2017). During this process, the human teacher often needs to kinesthetically guide or teleoperate the robot to show it exactly what actions it should take. On the other hand, RL methods require a substantial number of rollouts for robots to perform even simple tasks (Kober et al., 2013; Morales et al., 2021). Additionally, defining appropriate reward functions for reinforcement learning poses a challenge, particularly in unstructured environments (Dulac-Arnold et al., 2021).

In this paper we therefore study how robots can learn tasks by *watching* humans. The human provides a demonstration directly in the environment (e.g., physically picking up a cup), and the robot collects an RGB-D video of the human's demonstration. Our objective is for the robot to leverage this single video to learn the task and correctly manipulate the same object. The primary issue here lies in the overload of information conveyed by video demonstrations. Each video is comprised of thousands of image frames, and each frame contains numerous pixels. These raw pixel values — when seen in isolation — are not sufficient for the robot to determine what actions it should take (i.e., how to move the robot arm). Consequently, robots that learn from video demonstra-

✉ Ananth Jonnavittula
ananth@vt.edu

Sagar Parekh
sagarp@vt.edu

Dylan P. Losey
losey@vt.edu

[1] Mechanical Engineering Department, Virginia Tech, Blacksburg, USA

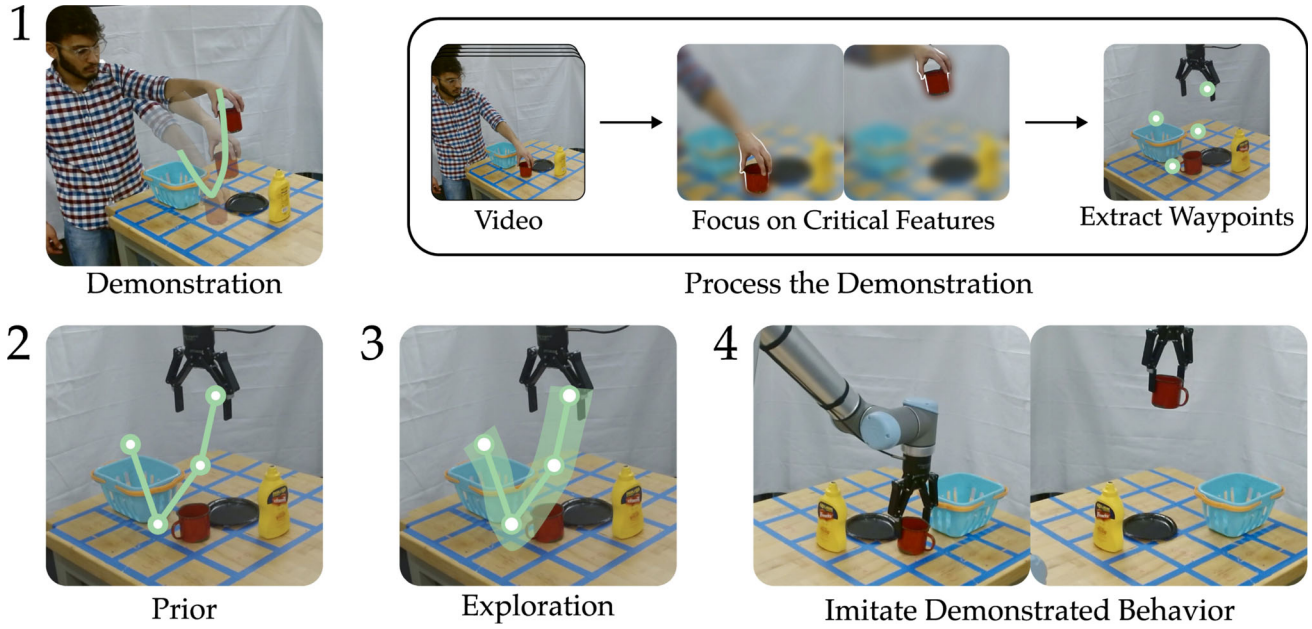

Fig. 1 Robot learning from visual demonstration. 1) A human demonstrates the task directly in the environment: here we use the example of picking up a cup. Under our proposed approach, the robot processes a single video of that demonstration to selectively focus on important features such as the human hand and the manipulated object. From these trajectories the robot obtains *waypoints* that capture the critical parts of the task (e.g., grasping the cup). 2) These extracted waypoints serve as a prior for the correct robot trajectory. 3) In practice, simply executing this prior rarely leads to task success due in part to the morphological differences between human and robot (in this case, the robot misses the cup entirely). Therefore, the robot must explore in a region around the initial waypoints to iteratively improve its trajectory. 4) After repetitively interacting with the environment, the robot learns to successfully imitate the behavior demonstrated in the human video

tions must extract pertinent information from a large amount of data.

To address this fundamental problem, our hypothesis is that robots do not need to reason over *all* the video data. Consider our motivating example of picking up a cup: when humans learn by watching other humans, we do not focus on environmental clutter or extraneous details. Instead, we just need to see where the human grabs the cup and how they carry it. At a high level, we can think about these critical parts of the task as *waypoints*: the cup's initial position, the human's hand configuration when grasping, key frames along the cup's motion, and where the human finally places the cup. Robots that can learn these waypoints from the human's video demonstration will be able to perform the overall task without having to reason over every single aspect of every video frame.

We leverage this hypothesis to develop VIEW: Visual Imitation lEarning with Waypoints (see Fig. 1). Our approach starts with a video of the human performing their desired manipulation task. We then process that video to get an initial trajectory (e.g., a best guess) of how the robot should complete the same task. To obtain this initial guess we extract the human's hand trajectory, and then autonomously identify the critical waypoints along that trajectory in visual and Cartesian spaces. In an idealized scenario, the robot could directly execute this initial trajectory and complete the task. However, the initial trajectory almost always fails because of (a) the morphological differences between human demonstrator and robot learner and (b) the sensor noise in the initial RGB-D video. Returning to our cup example, we often find that — even though the video shows the human picking up the cup — the robot's extracted trajectory misses that cup entirely.

Accordingly, the second part of VIEW focuses on iteratively improving the robot's prior and correctly completing the task. We develop sampling strategies so that the robot can intelligently explore around its waypoints. This includes waypoints where the robot needs to grasp an object (e.g., pick up the cup) and waypoints where the robot is manipulating that object (e.g., carrying the cup to a goal location). Again, we rely on our hypothesis: instead of reasoning about every aspect of the video, we focus on the object's location in the waypoint frames. This leads to an iterative learning process where the robot corrects its initial trajectory and eventually completes the original task shown in the video. Overall, VIEW enables robot arms to efficiently learn manipulation tasks such as picking, pushing, or moving objects, requiring fewer than 20 real-world trials and less than 30 minutes from demonstration collection to successful task execution. Additionally, our method enables robots to learn from long horizon videos that involve a combination of tasks — such as

moving a cup and dispensing tea into it, or placing multiple objects in a pan — using no additional information besides the initial human demonstration video.

Overall, we make the following contributions:

Condensed prior extraction We present a new approach for distilling a prior from video demonstrations that accurately reflects the human demonstrator's intent. We achieve this by extracting a concise set of waypoints that capture the human hand trajectory and its interaction with objects.

Agent agnostic rewards For sample efficient exploration, the robot requires effective feedback when exploring around the waypoints in the extracted prior. To provide this feedback, we propose an agent-agnostic reward function. Our reward model only focuses on the critical components of the task — i.e., movement of the object — regardless of which agent performs the task.

Sample efficient exploration Given morphological disparities and noise introduced during prior extraction, directly replicating the human trajectory is impractical. We introduce an algorithm that segments the prior into *grasping* and *task* phases, sequentially focusing on locating the object and replicating its movement.

Few shot adaptation While our approach can bridge the embodiment gap between human and robot through efficient exploration around the human prior, each new task requires starting from scratch. However, the robot gains valuable insights into the morphological differences and camera noise with each solved task. By integrating a residual learner that leverages this insight with our prior extraction, we demonstrate that the robot can achieve few-shot learning on new tasks.

Comparing VIEW to baselines We conduct a comparative analysis of our method against existing state-of-the-art approaches in visual imitation learning. Additionally, an ablation study is performed in a simulated environment to underscore the significance of each component within our framework. These comparative analyses and ablation studies collectively demonstrate our method's efficacy in enabling robots to quickly imitate a wide range of tasks based on video demonstrations.

2 Related work

We study how robots can efficiently learn to replicate a task based on a single video demonstration. Our approach builds upon existing learning from demonstration methods, particularly those that use videos, waypoints, and human activity recognition.

Learning from demonstrations (LfD) LfD is a general learning framework (Pomerleau, 1991; Schaal, 1996) that is used across domains such as autonomous driving (Pan et al., 2020; Chen et al., 2019), robotics (Jonnavittula and Losey, 2021, ?; Jonnavittula et al., 2024; Mehta and Losey, 2023; Ratliff et al., 2007), and video games (Amiranashvili et al., 2020; Schäfer et al., 2023; Scheller et al., 2020). In robotics, LfD has been employed to learn from teleoperated expert demonstrations (Jonnavittula et al., 2024; Ross et al., 2011; Kelly et al., 2019; Menda et al., 2019), extended to include imperfect demonstrations (Jonnavittula and Losey, 2021; Brown et al., 2020, 2019), and combined with other modalities such as preferences (Mehta and Losey, 2023; Taranovic et al., 2022; Habibian et al., 2022) or language (Shi et al., 2024; Lynch and Sermanet, 2020; Liu et al., 2024). A significant aspect of LfD in robotics involves the sourcing of demonstrations, predominantly obtained from human actions within the agent's environment. For example, in autonomous driving, demonstrations encompass steering controls similar to those the agent uses (Pan et al., 2020), while in robotic manipulation, demonstrations are acquired via direct teleoperation (Jonnavittula and Losey, 2021) or kinesthetic teaching (Mehta and Losey, 2023). This reliance on human provided demonstrations presents certain challenges, especially in robotics. Humans primarily use their hands for manipulation, whereas robots utilize end-effectors with distinct morphologies. This fundamental discrepancy limits the feasibility and diversity of the training data collected for robot learning.

To address the morphological disparities between humans and robots, some researchers have advocated for the use of tools such as reacher-grabbers that resemble grippers commonly employed in robotics (Pari et al., 2021; Shafiullah et al., 2023; Song et al., 2020; Young et al., 2021), or utilizing camera angles that reduce the effects of hand to gripper morphology (Kim et al., 2023; Duan et al., 2023). These tools facilitate the recording of demonstrations that can be more easily translated into actionable robot policies, without the need for teleoperation. While these approaches have proven effective, they do not ameliorate the underlying limitation: the demonstrations are inherently restricted in scope due to the specialized interface. In response to this challenge, there has been a shift towards compiling extensive robot demonstration datasets, like Open-X (Padalkar et al., 2023), aiming to establish a foundational resource akin to ImageNet for robot learning. But these datasets overlook the vast reservoir of already existing human video demonstrations, which could significantly expedite the learning process for robots. VIEW builds upon prior LfD works by learning from human demonstrations. However, VIEW focuses on learning directly from human videos, and does not rely on kinesthetic demonstrations, teleoperated inputs, or intermediary tools.

Learning from video demonstrations There has been a parallel research focus on teaching robots with videos of robots performing the desired task (Cetin and Celiktutan, 2021; Rafailov et al., 2021; Wen et al., 2021, 2022). In these methods a human teleoperates the robot in the video

demonstration (i.e., *the video demonstration is of the robot completing the task*), and the robot learns to imitate the resulting RGB-D video. These methods tackle the challenges that arise from a lack of explicit action information (Cetin and Celiktutan, 2021; Wen et al., 2021, 2022). A significant aspect of this research is the reduction of data complexity through an emphasis on keyframes (Wen et al., 2021, 2022), aiming to simplify robot learning by concentrating on achieving these specific frames. Nevertheless, these methods necessitate a large number of trials with the robot (Cetin and Celiktutan, 2021), and do not solve the key problem of obtaining generalized video demonstrations from humans or other agents.

Alternatively, some researchers have explored learning from *human video demonstrations* directly, with considerable effort dedicated to transforming human videos into a format applicable to the robot's domain (Smith et al., 2020; Li et al., 2021; Xiong et al., 2021; Sharma et al., 2019). To facilitate this transformation, these methods utilize cycle consistency networks (Isola et al., 2017) to translate human videos into equivalent robot videos (Xiong et al., 2021; Sharma et al., 2019). Once videos are translated, they extract key points from the videos, which serve as the basis for learning (Xiong et al., 2021; Liu et al., 2018). However, a significant drawback of this approach is the necessity for a vast collection of videos, showcasing both humans and robots performing tasks. This requirement poses a substantial limitation, adversely affecting the scalability of such methods.

Several approaches closely align with our method, focusing on *human-to-robot imitation* learning (Lee et al., 2022; Jain et al., 2024; Jin et al., 2020; Sieb et al., 2020; Alakuijala et al., 2023; Sermanet et al., 2017; Chane-Sane et al., 2023; Patel et al., 2022; Shaw et al., 2024; Bahl et al., 2022). These methods extract meaningful representations of a task from videos (Chane-Sane et al., 2023) or use neural networks to learn reward functions from the videos to facilitate reinforcement learning (Sermanet et al., 2017; Alakuijala et al., 2023). Despite the promise shown by many of these methods, they share a common challenge: the necessity for a substantial number of robot rollouts in real-world scenarios to learn tasks effectively. One particularly similar approach here is WHIRL (Bahl et al., 2022), which mirrors our method but employs a variational autoencoder-based exploration exploitation strategy. As we will show, WHIRL requires a large number of rollouts to converge, and struggles to scale for long horizon tasks. Additionally, this approach also requires video inpainting (Lee et al., 2019), where the human must be removed from each frame in the demonstration video, and the robot must be removed from each rollout frame before reward computation. Given that inpainting is highly GPU-intensive, this process can require powerful GPUs and lead to substantial wait times between rollouts. Our proposed method, VIEW, addresses these challenges by segmenting and sequentially solving the task, reducing computational demands. In our experiments, we will directly compare VIEW and WHIRL in terms of task success and training time to highlight these improvements.

Waypoint-based learning While the methods discussed so far primarily focus on learning action policies directly from demonstrations, a growing trend in robotics is a shift towards teaching robots to reach designated waypoints. This approach is gaining traction, particularly because it aligns well with the use of separate planning and control algorithms, allowing low-level controllers to reach goals set by high-level planners. This concept has seen application in reinforcement learning for tasks such as object pick-and-place and door opening (Mehta et al., 2024). However, the success of these algorithms hinges on the creation of meticulous reward functions to guide learning. In the domain of imitation learning, it has facilitated the learning of intricate tasks, such as operating a coffee machine (Shi et al., 2023). Nevertheless, similar to methods discussed on learning from video demonstrations, Shi et al. (Shi et al., 2023) learn from a video demonstration of the robot performing the task. This distinction is crucial, as robot demonstrations bypass the morphological differences encountered when learning from human videos. Our approach to prior compression bears similarities to these waypoint-focused methods. However, it differs in that VIEW learns directly from a single human video, where the human physically performs the task without a robot.

Human activity recognition Many of the methods discussed above rely on human intent and activity recognition for enabling robots to understand object affordance. Within the realm of robot manipulation, numerous studies have proposed methods that use annotated video datasets such as SomethingSomething (Goyal et al., 2017), YouCook (Das et al., 2013), ActivityNet (Caba Heilbron et al., 2015), or the 100 Days of Hands (100DOH) (Shan et al., 2020). The 100DOH dataset is particularly valuable due to its detailed object interaction annotations. Building upon prior works, our approach utilizes the 100DOH framework to extract crucial data on hand positioning and interactions with objects.

In contrast to the many of the methods discussed here, VIEW distinguishes itself by focusing on sample-efficient learning directly from human videos. Our approach aims to teach robots manipulation tasks — such as picking or moving objects — with minimal human supervision. The only interaction required from the human is providing a single video demonstration.

3 Problem statement

We consider single object manipulation tasks in unstructured environments. First a human teacher physically demonstrates

their desired task within the robot's workspace. During this demonstration the robot is moved out of the way (i.e., the human does not interact with the robot) and the robot records the human's behavior with a stationary RGB-D camera. After the demonstration is complete the video is provided to the robot, and the robot must learn how to replicate the same task based on this single video. We highlight that the human and robot have morphological differences — e.g., the human's hand is different from the robot's gripper — and so the way the human performed the task may not transfer directly to the robot arm.

Environment We formulate the robot's environment as a Markov Decision Process without rewards: $\mathcal{M} = \langle \mathcal{S}, \mathcal{A}, \mathcal{T} \rangle$. The robot's end-effector position in Cartesian space is its state s, and the robot's workspace becomes its state space $\mathcal{S}$. In every state the robot can take an action $a \in \mathcal{A}$ which is the end-effector velocity. This action moves the robot to a new state s' based on the environment transition probability $\mathcal{T}(s' \mid s, a)$. We assume the environment remains unchanged between the human demonstration and the robot's learning activity; that is, all objects maintain their positions. Additionally, we only consider scenarios where the environment is captured from a fixed camera perspective.

Video Demonstration The robot learns from a single video demonstration (V_i) of a human performing the task (τ_i). This video is captured using a stationary RGB-D camera that is set up such that the human and the object they manipulate are visible at all times. The human does not interact with the robot beyond providing this video. Although the human only provides one video for task τ_i, they may provide demonstrations for multiple tasks: e.g., the robot could receive a set of n videos $V_1, \ldots, V_n$ for n different tasks $\tau_1, \ldots, \tau_n$. The robot's objective is to map each video into a trajectory that completes the demonstrated task.

Tasks Our focus is primarily on tasks that involve manipulating a single object at a time. We consider tasks such as picking, pushing, or moving an object within the robot's workspace. While multi-object tasks are allowed, they must be decomposed into multiple sequential single-object manipulation tasks, similar to those mentioned above.

4 VIEW

Our approach to imitation learning from video demonstrations relies on our intuition that efficient learning requires focusing on critical waypoints. In this section we discuss our approach that consists of three main parts: first, extracting which object to pay attention to, how this object moves throughout the task. Second, designing a robust reward signal that compares the robot's behavior to the human's behavior. Third, exploring around the extracted waypoints in a sample-efficient manner. We refer to our method as **VIEW: V**isual **I**mitation l**E**arning with **W**aypoints. Refer to Fig. 2 for an overview.

4.1 Prior extraction

Since VIEW relies on information about both hand and object movements, we divide the extraction process into two components: Hand Trajectory Extraction and Object Trajectory Extraction. A summary of our overall prior extraction method can be found in Fig. 3.

Hand Trajectory Extraction Prior methods have extensively addressed the extraction of hand trajectories from video demonstrations, often leveraging open-source neural networks for this purpose (Bahl et al., 2022; Wang et al., 2020; Zhang et al., 2020). In our approach (see Fig. 3), we analyze each frame (v_t) in a video (V) using the 100 Days of Hands (100DOH) model (Shan et al., 2020). This model helps us identify the hand's location and whether it is interacting with any objects via bounding box coordinates $(b^h_{x_t}, b^h_{y_t})$ and contact information (c_t). A bounding box alone can be ambiguous with respect to hand orientation and direction. To resolve this ambiguity, we further refine our coordinates with the MANO hand model (Romero et al., 2017) to pinpoint the human's wrist position $(p^h_{x_t}, p^h_{y_t})$. We convert the 2D image coordinates into 3D world coordinates (x^h_t, y^h_t, z^h_t) using depth information from the camera. We use *waypoint* to refer to the 3D world coordinates of the human hand or the object.

So far our methodology bears a strong resemblance to that used in WHIRL (Bahl et al., 2022). However, WHIRL overlooks a significant issue: the abundance of points along the extracted trajectory. For instance, a mere ten-second demonstration recorded at 60 frames per second yields a total of 600 frames. While this amount of visual and trajectory data appears substantial, upon closer inspection, much of it is redundant. Returning to our example of picking up a cup, the video contains crucial waypoints such as the initial hand position and the cup's grasp position, but it also includes several redundant frames that interpolate between these key waypoints. This principle extends to various manipulation tasks; for instance, stirring a pan necessitates waypoints depicting the start location, spatula grasp, stirring locations, and final place location. All other intermediate points can be discarded.

To take advantage of this redundancy, we apply a trajectory compression algorithm called Spatial Quality Simplification Heuristic - Extended (SQUISHE) (Muckell et al., 2014). SQUISHE enables users to prioritize key waypoints by setting a maximum allowable error or target count, selectively removing points that do not affect the trajectory shape. To achieve this, it minimizes synchronized Euclidean distance by calculating the difference between each waypoint and its interpolated position, pruning points that fall within

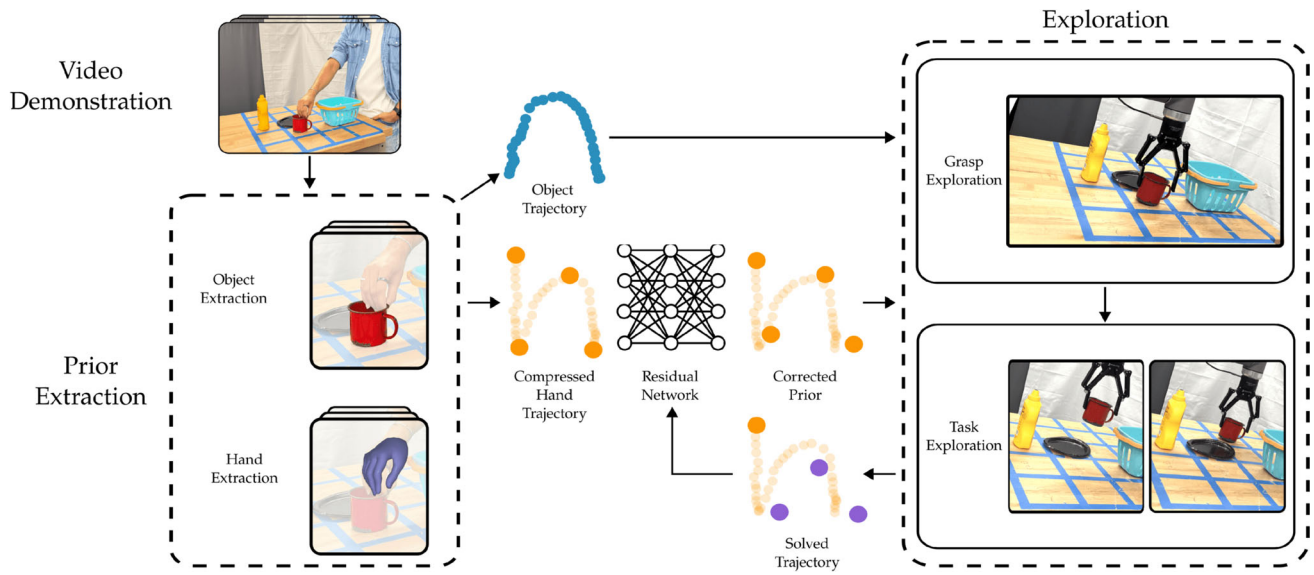

Fig. 2 Outline of VIEW, our proposed method for human-to-robot visual imitation learning. (Top Left) VIEW begins with a single video demonstration of a task. (Bottom Left) From this video we extract the object of interest, its trajectory, and the human's human trajectory. (Middle) We then perform compression to obtain a trajectory prior — a sequence of waypoints the robot arm should interpolate between to complete the task. Unfortunately, this initial trajectory is often imprecise due to the differences between human hands and robot grippers, as well as noise in the extraction process. We therefore refine the prior using a residual network, which is trained on previous tasks to de-noise the current data. (Right) The de-noised trajectory is then segmented into two phases: grasp exploration and task exploration. (Top Right) During grasp exploration, the robot determines how to pick up the object by modifying the pick point in its trajectory. (Bottom Right) Following a successful grasp, the robot proceeds to task exploration, where it simultaneously corrects the remaining waypoints of the trajectory. After completing exploration, the robot synthesizes a complete trajectory. (Middle) This solved trajectory, alongside the prior trajectory, is used to further train the residual network, thus enhancing the performance of our method in future tasks

an acceptable error range. For instance, if the movement between the hand's initial position and the cup grasp can be interpolated, SQUISHE removes intermediate waypoints. By condensing the trajectory in this way, we retain only significant trajectory changes, such as shifts in hand direction or contact with objects, often reducing the length from over 300 points to just 3 or 4.

Object Trajectory Extraction Thus far, we have focused on extracting a concise prior trajectory from the human's hand movements. However, merely mimicking human actions is not sufficient for the robot to solve the task. In reality, the critical aspect of these demonstrations lies in understanding how the human is interacting with and moving objects. Revisiting our cup example, the focus should not be on repeating the human's hand movements; instead, the robot must learn to move the cup to the correct location.

To extract the object's trajectory, we use the hand interactions detected by the 100DOH model to identify frames where the human's hand is in contact (c_t) with an object. We then create anchor boxes of varying sizes, similar to region proposal networks in image detection (Ren et al., 2016). Using these anchor boxes we extract objects that are in close proximity to the human's hand at points of interaction. While there may be frames where the human hand is in proximity to multiple objects, we hypothesize that the majority of the frames will only contain the human's intended object (tag).

Once the object is identified, we generate its trajectory (ζ_h^o) in the video demonstration. We accomplish this by generating bounding boxes around the intended object in pixel coordinates ($p_{x_t}^o, p_{y_t}^o$). We then apply the same deprojection technique used in hand trajectory extraction and use the depth information (δ_t) to translate two-dimensional image frame coordinates into three-dimensional world coordinates (x_t^o, y_t^o, z_t^o). This process outputs a 3D trajectory of the intended object, capturing its movement throughout the demonstration. Refer to Fig. 3 for a summary.

Overall our extracted prior provides us with three key pieces of information: a condensed trajectory representing the human's visited waypoints (ξ^h), a label identifying the object of interest (tag), and a trajectory indicating the object's movement (ξ_h^o), as summarized in Algorithm 2 in Appendix 1.

4.2 Agent-agnostic rewards

After we get the human's hand trajectory (ξ^h) from the video demonstration, the robot executes this trajectory in the environment to try and solve the task (i.e., the human's hand

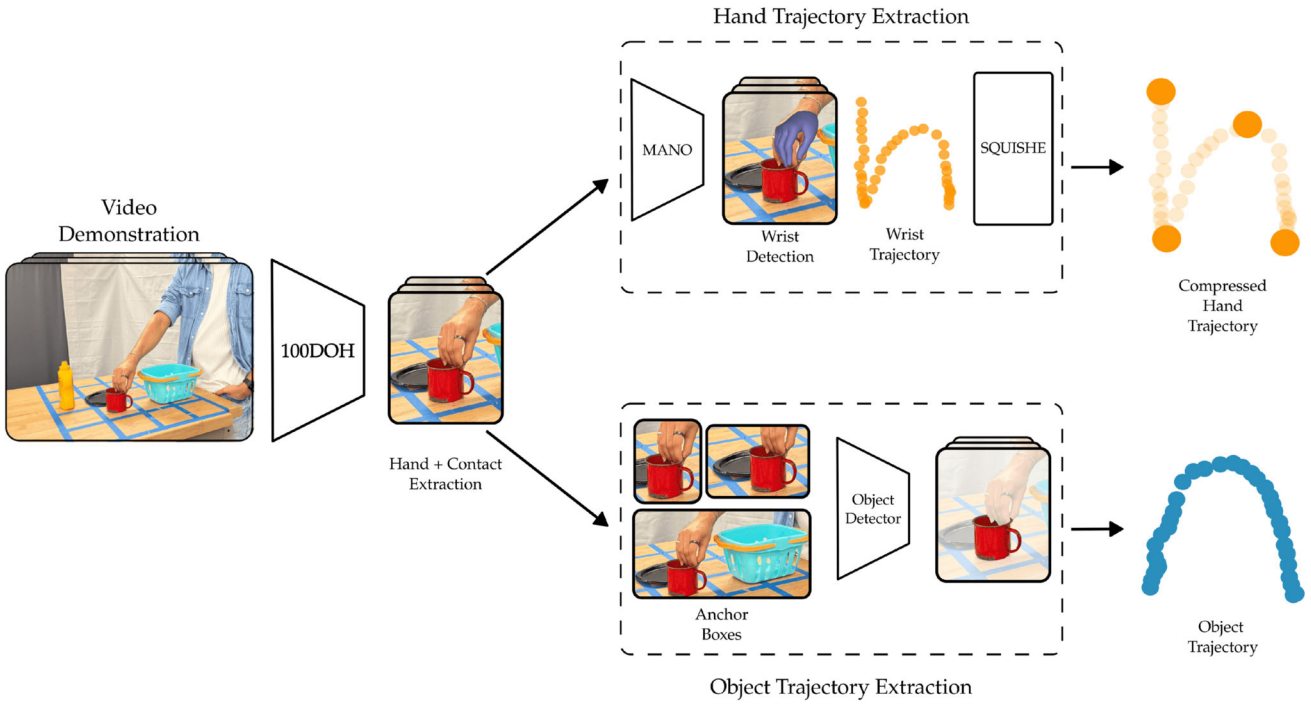

Fig. 3 An overview of our prior extraction method (Bottom Left in Fig. 2). Utilizing the 100 Days of Hands (100DOH) detector (Shan et al., 2020), we first identify the location of the hand and if it is in contact with any objects present in the frame. We then refine the human's hand trajectory using the MANO model (Romero et al., 2017) to capture wrist movements. Subsequently, to eliminate redundancy, we apply the SQUISHE algorithm (Muckell et al., 2014). This produces an initial trajectory with key waypoints that the robot should interpolate between. To pinpoint the object of interest amidst potential clutter, we analyze frames where hand-object contact occurs, creating anchor boxes that — in conjunction with an object detector — reveal the object the human interacts with most frequently. This identification enables us to construct an accurate object trajectory from the human's video

trajectory becomes the robot's initial trajectory). However, this trajectory almost always fails because of morphological differences and sensor noise. In order to improve the initial trajectory over repeated interactions, the robot explores around ξ^h to find the correct waypoints that solve the task (see Fig. 2). We will describe this exploration in detail in Sect. 4.3. But before we get to the exploration, we first need a feedback mechanism that allows the robot to differentiate between "good" and "bad" waypoints. More specifically, we design a reward model that compares how the robot is manipulating the target object to how the human manipulated the same object during their video demonstration.

Our prior from Sect. 4.1 contains the *tag* identifying the target object and its trajectory throughout the demonstration video. Similarly, we can take videos of the robot's interactions in the environment and extract the actual trajectory of the target object using the same procedure. To compare the movement of the object for the two agents, we take the mean square error (MSE) between the corresponding waypoints in their respective trajectories. The negative of this distance serves as our reward. For clarity, let the object trajectory from the prior ξ_h^o consist of waypoints $(p_{x_t}^h, p_{y_t}^h)$ and the object trajectory from the robot interaction ξ_r^o consist of waypoints $(p_{x_t}^r, p_{y_t}^r)$. Then, the reward corresponding to each waypoint is given as:

$$r_t = - \mid\mid \omega_t^r - \omega_t^h \mid\mid \tag{1}$$

$$\omega_t = (p_{x_t}, p_{y_t}) \tag{2}$$

To minimize transformation errors, we measure this distance in pixels. This approach is agent-agnostic: we extract object trajectories from both agents' videos and compare them directly, allowing the robot to align its behavior closely with that demonstrated by the human.

4.3 Exploration for iterative improvement

With a metric to assess the robot's performance, we can iteratively refine its trajectory. Starting with the initial trajectory derived from human hand movements (Fig. 2), the robot gradually improves this trajectory by exploring around the waypoints. In typical tasks, the robot first grasps an object and then manipulates it. Therefore, the task success depends on first establishing a secure grasp. In our running example of teaching the robot to pick up a cup, the robot cannot succeed

if it grabs the wrong object (e.g., picks up a plate), or if the robot does not grasp the object securely (e.g., drops the cup). We therefore divide the overall exploration into two parts: *grasp*, where the robot finds the grasp location for the object, and *task*, where the robot learns to imitate how the human manipulates that object.

Formally, the initial trajectory extracted from the human's video consists of a set of n waypoints and their corresponding contact information $\xi^h = \{(\omega^h_t, c_t) \mid t \in [t_1, t_2, \ldots, t_n]\}$ where each waypoint is a tuple (x, y, z) and c is the contact. Here x, y, z indicate the 3D position of the hand and c indicates if the hand is in contact with any objects. From this contact information we can determine when the human grasps and releases objects. For instance, let the waypoint where the contact begins be ω^h_{grasp}. We use this point to divide the prior into two trajectories: $\xi^h_{grasp} = \{(\omega^h_{t_1}, c_{t_1}), \ldots, (\omega^h_{grasp+1}, c_{grasp+1})\}$ and $\xi^h_{task} = \{(\omega^h_{grasp+1}, c_{grasp+1}), \ldots, (\omega^h_{t_n}, c_{t_n})\}$. For clarity, from now we will denote trajectories as a set of waypoints ω, however, each element of the trajectory is a tuple of a waypoint and its corresponding contact c. Under VIEW, the robot separately explores around these two trajectories to pick up the object and then perform the task. Below we discuss exploration strategies for iteratively improving *grasp* and *task*.

4.3.1 Correcting the grasp waypoint

In our first phase the robot explores around the prior $\xi^h_{grasp} = \{\omega^h_{t_1}, \ldots, \omega^h_{grasp}, \omega^h_{grasp+1}\}$. Although this prior contains multiple waypoints, the primary focus is on ω^h_{grasp}, the waypoint where it should grasp the target object. The approach used to reach the object is less important as long as the grasp is successful. Accordingly, in VIEW the robot uses the position where the human grasped the object as a prior (i.e., ω^h_{grasp}), and then the robot intelligently explores around this prior to find a grasp location that is effective for the robot arm and gripper.

Restricting the Exploration Space We define a search region around the waypoint ω^h_{grasp} using a bounding box $\mathcal{B}$. A simplistic approach centered on ω^h_{grasp} would include an area too large for efficient search. More explicitly, defining a range in the robot's coordinates with limits Δ: from $(x - \Delta, y - \Delta, z - \Delta)$ to $(x + \Delta, y + \Delta, z + \Delta)$, would create a bounding box with the waypoint ω^h_{grasp} at the center. However, such a bounding box may be unnecessarily large and span irrelevant parts of the robot workspace. In our running example of learning to pick up a cup, this bounding box could include part of the workspace which is farther away from the cup, as shown Fig. 4 (Top). Instead, we create a compact bounding box around both ω^h_{grasp} and the object location ω^o_{grasp}, extending the diagonal between these points by a limit Δ to account for sensor or model inaccuracies. This

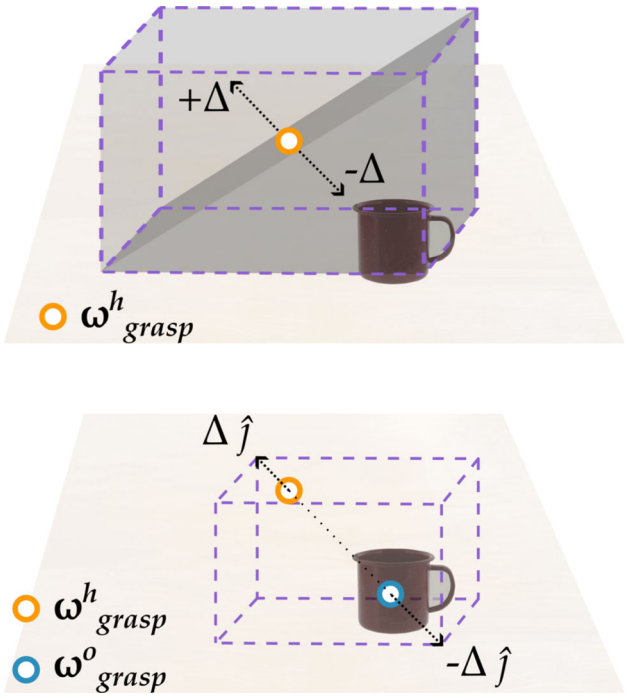

Fig. 4 Generating a bounding box for exploring grasp locations. We define a region around the waypoint $\omega^h_{grasp} = (x, y, z)$ where the human first interacted with the object in the video demonstration. (Top) A naive approach: the bounding box is centered around ω^h_{grasp} with limits Δ. The principal diagonal of the bounding box is defined by $(x - \Delta, y - \Delta, z - \Delta)$ and $(x + \Delta, y + \Delta, z + \Delta)$. (Bottom) Our approach leverages the estimated object location ω^o_{grasp} at the time of grasping to bias the search space. The principal diagonal of the bounding box are $\omega^h_{grasp} + \Delta\hat{j}$ and $\omega^o_{grasp} - \Delta\hat{j}$, here $\hat{j}$ is the unit vector parallel to the principal diagonal. This bounding box is typically smaller and is more likely to include an effective grasp location for the robot

bounding box, defined from $(\omega^o_{grasp} - \Delta\hat{j})$ to $(\omega^h_{grasp} + \Delta\hat{j})$, uses $\hat{j}$ as the unit vector between the two points. This focused region, shown in Fig. 4 (Bottom), allows for a more efficient search by targeting relevant areas.

Rewards for Grasp Exploration Once the exploration region $\mathcal{B}$ is established, the robot tests different waypoints within it to find an effective grasp. A grasp is only successful if it enables the robot to pick up and move the object. To verify this, we include the next waypoint $\omega^h_{grasp+1}$ in the exploration sequence, allowing the robot to grasp at a chosen location and proceed to the following waypoint. Since we know the object of interest (tag), we can confirm that a grasp is successful if the object remains close to the robot's end-effector at timestep $grasp + 1$.

Grasp Exploration The final step involves searching for the optimal grasp location within $\mathcal{B}$. This search is complicated by two main challenges. First, for most waypoints the robot does not move the object and the rewards are constant. Put another way, we have *sparse rewards*. Second, if the robot reaches a waypoint that is close to the object it may acciden-

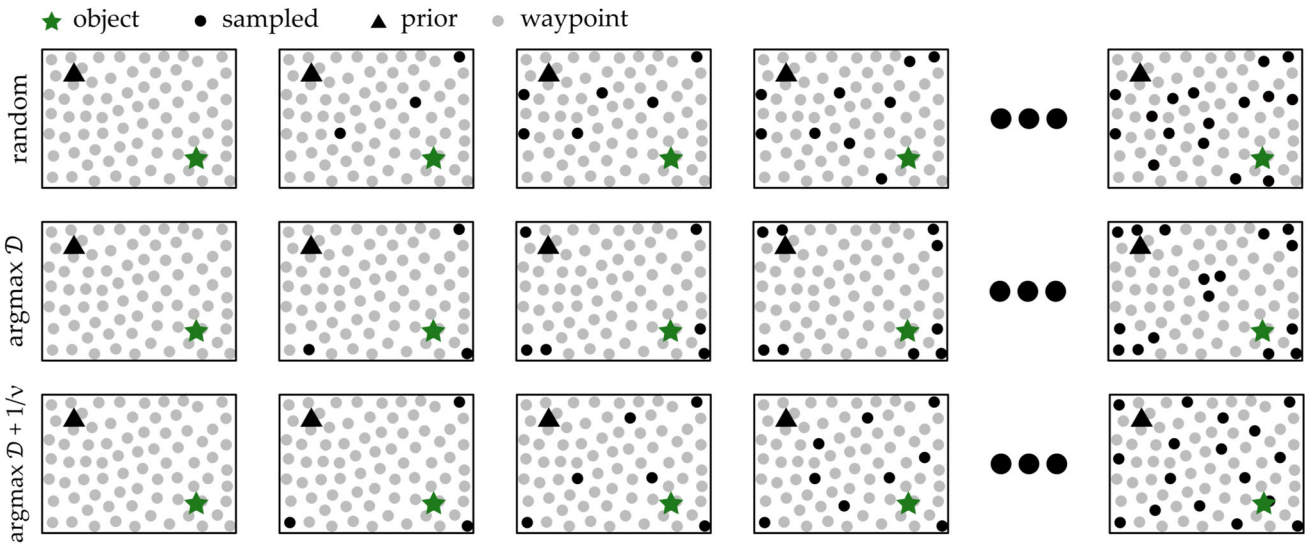

Fig. 5 Comparison of different sampling methods in our *high-level grasp exploration*. We show an example task in a two-dimensional space which is bounded around the prior (black triangle) and the object (green star). (Top) Each new high-level waypoint point is uniformly randomly sampled from our set of unvisited waypoints. This method can eventually reach the object with sufficient exploration. However, new samples may be close to previously tested points. (Middle) To quickly reduce the uncertainty about the unknown object location, we can sample high-level waypoints that maximize the distance to all previously visited waypoints. We expect that these waypoints will explore new regions of the search space. In practice, however, the distance-based estimation from Eq. 3 results in points that are clustered at the corners and center. (Bottom) Our proposed solution is to add a regularizing term in Eq. 5 to ensure that the next high-level waypoint is truly from an unexplored region of workspace. Our experiments show that this approach finds the grasp location more rapidly (Color figure online)

tally knock the object over or otherwise *lower its measured rewards*. Hence, waypoints that are actually close to a successful grasp could be penalized by the reward model.

Returning to our cup example, consider a scenario where the robot receives a baseline reward of +10 at waypoints that don't involve moving the cup. If the cup is supposed to be moved left as per the human's video demonstration, and the robot picks a point that hits the cup and moves it to the right, the reward might drop to +5. Conversely, moving the cup correctly to the left might increase the reward to +15. In both cases, the robot has gained valuable information: the chosen waypoint interacted with the object and may be near to a successful grasp location. This variation in rewards — regardless of whether the reward is increasing or decreasing — helps to pinpoint the object's location within the search space $\mathcal{B}$.

Put together, the sparse rewards at grasp locations and locally varying rewards around those locations make it difficult for the robot to efficiently optimize for successful grasps. We therefore propose a quality-diversity (QD) approach for intelligently searching the space $\mathcal{B}$. Our proposed QD algorithm is broken into a *high-level* search — which divides $\mathcal{B}$ into regions of interest — and a *low-level* search — which explores within those regions to pinpoint a successful grasp location. We summarize this overall method in Algorithm 3.

High-Level Grasp Exploration. To efficiently explore the bounding box $\mathcal{B}$, we discretize it into distinct regions using Centroidal Voronoi Tessellation (CVT) (Vassiliades et al., 2017). By numerically sampling points within $\mathcal{B}$ and applying k-means clustering, we generate M evenly distributed clusters, each representing a region for exploration. The centroids of these clusters form the set of potential high-level waypoints $\Omega_{unvisited} = \{\omega \mid i = 1, 2, \ldots, M\}$. The robot's task is to identify regions likely to contain a successful grasp location for targeted refinement.

A naive approach would be to uniformly sample waypoints from $\Omega_{unvisited}$. However, this risks selecting points close to already tested centroids, leading to redundant exploration (see Fig. 5 Top). Instead, we propose a sampling strategy that prioritizes unexplored regions by maximizing the distance between new waypoints and previously visited ones. We define the next high-level waypoint ω_{next} as:

$$\omega_{next} = \underset{\omega \in \Omega_{unvisited}}{\arg\max} \quad \mathcal{D}(\omega, \Omega_{visited}) \tag{3}$$

$$\mathcal{D}(\omega, \Omega_{visited}) = \frac{1}{k} \sum_{\omega_j \in N_i^k} || \omega - \omega_j || \tag{4}$$

Here, $\mathcal{D}(\omega, \Omega_{visited})$ is the mean distance between each unvisited waypoint and its k-nearest neighbors (N_i^k) in the visited set $\Omega_{visited}$, i.e., the set of tested centroids. This

ensures that new waypoints are chosen based on their separation from already tested locations.

To address potential clustering issues where a waypoint is close to one visited point but far from others (see Fig. 5 Middle), we introduce a regularization constraint. By calculating the variance ν of distances between an unvisited waypoint and all visited waypoints, we ensure equidistance. Our final optimization balances diversity and uniformity:

$$\omega_{next} = \underset{\omega \in \Omega_{unvisited}}{\arg\max} \quad \mathcal{D}(\omega, \Omega_{visited}) + \frac{1}{\nu} \tag{5}$$

This optimization ensures that the selected waypoint from $\Omega_{unvisited}$ maximizes distance from all waypoints in the set while attempting to be equidistant with the waypoints in $\Omega_{visited}$ (see Fig. 5 Bottom). In practice, the robot selects a high-level waypoint from $\Omega_{unvisited}$ using Eq. 5, and then executes a trajectory in the environment that attempts to grasp the object at that waypoint. We use the reward model from Eq. 1 to assess the performance of this grasp.

Low-Level Grasp Exploitation. To prioritize regions for refinement, the robot selects waypoints from the visited waypoint set ($\Omega_{visited}$) based on the magnitude of reward changes. Waypoints where rewards varied significantly — either increasing or decreasing — are more likely to be close to an optimal grasp. We sample a waypoint ω_{local} from $\Omega_{visited}$ with the probability distribution:

$$p_i = \frac{e^{\gamma \sigma_i}}{\sum e^{\gamma \sigma_j}} \tag{6}$$

where the denominator is summed across all visited waypoints, and σ_i is the normalized variation in reward between the high-level waypoint i and the reward R_0 from the initial trajectory:

$$\sigma_i = \frac{|| R_i - R_0 ||}{\max_j || R_j - R_0 ||} \tag{7}$$

This approach biases the robot's search toward high-level waypoints that showed the most significant changes in reward, ensuring focus on areas likely to contain a successful grasp location.

Once a high-level waypoint ω_{local} is selected, the robot defines a smaller bounding box $\mathcal{B}_{local} \subset \mathcal{B}$ centered around ω_{local} with a radius ϵ. This refined region narrows the search to the immediate vicinity of the promising waypoint. Within $\mathcal{B}_{local}$, the robot uses an optimization algorithm, such as Bayesian optimization (BO)[1] Snoek et al. (2012), to maximize the reward function. BO iteratively samples waypoints ω_{opt} from $\mathcal{B}_{local}$, evaluating each based on the reward model (Eq. 1). If a sampled waypoint ω_{opt} achieves a higher reward than the current ω_{local}, it is added to $\Omega_{visited}$ as the new candidate for refinement.

This two-step process — targeting promising regions through reward-based sampling and optimizing locally using BO — enables the robot to precisely identify and finetune the optimal grasp location within the bounding box $\mathcal{B}$. Algorithm 3 in Appendix 1 summarizes this exploration scheme.

Trading-off Between High- and Low-Level Search. Our overall exploration process for identifying a successful grasp trades-off between testing new high-level waypoints from $\Omega_{unvisited}$ and then exploiting the regions around relevant waypoints from $\Omega_{visited}$. We balance this exploration of new regions and exploitation of sampled regions using probability $p_{explore}$: with probability $p_{explore}$ we test an $\Omega_{unvisited}$ waypoint, and with probability $1 - p_{explore}$ the robot explores the region around a waypoint from $\Omega_{visited}$. The value of this probability is chosen based on the variance in the rewards of the high-level waypoints. Concretely, the exploration probability is calculated as $p_{explore} = \frac{\alpha}{\sigma_{\max}}$ where $\sigma_{\max}$ is the highest normalized variance in reward between the high-level waypoints and α is a hyperparameter. Intuitively, as the variance in the rewards increases, the probability of low-level exploitation increases proportionately. This search process ends once the robot identifies a waypoint that successfully grasps the target item.

In summary, grasp exploration works in a hierarchical manner. First the robot conducts a broad search across the bounding box $\mathcal{B}$ by dividing it into M evenly distributed high-level waypoints. The robot then conducts a more refined search in the vicinity of waypoints that are potentially close to the object — i.e., waypoints that incur a high variation in reward. Overall, our grasp exploration approach has similarities to the QD algorithm CMA-ME (Fontaine et al., 2020). The primary novelty of our approach as compared to Fontaine et al. (2020) is the sampling technique used for selecting the high-level waypoints. While CMA-ME relies on randomness to select a point from $\Omega_{unvisited}$, we propose a regularized entropy metric for selecting points that are evenly spaced across the search space. In Fig. 5 we show an example of why this high-level sampling approach is important, and how our proposed approach can more rapidly identify the grasp location. We also test this difference in our experiments.

4.3.2 Correcting the task waypoints

Once the robot has grasped the target object, it can now proceed to replicate how the human manipulated that object in the demonstration video. This process is more straightforward than identifying the correct grasp because here the rewards are *dense*: any change in the way the robot moves

[1] VIEW is not tied to a specific local optimization algorithm. While we use BO in our experiments, it can be replaced with any other optimizer.

the object will lead to a change in the object's position, and therefore a change in the measured rewards from Eq. 1. Accordingly, we can use off-the-shelf optimization methods to iteratively improve the waypoints along the initial trajectory $\xi^h_{task} = \{\omega^h_{grasp+1}, \ldots, \omega^h_{t_n}\}$ after the robot has learned to successfully grasp the target item.

Similar to our approach for grasp optimization, we start by drawing bounding boxes $\mathcal{B}$ around each waypoint in ξ^h_{task}. Here it is important to remember that the reward function from Eq. 1 is the distance between the object position in the human's video demonstration and the object position in the robot's task execution. Hence, the rewards associated with each waypoint are independent, and the robot can simultaneously explore and improve each task waypoint without affecting the results across other task waypoints. We therefore conduct $n - grasp$ search processes in *parallel*, one for each waypoint from $\omega^h_{grasp+1}$ to the final waypoint $\omega^h_{t_n}$. Let us denote the robot's updated trajectory as $\xi^r_{task} = \{\omega^r_{grasp+1}, \ldots, \omega^r_{t_n}\}$. To find an optimal waypoint ω_r, the samples a point within the corresponding bounding box and then rolls-out a trajectory that moves through that point in the environment. Here we use Bayesian optimization (although other methods are possible): for each $\omega^r_t \in \xi^r_{task}$, a separate instance of BO updates the robot's waypoint to better match the video demonstration. See Algorithm 4 in Appendix 1 for a summary.

4.4 Residual network

The process we have described so far in Sect. 4 enables the robot to learn a task from a *single* video. But when the robot gets a new video demonstration for a different task, we are faced with the question: does the robot need to restart VIEW from scratch, or can the robot leverage what it has learned on one task to accelerate learning on another task? Here we return to our example of learning to pick up a cup. Initially, the robot extracts an imperfect trajectory ξ^h from the video demonstration, which it refines through exploration to arrive at a successful trajectory ξ^*. Ideally, the robot would have extracted ξ^* directly as the initial trajectory. The discrepancy between ξ^h and ξ^* reflects the robot's systematic errors during prior extraction.

We hypothesize that the error can be modeled as additive noise, expressed as $\xi^* = \xi^h + \eta_{static} + \eta_{random}$. Here, η_{static} accounts for consistent errors — such as sensor inaccuracies, model misalignment, or morphological differences — that remain approximately constant across tasks. To enhance the accuracy of prior trajectory extraction, we propose training a residual network to estimate this noise. Given a dataset of k previously solved tasks $\mathcal{D} = (\xi^h_k, \xi^*_k)$, we train a model to estimate the noise η_{static}. More specifically, we train a residual $\Phi(\xi^h) = \eta$ to minimize the loss $\|\xi^* - \xi^h + \Phi(\xi)\|^2$ across the dataset. The robot then deploys this residual when it receives the $k+1$ video demonstration. The robot starts by compressing the new video using the steps from Sect. 4.1 to get ξ^h_{k+1}; we then add the residual $\Phi(\xi^h_{k+1})$ to push this prior towards the correct trajectory. In practice, we will show that the residual can improve the accuracy of the prior and reduce the number of iterations the robot needs to learn new tasks.

4.5 Incorporating multi-object scenarios

Our discussions thus far have dealt with scenarios where the human interacts with a single object. However, many real-world manipulation tasks involve handling multiple objects. For example, when making tea the human might carry a cup to a specific location, and then add tea from a kettle into the cup.

The proposed method VIEW readily adapts to such multi-object scenarios. Recall that our prior extraction process outputs a hand trajectory, providing the wrist location and contact information throughout the video. We can use the changes in the contact information to divide long trajectories — involving multiple objects — into distinct sub-trajectories for each subtask. Each subtask then involves interaction with only one object, which we can solve using the algorithms described above.

Consider the example of making tea. By segmenting the trajectory at points where contact changes, we can create separate, manageable segments: one for moving the cup and another for adding the tea. Each subtask is then solved separately using our algorithm, which includes dividing each individual subtask into grasp and task phases and solving them using our methods in Sect. 4.3.1 and Sect. 4.3.2. For example, we first address the cup's movement, and once complete, proceed to handle the kettle in a similar manner.[2] Overall, this modular strategy allows the robot to systematically learn long, multi-step tasks with visual imitation learning by concentrating on one subtask at a time.

5 Simulations

We proposed VIEW, a waypoint-based algorithm that can imitate humans by watching video demonstrations. We hypothesize that each component of VIEW will significantly impact the overall success of the robot. To test this hypothesis, in this section we conduct an ablation study that investigates how each part of VIEW contributes to the overall robot performance.

Experimental Setup The simulations are conducted in a Pybullet environment. To collect demonstrations, we con-

[2] See https://collab.me.vt.edu/view/ for videos showcasing VIEW learning these multi-object tasks.

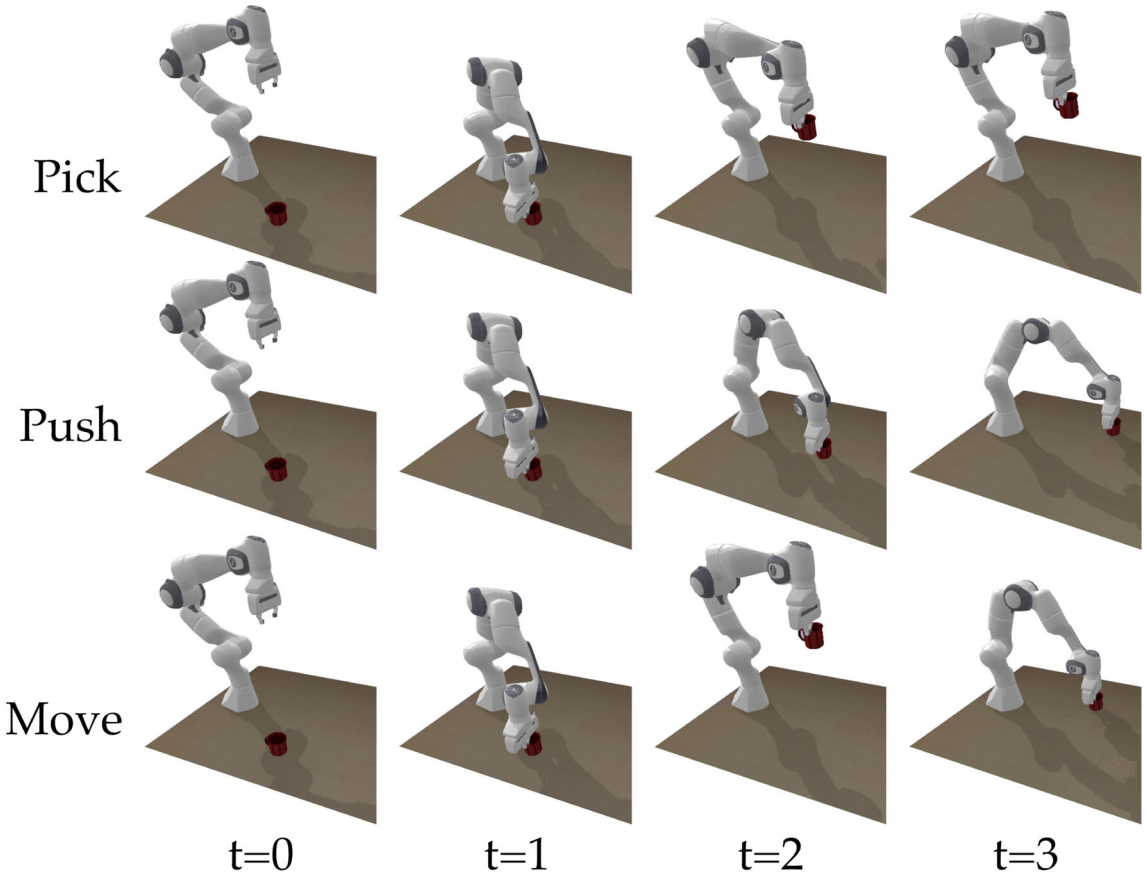

Fig. 6 Task demonstrations used in our simulations. (Top) During **Pick** the robot is teleoperated to pick up a cup placed on the table. (Middle) During **Push** the robot is teleoperated to grasp the cup and push it to a new specified location on the table. (Bottom) During **Move** the robot picks up the cup and places it at a specified new location on the table. In our ablation studies examining the effects of noise, trajectory compression, and exploration techniques, we utilize a single demonstration that is then perturbed using Gaussian noise. For assessing the influence of residual learning in our final simulation, we uniformly sample start and end points for each task and collect teleoperated demonstrations accordingly. These demonstrations are then perturbed using a deterministic noise function. We compile a dataset of 50 demonstrations for each task, either by introducing Gaussian noise to a single trajectory or by generating 50 distinct trajectories through uniform sampling

trol a simulated FrankaEmika robot arm and record frames at the rate of 20Hz. Demonstrations are collected for three tasks: picking up an object (**pick**), pushing an object (**push**), and picking and placing an object (**move**). A single object (a cup) is used for all evaluations (See Fig. 6). For the push and pick tasks, the initial trajectory has three waypoints after compression, while the move task yields four waypoints. To approximate noise in the real world, we distort these initial trajectories using either Gaussian noise or a fixed noise matrix. To understand our algorithm's performance in ideal conditions, no noise is injected into the reward function. The success criteria vary slightly between tasks: for **pick**, success means the robot has picked up the cup; for **push**, it is successful if it pushes the cup to the correct location; and for **move**, success requires the robot to pick up the cup and place it at the correct location on the table.

5.1 Impact of noise

In our first simulation, we study how noise in the extracted prior influences our algorithm's capability. Here increasing noise means that the robot's extraction of the human's hand trajectory is farther from the actual trajectory that the human followed. For each task we carry out a series of 50 trials. Since this simulation is designed to isolate the impact of noise on our exploration scheme, we do not include the residual network during these trials.

Our findings (refer to Fig. 7) reveal that VIEW can use exploration to overcome an incorrect prior. For noise variance between 0.05m and 0.15m, the robot is able to successfully imitate the simulated human demonstration in almost 100% of the trials. However, as the prior is distorted farther away from the correct trajectory, the performance of VIEW eventually decreases. At a noise variance of 0.2m we observed a

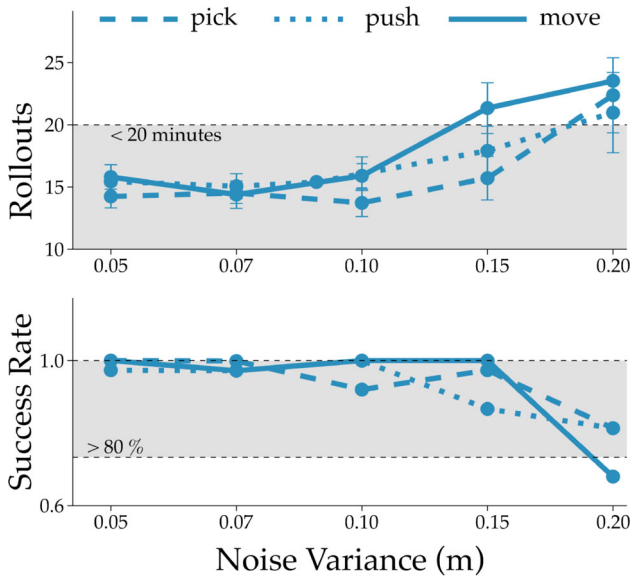

Fig. 7 Simulation results demonstrating the impact of noise on our algorithm. We test VIEW on three tasks — pick, push, move. For each task we collect the true initial trajectory, and then add Gaussian noise to distort that trajectory. This captures scenarios where the robot's prior is incorrect (e.g., misses the cup entirely), and the robot must explore around this prior to imitate the demonstrated task. Our results are shown across 50 trials. The shaded region in the top plot indicates less than 20 minutes of learning time to successfully imitate the task. The shaded region in the bottom plot indicates more than 80% success rate. The bars indicate standard error of the mean

notable decrease in the success rate. This observation aligns with our expectation that larger distortions lead to longer search times, potentially resulting in timeouts before solutions are found. This pattern is also evident in the number of exploration rollouts required for convergence: in general, the more noise in the prior the more exploration rollouts the robot needed to correct its waypoints. Finally, we note that the number of waypoints can impact performance: tasks involving more waypoints (**move**) generally required more rollouts than tasks with fewer waypoints (**pick** and **push**). In practice, this simulation suggests that VIEW's exploration steps are critical to success, and the robot can use exploration to overcome errors in its initial guess of the correct trajectory.

5.2 Impact of trajectory compression

In our second simulation we assess the importance of trajectory compression within our algorithm. In stead of our proposed compression algorithm, simpler methods are also possible: for instance, we could simply sample the demonstration at a reduced rate, and use the sampled points as the initial trajectory (e.g., down-sample the video every 100 frames). Here we compare the impact of this alternative method against our compression algorithm.

To generate these alternative compressions, we first distort the correct initial trajectory using a noise variance of 0.15m. We then resample this modified demonstration at a fixed sample rate. We tested sampling rates from 20Hz to 5Hz. Our original demonstrations comprised approximately 40 waypoints: hence, the compression could range from 40 waypoints at the highest sampling rate to 10 waypoints at the lowest sampling rate. We did not sample lower than 10 waypoints using a fixed sampling rate because this caused the trajectory to skip critical waypoints, such as the pick point. Similar to the previous subsection, we assessed the impact of compression using the success rate and the number of rollouts required for convergence across 50 trials.

Our results, depicted in Fig. 8, provide two important outcomes. First, as the number of waypoints in the robot's trajectory increases (higher sampling rate), the number of rollouts required for convergence also rises. This suggests that compression is indeed important — we can accelerate the robot's visual imitation learning by focusing on a smaller number of waypoints. Second, using simplistic compression algorithms that down-sample the demonstration at a fixed rate perform worse than our VIEW approach. The key difference here is that sampling at a fixed rate may cause the robot to miss a critical point along the demonstration (such as the frame where the human grasps the cup). Using VIEW, the robot minimizes the number of waypoints, while also ensuring that those waypoints retain critical aspects of the demonstration.

5.3 Impact of our exploration approach

In our third simulation we investigate the effectiveness of our exploration strategy. As an alternative to our proposed exploration strategy, we examine the performance of a unified optimization approach. This baselines does not separate task into grasp and manipulation phases; instead, it performs Bayesian Optimization to de-noise the *entire* trajectory. We maintain the noise level at 0.15m, and evaluate success rates and convergence rollouts for across 50.

The outcomes of this experiment are depicted in Fig. 9. As anticipated, we observe a substantial reduction in success rates when the waypoints are not split into grasping and manipulation phases. The highest success rate achieved without splitting is approximately 70% for the **push** task, whereas our segmented approach with VIEW achieves a minimum of 92% on the same task. VIEW also decreases the number of environmental rollouts required for convergence. Finally, we noticed that these results are impacted by the number of waypoints in the trajectory. For instance, in the **move** task — which has four waypoints instead of the three in **pick** and **push** — the success rate of the baseline approaches zero. These results indicate that forgoing waypoint segmentation during exploration not only leads to inferior performance,

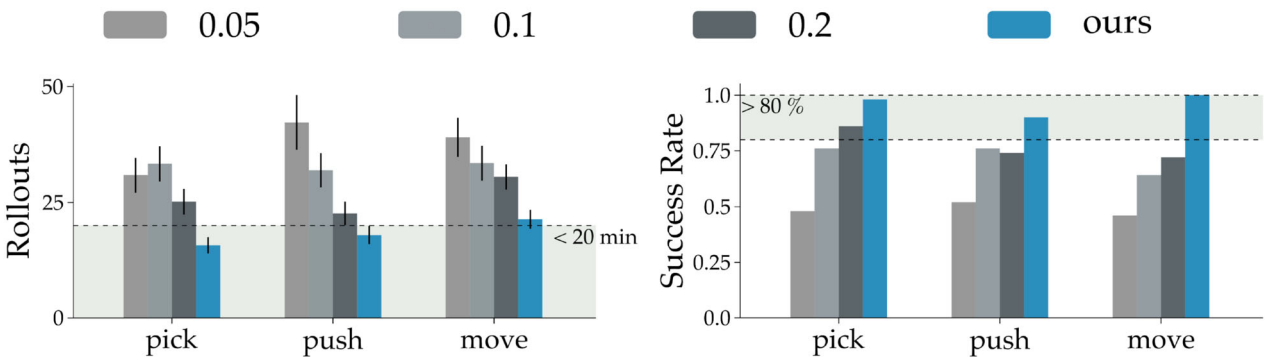

Fig. 8 Simulation results examining the impact of trajectory compression. Within VIEW we compress the prior trajectory to minimize the number of waypoints while maximizing the accuracy of the compressed trajectory. We compare this method with an alternative approach in which the prior is sampled at a lower frequency to limit the number of points in the trajectory. We vary the sampling frequency of the prior trajectory to be 5Hz, 10Hz, or 20Hz. The plot on the left shows the average number of rollouts it takes to learn each task over 50 trials, and the shaded region indicates less than 20 minutes of training time. The plot on the right shows the success rate for each task across 50 trials, and the shaded region shows a success rate higher than 80%. The bars indicate standard error of the mean

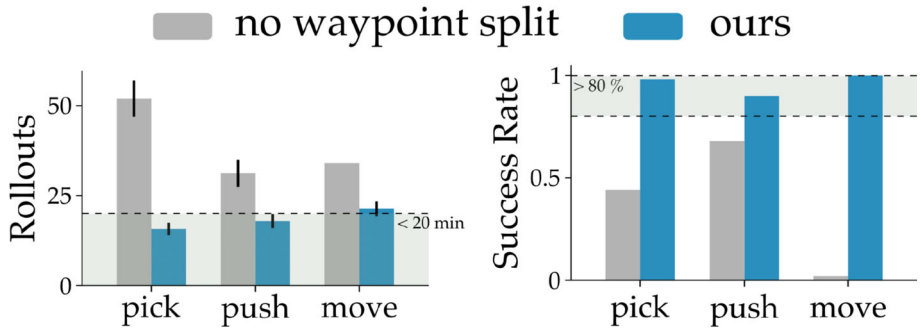

Fig. 9 Simulation results examining how separating the waypoints into grasping and manipulation phases affects performance. Under VIEW the robot autonomously splits the task into separate parts: first the robot learns to grasp the object, and then it learns how to manipulate that object and complete the task. We compare this division against a unified approach that solves the entire task simultaneously. We measure the average number of rollouts taken to solve the task (Left) and the success rate (Right) over 50 trials. The shaded regions indicate less than 20 minutes of training time and over 80% success rate, respectively. The bars indicate standard error of the mean

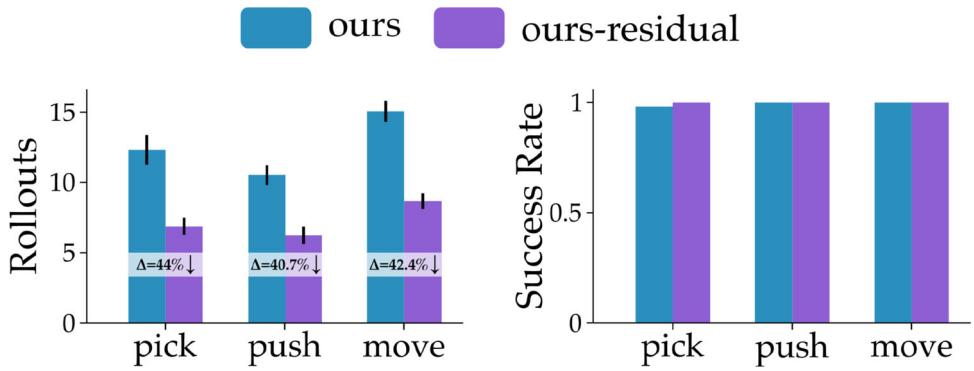

Fig. 10 Simulation results for VIEW with and without the residual. We examine if the robot can utilize previous experiences to more rapidly imitate new tasks. In this simulation we sample 50 random locations from the robot's workspace and their corresponding distortions from a noise matrix (Eq. 8). We then use these samples to train a residual network that de-noises the distorted prior. The plots above compare the performance of our approach with and without using this residual. (Left) The number of rollouts taken to solve the task averaged over 50 trials. We also list the percentage decrease in rollouts when the residual is present. (Right) The success rate for each task. The bars indicate standard error of the mean

but also fails to scale effectively with an increasing number of waypoints.

5.4 Impact of residual

In our final simulation, we assess how the residual network can accelerate the robot's learning, contrasting it with previous simulations where each task or demonstration was treated independently. Earlier, we used Gaussian noise to introduce various distortions, while keeping the cup's location fixed to ensure consistent analysis — only the robot's initial trajectory was modified. However, Gaussian noise is unsuitable here, as it would produce inconsistent offsets at each waypoint for the same object in the same location. To model a more consistent offset similar to η_{stable} in real-world conditions, we apply a nonlinear noise matrix to introduce consistent distortions across the robot's entire workspace:

$$\eta = \tanh \frac{\xi - \mathcal{C}}{\lambda} \tag{8}$$

We utilize tanh to introduce distortions into the trajectory waypoints, adjusting their positions based on their proximity to a centroid ($\mathcal{C}$). The degree of distortion is modulated by the regularizer λ. This noise is then added to the demonstration to get a distorted initial trajectory ξ^h. We adjust the location of $\mathcal{C}$ and the value of λ to ensure that the distortions range from 4cm to 30cm across all waypoints. To mitigate against any bias, we do not use a fixed cup location; instead, we sample the cup's location from a uniform distribution across the table, and then collect demonstrations for each task. We gather a total of 50 random demonstrations from the environment, each distorted via the noise matrix, to form our dataset. Specifically, our dataset $\mathcal{D}$ for training the residual consists of 50 pairs of initial trajectories ξ^h and their corresponding ground truths ξ^*. Our algorithm's performance — with and without the integration of the residual network — is then evaluated across 50 new and unexplored cup locations for each task.

Our results are illustrated in Fig. 10. Across all tasks, VIEW with the residual demonstrated the ability to few-shot learn new object locations. We observed a reduction of over 40% in the number of rollouts required for the robot to learn each task. Indeed, VIEW with the residual network needed fewer than 10 trials on average to learn the correct behavior from distorted input trajectories. These results suggest that VIEW is not only effective when learning from scratch; we can also leverage the tasks that VIEW learned across previous video demonstrations to accelerate learning on a new video demonstration in the same workspace.

6 Experiments

In the previous section we explored the components of VIEW through an ablation study in a simulated environment. In this section we now test our overall method in the real-world with human video demonstrations. We apply VIEW on video demonstrations for various tasks such as picking up a cup or moving a basket. To see videos of these demonstrations and VIEW's learning process, visit: https://collab.me.vt.edu/view/

Tasks Our real-world tasks span different skills and objects. We focus on three primitive skills — push, pick, move (Fig. 11 shows a demonstration for each skill). A full list of the objects used in our experiments is found in the Appendix: these objects include household items such as foods, cups, and containers. We start with simple tasks where the robot must learn the primitive skills in *Uncluttered* environments where no other items are present. Next, we provide video demonstrations in *Cluttered* settings with multiple objects, and the robot must learn to imitate the demonstrated task despite this environmental clutter.

In *Uncluttered* tasks we test all three fundamental skills. However, in *Cluttered* tasks we test only the **Move** skill: here the robot must reach for and move the correct object while avoiding and ignoring the environmental clutter. Different objects may be placed close together in the environment to visually saturate the robot's camera or constrain the grasp locations.

Our method can scale to arbitrarily long tasks that involve manipulating multiple objects, as shown in our supplemental videos. However, for the purposes of this experiment, we only focus on single object manipulation tasks. Our aim is to test VIEW's ability to imitate manipulation tasks from a single video demonstration, and to compare VIEW to relevant baselines.

Baselines Our primary baseline is **WHIRL** Bahl et al. (2022), a state-of-the-art method for visual imitation learning from human demonstrations. However, our implementation of WHIRL differs slightly from the original method. Within the original work, WHIRL calculates rewards by comparing agent-agnostic representations of the human demonstration and robot interaction (similar to VIEW). WHIRL finds this agent-agnostic representation by inpainting the human and the robot from the videos using Copy-Paste Networks (Lee et al., 2019), and then using the action-recognition model of Monfort et al. (2021) to calculate its representation. In our experiments, however, Copy-Paste Networks could not reliably inpaint the robot, despite extensive fine-tuning on a custom dataset.[3] To ensure a fair comparison, we replaced WHIRL's original reward model with our object-centric reward model from Sect. 4.2. This substitution is reason-

[3] See the Appendix for more details.

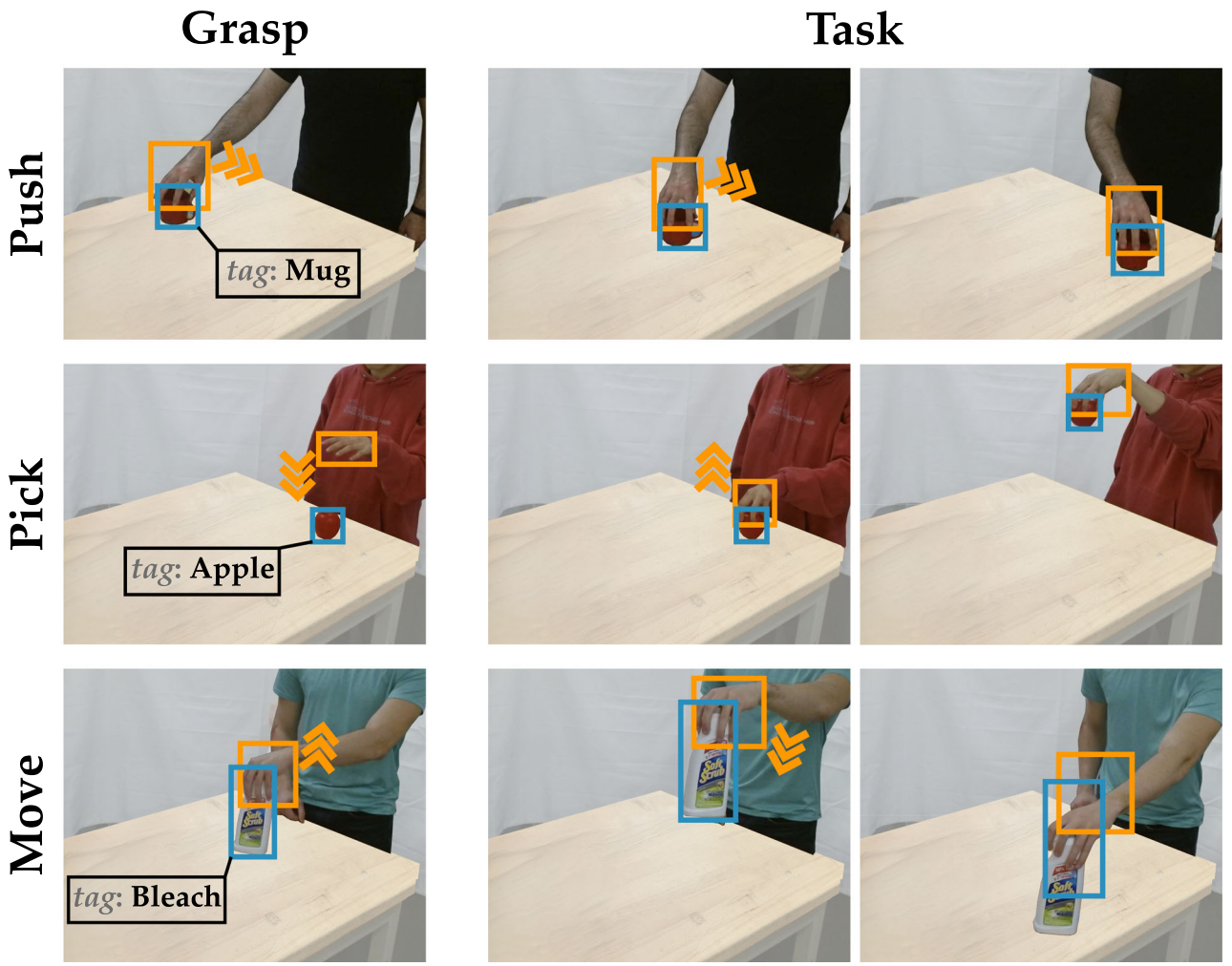

Fig. 11 Manipulation tasks from our experiments. People (including the authors and external participants) provided video demonstrations of three fundamental skills necessary for more complex tasks (Padalkar et al., 2023): **push**, **pick**, **move**. Here we show examples frames where VIEW detected the human hand and the intended object, i.e., the object human is interacting with. VIEW used these frames to extract a prior trajectory for the human hand and object

able as our reward model provides more explicit feedback by directly comparing target object movements rather than using high-dimensional action representations, as in the original WHIRL. The rest of the WHIRL algorithm matches the original manuscript.

Our other experimental baselines are ablations of our approach. At one extreme we have **Prior**, a method that extracts the human hand trajectory from the video demonstration and then replays that trajectory on the robot arm. This corresponds to VIEW without any exploration or residual. In practice, **Prior** will only succeed if the initial trajectory the robot extracts is sufficient to successfully imitate the task. At the other extreme we tested **ours-BO**, an ablation of VIEW that leverages a different exploration scheme. **ours-BO** is a variant of VIEW that does not use the QD algorithm: instead, the robot employs Bayesian Optimization to separately identify the grasp and match the human's behavior. Finally, when the robot is learning multiple tasks, we also test **ours-residual**. This is our full VIEW algorithm that leverages previously solved tasks to improve its prior extraction.

Experimental Setup and Procedure Experiments were conducted on a Universal Robots UR10 manipulator with 6 Degrees-of-Freedom. The human's video demonstrations were recorded with a RealSense D435 RGB-D camera at 60 frames per second. We also recorded the robot's interactions with the environment as it iteratively tried to imitate the demonstrated behavior.

We collected 13 video demonstrations across all tasks, where 9 were in *Uncluttered* environments and 4 were in *Cluttered* environments. For the *Uncluttered* tasks the 9 total demonstrations were divided into 3 videos for each skill — *move-uncluttered*, *pick-uncluttered*, and *push-uncluttered*.

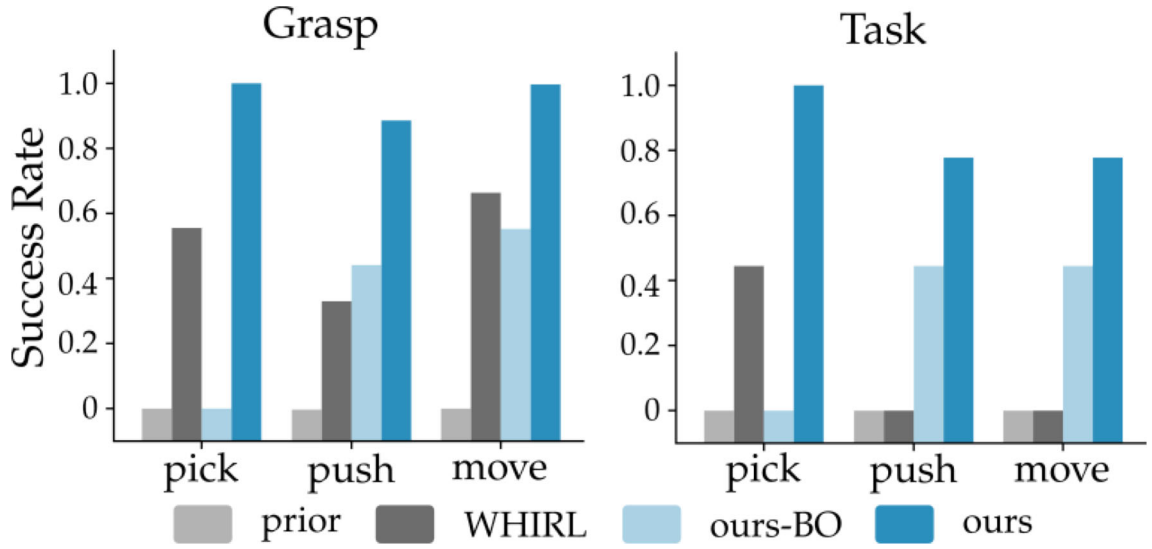

Fig. 12 Experiment results for *Uncluttered* tasks. (Left) How frequently the robot grasped the object from the human's video demonstration. (Right) How frequently the robot learned to imitate the human's video demonstration. Results are calculated across 9 separate video demonstrations and 3 trials per video demonstration. Note that the results for **pick** are the same in both grasping and task exploration, since here the objective is just to pick up (i.e., grasp) an item

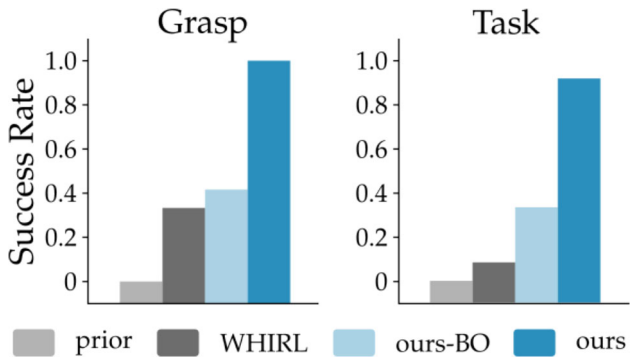

Fig. 13 Experiment results for *Cluttered* tasks. Here the environment contained multiple extraneous items in addition to the target object the human manipulated. (Left) How frequently the robot learned to grasp the correct item. (Right) How frequently the robot correctly imitated the entire video demonstration. These results were taken across 4 separate video demonstrations and 3 trials per video demonstration (for a total of 12 datapoints)

The 4 videos in four different *Cluttered* environments all demonstrated the same skill *move-cluttered*. We conducted three trials on the robot for every demonstration, totalling 39 trials per method.

6.1 Results

Uncluttered Tasks The results from our experiments on *Uncluttered* tasks are shown in Fig. 12. These plots display the results across the two phases of each task: the success rate for learning to grasp the object, and the success rate for learning to correctly manipulate that object. The robot is said to have succeeded in grasping if it was able to pick up the object. The definition of successful task completion varied between the different tasks: for *move-uncluttered*, the task was considered a success if the robot placed the object close to the same location as the human. In *Pick-uncluttered*, the robot was successful it if grasped the target object and lifted it off the table, while in *push-uncluttered*, the robot successfully completed the task if it pushed the item to the human's demonstrated location.

We observed that the prior trajectory was typically distorted by 10-15 cm from the correct pick location, and simply replaying the trajectory extracted from the human's video demonstration was consistently unsuccessful. Across all tasks, **Prior** was not able to either grasp or manipulate the target object. The state-of-the-art visual imitation learning baseline **WHIRL** was more effective, particularly in learning to grasp the target item. But **our** proposed VIEW algorithm surpassed this baseline, reaching more than twice the success rate of **WHIRL** for the push task and achieving a 100% success percentage in the pick task. For *push-uncluttered* and *move-uncluttered*, **WHIRL** was able to grasp the target object in some trials, but it did not learn to correctly manipulate that object within the limit of 100 rollouts in the environment (roughly 45 minutes). For these same tasks **our** VIEW algorithm reached an 80% success rate, learning to

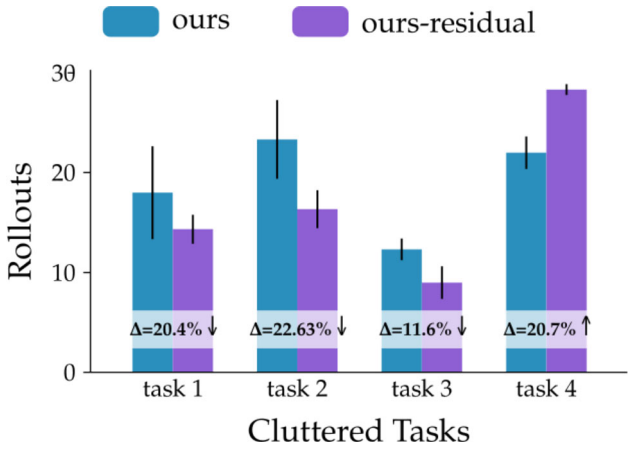

Fig. 14 Experiment results for learning from multiple tasks. Here the robot has previously solved the *Uncluttered* tasks, and it is now trying to learn a new *Cluttered* task. We compare our VIEW algorithm without the residual (*ours*) to our full VIEW algorithm with the residual (*ours-residual*). There are a total of four video demonstrations for different *Cluttered* tasks. We plot the average number of rollouts needed for VIEW to solve each of these tasks. Δ is the percentage change in the number of rollouts with and without the residual. The bars indicate standard error of the mean

replicate the human's video demonstrations in less than 30 minutes.[4]

Finally, we compared the performance of **ours-BO** and **ours**. Across the board, we found that **ours-BO** is less effective at visual imitation learning than our full VIEW algorithm, and in the *pick-uncluttered* environment this baseline performs significantly worse than **WHIRL**. These results highlight the importance of our high-level and low-level QD search algorithms for exploring how to grasp the object: without the ability to learn effective grasps, **ours-BO** struggles to imitate the rest of the manipulation task.

Cluttered Tasks We present the results for the *Cluttered* task trials in Fig. 13. As before, the plot on the left shows the grasp success rate, and the plot on the right shows the full task success rate. Our results followed the same trends as in *Uncluttered* tasks. Directly executing the extracted prior never led to success for either grasping or manipulation. The baselines **WHIRL** and **ours-BO** were roughly similar, reaching success percentages of less than 50% across a maximum of 100 real-world rollouts (roughly 45 minutes). We were not surprised that **WHIRL** struggled with cluttered environments: it does not split the exploration into separate parts for grasping and manipulation; even if the robot grasps the object in an interaction, it can fail to grasp it again in the subsequent repetitions.[5] Overall, **our** VIEW method was effective across the cluttered settings, grasping and manipulating the correct object to match the human's video demonstration in almost 100% of the trials. VIEW solved each task in less than 30 minutes.

Learning from Multiple Tasks In Fig. 14 we summarize the results from our final experiment. This experiment focused on how VIEW can leverage the tasks it has previously solved to improve its prior and accelerate its learning on new tasks. To quantify this acceleration, we measured the number of rollouts it took for the robot to successfully imitate a video demonstration in the *Cluttered* environment. Both **ours** and **ours-residual** used VIEW, but **ours-residual** included the full VIEW algorithm with the residual component. We found that for 3 out of the 4 *Cluttered* demonstrations, applying the residual significantly reduced the number of interactions needed to learn the task (roughly 25% fewer rollouts). Interestingly, for the fourth demonstration the residual actually had the opposite effect, and slowed down the robot's learning. On further examination, we believe this decrease in performance occurred because the initial hand trajectory ξ^h lied outside the distribution of the data used to train the residual. Since the residual had not seen a demonstration that operated in the same part of the workspace, it was not able to de-noise the prior and accelerate the robot's learning. This suggests that — while the residual can be useful — it should be carefully applied. Learning a robust residual necessitates an expansive dataset that includes waypoints spanning the workspace.

7 Limitations and future work

Our method has demonstrated the capability to expedite the learning process from human demonstrations, significantly reducing the required time from several hours (Bahl et al., 2022) to less than 30 minutes. However, these results are achieved with certain limitations that present opportunities for future enhancements.

One primary limitation of our approach is its inability to incorporate orientation within the extracted trajectories. This restricts VIEW from handling tasks where orientation is crucial, such as pouring. The limitation stems from

[4] WHIRL was shown to work for similar tasks in the original paper (Bahl et al., 2022). However, we were unable to reproduce these results. We acknowledge that we replaced WHIRL's original reward model with our own agent-agnostic reward. However, this new reward provides more explicit feedback about the task and exploration. See Appendix for more details.

[5] It is important to note that our reward model provides explicit feedback about the *tagged* object: the rewards do not change if WHIRL moves any object other than the *tagged* object. In contrast, the original reward model in WHIRL compares the agent-agnostic action embeddings. This would pose a significant challenge in a cluttered environment since the robot would still be executing the right behavior, but manipulating the wrong object. For instance, if the robot were to move the kettle instead of the cup, it performs the same action — moving — and would receive a high reward even though it actually fails to complete the task.

our discretization process for grasp exploration, which uses Centroidal Voronoi Tessellations to identify potential grasp points. This process does not extend to non-affine elements, such as orientation. Addressing this limitation in future work could involve incorporating Euler-Rodrigues formulas (Dai, 2015) to handle orientation changes in an affine space, allowing VIEW to extend to tasks requiring both positional and rotational considerations. By integrating orientation, the system could more effectively perform complex manipulation tasks.

Another limitation is VIEW's dependence on a fixed camera pose between the human demonstration and the robot's rollouts. Since our reward computation relies on object detection and pixel-based comparisons of object centroid locations, consistency in camera setup is essential to ensure accurate alignment between human demonstrations and robot actions. A promising direction to address this limitation is to integrate an object detector capable of pose estimation, such as PoseCNN (Chéron et al., 2015) or Bundle-SDF (Wen et al., 2023), which would allow for pose-based rather than purely pixel-based comparisons. By transforming object poses into a world-coordinate system that is invariant to camera angles, VIEW could adapt to scenarios with variable camera positions, further broadening its applicability.

Lastly, VIEW relies on environmental consistency between the demonstration and learning environments, which constrains the approach to task-specific setups. For example, if a demonstration shows a human picking up a cup from a specific location, the robot learns to perform the action from that exact position; any change in object location requires a new demonstration. To overcome this, future work could consider defining waypoints relative to object positions rather than in fixed 3D coordinates. Combined with a pose-based reward function, this would enable the system to perform tasks even if object positions change.

Despite these limitations, our method presents a promising step toward one-shot visual imitation learning. In addition, our method can also serve as a translation layer for downstream policy learning. Recall that a key limitation of policy learning approaches is the requirement for state-action pairs. VIEW can quickly generate these state-action pairs from video demonstrations, and with a robust residual network, we believe it can do so without the need for exploration. These state-action pairs could then be used in behavior cloning or a policy learning frameworks to learn more adaptable policies. This integration would enable VIEW to data-efficiently translate human demonstrations into imitation learning policies that can respond to changes in the world state, serving as a foundational layer for broader task generalization.

Fig. 15 Objects manipulated in our real-world experiments. (From left to right) We use a bottle of bleach, a kettle, a mug, a banana, an apple, a bottle of mustard, and a basket. These seven distinct items were systematically selected for assessment based on their varying shapes, sizes, and colors to provide a comprehensive evaluation of our algorithm

8 Conclusion

State-of-the-art visual imitation learning methods rely on intricate architectures to manage the complexities present in video demonstrations. This paper introduces an alternative framework designed to streamline the learning process by compressing video data and honing in on crucial features and waypoints. We show that by concentrating on these essential aspects, robots can more rapidly learn tasks from human video demonstrations. Our method, VIEW, incorporates distinct modules for (a) generating a condensed prior that captures the key aspects of the human demonstrator's intent, (b) facilitating targeted exploration around the waypoints in the prior through a division into grasp and task execution phases, and (c) employing a residual model to enhance learning efficiency by drawing on insights from previously completed tasks.

Through an ablation study in a simulated environment, we examine the contribution of each module to VIEW's overall efficacy. Our method achieves over 80% success rate when using our proposed trajectory compression, and it achieves over 90% success rate with our exploration approach. Further, using our residual network leads to a dramatic decrease in the number of rollouts (more than 40%) needed to solve each task. Subsequent real-world experiments, utilizing videos of human demonstrations, further validate our method's capability to effectively learn from such demonstrations. In particular, our method achieves over 80% success rate in *Uncluttered* tasks and achieves 100% success rate in *Cluttered* tasks, outperforming the state-of-the-art baseline. The combined results from our simulation studies and real-world testing indicate that VIEW can efficiently learn tasks demonstrated using a single video, typically requiring under

30 minutes and fewer than 20 real-world trials. Additionally, we advance the capabilities of human-to-robot visual imitation learning by showing that VIEW can learn from arbitrarily long video demonstrations involving multiple object interactions. These findings are illustrated in our supplemental videos, available here: https://collab.me.vt.edu/view/

A Appendix

A.1 Implementation details

A public repository of our code can be found here: https://github.com/VT-Collab/view.

Data collection In our experiments, we use the Intel RealSense D435 RGB-D camera to capture video demonstrations, recording both RGB and depth data at 60 frames per second for each demonstration and rollout. While the depth data can be noisy, the Intel RealSense SDK offers several filters to enhance quality; we found that the hole-filling filter provided the most significant improvement. Despite this filtering, noise still introduced deviations of 5 to 19 cm from the ground truth waypoints. All our experiments incorporated this level of noise in the extracted waypoints, and our method, VIEW, effectively denoised these waypoints to achieve reliable results.

Hand trajectory extraction In line with the methodology described in WHIRL (Bahl et al., 2022), we utilize the 100 Days of Hands (100DOH) detector from https://github.com/ddshan/hand_detector.d2 for identifying hand-object contact points. For wrist detection, we integrate this with FrankMocap, as documented in Rong et al. Rong et al. (2021), without any model fine-tuning. The implementation for FrankMocap can be found here: https://github.com/facebookresearch/frankmocap. To obtain compressed trajectories, we combine the output of the FrankMocap model with SQUISHE. We develop our own version of SQUISHE based on the description provided in Muckell et al. (2014). Our implementation can be accessed in the code repository.

Object trajectory extraction To identify objects within the scene, we use Mask R-CNN, as detailed by He et al. He et al. (2017), through its implementation in Detectron2 (https://github.com/facebookresearch/detectron2). Following the methodologies outlined in Gouda et al. (2023), we initially pretrain our model using the Nvidia Falling Things dataset (Tremblay et al., 2018) and the DoPose-6D dataset (Gouda et al., 2022). We then finetune the model on a custom dataset containing 21 objects, with a subset of 7 being directly relevant to our final evaluations. This subset includes standard objects from the YCB object dataset (Calli et al., 2017) and others that are commonly found in kitchen environments. The complete list of objects used in our evaluation is shown in Fig. 15.

Object detection during runtime Given the challenges of ensuring object visibility and detection in every frame, we detect the object's location only at key waypoints identified through compression with SQUISHE, interpolating for all other points along the trajectory. This approach allows us to perform object detection at select locations, significantly reducing the computational load during runtime.

Residual network For our residual network, we employ a fully connected multi-layer perceptron with two hidden layers, utilizing ReLU as the activation function and mean squared error (MSE) for loss calculation. We use the Adam optimizer and train the network for 100 epochs. The initial learning rate is set at 0.1, with a decay factor of 0.15. Our network takes each waypoint as an input and then outputs the corrected waypoint. We use the same residual network across multiple tasks and multiple objects. For more detailed information on our training parameters, please refer to our code repository.

Robot Experiments In our robot experiments, during the grasp exploration the robot searches for a good grasp location. At the end of each rollout, proctors manually reset the objects to their original positions ensuring minimal errors in the object positions. Once the robot identifies a successful grasp location, the environment reset is automated, with the robot itself returning the item to its original position. The robot movement in these exploration rollouts are executed with a compliance controller to present any damages to the robot or the objects.

A.2 Challenges with WHIRL

Because of the lack of publicly available implementations of WHIRL, we developed our version based on the algorithms provided in WHIRL's publication (Bahl et al., 2022). As described, we used a four-layer MLP, implemented as a Variational Autoencoder and optimized via KL divergence loss. Initially — consistent with the guidelines in WHIRL's manuscript — we employed Copy-Paste Networks for inpainting (Lee et al., 2019) and the moment model from Monfort et al. Monfort et al. (2021) for calculating rewards.

However, during our experiments, we encountered two major challenges with WHIRL (see Fig. 16). The first issue was the inconsistency observed in the video inpainting performance, where the Copy-Paste Network failed to fully remove the robot from several frames (See Fig. 16 Top). This inconsistency persisted even after we fine-tuned the model on 400 custom images of our robot. Having consistent images with the human and the robot removed are particularly critical because WHIRL's exploration strategy relies heavily on comparing embeddings across frames. Due to the erratic inpainting results, the robot often converged to suboptimal positions, distant from the target object (See Fig. 16 Bottom Left). The second issue pertained to the rewards linked with

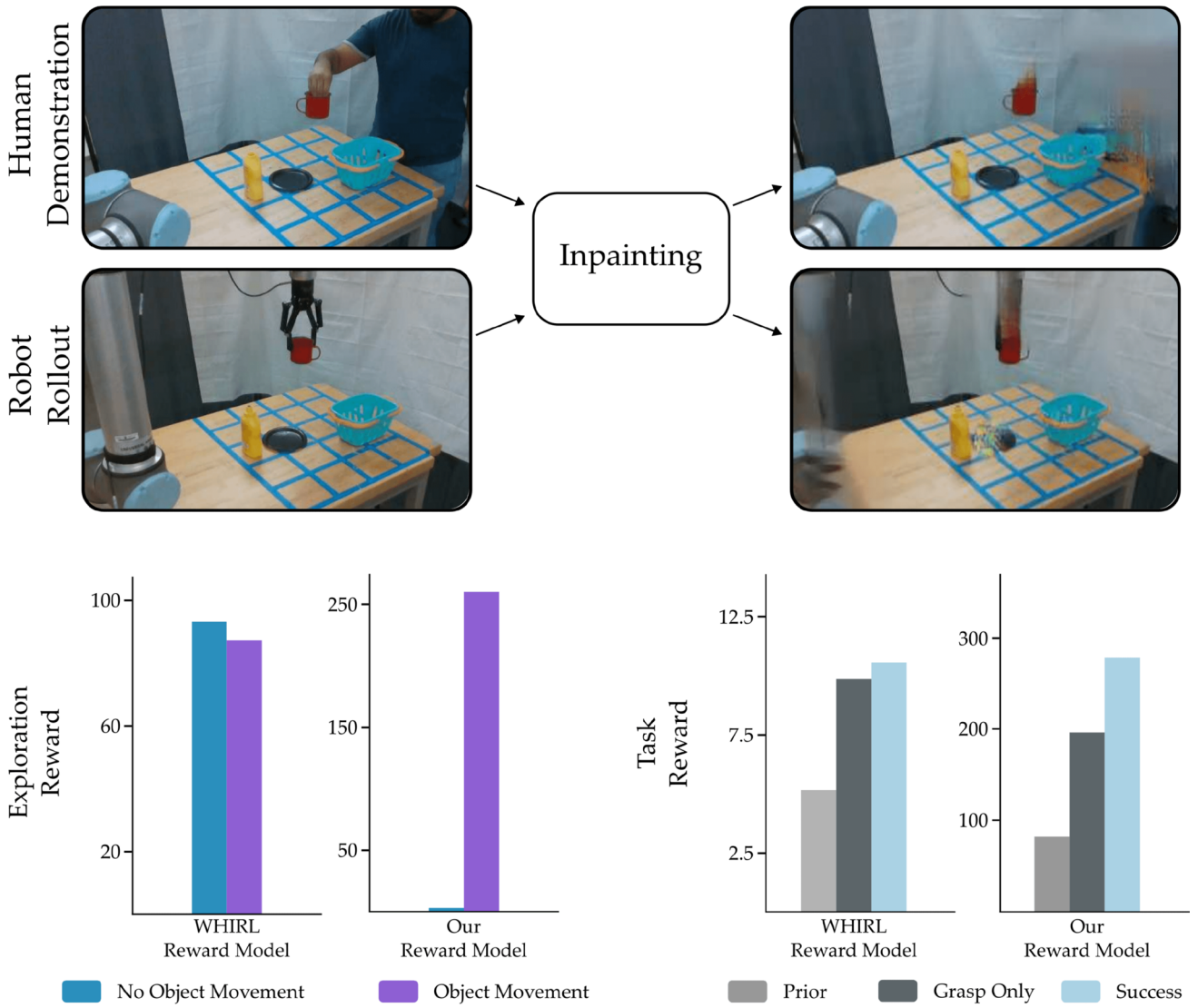

Fig. 16 Challenges with WHIRL Evaluation. (Top) As described in Bahl et al. (2022), we utilized Copy-Paste Networks (Lee et al., 2019) for the purpose of video inpainting, with the aim of removing both the human demonstrator and the robot arm from the video frames. This process is critical for enabling the comparison of frames through moment models (Monfort et al., 2021), which in turn facilitates the computation of agent-agnostic rewards. However, in our evaluations we encountered consistency issues with the inpainted images, leading to highly variable reward signals. (Bottom Left) The inconsistency in reward signals led to scenarios where the robot received high exploration rewards without actually moving the object. This is problematic because the robot relies on these rewards to identify waypoints that are near the object, which are necessary for successful grasping. In contrast, WHIRL with our reward model produces low exploration rewards when there is no object movement, and rewards increase significantly only when the object is displaced. This variability in the WHIRL reward model often caused the robot's learning trajectory to converge prematurely at a suboptimal point, usually far from the target object. (Bottom Right) When the robot managed to overcome the variability in exploration rewards and successfully grasped the object, we observed that the reward difference between just grasping the object and completing the entire task was minimal. WHIRL with our reward model provided a clearer distinction between these different phases of the task. The lack of clear reward differentiation in WHIRL's reward model frequently hindered the robot's ability to fully learn the task, often resulting in the robot only learning to pick up the object without completing subsequent steps. Based on these results, in our experiments from Sect. 6 we used WHIRL with our proposed reward model instead of WHIRL with its original reward model

task completion. There was a marked lack of differentiation in rewards between trajectories where the robot only grasped the object and those where it successfully completed the task (see Fig. 16 Bottom Right). This similarity in rewards often caused the robot to become stuck in a local minimum, proficient at object pickup but failing to complete the rest of the manipulation task.

In response to these issues, we replaced the reward model from WHIRL with the reward model described in Sect. 4.2. While WHIRL calculates exploration rewards based on the variance in frame embeddings and task rewards through the difference in video embeddings, our method takes a different approach. We explicitly compute exploration rewards using changes in the object's position and gauge task completion by measuring the object's proximity to the demonstrated trajectory. Our reward structure therefore capitalizes on direct and pertinent information (i.e., the location of the target object) rather than an indeterminate high-dimensional representation. We conducted a limited set of experiments to ensure that reward responses from our model were comparable to those from the original moment model. We believe that this modification does not fundamentally alter the functionality of WHIRL, and is a reasonable baseline for comparison.

A.3 Algorithms

Algorithm 1 VIEW

Input: Video of human interaction V
Residual network Φ
Dataset of previous corrections $\mathcal{D}$
Output: Successful robot trajectory ξ^r
Updated Residual network Φ

ξ^h = ExtractHandTraj(V)
ξ^o, tag = ExtractObjectTraj(V, ξ^h)
$\xi^h_c = \xi^h + \Phi(\xi^h)$
ξ^h_{grasp}, ξ^h_{task} = DivideTraj(ξ^h_c)
ξ^*_{grasp} = GraspExplore(ξ^h_{task}, ξ^o, tag)
if ξ^*_{grasp} is success **then**
  ξ^*_{task} = TaskExplore(ξ^h_{task}, ξ^*_{grasp}, ξ^o, tag)
  if ξ^*_{task} is success **then**
    ξ^* = CombineTraj(ξ^*_{grasp}, ξ^*_{task})
    Add (ξ^h , ξ^*) to $\mathcal{D}$
    Retrain Φ on $\mathcal{D}$
  end if
end if

Algorithm 2 Object Trajectory Extraction

Input: Video of human interaction V
Depth information D^h
Human hand trajectory ξ^h
Object detection model OD
Set of Anchor boxes A
Camera intrinsic and extrinsic parameters $\mathcal{C}^c_r$
Output: Object tag
Object trajectory in pixel space ζ^o_h
Object trajectory in 3D space ξ^o_h

Initialize OD
Initialize object count
for contact information c_t in ξ^h if c_t = True **do**
  v_t = ExtractVideoFrame(V, c_t)
  for Anchor box α in A **do**
    objects = OD(α)
    Update object count
  end for
end for
tag = Max(object count)
ξ^o_h, ζ^o_h = ExtractObjectTraj(V ,tag)
return ξ^o_h, ζ^o_h, tag

function ExtractObjectTraj(V, tag)
  Initialize an empty lists ξ_h, ζ_h
  for Video frame v_t in V **do**
    $p^o_{x_t}$, $p^o_{y_t}$ = $OD(v_t, tag)$
    Append ($p^o_{x_t}$, $p^o_{y_t}$) to ζ_h
    $\delta^o_t = D^h(p^o_{x_t}, p^o_{y_t})$
    $x^o_t, y^o_t, z^o_t = \mathcal{C}^c_r(p^o_{x_t}, p^o_{y_t}, \delta^o_t)$
    Append (x^o_t, y^o_t, z^o_t) to ξ^o_i
  end for
  return ξ_h, ζ_h
end function

Algorithm 3 Grasp Exploration

Input: Prior trajectory of grasping ξ^h_{grasp}
object location in robot coordinates ω^o

Initialize δ, ϵ, $p_{explore}$
Initialize flag $local = False$
Initialize an empty list $\Omega^r_{visited}$
Get the point where human grasps the object ω^h_{grasp} from prior ξ^h_{grasp}
Define bounding box $\mathcal{B}$ that circumscribes the points $\omega^o_{grasp} - \Delta\hat{j}$ and $\omega^h_{grasp} + \Delta\hat{j}$
Sample random $points$ from $\mathcal{B}$ and initialize the set $\Omega^r_{unvisited} =$ K- MEANS$(points)$
Initialize Bayesian optimizer BO

function ASK
    Generate p from uniform distribution [0, 1]
    if $p < p_{explore}$ **then**
        Sample high-level waypoint ω^r from $\Omega^r_{unvisited}$ using Equation 5
        Change flag $local = False$
        return ω^r
    else
        Sample high-level waypoint from $\Omega^r_{visited}$ with probability distribution from Equation 6
        Start low-level search by defining a bounding box around this waypoint with limits ϵ
        Query BO for ω^r
        Change flag $local = True$
        return ω^r
    end if
end function

function TELL(ω^r_i, R_i)
    if $local$ **then**
        Update BO with ω^r_i, R_i
    else
        Remove ω^r_i from $\Omega^r_{unvisited}$
        Add (ω^r_i, R_i) to $\Omega^r_{visited}$
    end if
end function

while *grasp* is not successful **do**
    Sample a waypoint $\omega^r =$ ASK
    Execute trajectory $\xi^r = \{\omega^r_{t_1}, \omega^r_{t_2}, \ldots \omega^r, \omega^r_{grasp+1}\}$
    Get the reward R using Equation 1
    Inform the explorer TELL(ω^r, R)
end while

Algorithm 4 Task Exploration

Input: Prior trajectory of task ξ^h_{task}
Define bounding box for each waypoint $\omega^r \in \xi^r_{task}$
Initialize a separate Bayesian optimizer BO for each waypoint in task

function ASK
    Initialize an empty list ξ^r_{task}
    for $\omega^h_i \in \xi^h_{task}$ **do**
        Query BO for ω^r_i
        Add ω^r_i to ξ^r_{task}
    end for
    return ξ^r_{task}
end function

function TELL(ξ^r_{task}, R)
    for $i = 1, \ldots, n$ **do**
        Update corresponding BO with $\omega^r_i \in \xi^r_{task}, R_i \in R$
    end for
end function

while *task* is not successful **do**
    Sample trajectory $\xi^r_{task} =$ ASK
    Execute trajectory ξ^r_{task} in environment
    Get the reward R for each waypoint in the trajectory using Equation 1
    Inform the explorer TELL(ξ^r_{task}, R)
end while

Supplementary Information The online version contains supplementary material available at https://doi.org/10.1007/s10514-024-10188-y.

Acknowledgements We thank Heramb Nemlekar for his valuable feedback on our manuscript.

Author Contributions A.J. led the algorithm development for prior extraction and agent agnostic reward computation. S.P. led the development for exploration. A.J. and S.P. wrote the first manuscript draft. A.J. ran the simulations and S.P. conducted the physical experiments. D.L. supervised the project, helped develop the method, and edited the manuscript.

Funding This research was supported in part by the USDA National Institute of Food and Agriculture, Grant 2022-67021-37868.

Declarations

Conflict of interest The authors declare that they have no conflict of interest.

Ethical Statement All physical experiments that relied on interactions with humans were conducted under university guidelines and followed the protocol of Virginia Tech IRB #20-755.

References

Alakuijala, M., Dulac-Arnold, G., Mairal, J., Ponce, J., & Schmid, C. (2023). Learning reward functions for robotic manipulation by observing humans. In *IEEE international conference on robotics and automation* (pp. 5006–5012).

Amiranashvili, A., Dorka, N., Burgard, W., Koltun, V., & Brox, T. (2020). Scaling imitation learning in minecraft. arXiv preprint arXiv:2007.02701

Bahl, S., Gupta, A., & Pathak, D. (2022) Human-to-robot imitation in the wild. In *Robotics: Science and systems*.

Brown, D., Goo, W., Nagarajan, P., & Niekum, S. (2019). Extrapolating beyond suboptimal demonstrations via inverse reinforcement learning from observations. In *International conference on machine learning* (pp. 783–792).

Brown, D. S., Goo, W., & Niekum, S. (2020). Better-than-demonstrator imitation learning via automatically-ranked demonstrations. In *Conference on robot learning* (pp. 330–359).

Caba Heilbron, F., Escorcia, V., Ghanem, B., & Carlos Niebles, J. (2015) ActivityNet: A large-scale video benchmark for human activity understanding. In *IEEE conference on computer vision and pattern recognition* (pp. 961–970).

Calli, B., Singh, A., Bruce, J., Walsman, A., Konolige, K., Srinivasa, S., Abbeel, P., & Dollar, A. M. (2017). Yale-CMU-Berkeley dataset for robotic manipulation research. *The International Journal of Robotics Research, 36*(3), 261–268.

Cetin, E., & Celiktutan, O. (2021). Domain-robust visual imitation learning with mutual information constraints. In *International conference on learning representations*.

Chane-Sane, E., Schmid, C., & Laptev, I. (2023) Learning video-conditioned policies for unseen manipulation tasks. In *International conference on robotics and automation* (pp. 909–916).

Chen, J., Yuan, B., & Tomizuka, M. (2019) Deep imitation learning for autonomous driving in generic urban scenarios with enhanced safety. In *IEEE/RSJ International conference on intelligent robots and systems* (pp. 2884–2890).

Chéron, G., Laptev, I., & Schmid, C. (2015) P-CNN: Pose-based CNN features for action recognition. In *IEEE international conference on computer vision* (pp. 3218–3226).

Dai, J. S. (2015). Euler-Rodrigues formula variations, quaternion conjugation and intrinsic connections. *Mechanism and Machine Theory, 92*, 144–152.

Das, P., Xu, C., Doell, R. F., & Corso, J. J. (2013). A thousand frames in just a few words: Lingual description of videos through latent topics and sparse object stitching. In *IEEE conference on computer vision and pattern recognition* (pp. 2634–2641).

Duan, J., Wang, Y. R., Shridhar, M., Fox, D., & Krishna, R. (2023) AR2-D2: Training a robot without a robot. arXiv preprint arXiv:2306.13818

Dulac-Arnold, G., Levine, N., Mankowitz, D. J., Li, J., Paduraru, C., Gowal, S., & Hester, T. (2021). Challenges of real-world reinforcement learning: Definitions, benchmarks and analysis. *Machine Learning, 110*(9), 2419–2468.

Fang, B., Jia, S., Guo, D., Xu, M., Wen, S., & Sun, F. (2019). Survey of imitation learning for robotic manipulation. *International Journal of Intelligent Robotics and Applications, 3*, 362–369.

Fontaine, M. C., Togelius, J., Nikolaidis, S., & Hoover, A. K. (2020). Covariance matrix adaptation for the rapid illumination of behavior space. In *Genetic and evolutionary computation conference* (pp. 94–102).

Gouda, A., Ghanem, A., & Reining, C. (2022). DoPose-6D dataset for object segmentation and 6D pose estimation. In *IEEE international conference on machine learning and applications* (pp. 477–483).

Gouda, A., & Roidl, M. (2023) DoUnseen: Zero-shot object detection for robotic grasping. arXiv preprint arXiv:2304.02833

Goyal, R., Ebrahimi Kahou, S., Michalski, V., Materzynska, J., Westphal, S., Kim, H., Haenel, V., Fruend, I., Yianilos, P., & Mueller-Freitag, M. (2017) The" something something" video database for learning and evaluating visual common sense. In *IEEE international conference on computer vision* (pp. 5842–5850).

Habibian, S., Jonnavittula, A., & Losey, D. P. (2022). Here's what I've learned: Asking questions that reveal reward learning. *ACM Transactions on Human-Robot Interaction (THRI), 11*(4), 1–28.

He, K., Gkioxari, G., Dollár, P., & Girshick, R. (2017). Mask R-CNN. In *IEEE international conference on computer vision* (pp. 2961–2969).

Hussein, A., Gaber, M. M., Elyan, E., & Jayne, C. (2017). Imitation learning: A survey of learning methods. *ACM Computing Surveys, 50*(2), 1–35.

Isola, P., Zhu, J. Y., Zhou, T., & Efros, A. A. (2017) Image-to-image translation with conditional adversarial networks. In *IEEE conference on computer vision and pattern recognition* (pp. 1125–1134).

Jain, V., Attarian, M., Joshi, N. J., Wahid, A., Driess, D., Vuong, Q., Sanketi, P. R., Sermanet, P., Welker, S., Chan, C., et al. (2024). Vid2robot: End-to-end video-conditioned policy learning with cross-attention transformers. arXiv preprint arXiv:2403.12943

Jin, J., Petrich, L., Dehghan, M., & Jagersand, M. (2020) A geometric perspective on visual imitation learning. In *IEEE/RSJ international conference on intelligent robots and systems* (pp. 5194–5200).

Jonnavittula, A., & Losey, D. P. (2021). I know what you meant: Learning human objectives by (under) estimating their choice set. In *IEEE international conference on robotics and automation* (pp. 2747–2753).

Jonnavittula, A., & Losey, D. P. (2021). Learning to share autonomy across repeated interaction. In *IEEE/RSJ international conference on intelligent robots and systems* (pp. 1851–1858).

Jonnavittula, A., Mehta, S. A., & Losey, D. P. (2024). SARI: Shared autonomy across repeated interaction. *ACM Transactions on Human-Robot Interaction, 13*(2), 1–36.

Kelly, M., Sidrane, C., Driggs-Campbell, K., & Kochenderfer, M. J. (2019) HG-DAgger: Interactive imitation learning with human experts. In *IEEE international conference on robotics and automation* (pp. 8077–8083).

Kim, M. J., Wu, J., & Finn, C. (2023). Giving robots a hand: Learning generalizable manipulation with eye-in-hand human video demonstrations. arXiv preprint arXiv:2307.05959

Kober, J., Bagnell, J. A., & Peters, J. (2013). Reinforcement learning in robotics: A survey. *The International Journal of Robotics Research, 32*(11), 1238–1274.

Lee, R., Abou-Chakra, J., Zhang, F., & Corke, P. (2022). Learning fabric manipulation in the real world with human videos. arXiv preprint arXiv:2211.02832

Lee, S., Oh, S. W., Won, D., & Kim, S. J. (2019). Copy-and-paste networks for deep video inpainting. In *IEEE/CVF international conference on computer vision* (pp 4413–4421).

Li, J., Lu, T., Cao, X., Cai, Y., & Wang, S. (2021). Meta-imitation learning by watching video demonstrations. In *International conference on learning representations*.

Liu, P., Orru, Y., Paxton, C., Shafiullah, N. M. M., & Pinto, L. (2024) OK-Robot: What really matters in integrating open-knowledge models for robotics. arXiv preprint arXiv:2401.12202

Liu, Y., Gupta, A., Abbeel, P., & Levine, S. (2018) Imitation from observation: Learning to imitate behaviors from raw video via context translation. In *IEEE International conference on robotics and automation* (pp. 1118–1125).

Lynch, C., & Sermanet, P. (2020). Language conditioned imitation learning over unstructured data. In *Robotics: Science and systems*.

Mehta, S. A., Habibian, S., & Losey, D. P. (2024). Waypoint-based reinforcement learning for robot manipulation tasks. arXiv preprint arXiv:2403.13281

Mehta, S. A., & Losey, D. P. (2023). Unified learning from demonstrations, corrections, and preferences during physical human-robot interaction. *ACM Transactions on Human-Robot Interaction, 13*(3), 1–25.

Menda, K., Driggs-Campbell, K., & Kochenderfer, M. J. (2019). EnsembleDAgger: A Bayesian approach to safe imitation learning. In: *IEEE/RSJ international conference on intelligent robots and systems* (pp. 5041–5048).

Monfort, M., Pan, B., Ramakrishnan, K., Andonian, A., McNamara, B. A., Lascelles, A., Fan, Q., Gutfreund, D., Feris, R. S., & Oliva, A. (2021). Multi-moments in time: Learning and interpreting models for multi-action video understanding. *IEEE Transactions on Pattern Analysis and Machine Intelligence, 44*(12), 9434–9445.

Morales, E. F., Murrieta-Cid, R., Becerra, I., & Esquivel-Basaldua, M. A. (2021). A survey on deep learning and deep reinforcement learning in robotics with a tutorial on deep reinforcement learning. *Intelligent Service Robotics, 14*(5), 773–805.

Muckell, J., Olsen, P. W., Hwang, J. H., Lawson, C. T., & Ravi, S. (2014). Compression of trajectory data: A comprehensive evaluation and new approach. *GeoInformatica, 18*, 435–460.

Padalkar, A., Pooley, A., Jain, A., Bewley, A., Herzog, A., Irpan, A., Khazatsky, A., Rai, A., Singh, A., Brohan, A., et al. (2023). Open X-embodiment: Robotic learning datasets and RT-X models. arXiv preprint arXiv:2310.08864

Pan, Y., Cheng, C. A., Saigol, K., Lee, K., Yan, X., Theodorou, E. A., & Boots, B. (2020). Imitation learning for agile autonomous driving. *The International Journal of Robotics Research, 39*(2–3), 286–302.

Pari, J., Shafiullah, N. M., Arunachalam, S. P., & Pinto, L. (2021). The surprising effectiveness of representation learning for visual imitation. In *Robotics: Science and systems*.

Patel, A., Wang, A., Radosavovic, I., & Malik, J. (2022). Learning to imitate object interactions from internet videos. arXiv preprint arXiv:2211.13225

Pomerleau, D. A. (1991). Efficient training of artificial neural networks for autonomous navigation. *Neural Computation, 3*(1), 88–97.

Rafailov, R., Yu, T., Rajeswaran, A., & Finn, C. (2021). Visual adversarial imitation learning using variational models. *Advances in Neural Information Processing Systems, 34*, 3016–3028.

Ratliff, N., Bagnell, J. A., & Srinivasa, S. S. (2007). Imitation learning for locomotion and manipulation. In *IEEE-RAS international conference on humanoid robots* (pp. 392–397).

Ren, S., He, K., Girshick, R., & Sun, J. (2016). Faster R-CNN: Towards real-time object detection with region proposal networks. *IEEE Transactions on Pattern Analysis and Machine Intelligence, 39*(6), 1137–1149.

Romero, J., Tzionas, D., & Black, M. J. (2017). Embodied hands: Modeling and capturing hands and bodies together. *ACM Transactions on Graphics, 36*(6), 245.

Rong, Y., Shiratori, T., & Joo, H. (2021). FrankMocap: A monocular 3d whole-body pose estimation system via regression and integration. In *IEEE international conference on computer vision workshops* (pp. 1749–1759).

Ross, S., Gordon, G., & Bagnell, D. (2011). A reduction of imitation learning and structured prediction to no-regret online learning. In *International conference on artificial intelligence and statistics* (pp. 627–635).

Schaal, S. (1996). Learning from demonstration. *Advances in Neural Information Processing Systems, 9*, 1040–1046.

Schäfer, L., Jones, L., Kanervisto, A., Cao, Y., Rashid, T., Georgescu, R., Bignell, D., Sen, S., Gavito, A. T., & Devlin, S. (2023). Visual encoders for data-efficient imitation learning in modern video games. arXiv preprint arXiv:2312.02312

Scheller, C., Schraner, Y., & Vogel, M. (2020) Sample efficient reinforcement learning through learning from demonstrations in minecraft. In *NeurIPS competition and demonstration track* (pp. 67–76).

Sermanet, P., Xu, K., & Levine, S. (2017). Unsupervised perceptual rewards for imitation learning. In *Robotics: Science and systems*.

Shafiullah, N. M. M., Rai, A., Etukuru, H., Liu, Y., Misra, I., Chintala, S., & Pinto, L. (2023). On bringing robots home. arXiv preprint arXiv:2311.16098

Shan, D., Geng, J., Shu, M., & Fouhey, D. F. (2020) Understanding human hands in contact at internet scale. In *IEEE/CVF conference on computer vision and pattern recognition* (pp. 9869–9878).

Sharma, P., Pathak, D., & Gupta, A. (2019). Third-person visual imitation learning via decoupled hierarchical controller. In *Advances in neural information processing systems* (vol. 32).

Shaw, K., Bahl, S., Sivakumar, A., Kannan, A., & Pathak, D. (2024). Learning dexterity from human hand motion in internet videos. *The International Journal of Robotics Research, 43*(4), 513–532.

Shi, L. X., Hu, Z., Zhao, T. Z., Sharma, A., Pertsch, K., Luo, J., Levine, S., & Finn, C. (2024) Yell at your robot: Improving on-the-fly from language corrections. arXiv preprint arXiv:2403.12910

Shi, L. X., Sharma, A., Zhao, T. Z., & Finn, C. (2023) Waypoint-based imitation learning for robotic manipulation. In *Conference on Robot learning*

Sieb, M., Xian, Z., Huang, A., Kroemer, O., & Fragkiadaki, K. (2020) Graph-structured visual imitation. In *Conference on Robot learning* (pp. 979–989).

Smith, L., Dhawan, N., Zhang, M., Abbeel, P., & Levine, S. (2020) AVID: Learning multi-stage tasks via pixel-level translation of human videos. In *Robotics: Science and systems*.

Snoek, J., Larochelle, H., Adams, R. P. (2012). Practical Bayesian optimization of machine learning algorithms. In *Advances in neural information processing systems* (vol. 25).

Song, S., Zeng, A., Lee, J., & Funkhouser, T. (2020). Grasping in the wild: Learning 6dof closed-loop grasping from low-cost demonstrations. *IEEE Robotics and Automation Letters, 5*(3), 4978–4985.

Taranovic, A., Kupcsik, A. G., Freymuth, N., & Neumann, G. (2022). Adversarial imitation learning with preferences. In *International conference on learning representations*.

Tremblay, J., To, T., & Birchfield, S. (2018). Falling things: A synthetic dataset for 3d object detection and pose estimation. In *IEEE conference on computer vision and pattern recognition workshops* (pp. 2038–2041).

Vassiliades, V., Chatzilygeroudis, K., & Mouret, J. B. (2017). Using centroidal Voronoi tessellations to scale up the multidimensional archive of phenotypic elites algorithm. *IEEE Transactions on Evolutionary Computation, 22*(4), 623–630.

Wang, J., Mueller, F., Bernard, F., Sorli, S., Sotnychenko, O., Qian, N., Otaduy, M. A., Casas, D., & Theobalt, C. (2020). Rgb2hands: Real-time tracking of 3D hand interactions from monocular RGB video. *ACM Transactions on Graphics, 39*(6), 1–16.

Wen, B., Lian, W., Bekris, K., & Schaal, S. (2022) You only demonstrate once: Category-level manipulation from single visual demonstration. In *Robotics: Science and systems*.

Wen, B., Tremblay, J., Blukis, V., Tyree, S., Müller, T., Evans, A., Fox, D., Kautz, J., & Birchfield, S. (2023). BundleSDF: Neural 6-DOF tracking and 3D reconstruction of unknown objects. In *IEEE/CVF conference on computer vision and pattern recognition* (pp. 606–617).

Wen, C., Lin, J., Qian, J., Gao, Y., & Jayaraman, D. (2021). Keyframe-focused visual imitation learning. In *International conference on machine learning* (vol. 139, pp. 11123–11133).

Xiong, H., Li, Q., Chen, Y. C., Bharadhwaj, H., Sinha, S., & Garg, A. (2021). Learning by watching: Physical imitation of manipulation skills from human videos. In *IEEE/RSJ international conference on intelligent robots and systems* (pp. 7827–7834).

Young, S., Gandhi, D., Tulsiani, S., Gupta, A., Abbeel, P., & Pinto, L. (2021). Visual imitation made easy. In *Conference on Robot learning*.

Zhang, F., Bazarevsky, V., Vakunov, A., Tkachenka, A., Sung, G., Chang, C. L., & Grundmann, M. (2020). Mediapipe hands: On-device real-time hand tracking. In *CVPR workshop on computer vision for augmented and virtual reality*.

Ananth Jonnavittula is an AI Researcher at Foundation Inc. He received his doctoral degree from Virginia Tech in 2024, master's degree from Worcester Polytechnic Institute in 2017, and bachelor's degree from SASTRA University in 2015. His research focuses on Imitation Learning for robot manipulation.

Sagar Parekh is a Ph.D. student in Mechanical Engineering at Virginia Tech, USA. His research interests include human-robot collaboration and robot learning. He was previously a research assistant at the Indian Institute of Technology (IIT) Gandhinagar, India until 2021. He received his bachelors degree from Nirma University,India.

Dylan P. Losey is an Assistant Professor in the Department of Mechanical Engineering at Virginia Tech. His research group develops learning and control algorithms for robots that interact with people. Dylan was previously a postdoctoral scholar at Stanford University. He received his doctoral degree from Rice University in 2018 and his bachelor's degree from Vanderbilt University in 2014. Dylan is an NSF CAREER recipient and has won best paper awards from the Conference on Robot Learning, the IEEE/ASME Transactions on Mechatronics, and the IEEE Transactions on Haptics.