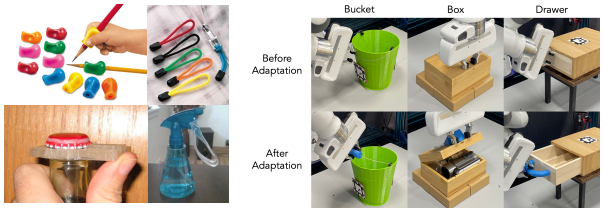
Learning to Design 3D Printable Adaptations on Everyday Objects for Robot Manipulation

Michelle Guo[1], Ziang Liu[1], Stephen Tian[1], Zhaoming Xie[1], Jiajun Wu[1], and C. Karen Liu[1]

***Abstract*— Advancements in robot learning for object manipulation have shown promising results, yet certain everyday objects remain challenging for robots to effectively interact with. This discrepancy arises from the fact that human-designed objects are optimized for human use rather than robot manipulation. To address this gap, we propose a framework to automatically design 3D printable adaptations that can be attached to hard-to-use objects, thus improving "robot ergonomics". Our learning-based framework formulates the adaptation design and control as a dual Markov decision process and is able to improve robot-object interactions for various robot end effectors and objects. We further validate our designs in the real world with a Franka Panda robot. Please see the supplementary video and `https://object-adaptation.github.io` for additional visualizations.**



(a) Human adaptations. (b) Robot adaptations.

Fig. 1: Inspired by human examples of adaptation (a), we focus on adapting objects for robots to use in an *automated* way via learning (b).

I. Introduction

Recent advancements in robot learning algorithms for object manipulation have yielded promising results [1]. Yet, many objects remain challenging for robots to effectively manipulate or utilize [2], including everyday objects commonly found in household environments—an important application of robot learning [3]. To circumvent this problem, researchers may selectively focus on robot-friendly objects, or instrument them in ways that facilitate successful interactions, such as by attaching wrappers around thin handles or making object states more accessible (e.g., initializing a laptop lid slightly open for easier laptop opening).

Why do robots struggle with the manipulation of everyday objects? One compelling explanation is that the design and creation of everyday objects have historically been driven by human needs and intended for human use, not robots. Therefore we hypothesize that the primary obstacle lies in the grasp affordances of these objects, which are designed for human hands rather than the robot end-effectors commonly employed today. To enable robots to interact seamlessly with a wider range of objects, it is crucial to address the disparity between human-designed objects and the current morphologies and control capabilities of robots.

Promising avenues for bridging this gap include re-designing either the robot end-effector or the object itself; however, these approaches suffer from practical limitations. Recent methods for end-effector design [4], [5] require frequent switching between specialized "hands" tailored for different objects, compromising long-horizon task efficiency. On the other hand, re-designing the object [6], [7] entails inventing a second set of objects optimized for robotic use, creating geometries and kinematics distinct from their human counterparts but serving similar functions. This approach is resource-intensive due to the large design space required to flexibly explore and discover the optimal forms that support the desired functions. Moreover, the notion of maintaining separate sets of objects for robots and humans contradicts the goals of human-robot interaction, where robots must still handle the diverse array of human objects encountered in tasks like hand-over interactions [8].

Drawing inspiration from human-computer interaction [9], [10], [11], [12] and human ergonomics [13], [14], we examine the problem through a different lens, by viewing this as a "robot ergonomics" problem [15], [16], [17]. Rather than resorting to replacing robot end-effectors or entire objects, we argue for a simpler and more practical solution: designing 3D printable *adaptations* that can be attached to hard-to-use objects. For example, humans often adapt objects or tools when the original design is inadequate for certain users or use cases (e.g., attaching a larger zipper pull or affixing a rubber grip to pencils for toddlers to write; see Figure 1a) [18], [19], [20]. Similarly, adaptations can enhance robot ergonomics while leaving the rest of the object unchanged, allowing robots to use human objects more effectively (Figure 1b). This approach has the potential to expand the set of human objects that robots can use, and ultimately help bridge the design gap between robots and humans. However, current approaches for designing adaptations [10], [17] require significant human intervention to define and customize the designs, which is cumbersome and difficult to scale.

To tackle this issue, we adopt a learning-based approach to automatically design object adaptations for robot morphologies, while leveraging just task progress as the only learning signal. Because the adaptation design and the way it is used are interdependent, we formulate the design and control as a dual Markov decision process (MDP) and adopt a reinforcement learning framework to optimize them jointly. Our design adaptation allows for a simple and general reward

[1]Stanford University

function agnostic to the choice of end-effector or task. We validate our framework by learning design and control in simulation with a Franka Panda robot and a Barrett three-finger hand, using objects that are hard to manipulate without object adaptation, such as a bucket handle or a box.

In summary, the contributions of our work include:

- Propose a new problem setting, object adaptation, for robotic manipulation,
- Develop three design parameterizations and six simulated robotic manipulation environments for object adaptation,
- Demonstration of the effectiveness of object adaptation, with analyses across different design spaces and approaches, and
- Evaluation on real-world hardware (Franka Panda robot) with physically fabricated adaptations.

II. Related Work

End-effector and tool design. Standard parallel-jaw grippers cannot stably grasp objects with arbitrary geometries. Several works attempt to overcome this issue by developing automated methods for designing robot morphologies specialized to different tasks or objects [4], [5], [21]. However, deploying these methods in practice requires end-effectors to be specially instrumented before manipulating each object of interest, which is time-consuming.

Another approach is designing or constructing additional *tools* to provide agents the necessary affordances [6], [7], [22]. However, existing tool design works do not address how robots should grasp tools, and assume that a tool is already grasped in the hand when execution begins. Conversely, in this work we consider how adaptations should be designed and optimized to enable objects to be easily grasped by the agent's original end-effector.

Extrinsic dexterity, or the use of external forces or contacts to enable manipulation, is another avenue towards exposing grasp affordances [23], [24]. While extrinsic dexterity can be effective when the external environment enables appropriate behaviors, our approach directly adapts objects to be easier to use in any environment, and can also be used in combination with extrinsic dexterity methods.

Object adaptations. Prior works have studied the effect of tool or object design on comfort and effectiveness in the context of human ergonomics [13], [14]. This motivates us to apply the same concept to robots, known in the literature as "robot ergonomics" [25], [15], [16], [17], but we take a learning approach to design that requires minimal human intervention as opposed to previously studied manual designs. Shao et al. [26] apply deep reinforcement learning to learn scaffolding for learning robotic manipulation, but only allow the pose of the fixture to be learned, while we parameterize the geometry. Furthermore, in our work the designs are used directly to improve task feasibility rather than as aids of learning progress.

In this work, we use 3D printing to rapidly fabricate designed adaptations. A variety of methods have been developed to extend, modify, or adapt existing objects via 3D printing [9], [10], [11], [12], but pursue the orthogonal direction of improving the printing and fabrication process itself, while we focus on learning the designs themselves.

III. Learning to Design Object Adaptations

A. Problem definition

The objective of our study is to create object adaptations that can expand the set of objects that an agent can interact with in the world. To accommodate for various design and control spaces, we represent the agent's environment as a dual Markov decision process (MDP), consisting of a design MDP and a control MDP, visualized in Figure 2. During the design MDP, a design action vector applies a transformation to the default object adaptation. The transformed adaptation is then attached to the object, and the end effectors are commanded by the control policy during the control MDP to manipulate the objects. Given this dual MDP, we can optimize the design action and control policy jointly using reinforcement learning algorithms.

Design MDP. Each episode begins in the design MDP. The state space $\mathcal{S}^D$ for the design MDP specifies the shape of the adapted object and is initialized with default shape parameters. While the design MDP can contain multiple design steps, in practice we employ a single environment transition. Thus, we only require learning a design action vector $\boldsymbol{a}^D \in \mathcal{A}^D$ (instead of a design policy) that specifies the design parameters that will be used to transform the shape parameter. During the design MDP, we assign zero reward; the usefulness of a design is determined solely by task progress during the control MDP. This means that any signal for the utility of an adaptation depends only on how well the robot can use it to solve the task.

Control MDP. We enter the control MDP after the end of the design MDP. The control action $\boldsymbol{a}^C \in \mathcal{A}^C$ represents the control command to the robot's end-effector. The control policy $\pi_C(\boldsymbol{a}^C|\boldsymbol{s}^C)$ will generate the control action based on $\boldsymbol{s}^C \in \mathcal{O}_{robot} \times \mathcal{O}_{object} \times \mathcal{S}^D$, where $\mathcal{O}_{robot}$ is the state of the robot including end-effector position and orientation $\boldsymbol{q}_{\text{end}} = [\boldsymbol{p}_{\text{end}}, \boldsymbol{\theta}_{\text{end}}]$, gripper state $\boldsymbol{s}_{\text{gripper}} \in \{\text{close}, \text{open}\}$, and binary flags $\boldsymbol{s}_{\text{finger}} \in \{0,1\}^{N_{\text{finger}}}$ indicating which of the N_{finger} fingers are in contact. $\mathcal{O}_{object}$ is the full state $\boldsymbol{q}_{\text{object}}$ of the object to be manipulated, including its position and orientation, as well as the joint states if articulated joints are present. The design parameter $\boldsymbol{s}^D \in S^D$ from the design MDP is also used to allow the control policy to adapt its strategy to different design parameters. The control policy also receives a task reward r_t after each control time step t. The control MDP ends when the task is completed or when the time limit of an episode is reached.

B. Instantiating our framework

Design action. Given an articulated object we wish to manipulate, we first identify the link with which the robot's end-effector needs to interact (e.g., the handle on a water bucket or the lid of a box). Then, the adaptation shape is generated based on the design action and is attached on the link. We explore three design parameterizations. Primitives

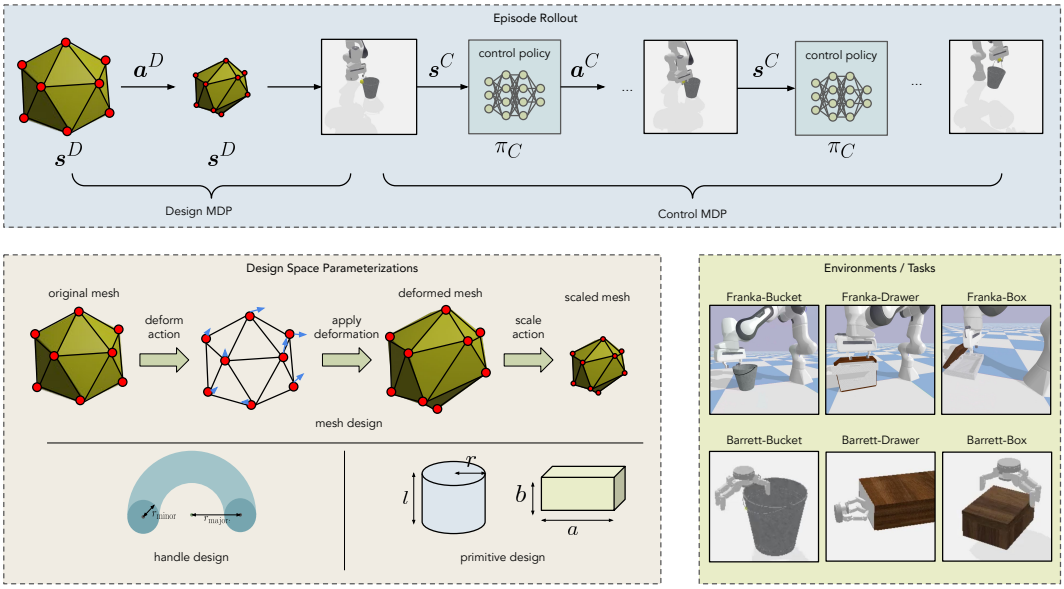


Fig. 2: Overview of our framework. We represent the agent's environment as a dual Markov decision process (MDP) (top box), consisting of a design MDP (top box, left) and control MDP (top box, right). During the design MDP, a design action vector $\boldsymbol{a}^D \in \mathcal{A}^D$ applies a transformation to the default object adaptation. The transformed adaptation is then attached to the object, and the end effectors are commanded by the control policy during the control MDP to manipulate the objects. Given this dual MDP, we optimize the design action and control policy jointly using reinforcement learning algorithms. We also show different design parameterizations (bottom left box) and different environments and tasks that we evaluate our framework on (bottom right box).

and handles are biased towards designs frequently seen in the real world, and meshes allow flexibility for the control policy to explore more broadly. Fig. 2 illustrates some examples of the effect of the design action. We describe them in detail below:

- *Primitives*: We use cylinder and cuboid (with a square base) as primitive shapes for the adaptation. The design action is 3-dimensional, where the first dimension chooses the appropriate primitive and the rest modify the parameters of the primitives, i.e., the length and width a, b of the cuboid and the radius and length r, l of the cylinder.
- *Handles*: To generate shapes more commonly seen in the human environment, we initialize the adapted object to be handle-like with the shape of a half-torus. Specifically, given a point in the object coordinate $\boldsymbol{p} = [x, y]$ on the surface we want to attach the adapted object, we define a radius r_{major}, and attached two circles of radius r_{minor} at location $[x - r_{\text{major}}, y]$ and $[x + r_{\text{major}}, y]$ (Figure 2). The design action is 2-dimensional and contains r_{major} and r_{minor}. A half-torus is then generated using these two parameters and attached to the object at the two endpoints of the half-torus.
- *Meshes*: We initialize the adaptation as a level-zero icosphere, with 12 vertices $\boldsymbol{V} \in \mathbb{R}^{12\times 3}$. The dimensionality of the design action is 37, where the first 36 dimensions specify the amount of deformation $\Delta\boldsymbol{V}$ desired, and the last dimension applies scaling λ to the mesh to adjust its size. The final mesh vertices $\boldsymbol{V}'$ are computed as $\boldsymbol{V}' = \lambda(\boldsymbol{V} + \Delta\boldsymbol{V})$. To avoid self-intersections and highly non-convex shapes, we compute the convex hull of the deformed mesh to be the final mesh.

Control action. The action space during the control MDP consists of a delta (change in) pose for the end-effector of the robot as well as the gripper state. Specifically, given the current pose of the robot end-effector $\boldsymbol{q}_{\text{end}} = [\boldsymbol{p}_{\text{end}}, \boldsymbol{\theta}_{\text{end}}]$ that specifies the current position and orientation of the robot end-effector in the world frame, the control action $\boldsymbol{a}^C = [\delta\boldsymbol{p}_{\text{end}}, \delta\boldsymbol{\theta}_{\text{end}}, a_{\text{gripper}}]$, where $[\delta\boldsymbol{p}_{\text{end}}, \delta\boldsymbol{\theta}_{\text{end}}]$ predicts a desired displacement of the end-effector pose. The desired next end-effector pose is then computed via $\mathbf{q}^d_{\text{end}} = [\boldsymbol{p}_{\text{end}} + \delta\boldsymbol{p}_{\text{end}}, \boldsymbol{\theta}_{\text{end}} + \delta\boldsymbol{\theta}_{\text{end}}]$. A proportional-derivative (PD) controller then drives the end-effector to the desired pose. $a_{\text{gripper}} \in \{\texttt{close}, \texttt{open}\}$ predicts the binary gripper state to determine whether to open or close the gripper and another PD controller drives the gripper to the desired open or close state.

Reward design. During the control MDP, we use a dense reward function (following [2]) to guide the policy to accomplish the desired task by manipulating the adapted object. This reward function is task and end-effector agnostic, and requires knowing only the final goal state of the object. We decompose the task into two phases. In the first phase (Approaching Phase), the agent is rewarded for moving the robot end-effector toward the adaptation or the object part to be manipulated (e.g., bucket handle, box lid). In the second phase (Manipulation Phase), the agent is rewarded for moving the adapted object toward a target state.

Throughout the whole episode, the following reward is used to encourage the end-effector to stay close to the link to be manipulated:

$$r_{\text{obj}} = -\min_{\boldsymbol{p} \in \mathcal{P}_{\text{link}}} (\|\boldsymbol{p}_{\text{end}} - \boldsymbol{p}\|),$$

where we penalize the minimum distance of the end-effector to all the points $\mathcal{P}_{\text{link}}$ on the link we want to manipulate. During the Approaching Phase, this rewards the agent to move the robot end-effector toward the link, and during the Manipulation Phase, this penalizes the agent for moving the end-effector away from the link, e.g., dropping the object while lifting it up.

We also use a "holding" detector to determine whether the gripper is in contact with the link. In particular, the detector returns true when

$$\sum_{\text{finger}} \texttt{IsContact}(\text{finger}, \text{link}) = N_{\text{finger}},$$

where `IsContact` is a boolean function that returns whether a given finger of the end-effector is in contact with the desired link and N_{finger} is the total number of fingers. This encourages the gripper to close around the link once the end-effector gets close enough. A reward of $r_{\text{holding}} = 10$ is rewarded the first time holding is detected, and the MDP transitions from the Approaching Phase to the Manipulation Phase.

During the Manipulation Phase, in addition to r_{obj}, we also supply a progress-based reward

$$r_{\text{target}} = 100(\|\boldsymbol{s}_{\text{obj}} - \boldsymbol{s}_{\text{target}}\| - \|\boldsymbol{s}_{\text{obj}}^{\text{prev}} - \boldsymbol{s}_{\text{target}}\|),$$

where $\boldsymbol{s}_{\text{target}}$ is the target state of the object, and $\boldsymbol{s}_{\text{obj}}$ and $\boldsymbol{s}_{\text{obj}}^{\text{prev}}$ is the object state at the current and previous time step. This encourages the agent to move the object toward the target state.

When the object state is within a threshold distance from the target state, the task is considered finished and the episode ends with a completion bonus of $r_{\text{success}} = 100$. Otherwise, the episode runs until a time limit of $t_{\text{limit}} = 5$ seconds is reached and the task is considered failed.

Reinforcement learning. At each training iteration we collect interaction trajectories that span the design and control MDP. The design action vector and the control policy are then optimized using Proximal Policy Optimization (PPO) [27].

IV. Experiments

In this section, we present our experimental evaluation of the proposed framework in both simulation and the real world. We aim to address the following key questions: i) Does object adaptation significantly improve task performance? ii) Which design parameterizations play a larger role in the effectiveness of adaptations for each task? iii) Do object adaptations translate to real-world environments?

A. Environments

We evaluate our framework in six manipulation environments, visualized in Figure 2, that cover three tasks (Bucket, Drawer, Box) and two end-effectors (a 7-DoF Franka Panda robot arm equipped with its standard parallel-jaw gripper (PJG), and a three-finger Barrett hand). The tasks feature articulated objects commonly present in household environments, but challenging for non-dexterous end-effectors to manipulate. We implement the environments using the PyBullet simulation engine [28]. We describe the tasks below:

- **Bucket.** The robot must lift a bucket from the ground. The challenge lies in the thin bucket handle, initially flush against the side of the bucket. The robot must exert an upward force on the handle, creating space between the bucket and the handle, and then lift the handle, possibly by establishing force closure with the handle or caging the handle by hooking its fingers around it. The task progress is measured by the distance between the bucket handle and the desired goal height, and is considered solved when the bucket handle is lifted to a sufficient height.
- **Drawer.** The robot must pull open a drawer (without an explicit handle) from a closed configuration. Task progress is quantified by the extent to which the drawer is open, and the task is considered successfully completed when the drawer is open to a certain extent.
- **Box.** The robot's objective is to open a closed box. Opening the box poses a challenge for the robot gripper since it lacks an explicit handle or knob. Humans may solve this task by pushing the box lid upwards with their fingers, which is a strategy unavailable to PJGs and difficult to discover with RL for the Barrett hand. Task progress is indicated by the change in the lid angle, and the task is considered successfully completed when the lid is open to 90 degrees.

Assets. We use models of articulated objects from the PartNet-Mobility dataset [29]. To ensure realistic object sizes relative to the robot, we manually annotate the object scales. The initial state distribution is designed to prevent self-penetration or collision between the object, the attached adaptation design, and the robot. To obtain the collision geometry, we employ Volumetric Hierarchical Approximate Decomposition (V-HACD) [30]. We attach the adaptation design to the object by creating a new joint in the object's Unified Robotics Description Format (URDF) file, connecting the parent link (e.g., bucket handle, box lid) to the child link representing the adaptation design.

B. Task performance evaluation with and without adaptations

To assess the impact of object adaptations on task performance, we perform experiments with varying combinations of method, design parameters (primitive, handle, mesh), end-effector (Franka or Barrett), and task. Methods include our proposed system ("Ours"), a "Random" baseline method,

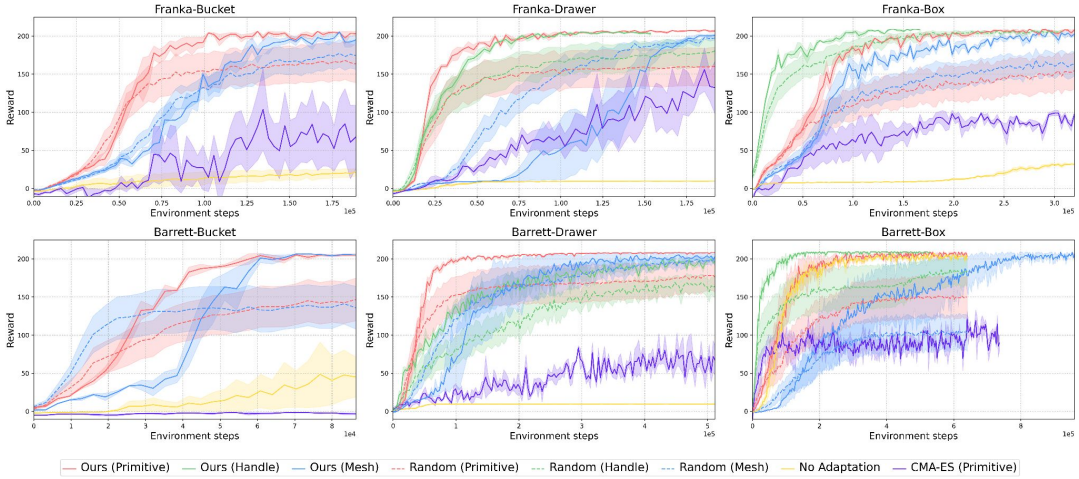


Fig. 3: Learning curves for varying design strategies. Each plot corresponds to one of the six environments that we evaluate our framework on. We show results for our method ("Ours") vs. random designs ("Random"), across different design representations ("Primitive", "Handle", and "Mesh"). We additionally compare against no adaptation baselines, as well as a CMA-ES baseline that directly optimizes the design and control actions. Shaded regions indicate standard error across a minimum of three random seeds.

where a design is sampled uniformly from the design space and is fixed throughout training, and a "No Adaptation" baseline method where the control policies learn to manipulate the original object without adaptation. We also compare against CMA-ES, which optimizes the design and control actions directly for the reward function.

Figure 3 presents the training curves, showcasing the improvements achieved with adaptations compared to the no-adaptation baseline. The learned adaptation designs for different end-effectors and design representations are shown in Figure 5. For all of the tasks and all design parameterizations, optimized designs outperform random designs, except for Barrett-Drawer with Mesh. This indicates our dual MDP is able to efficiently find design parameters that can improve task performance. Compared to the CMA-ES baseline, our method achieves better converged performance and sample efficiency.

We also note that different design parameterizations can be more effective depending on the task. For example, the primitive parameterization performs better for the Drawer task, while the handle parameterization performs better for the Box task. In all cases, the mesh parameterization performs worst. This is expected as it has more parameters to optimize and more easily succumbs to local minima. However, it is the most flexible parameterization, and can in principle produce a superset of primitive or handle shapes given the right optimization techniques. Further exploration of how to better optimize mesh parameters is an exciting direction for future work. It is also worth noting that the Barrett hand can occasionally learn to complete the tasks without adaptation. This is because the Barrett hand is closer to the human hand in terms of morphology and capability compared to the parallel gripper. However, the Barrett hand can still use adaptation in order to learn faster.

Finally, we analyze the effect of physical parameters on the performance. We evaluate the setting where the friction coefficient between adaptation and the end-effector is reduced, and show the results in Figure 4. We find that, with the task being more difficult, the gap between "Ours" and "Random Design" is much larger than the gaps in Figure 3. We also evaluate a two-stage optimization procedure, where design and control are optimized separately. "Control UP" corresponds to training a universal policy (UP) for control with PPO, where every episode is initialized with a random design. This tries to learn a control policy that can use *any* design. "Control UP + CMA-ES" uses CMA-ES to optimize the design parameters. The reward is obtained by rolling out a pre-trained control UP on each candidate design. While separate optimization of design and control also solves the task, our method achieves higher sample efficiency.

C. Real world evaluation

To validate the effectiveness of the designed adaptations in real-world scenarios, we roll out the control policies in simulation for all the tasks and optimized designs, and execute the joint trajectories of the Franka Panda robot arm and gripper in the real world. We track objects with the AprilTag fiducial system [31], using two RealSense D415 RGB-D cameras in order to handle potential self- and robot-arm-occlusions of the object. The adaptation links are 3D printed using Creality's Ender 3 and 5 fused deposition modeling (FDM) 3D printers, with Polylactic Acid (PLA) filament, and physically attached to the objects. Real-world rollouts are shown in Figure 6, and the success rates are reported in Table I. Videos of the real-world rollouts can be found in the supplementary video. Without adaptations, the robot fails to complete the tasks. Meanwhile, all the designs optimized from the primitive and handle parameterization can be used by the physical Franka Panda robot. However,

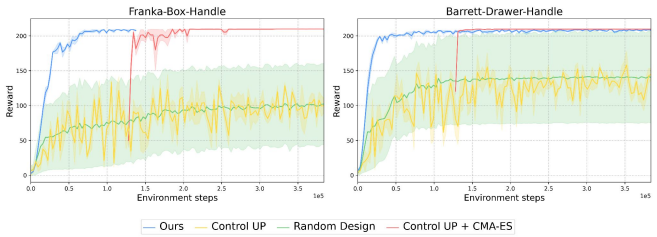


Fig. 4: Learning curves for the low-friction setting. The gap between "Ours" and "Random Design" is much larger than the gaps in Figure 3. "Control UP" corresponds to training a universal policy (UP) for control with PPO, where every episode is initialized with a random design. This tries to learn a control policy that can use *any* design. "Control UP + CMA-ES" uses CMA-ES to optimize the design parameters. The reward is obtained by rolling out a pre-trained control UP on each candidate design. Shaded regions indicate standard error across a minimum of three random seeds.

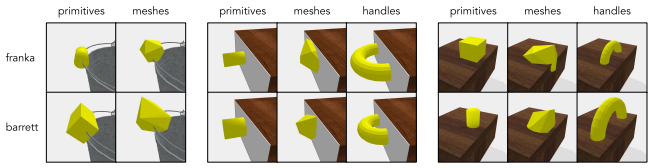


Fig. 5: Visualization of optimized designs. Each image shows an optimized design (yellow shape) that is generated by our method. The left panel corresponds to the Bucket task, the middle panel corresponds to the Drawer task, and the right panel corresponds to the Box task. Rows represent different hands (Franka and Barrett), and columns represent different design parameterizations (primitives, handles, meshes).

the mesh design only works for the bucket and box but not for the drawer task. The shapes of the meshes optimized for the drawer are irregular and the control policies utilize unrealistic frictions to accomplish the tasks in simulation, while in the real world, it can only have partial success by pulling out roughly 30% of the desired drawer length.

V. Conclusion

In this paper, we propose a framework to enable robots to effectively manipulate everyday objects designed for human use. By approaching the problem from a "robot ergonomics" perspective, we propose a learning-based approach for automatically designing object adaptations that better fit robot affordances. Our simulated studies and real-world experiments demonstrate the effectiveness of our approach in the context of robot manipulation. The generated adaptations outperform baseline naive designs, showcasing the potential of our method to bridge the gap between human-designed objects and robot capabilities. Moreover, our analysis of design parameters highlights the importance of design dimensions in shaping the effectiveness of the adaptations. Our findings pave the way for further advancements in robot ergonomics and the integration of robots into human-centric environments, fostering improved human-robot collaboration and expanding the range of objects in the world that robots can effectively interact with.

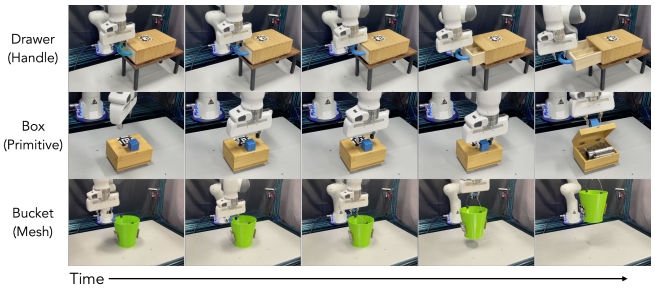


Fig. 6: Real world rollouts. 3D-printed adaptation parts, outlined in the left-most frame in each row, allow the robot to successfully complete each of the three tasks.

Design	Bucket	Drawer	Box
No Adaptation	0/3	0/3	0/3
Primitive	3/3	3/3	3/3
Handle	-	3/3	3/3
Mesh	3/3	0/3	3/3

TABLE I: Real world success rates on each task when deploying object adaptations in the real world and performing control with a Franka Panda robot.

Acknowledgments

This work was in part supported by the Stanford Institute for Human-Centered AI (HAI), the Center for Integrated Facility Engineering (CIFE), NSF RI #2211258 and #2338203, and ONR MURI N00014-22-1-2740.

Appendix

VI. Implementation Details

See Table II for a summary of hyperparameters used for PPO training. We use an MLP to model the policy.

Parameter	Setting
Max. Episode Steps	40
Pol. Hidden	[1024, 1024]
Val. Hidden	[128, 128]
Std	0.08208
Gamma	0.99
Lambda	0.95
Policy Learning Rate	1e-5
Value Learning Rate	1e-4
Policy Clip	0.2
Batch Size	160

TABLE II: Hyperparameters used for PPO training.

REFERENCES

[1] O. Kroemer, S. Niekum, and G. Konidaris, "A review of robot learning for manipulation: Challenges, representations, and algorithms," *The Journal of Machine Learning Research*, 2021.

[2] C. Bao, H. Xu, Y. Qin, and X. Wang, "Dexart: Benchmarking generalizable dexterous manipulation with articulated objects," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2023.

[3] C. Li, R. Zhang, J. Wong, C. Gokmen, S. Srivastava, R. Martín-Martín, C. Wang, G. Levine, M. Lingelbach, J. Sun, *et al.*, "Behavior-1K: A benchmark for embodied ai with 1,000 everyday activities and realistic simulation," in *Conference on Robot Learning (CoRL)*. PMLR, 2023.

[4] H. Ha, S. Agrawal, and S. Song, "Fit2form: 3D generative model for robot gripper form design," in *Conference on Robot Learning (CoRL)*. PMLR, 2021.

[5] M. Kodnongbua, I. Good, Y. Lou, J. Lipton, and A. Schulz, "Computational design of passive grippers," *ACM Transactions on Graphics (TOG)*, 2022.

[6] Y. Wu, S. Kasewa, O. Groth, S. Salter, L. Sun, O. P. Jones, and I. Posner, "Imagine that! Leveraging emergent affordances for 3D tool synthesis," *arXiv preprint arXiv:1909.13561*, 2019.

[7] M. Li, R. Antonova, D. Sadigh, and J. Bohg, "Learning tool morphology for contact-rich manipulation tasks with differentiable simulation," in *IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2023.

[8] C. Wang, C. Pérez-D'Arpino, D. Xu, L. Fei-Fei, K. Liu, and S. Savarese, "Co-gail: Learning diverse strategies for human-robot collaboration," in *Conference on Robot Learning (CoRL)*. PMLR, 2022.

[9] X. Chen, S. Coros, J. Mankoff, and S. E. Hudson, "Encore: 3D printed augmentation of everyday objects with printed-over, affixed and interlocked attachments," in *Proceedings of the 28th Annual ACM Symposium on User Interface Software & Technology (UIST)*, 2015.

[10] X. Chen, J. Kim, J. Mankoff, T. Grossman, S. Coros, and S. E. Hudson, "Reprise: A design tool for specifying, generating, and customizing 3D printable adaptations on everyday objects," in *Proceedings of the 29th Annual ACM Symposium on User Interface Software & Technology (UIST)*, 2016.

[11] Y. Koyama, S. Sueda, E. Steinhardt, T. Igarashi, A. Shamir, and W. Matusik, "Autoconnect: computational design of 3D-printable connectors," *ACM Transactions on Graphics (TOG)*, 2015.

[12] A. Teibrich, S. Mueller, F. Guimbretière, R. Kovacs, S. Neubert, and P. Baudisch, "Patching physical objects," in *Proceedings of the 28th Annual ACM Symposium on User Interface Software & Technology (UIST)*, 2015.

[13] G. Fellows and A. Freivalds, "Ergonomics evaluation of a foam rubber grip for tool handles," *Applied Ergonomics*, 1991.

[14] I. Halim, R. Z. R. Umar, M. S. S. Mohamed, N. Ahmad, V. Padmanathan, and A. Saptari, "The influence of hand tool design on hand grip strength: A review," *International Journal of Integrated Engineering*, 2019.

[15] N. Melenbrink, C. Teeple, and J. Werfel, "A robot factors approach to designing modular hardware," in *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2022.

[16] R. Sosa, M. Montiel, E. B. Sandoval, R. E. Mohan, *et al.*, "Robot ergonomics: Towards human-centred and robot-inclusive design," in *DS 92: Proceedings of the DESIGN 15th International Design Conference*, 2018.

[17] Z. Xu and M. Cakmak, "Robot factors: an alternative approach for closing the gap in human versus robot manipulation," in *Workshop on Human versus Robot Grasping and Manipulation–How Can We Close the Gap*, 2014.

[18] J. R. Plaxen, "Adapt my world: Homemade adaptations for people with disabilities," *Scoliosis*, 2023.

[19] S. Robitaille, *The Illustrated Guide to Assistive Technology and Devices: Tools and Gadgets for Living Independently: Easyread Super Large 18pt Edition*. ReadHowYouWant. com, 2010.

[20] T. Willkolmm, "Assistive technology solutions in minutes ii: Ordinary items, extraordinary solutions," 2013.

[21] Y. Yuan, Y. Song, Z. Luo, W. Sun, and K. Kitani, "Transform2act: Learning a transform-and-control policy for efficient agent design," in *International Conference on Learning Representations*, 2022.

[22] Z. Liu, S. Tian, M. Guo, C. Liu, and J. Wu, "Learning to design and use tools for robotic manipulation," in *Conference on Robot Learning (CoRL)*. PMLR, 2023.

[23] N. Chavan-Dafle, A. Rodriguez, R. Paolini, B. Tang, S. Srinivasa, M. Erdmann, M. T. Mason, I. Lundberg, H. Staab, and T. Fuhlbrigge, "Extrinsic dexterity: In-hand manipulation with external forces," in *IEEE International Conference on Robotics and Automation (ICRA)*, May 2014.

[24] W. Zhou and D. Held, "Learning to grasp the ungraspable with emergent extrinsic dexterity," in *Conference on Robot Learning (CoRL)*. PMLR, 2023.

[25] J. Li, A. Samoylov, J. Kim, and X. Chen, "Roman: Making everyday objects robotically manipulable with 3D-Printable add-on mechanisms," in *Proceedings of the CHI Conference on Human Factors in Computing Systems*, 2022.

[26] L. Shao, T. Migimatsu, and J. Bohg, "Learning to scaffold the development of robotic manipulation skills," in *IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2020.

[27] J. Schulman, F. Wolski, P. Dhariwal, A. Radford, and O. Klimov, "Proximal policy optimization algorithms," *arXiv preprint arXiv:1707.06347*, 2017.

[28] E. Coumans and Y. Bai, "Pybullet, a python module for physics simulation for games, robotics and machine learning," 2016.

[29] F. Xiang, Y. Qin, K. Mo, Y. Xia, H. Zhu, F. Liu, M. Liu, H. Jiang, Y. Yuan, H. Wang, *et al.*, "Sapien: A simulated part-based interactive environment," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2020.

[30] K. Mamou, E. Lengyel, and A. Peters, "Volumetric hierarchical approximate convex decomposition," in *Game Engine Gems 3*. AK Peters, 2016.

[31] E. Olson, "AprilTag: A robust and flexible visual fiducial system," in *IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2011.