Appendix

A Table of Contents

B Frequently Asked Questions (FAQ)

1. **How did you select those specific baselines and ablations in Sec. 6?**

   Our experiments showcase the capabilities of HITL-TAMP as (1) a scalable demonstration collection system and (2) an efficient learning and control framework. To show its value in collecting human demonstrations over an alternative, we compared it extensively against a widely-adopted conventional teleoperation paradigm used in prior works that collect and learn from human demonstrations [1, 6, 2, 8, 42, 21, 10, 37, 38, 43, 11, 16, 17, 12] (see Table 1 and Fig. 6).

   To show its value in learning policies for manipulation tasks, we investigated the value of the core component - the TAMP-gated control mechanism (described in Appendix I). We showed that even policies trained on conventional teleoperation data benefit substantially from incorporating the TAMP-gated control mechanism (Fig. 6). Our TAMP-gated control is a novel control algorithm made possible by key technical components of HITL-TAMP (as described in Sec. 3).

   There are other systems that are designed for specific contact-rich manipulation (such as peg insertion [44, 45]), but HITL-TAMP was not designed to be specialized for any specific task. Rather, it was meant to be a general-purpose system that can be applied to any contact-rich, long-horizon manipulation task, as long as the task can be demonstrated by a human operator, and described in PDDLStream.

2. **How does this work compare with other works that combine imitation learning and TAMP?**

   Prior works, such as [46], trained agents in simulation to imitate demonstration data provided by a TAMP supervisor in simulation. In this way, during deployment, an agent can operate without privileged information (such as object poses) required by TAMP. However, this setting makes a strong assumption that the TAMP system can already solve the target tasks. By contrast, our work extends a TAMP system's capabilities using an agent trained on human demonstration segments collected by HITL-TAMP (training details in Appendix J) in order to solve complex contact-rich tasks in the real world. Training an agent on the TAMP segments collected by HITL-TAMP in order to enable TAMP-free policy deployments is an exciting application for future work. However, it is orthogonal to the main contributions in this paper.

3. **What are the trade-offs between effort to provide demos and effort to design models/-controllers?**

   Collecting a large number of human demos can be labor and time intensive [12, 8, 38], but extensive modeling of a task for TAMP can similarly be time-consuming. Our system achieves a good tradeoff, by lessening the modeling burden for TAMP by deferring difficult task segments to the human, and lessening the human operator burden by only asking them to operate small segments of a task. When deploying HITL-TAMP (especially in real-world settings), there is significant flexibility in deciding what information is available to the TAMP system in order to automate portions of a task, and which portions of a task should instead be deferred to a human operator (or trained agent).

4. **How does the TAMP system determine which parts of a task plan require a human operator?**

   We formalize human-teleoperated TAMP skills in Sec. 3.1. While their discrete structure is provided by a human (e.g. which objects are involved), our novel action constraint learning technique (Sec. 3.2) characterizes their continuous action parameters. Human modelers have flexibility in deciding which skills should be teleoperated based on the contact-richness and required precision of the interaction. Fig. E.1 (in Appendix E) showcases the parts of each task that are handled by the TAMP system and the parts that are handled by the human (or trained agent).

5. **What assumptions are needed to apply HITL-TAMP to real-world settings, as opposed to simulation?**

   Typically, TAMP systems place a high burden on real-world perception, as accurate perception and dynamics models are often needed by TAMP for planning. Part of the motivation of our work was to reduce this requirement. While we do assume knowledge of crude object models and the ability to associate objects (see Sec. 6.3), we use a very simple perception pipeline in this work. We show that this simple pipeline suffices, **even for the challenging Tool Hang task in the real-world** since a human or an end-to-end trained policy handles the most challenging, contact-rich interactions.

C Limitations

In this section, we discuss some limitations of HITL-TAMP, which future work can address.

1. **Applicable tasks.** Our general-purpose system can be deployed on any tasks that (1) can be described in PDDLStream and (2) human operators can demonstrate. We did not engineer the system for any specific task — our system greatly extends the set of tasks that can be solved when compared to TAMP alone.
2. **Task variety.** The tasks in this work are focused on tabletop domains, and there is limited object variety in each task. Scaling HITL-TAMP to work for more scenes and objects requires a richer set of assets and scenes (in simulation) and a more robust perception pipeline in the real world.
3. **Prior information on what is difficult for TAMP.** HITL-TAMP requires prior information (at a high-level) on which task portions will be difficult for TAMP. Being able to automatically identify when human demonstrations are needed (e.g. based on uncertainty estimates from perception) is left for future work.
4. **Perception for TAMP.** We assume access to coarse object models and approximate pose estimation in order to conduct the TAMP segments. Future work could relax this assumption by integrating TAMP methods that do not require object models [36].

D Related Work

D.1 Demonstration Collection Systems for Robot Manipulation

Recent studies have shown the effectiveness of teaching robots manipulation skills through human demonstration [6, 1, 7, 8, 9, 10]. High-quality, large-scale demonstrations are crucial to this success [7]. Although recent advancements have made demonstration collection systems more scalable and user-friendly [6, 37], collecting a substantial amount of high-quality, long-horizon demonstrations remains time-consuming and labor-intensive [7]. On the other hand, intervention-based systems [47, 43] allow the demonstrator to proactively correct for near-failure cases. However, such systems require users to constantly monitor robot task executions, which is equally time-consuming and sometimes more cognitively-demanding than demonstrating a task [48]. Our system uses a TAMP-gated mechanism that automatically switches control between the robot and the demonstrator. The mechanism also enables a user to demonstrate for multiple sessions asynchronously, dramatically increasing the throughput of task demonstration.

A number of recent works have also investigated automatic control hand-offs in the context of online imitation learning [13, 14, 15, 16, 17]. These works have largely focused on iteratively improving a single learned policy, and the gating mechanisms rely on predicting task performances and action uncertainties, which are often policy and data-specific. Our work instead proposes to augment a TAMP system with imitation-learned policies. The symbolic abstractions of the TAMP system readily delineate TAMP's capabilities and can be used to determine the conditions for control hand-offs.

Our HITL-TAMP also acts as a TAMP-assisted teleoperation system. However, unlike most prior works in assisted robot teleoperation, for which the aims are for humans to provide high-level guidance for low-level autonomous control [49, 50, 51], HITL-TAMP focuses on allowing human teleoperators to ”fill the gap” for a TAMP system to complete goal-directed tasks and enabling the system to become more autonomous by learning skills from the human demonstrations.

D.2 Learning for Task and Motion Planning

Task and Motion Planning (TAMP) is a powerful approach for solving challenging manipulation tasks by breaking them into smaller, easier to solve symbolic-continuous search problems [5, 23, 4, 24]. However, TAMP requires prior knowledge of skills and environment models, making it unsuitable for contact-rich tasks where hand-defining models is difficult. Recent works have proposed to learn environment dynamic models [25, 26, 27], skill operator models [28, 29], and skill samplers [30, 31]. However, these methods still require a complete set of hand-crafted skills. Closest to our work are LEAGUE [32] and Silver *et al.* [33] that learn TAMP-compatible skills. However, both works are limited in their real-world applicability. LEAGUE relies on hand-defined TAMP plan sampler and expensive RL procedures to learn skills in simulation, while Silver *et al.* requires hard-coded demonstration policies that can already solve the target tasks. Our work instead leverage human demonstrations to both train visuomotor skills and informing TAMP plan sampling. We empirically show that HITL-TAMP can efficiently solve challenging tasks such as making coffee in the real world.

D.3 Imitation Learning from Human Demonstrations

Imitation learning techniques based on deep neural networks have shown remarkable performances in solving real-world manipulation tasks [6, 1, 11, 7, 8, 12]. We take a data-centric view [9, 7, 12] to scaling up imitation learning — HITL-TAMP speeds up demonstration collection for a wide range of contact-rich manipulation tasks. A trained HITL-TAMP also acts as a hierarchical policy [52]. The key difference to pure data-driven approaches [11, 52, 40, 9, 53] is that in HITL-TAMP, the TAMP framework directly drives the hierarchy to ensure that the learned skills are modular and compatible. Similarly, our work builds on research in combining learned and predefined skills [18, 19, 20, 21, 22] and formalizes human demonstrations and learned skills within a TAMP framework.

E Tasks

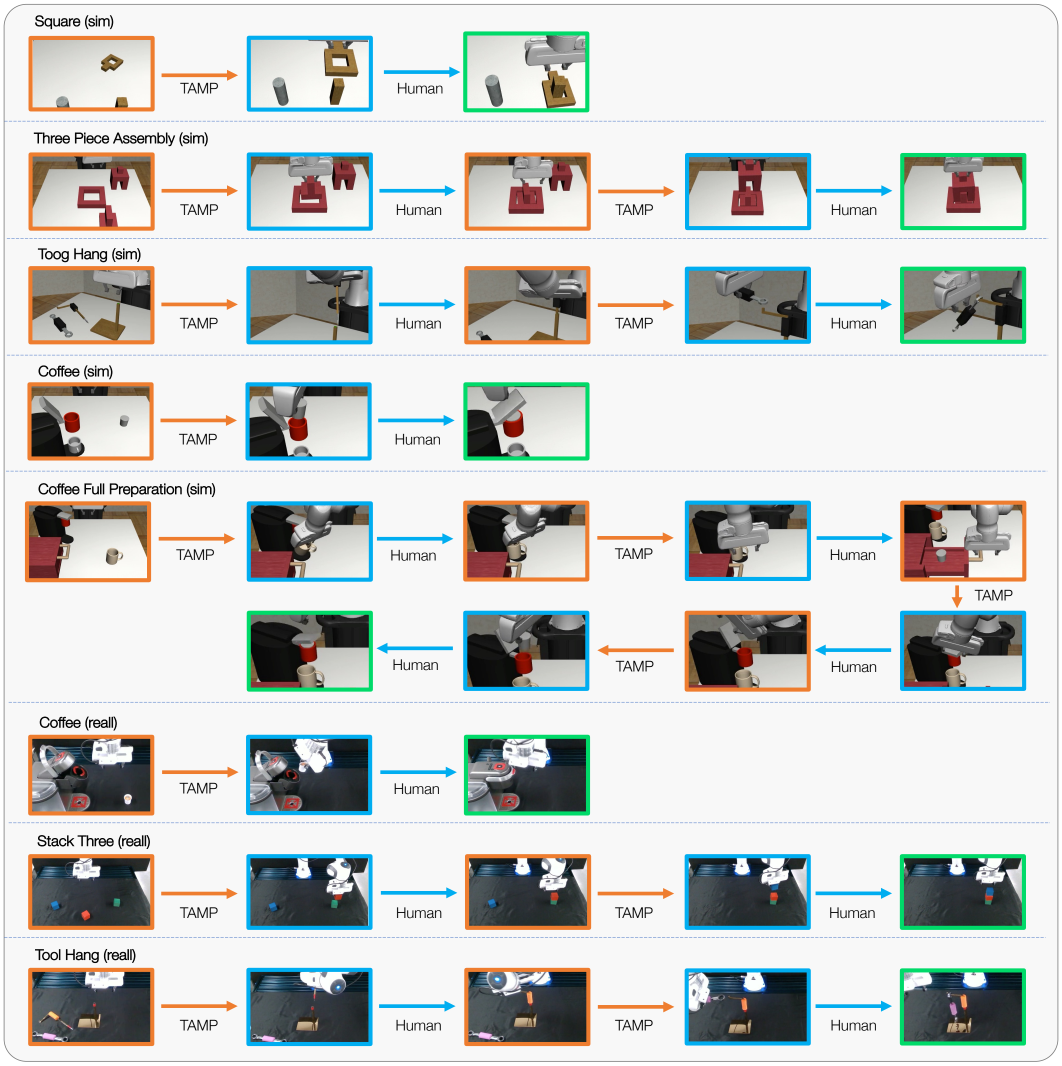

Figure E.1: **Task Segments.** We show the human and TAMP segments for each task.

In this section, we present extended task descriptions for each task, including a breakdown of which segments the human controls and which TAMP handles (see Fig. E.1).

Stack Three (real). The robot must stack 3 randomly placed cubes. The task consists of 4 total segments — TAMP handles grasping each cube and approaching the stack, and the human handles the placement of the 2 cubes on top of the stack.

Square [54, 1] (sim). The robot must pick a nut and place it onto a peg. The nut is initialized in a small region and the peg never moves. This task consists of two segments — TAMP grasps the nut and approaches the peg, and the human inserts the nut onto the peg.

Square Broad (sim).: The nut and peg are initialized anywhere on the table.

Coffee [43] (sim + real). The robot must pick a coffee pod, insert it into a coffee machine, and close the lid. The pod starts at a random location in a small, box-shaped region, and the machine is fixed. The task has two segments — TAMP grasps the pod and approaches the machine, and the human inserts the pod and closes the lid.

Coffee Broad (sim + real). The pod and the coffee machine have significantly larger initialization regions. With 50% probability, the pod is placed on the left of the table, and the machine on the right side, or vice-versa. Once a side is chosen for each, the machine location and pod location are further randomized in a significant region.

Three Piece Assembly (sim). The robot must assemble a structure by inserting one piece into a base and then placing a second piece on top of the first. The two pieces are placed around the base, but the base never moves. The tasks consists of four segments — TAMP grasps each piece and approaches the insertion point while the human handles each insertion.

Three Piece Assembly Broad (sim). The pieces are placed anywhere in the workspace.

Tool Hang [1] (sim + real). The robot must insert an L-shaped hook into a base piece to assemble a frame, and then hang a wrench off of the frame. The L-shaped hook and wrench vary slightly in pose, and the base piece never moves. The task has four segments — TAMP handles grasping the L-shaped hook and the wrench, and approaching the insertion / hang points, while the human handles the insertions.

Tool Hang Broad (sim). All three pieces move in larger regions of the workspace.

Coffee Full Preparation (sim). The robot must place a mug onto a coffee machine, retrieve a coffee pod from a drawer, insert the pod into the machine, and close the lid. The task has 8 segments — first TAMP grasps the mug and approaches the placement location, then the human places the mug on the coffee machine (the placement requires precision due to the arm size and space constraints). Next, TAMP approaches the machine lid, and the human opens the lid (requires extended contact with an articulated mechanism). Then, TAMP approaches the drawer handle, and the human opens the drawer. Finally, TAMP grasps the pod from inside the drawer and approaches the machine, and the human inserts the pod and closes the machine lid.

F Additional Data Throughput Comparisons

Task	**HITL-TAMP Time (min)**	**Conventional Time (min)**
Square	**13.5**	35.0
Square Broad	**14.0**	48.0
Coffee	**22.6**	46.4
Coffee Broad	**28.8**	57.8
Tool Hang	**48.0**	97.1
Tool Hang Broad	**51.5**	109.8
Three Piece Assembly	**30.0**	60.0
Three Piece Assembly Broad	**34.9**	68.3
Coffee Preparation	**78.4**	132.7
Total	**321.7**	655.1

Table F.1: **Collection time comparison to conventional teleoperation datasets.** An extended comparison of data collection time for 200 demos across several tasks for both HITL-TAMP and the conventional teleoperation system. Some items were estimated using the time spent collecting 10 human demonstrations.

In this section, we compare how long it would have taken to collect our 2.1K+ HITL-TAMP demonstrations with a conventional teleoperation system. The results are shown in Table F.1. Several of the numbers were estimated by collecting 10 human demonstrations and multiplying by 20 (due to the time burden of collecting 200 human demonstrations across all tasks with a conventional teleoperation system). In most cases, HITL-TAMP takes more than 2x fewer minutes to collect 200 demos than the conventional system.

G Demonstration Statistics

Task	**Human**	**Trajectory (HT)**	**Trajectory (C)**
Square	19.8	582.2	150.8
Square Broad	24.2	647.8	167.9
Coffee	71.6	472.0	199.3
Coffee Broad	90.6	663.7	273.8
Tool Hang	70.4	1297.9	479.8
Tool Hang Broad	71.3	1485.8	522.6
Three Piece Assembly	35.3	897.9	260.1
Three Piece Assembly Broad	39.6	1174.1	342.0
Coffee Preparation	43.8	1328.6	593.2
Stack Three (real)	60.9	499.2	-
Coffee (real)	295.3	494.9	-
Coffee Broad (real)	326.5	548.3	-
Tool Hang (real)	124.3	1144.5	-

Table G.1: **Demonstration Lengths.** For each task, we report the average length (time steps) of the human segment, the average trajectory length of our HITL-TAMP datasets (HT), and as a point of comparison, the average trajectory length of the conventional system data (C). Note that if a trajectory contains multiple human segments, we average them.

In Table G.1, we present the average length (time steps) of the human-provided segment, the average trajectory length of our HITL-TAMP datasets (HT), and as a point of comparison, the average trajectory length of the conventional system data (C). Note that if a trajectory contains multiple human segments, we average across them, and that some of the conventional system lengths are estimates based on collecting 10 trajectories (the same ones used for the analysis in Appendix F). We see that the average human segment is small compared to the entire trajectory length — this might help explain the efficacy of our TAMP-gated policy, since the policy is only responsible for short-horizon, contact-rich behaviors.

H Queueing System Analysis

In Sec. 4 and Fig. 3, we discussed our queueing system, which enables scalable data collection with HITL-TAMP by allowing a single human operator to manage a fleet of N_{robot} robot arms and ensuring that the human operator is always kept busy. In this section, we provide some additional derivations and analysis on how the choice of the number of robot arms influences data throughput.

Assuming that the human has an average queue consumption rate (number of task demonstrations completed per unit time) of R_H and the TAMP system has an average queue production rate (number of task segments executed successfully per unit time) of R_T, we would like the effective rate of production to match or exceed the rate of consumption,

$$R_T(N_{\text{robot}} - 1) \geq R_H.$$

Here, the minus 1 is because 1 robot is controlled by the human. Rearranging, we obtain $N_{\text{robot}} \geq 1 + \frac{R_H}{R_T}$. Thus, the size of the fleet should be at least one more than the ratio between the human rate of producing demonstration segments and the TAMP rate of solving and executing segments.

This number is often limited by either the amount of system resources (in simulation) or the availability of hardware (in real world). In practice, human operators also need to take breaks and have an effective "duty cycle" where they are kept busy $X\%$ of the time. HITL-TAMP can support this extension as well. Assume that the human is operating the system for T_{on} and resting for T_{off}. The human consumes items in the queue during T_{on} at an effective rate of

$$R_H - R_T(N_{\text{robot}} - 1),$$

and has the queue filled up during T_{off} at a rate of $R_T(N_{\text{robot}} - 1)$. Ensuring that the human consumption rate is less than or equal to the production rate, we have

$$T_{\text{on}}(R_H - R_T(N_{\text{robot}} - 1)) \leq T_{\text{off}} R_T(N_{\text{robot}} - 1).$$

After rearranging we arrive at

$$N_{\text{robot}} \geq 1 + \frac{R_H}{R_T} \frac{X}{100},$$

where

$$\frac{X}{100} = \frac{T_{\text{on}}}{(T_{\text{on}} + T_{\text{off}})}$$

is the human duty cycle ratio.

I Additional Details on TAMP-Gated Teleoperation

We provide additional details on how TAMP-gated teleoperation works. The TAMP system decides when to execute portions of a task, and when a human operator should complete a portion. Each teleoperation episode consists of one or more *handoffs* where the TAMP system prompts a human operator to control a portion of a task, or where the TAMP system takes control back after it determines that the human has completed their segment.

Algorithm 1 displays the pseudocode of the HITL-TAMP system: TAMP-GATED-CONTROL. It takes as input goal formula G. On each TAMP iteration, it observes the current state s. If it satisfies the goal, the episode terminates successfully. Otherwise, the TAMP system solves for a plan $\vec{a}$ using PLAN-TAMP from current state s to the goal G. We implement PLAN-TAMP using the *adaptive* PDDLStream algorithm [24]. The TAMP system then deploys its controller EXECUTE-JOINT-COMMANDS and issues joint position commands to the robot to carry out planned motions until reaching an action a that requires the human. At this time, control switches into teleoperation mode, where the human has full 6-DoF control of the end effector. We use a smartphone interface and map phone pose displacements to end effector displacements, similar to prior teleoperation systems [37, 38, 11]. The robot end effector is controlled using an Operational Space Controller [39]. As in [43], we apply phone pose differences as relative pose commands to the current end effector pose. This allows control to be decoupled from the current configuration of the robot arm, which is important as the TAMP system can prompt the human to takeover in diverse configurations. While the human is controlling the robot, the TAMP system monitors whether the state satisfies the planned action postconditions $a.\mathit{effects}$. Once satisfied, control switches back to the TAMP system, which replans.

Algorithm 1 TAMP-Gated Teleoperation

```
procedure TAMP-GATED-CONTROL($G$)
    while True do
        $s \leftarrow$ OBSERVE()                                ▷ Estimate or observe state
        if $s \in G$ then                                       ▷ State satisfies goal
            return True                                         ▷ Success!
        $\vec{a} \leftarrow$ PLAN-TAMP($s, G$)                  ▷ Solve for a plan $\vec{a}$
        for $a \in \vec{a}$ do                                  ▷ Iterate over actions
            if not IS-HUMAN-ACTION($a$) then
                EXECUTE-JOINT-COMMANDS($a$)
            else
                while OBSERVE() $\notin a$.eff do
                    EXECUTE-TELEOP()                            ▷ Teleoperation
                break                                           ▷ Re-observe and re-plan
```

I.1 Example Plan

Consider a plan found by the TAMP system for the **Tool Hang** task on the first planning invocation:

$$\vec{a}_1 = [\texttt{move}(\boldsymbol{q_0}, \tau_1, q_1), \texttt{pick}(\mathit{frame}, g^f, \boldsymbol{p}^f_{\boldsymbol{0}}, q_1), \texttt{move}(q_1, \tau_2, q_2), \underline{\texttt{attach}(\mathit{frame}, g^f, p_2, q_2, \widehat{p}^f_2, \widehat{q}_2, \mathit{stand})},$$
$$\texttt{move}(\widehat{q}_2, \widehat{\tau}_3, q_3), \texttt{pick}(\mathit{tool}, g^t, \boldsymbol{p}^t_{\boldsymbol{0}}, q_3), \texttt{move}(q_3, \tau_4, q_4), \texttt{attach}(\mathit{tool}, g^t, p_4, q_4, \widehat{p}^t_4, \widehat{q}_4, \mathit{frame})].$$

The values in bold represent constants present in the initial state; the non-bold values are parameter values selected by the planner. The learned preimages enable the TAMP system to plan not only a trajectory τ_1 to the first manipulation but also to the second manipulation τ_2. However, because the third trajectory $\widehat{\tau}_3$ depends on the resultant configuration $\widehat{q}_2$, planning for it is deferred. Upon successfully achieving $\texttt{Attached}(\mathit{frame}, \mathit{stand})$, replanning produces a new plan.

J Policy Training Details

In this section, we detail how we train policies via imitation learning from the human segments of HITL-TAMP datasets. Many choices are mirrored from Mandlekar *et al.* [1].

J.1 Observation Spaces

In our experiments, policies are either trained on low-dim state observations or image observations — this kind of flexibility is advantageous as it eases the burden of perception for deploying TAMP systems in the real world. Low-dim observations include ground-truth object poses, while image observations consist of RGB images from a front-view camera and a wrist-mounted camera. Both observations include proprioception (end-effector pose and gripper finger width). In simulation, the image resolution is 84x84, while in real world tasks, we use a resolution of 120x160 for Stack Three, Coffee, and Coffee Broad, and a resolution of 240x240 for Tool Hang. Our real-world agents are all image-based, since we do not assume that objects can be tracked. The real-world Tool Hang agent did not use the wrist-view in observations, since we found that it was completely occluded during the human portions of the task. The TAMP system only estimates poses at the start of each episode. We use a simple perception pipeline consisting of RANSAC plane estimation to segment the table from the point cloud, DBSCAN [55] to cluster objects, color-based statistics to associate objects, and Iterative Closest Point (ICP) to estimate object poses. For image-based agents, we apply pixel shift randomization (up to 10% of each image dimension) as a data augmentation technique (as in Mandlekar *et al.* [1]).

J.2 Training and Evaluation

We use BC-RNN with default hyperparameters from Mandlekar *et al.* [1] with the exception of an increased learning rate of 10^{-3} for policies trained on low-dim observations, to train policies from the human segments in each dataset. We follow the policy evaluation convention from Mandlekar *et al.* [1], and report the maximum Success Rate (SR) across all checkpoint evaluations over 3 seeds, which is evaluated over 50 rollouts. However, the TAMP system can fail during a rollout. To decouple TAMP failures from policy failures, we keep conducting rollouts for each checkpoint until 50 rollouts with no TAMP failures have been collected, and compute policy success rate over those rollouts (discussion in Appendix L). In the real world, we take the final policy checkpoint from training, and use it for evaluation.

K Low-Dim Policy Training Results

Task	**Time (min)**	**SR (im)**	**TAMP-gated SR (im)**
Square (C)	25.0	84.0 ± 0.0	91.3 ± 5.2
Square (HT)	$\mathbf{13.5}$	$\mathbf{100.0 \pm 0.0}$	$\mathbf{100.0 \pm 0.0}$
Square Broad (C)	48.0	29.3 ± 0.0	88.0 ± 1.6
Square Broad (HT)	$\mathbf{14.0}$	$\mathbf{100.0 \pm 0.0}$	$\mathbf{100.0 \pm 0.0}$
Three Piece Assembly (C)	60.0	55.3 ± 0.0	$\mathbf{96.0 \pm 2.8}$
Three Piece Assembly (HT)	$\mathbf{30.0}$	$\mathbf{100.0 \pm 0.0}$	$\mathbf{100.0 \pm 0.0}$
Tool Hang (C)	80.0	29.3 ± 0.0	60.0 ± 19.6
Tool Hang (HT)	$\mathbf{48.0}$	$\mathbf{80.7 \pm 1.9}$	$\mathbf{80.7 \pm 1.9}$

Table K.1: **Comparison to conventional teleoperation datasets (low-dim).** We trained normal and TAMP-gated policies using conventional teleoperation (C) and compared them to HITL-TAMP (HT). TAMP-gating makes policies trained on the data comparable to HITL-TAMP data, but data collection still involves significantly higher operator time.

In Table 6 and Sec. 6.2, we only presented results with image policies. In this section, we show that HITL-TAMP still compares favorably to conventional teleoperation data when trained on low-dim observations. The results are presented in Table K.1.

L TAMP Success Analysis

Task	Time (min)	SR (low-dim)	SR (image)	TAMP SR (low-dim)	Raw SR (low-dim)	TAMP SR (image)	Raw SR (image)
Square	13.5	100.0 ± 0.0	100.0 ± 0.0	77.7 ± 1.5	77.7 ± 1.5	82.0 ± 1.9	82.0 ± 1.9
Square Broad	14.0	100.0 ± 0.0	100.0 ± 0.0	81.2 ± 2.7	81.2 ± 2.7	76.1 ± 5.1	76.1 ± 5.1
Coffee	22.6	100.0 ± 0.0	100.0 ± 0.0	100.0 ± 0.0	100.0 ± 0.0	100.0 ± 0.0	100.0 ± 0.0
Coffee Broad	28.8	99.3 ± 0.9	96.7 ± 0.9	98.1 ± 1.6	97.4 ± 0.9	97.4 ± 0.9	94.2 ± 0.1
Tool Hang	48.0	80.7 ± 1.9	78.7 ± 0.9	97.4 ± 1.8	78.6 ± 2.9	97.4 ± 1.8	76.6 ± 1.2
Tool Hang Broad	51.5	49.3 ± 1.9	40.7 ± 0.9	88.8 ± 1.9	43.8 ± 0.8	93.8 ± 0.8	38.1 ± 1.1
Three Piece Assembly	30.0	100.0 ± 0.0	100.0 ± 0.0	96.2 ± 1.5	96.2 ± 1.5	95.0 ± 2.3	95.0 ± 2.3
Three Piece Assembly Broad	34.9	84.7 ± 4.1	82.0 ± 1.6	71.4 ± 0.0	60.5 ± 2.9	76.0 ± 4.0	62.3 ± 4.3
Coffee Preparation	78.4	96.0 ± 3.3	100.0 ± 0.0	80.9 ± 4.8	77.6 ± 4.4	83.8 ± 1.8	83.8 ± 1.8

Table L.1: **Analyzing TAMP Success Rates during Policy Evaluations.** A more complete set of results from Table 6 on HITL-TAMP datasets to demonstrate that policy evaluations do not have significant bias by only evaluating in regions where TAMP is successful. All TAMP success rates are high (above 70%) and most are above 88%.

Recall that when evaluating a trained policy, to decouple TAMP failures from policy failures, we keep conducting rollouts for each checkpoint until 50 rollouts with no TAMP failures have been collected, and compute policy success rate over those rollouts. In certain cases, this procedure could lead to biased evaluations — for example, if TAMP is only successful for an object in a limited region of the robot workspace. In this section, we present the TAMP success rates and raw success rates (including TAMP failures) for the policies in Table 6 (left), and demonstrate that it is unlikely that such bias exists in our evaluations. We present the results in Table L.1 — note that the Time and SR columns are reproduced from Table 6 (right) for ease of comparison. We see that all TAMP success rates are high (above 70%) and most are above 88%.

M Supplemental Video Overview

The supplemental video contains:

1. Real-World HITL-TAMP policies on Tool Hang, Coffee Broad, Stack Three, and Coffee.
2. HITL-TAMP dataset trajectories visualized across the 9 simulated tasks. The red border indicates human control (and the lack of one indicates TAMP control).
3. HITL-TAMP initial state distribution visualizations across the 9 simulated tasks.

References

[1] A. Mandlekar, D. Xu, J. Wong, S. Nasiriany, C. Wang, R. Kulkarni, L. Fei-Fei, S. Savarese, Y. Zhu, and R. Martín-Martín. What matters in learning from offline human demonstrations for robot manipulation. In *Conference on Robot Learning (CoRL)*, 2021.

[2] A. Brohan, N. Brown, J. Carbajal, Y. Chebotar, J. Dabis, C. Finn, K. Gopalakrishnan, K. Hausman, A. Herzog, J. Hsu, et al. Rt-1: Robotics transformer for real-world control at scale. *arXiv preprint arXiv:2212.06817*, 2022.

[3] H. Ravichandar, A. S. Polydoros, S. Chernova, and A. Billard. Recent advances in robot learning from demonstration. *Annual review of control, robotics, and autonomous systems*, 3: 297–330, 2020.

[4] M. A. Toussaint, K. R. Allen, K. A. Smith, and J. B. Tenenbaum. Differentiable physics and stable modes for tool-use and manipulation planning. 2018.

[5] C. R. Garrett, R. Chitnis, R. Holladay, B. Kim, T. Silver, L. P. Kaelbling, and T. Lozano-Pérez. Integrated task and motion planning. *Annual review of control, robotics, and autonomous systems*, 4:265–293, 2021.

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

[7] A. Brohan, N. Brown, J. Carbajal, Y. Chebotar, J. Dabis, C. Finn, K. Gopalakrishnan, K. Hausman, A. Herzog, J. Hsu, J. Ibarz, B. Ichter, A. Irpan, T. Jackson, S. Jesmonth, N. Joshi, R. Julian, D. Kalashnikov, Y. Kuang, I. Leal, K.-H. Lee, S. Levine, Y. Lu, U. Malla, D. Manjunath, I. Mordatch, O. Nachum, C. Parada, J. Peralta, E. Perez, K. Pertsch, J. Quiambao, K. Rao, M. Ryoo, G. Salazar, P. Sanketi, K. Sayed, J. Singh, S. Sontakke, A. Stone, C. Tan, H. Tran, V. Vanhoucke, S. Vega, Q. Vuong, F. Xia, T. Xiao, P. Xu, S. Xu, T. Yu, and B. Zitkovich. Rt-1: Robotics transformer for real-world control at scale. In *arXiv preprint arXiv:2212.06817*, 2022.

[8] E. Jang, A. Irpan, M. Khansari, D. Kappler, F. Ebert, C. Lynch, S. Levine, and C. Finn. Bc-z: Zero-shot task generalization with robotic imitation learning. In *Conference on Robot Learning*, pages 991–1002. PMLR, 2022.

[9] C. Lynch, M. Khansari, T. Xiao, V. Kumar, J. Tompson, S. Levine, and P. Sermanet. Learning latent plans from play. In *Conference on robot learning*, pages 1113–1132. PMLR, 2020.

[10] C. Lynch, A. Wahid, J. Tompson, T. Ding, J. Betker, R. Baruch, T. Armstrong, and P. Florence. Interactive language: Talking to robots in real time. *arXiv preprint arXiv:2210.06407*, 2022.

[11] A. Mandlekar, D. Xu, R. Martín-Martín, S. Savarese, and L. Fei-Fei. Learning to generalize across long-horizon tasks from human demonstrations. *arXiv preprint arXiv:2003.06085*, 2020.

[12] M. Ahn, A. Brohan, N. Brown, Y. Chebotar, O. Cortes, B. David, C. Finn, K. Gopalakrishnan, K. Hausman, A. Herzog, et al. Do as i can, not as i say: Grounding language in robotic affordances. *arXiv preprint arXiv:2204.01691*, 2022.

[13] R. Hoque, A. Balakrishna, C. Putterman, M. Luo, D. S. Brown, D. Seita, B. Thananjeyan, E. Novoseller, and K. Goldberg. Lazydagger: Reducing context switching in interactive imitation learning. In *2021 IEEE 17th International Conference on Automation Science and Engineering (CASE)*, pages 502–509. IEEE, 2021.

[14] R. Hoque, L. Y. Chen, S. Sharma, K. Dharmarajan, B. Thananjeyan, P. Abbeel, and K. Goldberg. Fleet-dagger: Interactive robot fleet learning with scalable human supervision. In *6th Annual Conference on Robot Learning*.

[15] J. Zhang and K. Cho. Query-efficient imitation learning for end-to-end autonomous driving. *arXiv preprint arXiv:1605.06450*, 2016.

[16] R. Hoque, A. Balakrishna, E. Novoseller, A. Wilcox, D. S. Brown, and K. Goldberg. Thriftydagger: Budget-aware novelty and risk gating for interactive imitation learning. *arXiv preprint arXiv:2109.08273*, 2021.

[17] S. Dass, K. Pertsch, H. Zhang, Y. Lee, J. J. Lim, and S. Nikolaidis. Pato: Policy assisted teleoperation for scalable robot data collection. *arXiv preprint arXiv:2212.04708*, 2022.

[18] T. Silver, K. Allen, J. Tenenbaum, and L. Kaelbling. Residual policy learning. *arXiv preprint arXiv:1812.06298*, 2018.

[19] T. Johannink, S. Bahl, A. Nair, J. Luo, A. Kumar, M. Loskyll, J. A. Ojea, E. Solowjow, and S. Levine. Residual reinforcement learning for robot control. In *2019 International Conference on Robotics and Automation (ICRA)*, pages 6023–6029. IEEE, 2019.

[20] A. Kurenkov, A. Mandlekar, R. Martin-Martin, S. Savarese, and A. Garg. Ac-teach: A bayesian actor-critic method for policy learning with an ensemble of suboptimal teachers. *arXiv preprint arXiv:1909.04121*, 2019.

[21] O. Mees, J. Borja-Diaz, and W. Burgard. Grounding language with visual affordances over unstructured data. *arXiv preprint arXiv:2210.01911*, 2022.

[22] E. Valassakis, N. Di Palo, and E. Johns. Coarse-to-fine for sim-to-real: Sub-millimetre precision across wide task spaces. In *2021 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 5989–5996. IEEE, 2021.

[23] L. P. Kaelbling and T. Lozano-Pérez. Hierarchical task and motion planning in the now. In *ICRA*, 2011.

[24] C. R. Garrett, T. Lozano-Pérez, and L. P. Kaelbling. Pddlstream: Integrating symbolic planners and blackbox samplers via optimistic adaptive planning. In *Proceedings of the International Conference on Automated Planning and Scheduling*, volume 30, pages 440–448, 2020.

[25] G. Konidaris, L. P. Kaelbling, and T. Lozano-Perez. From skills to symbols: Learning symbolic representations for abstract high-level planning. *Journal of Artificial Intelligence Research*, 61: 215–289, 2018.

[26] Z. Wang, C. R. Garrett, L. P. Kaelbling, and T. Lozano-Pérez. Learning compositional models of robot skills for task and motion planning. *The International Journal of Robotics Research*, 40(6-7):866–894, 2021.

[27] J. Liang, M. Sharma, A. LaGrassa, S. Vats, S. Saxena, and O. Kroemer. Search-based task planning with learned skill effect models for lifelong robotic manipulation. In *2022 International Conference on Robotics and Automation (ICRA)*, pages 6351–6357. IEEE, 2022.

[28] H. M. Pasula, L. S. Zettlemoyer, and L. P. Kaelbling. Learning symbolic models of stochastic domains. *Journal of Artificial Intelligence Research*, 29:309–352, 2007.

[29] T. Silver, R. Chitnis, J. Tenenbaum, L. P. Kaelbling, and T. Lozano-Pérez. Learning symbolic operators for task and motion planning. In *2021 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 3182–3189. IEEE, 2021.

[30] R. Chitnis, D. Hadfield-Menell, A. Gupta, S. Srivastava, E. Groshev, C. Lin, and P. Abbeel. Guided search for task and motion plans using learned heuristics. In *ICRA*. IEEE, 2016.

[31] B. Kim, L. Shimanuki, L. P. Kaelbling, and T. Lozano-Pérez. Representation, learning, and planning algorithms for geometric task and motion planning. *IJRR*, 41(2), 2022.

[32] S. Cheng and D. Xu. Guided skill learning and abstraction for long-horizon manipulation. *arXiv preprint arXiv:2210.12631*, 2022.

[33] T. Silver, A. Athalye, J. B. Tenenbaum, T. Lozano-Pérez, and L. P. Kaelbling. Learning neuro-symbolic skills for bilevel planning. In *6th Annual Conference on Robot Learning*.

[34] D. A. Pomerleau. Alvinn: An autonomous land vehicle in a neural network. In *Advances in neural information processing systems*, pages 305–313, 1989.

[35] M. Sundermeyer, A. Mousavian, R. Triebel, and D. Fox. Contact-graspnet: Efficient 6-dof grasp generation in cluttered scenes. In *2021 IEEE International Conference on Robotics and Automation (ICRA)*, pages 13438–13444. IEEE, 2021.

[36] A. Curtis, X. Fang, L. P. Kaelbling, T. Lozano-Pérez, and C. R. Garrett. Long-horizon manipulation of unknown objects via task and motion planning with estimated affordances. In *2022 International Conference on Robotics and Automation (ICRA)*, pages 1940–1946. IEEE, 2022.

[37] A. Mandlekar, Y. Zhu, A. Garg, J. Booher, M. Spero, A. Tung, J. Gao, J. Emmons, A. Gupta, E. Orbay, S. Savarese, and L. Fei-Fei. RoboTurk: A Crowdsourcing Platform for Robotic Skill Learning through Imitation. In *Conference on Robot Learning*, 2018.

[38] A. Mandlekar, J. Booher, M. Spero, A. Tung, A. Gupta, Y. Zhu, A. Garg, S. Savarese, and L. Fei-Fei. Scaling robot supervision to hundreds of hours with roboturk: Robotic manipulation dataset through human reasoning and dexterity. *arXiv preprint arXiv:1911.04052*, 2019.

[39] O. Khatib. A unified approach for motion and force control of robot manipulators: The operational space formulation. *IEEE Journal on Robotics and Automation*, 3(1):43–53, 1987.

[40] A. Mandlekar, F. Ramos, B. Boots, S. Savarese, L. Fei-Fei, A. Garg, and D. Fox. Iris: Implicit reinforcement without interaction at scale for learning control from offline robot manipulation data. In *IEEE International Conference on Robotics and Automation (ICRA)*, pages 4414–4420. IEEE, 2020.

[41] S. G. Hart and L. E. Staveland. Development of nasa-tlx (task load index): Results of empirical and theoretical research. In *Advances in psychology*, volume 52, pages 139–183. Elsevier, 1988.

[42] O. Mees, L. Hermann, E. Rosete-Beas, and W. Burgard. Calvin: A benchmark for language-conditioned policy learning for long-horizon robot manipulation tasks. *IEEE Robotics and Automation Letters*, 7(3):7327–7334, 2022.

[43] A. Mandlekar, D. Xu, R. Martín-Martín, Y. Zhu, L. Fei-Fei, and S. Savarese. Human-in-the-loop imitation learning using remote teleoperation. *arXiv preprint arXiv:2012.06733*, 2020.

[44] K. Van Wyk, M. Culleton, J. Falco, and K. Kelly. Comparative peg-in-hole testing of a force-based manipulation controlled robotic hand. *IEEE Transactions on Robotics*, 34(2):542–549, 2018.

[45] H. Park, J. Park, D.-H. Lee, J.-H. Park, and J.-H. Bae. Compliant peg-in-hole assembly using partial spiral force trajectory with tilted peg posture. *IEEE Robotics and Automation Letters*, 5(3):4447–4454, 2020.

[46] M. J. McDonald and D. Hadfield-Menell. Guided imitation of task and motion planning. In *Conference on Robot Learning*, pages 630–640. PMLR, 2022.

[47] M. Kelly, C. Sidrane, K. Driggs-Campbell, and M. J. Kochenderfer. Hg-dagger: Interactive imitation learning with human experts. In *2019 International Conference on Robotics and Automation (ICRA)*, pages 8077–8083. IEEE, 2019.

[48] J. S. Warm, R. Parasuraman, and G. Matthews. Vigilance requires hard mental work and is stressful. *Human factors*, 50(3):433–441, 2008.

[49] R. Tedrake, M. Fallon, S. Karumanchi, S. Kuindersma, M. Antone, T. Schneider, T. Howard, M. Walter, H. Dai, R. Deits, et al. A summary of team mit's approach to the virtual robotics challenge. In *2014 IEEE International Conference on Robotics and Automation (ICRA)*, pages 2087–2087. IEEE, 2014.

[50] R. Luo, C. Wang, E. Schwarm, C. Keil, E. Mendoza, P. Kaveti, S. Alt, H. Singh, T. Padir, and J. P. Whitney. Towards robot avatars: Systems and methods for teleinteraction at avatar xprize semi-finals. In *2022 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 7726–7733. IEEE, 2022.

[51] J. M. Marques, N. Patrick, Y. Zhu, N. Malhotra, and K. Hauser. Commodity telepresence with the avatrina nursebot in the ana avatar xprize semifinals. In *RSS 2022 Workshop on "Towards Robot Avatars: Perspectives on the ANA Avatar XPRIZE Competition*, 2022.

[52] H. Le, N. Jiang, A. Agarwal, M. Dudík, Y. Yue, and H. Daumé III. Hierarchical imitation and reinforcement learning. In *International conference on machine learning*, pages 2917–2926. PMLR, 2018.

[53] K. Shiarlis, M. Wulfmeier, S. Salter, S. Whiteson, and I. Posner. Taco: Learning task decomposition via temporal alignment for control. In *International Conference on Machine Learning*, pages 4654–4663. PMLR, 2018.

[54] Y. Zhu, J. Wong, A. Mandlekar, and R. Martín-Martín. robosuite: A modular simulation framework and benchmark for robot learning. In *arXiv preprint arXiv:2009.12293*, 2020.

[55] M. Ester, H.-P. Kriegel, J. Sander, and X. Xu. A density-based algorithm for discovering clusters in large spatial databases with noise. In *KDD*, 1996.