Interpretable Human-in-the-Loop In-Context Preference Learning Via Preference Boundaries

Valerie K. Chen
Massachusetts Institute of Technology
vkchen@csail.mit.edu

Julie A. Shah
Massachusetts Institute of Technology

Andreea Bobu
Massachusetts Institute of Technology

***Abstract*—While imitation learning-based methods have gained popularity for enabling robots to perform complex tasks, they lack flexibility for adapting to preferences and general interpretability. Drawing insights from feature-based reward learning literature, which emphasizes alignment to human intent, we propose a novel abstraction called a "preference boundary" to reusably represent preferences. We also propose a method for validating alignment with the user's preferences and provide preliminary results evaluating these methods. We conclude with discussions of insights, next steps, and limitations.**

I. Introduction

Robot learning has been gaining traction, in part for the abilities to handle complex tasks and adapt to novel scenarios [13]. Following the successes of reinforcement learning from human feedback [17] in training large models in language and vision [24, 25], attention has turned to creating generalist robot policies. In particular, imitation learning [38, 18, 9, 7]—including human-in-the-loop formulations [31, 21, 12, 23]—has recently produced exciting results for robotics challenges.

However, imitation-learning-based methods such as behavior cloning lack interpretability and align with a human's intent only when that intent is clearly present in the training data. What if a person has a different preference, such as ordering books on their bookshelf by color rather than author name? One option is to finetune the policy with data reflecting the preference, but collecting sufficient data would be time-intensive [19, 33], placing a high workload on the human.

On the other hand, feature-based reward learning methods have the ability to more flexibly adapt to user preferences. By representing semantically-meaningful elements of the task as individual reward features [4], these methods improve alignment to the user's goals in a more interpretable manner [6]. However, the reward must be relearned and the policy retrained each time a new feature is learned, resulting in a cumbersome policy adaptation process. Moreover, these methods struggle with accurate reward estimation in high-dimensional spaces, making them less suited than imitation-learning-based counterparts for complex, dexterous tasks [2].

In this work, we aim to explore whether we can maintain the best of both imitation learning and feature-based reward learning methods, combining the generalist performance of the former with the flexibility and increased alignment of the latter. In particular, we explore the research question: **Can imposing boundaries using methods inspired by feature-based reward learning enable pretrained imitation learning policies to flexibly adapt to user requirements?**

In this work, we contribute:

1) A method for learning and updating individual elements of a user's task preferences using a reusable abstraction called a "preference boundary" inspired by prior work in feature-based reward learning;
2) A protocol for closed-loop validation of a policy's alignment to preferences using preference boundaries;
3) Preliminary evaluation of these two methods and discussion of the inclusion of these strategies in a full system with a black-box policy, their challenges, and next steps.

II. Related Work

A. Preference Learning

Prior work has explored teaching preferences as individual features in the reward [4] and adapting pre-trained preference models for new tasks [16]. Preferences can be learned via feature traces [3], through comparisons over states or features [5, 30], or by adapting pre-trained preference models [16]. Selection and incorporation of relevant reward features based on a user's preferences have also been automated using LLMs [29, 14]. All of these methods assume access to and ability to modify the policy reward, which are not possible when working with a black-box policy. In this work, we explore abstracting and learning individual elements of preferences [3, 4] and leveraging large models to propose missing elements [29] without access to policy reward.

B. Policy Steering and Shielding

Steering robot policies via in-context learning enables modification of policy behavior without expensive finetuning. Researchers have explored encoding visual observations and trajectories as sequences of tokens [26], biasing sampling processes with human input [34], and using autoregressive prediction on trajectories using a causal transformer [11] to facilitate in-context learning. However, these approaches require explicit training on the final task. Ma et al. [20] leverage the sequential structure of robot trajectories to output value predictions for in-context learning; however, this notion of value is general rather than being user-specific.

To prevent misalignment to user preferences, we draw on the idea of shielding from the SafeRL community, which uses a learned reactive system to ensure an RL policy adheres to

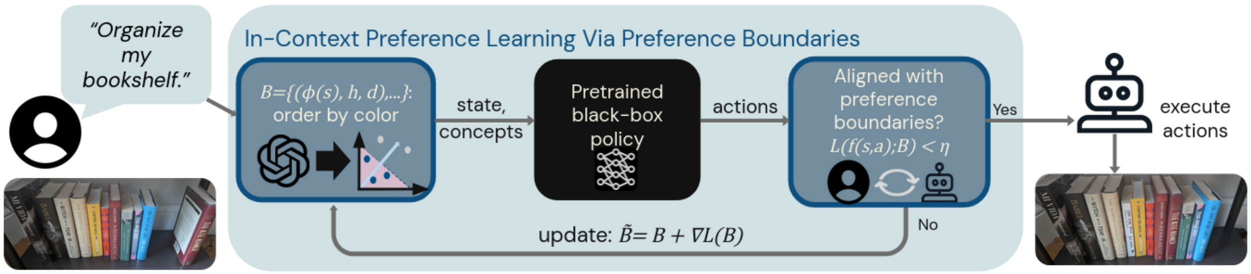

Fig. 1. We present methods for synthesizing a reactive system that wraps around a pretrained policy and adapts its behavior to human preferences. Preferences are first extracted and abstracted using "preference boundaries," which are also used to validate that the output of the pretrained policy aligns with user preferences. This paper examines the preference boundary extraction and alignment modules of this system, outlined in navy.

safety requirements [1, 8, 36]. The shield acts as a supervisor, monitoring the policy's behavior and intervening when certain specifications—pre-defined [1] or learned [8]—are violated.

C. LLMs and VLMs For Robotics

Recent works leverage the commonsense knowledge of LLMs or VLMs for robotics, despite these models lacking training on data representative of real-world physical tasks. Two approaches address this gap by using latent representations to address real-world physics [35] and aligning language to goal images [22], but both require training on data from the deployment environment, limiting flexibility.

Another strategy for enabling VLMs to contribute to real-world tasks without having to explicitly reason over spatial domains is by abstracting real-world semantics using keypoints. For example, previous approaches have encoded spatial constraints as constraints over keypoints [15], or used a language model to generate code defining reward over keypoints [27]. Our work will apply some of these insights.

III. Method

A. Desiderata and Overview

The two main desiderata are that the method 1) interpretably adapts to user preferences and 2) wraps around and steers a black-box policy.

Since there is no access to policy reward, interpretability cannot be addressed through prior methods on inducing alignment via modification or addition of reward features. We propose an abstraction called a "preference boundary" for capturing preferences in a reusable way (Section III-B) and an alignment verification step with a separate reward (Section III-C) to address this challenge. Assuming no access to policy weights for more flexible use with different policies means the policy cannot be finetuned [32, 28, 35]. We explore in-context learning to address this challenge [11, 34, 37].

We envision a final system framework as shown in Figure 1. Given corrective human input—e.g., placing a red book by others with red spines—the collection of preference boundaries is updated to better capture the person's preferences (Section III-B)—e.g., organize books by color of spines. Then, the preference boundaries are provided as contextual input to the black-box policy, and the output trajectory is verified (Section III-C)—e.g., a blue book is placed with other blue books. We leave adaptation of the preference boundaries when verification fails to future work as detailed in Section V.

B. Representing Preferences With "Preference Boundaries"

We formalize a **preference boundary** b as a function of the state s encoding both the notions of an expressed attribute $\phi(s)$, similar to a reward feature, as well a threshold h and direction d representing the permissible amount of that attribute according to the preference.

$$b = (\phi(s), h, d) \tag{1}$$

Preference boundaries function as abstractions that influence the steering of the policy according to interpretable representations of a person's preferences. An example of a preference boundary encoding allowable driving aggression is given in Figure 2. A key advantage of this representation is that $\phi(s)$ represents meaningful attributes that may also apply with variation to future tasks; if the user's preference does not directly apply out of the box—for example, they tolerate a more aggressive driving style in a city than in a suburban setting—only the threshold h needs to be adjusted.

We leverage the commonsense knowledge of a VLM similar to prior work [29], but use it to propose and systematically extract both the $\phi(s)$ of missing preference boundaries and the anchoring (h, d). The specific prompt to guide extraction and updates is given in Appendix A.

1) "Bag of Preference Boundaries": As each preference boundary represents an elementary attribute of a preference, we define a collection, a "bag" B, of m preference boundaries

$$B = (b_1(s), b_2(s), \ldots, b_m(s)) \tag{2}$$

to represent the suite of a person's preferences (e.g., organize books by color, but place oversized books on a separate shelf). Each preference boundary effectively represents a cut in state space delineating allowable states, so the bag of preference boundaries intuitively restricts the traversable state space.

There are two categories of updates to B: adding/removing a preference boundary, or modifying an existing preference boundary. Preference boundaries must remain distinct following updates to avoid overparameterizing the space. Namely,

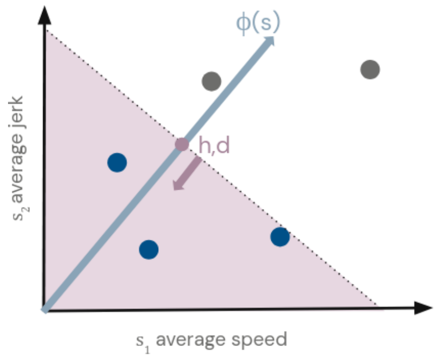

Fig. 2. Toy example of a preference boundary representing the acceptable level of driving aggression. $\phi(s)$ is a vector delineating the idea of "aggression" given average speed and average jerk, and (h, d) are a threshold and direction pair characterizing allowable amount of aggression.

the ϕ vectors must be sufficiently different:

$$dist(\phi_i, \phi_j) > \eta, \forall (i, j) \in [1, m], i \neq j \tag{3}$$

for some η threshold given a distance metric. A modification on any ϕ_i must trigger an iterative check on all other ϕ_j vectors to ensure this property holds. If such a solution is infeasible, then redundant vectors may need to be consolidated.

Here, since preference boundaries are represented in the language domain, the VLM is used to approximate the distance metric as a first baseline approach. While language models' next-token prediction does not allow for rigorous computation of distances, prompting techniques such as sampling or prompting for relative comparisons can increase reliability.

C. Keypoint-Based Alignment Validation

In this section, we present a method for validating that policy output is aligned with learned preference boundaries.

A key challenge to alignment verification is translating from the language domain, where preference boundaries are collected, to the spatial domain of real-world robot trajectories. We build off of recent work on encoding spatial elements relevant to task completion as keypoints in image space [15, 27]. Thus, the model is only required to reason in 2D cartesian space, as opposed to reasoning over poses in SE(3) space.

We propose the following process for alignment validation. First, keypoints are extracted keypoints using an off-the-shelf visual feature extractor such as DINOv2 [25]. Then, keypoints are downsampled to keep a sparse number of semantically-meaningful ones [15]. Next, the VLM is prompted to generate code calculating alignment cost over keypoints, similar to the method proposed by Patel et al. [27]. Finally, this code is executed in-the-loop (along with keypoint tracking) during execution of the robot trajectory to evaluate alignment. If alignment is low, then the robot iteratively refines the learned preference boundaries or queries the person for help. Returning to the bookshelf organizing example, keypoints may each represent a single book, and the cost may be calculated as the sum of euclidean distances between the keypoint of the book to be moved and its two closest matches (Appendix C).

Fig. 3. Evaluation of ability to extract preference boundaries in a bookshelf organization task. The latent preference is to organize books by color. In the baseline method (left), the VLM is given only the correction of where *The Kite Runner* should be placed. In the proposed method (right), the VLM is asked to extract the relevant preference boundary feature ϕ and anchoring (h, d) after being given the same correction. Books are indeed ordered by color in the proposed case, which is untrue in the baseline case.

The prompt for alignment cost is given in Appendix B.

IV. Preliminary Evaluation

Preliminary evaluation was performed using GPT-4o [24] and focuses on validating preference boundary extraction and alignment verification using a VLM; evaluation with a full pipeline including robot policy is left to future work.

A. Preference Boundary Extraction and Updates

The three main goals of evaluation for preference boundary extraction were: first, to extract both the features and anchorings of preference boundaries using a VLM with a human in the loop; second, to validate bag of preference boundaries updates; and third, to ensure that extracted preference boundaries meaningfully influence task outputs. For the third, the VLM was used as a proxy "agent" for preliminary evaluation.

Three tasks were evaluated: organizing a bookshelf, sorting laundry into delicates versus normal wash, and gathering matching stationery. Each task incorporated a latent preference: organize by color, consider any decorations on top of original fabric to be delicate, and match by thematic element.

In the experimental cases, the model was prompted to propose five most likely missing preference boundaries following a correction, then to update the bag of preference boundaries with the human-labeled closest match. The VLM was instructed that latent preferences should be represented as preference boundaries and given an example. The same number of corrections were given for the baseline cases, in which general instructions about latent preferences were given without knowledge of preference boundaries.

Below, we report the accuracy of predictions for each task for the proposed and baseline methods ($n = 5$), the maximum number of corrections provided for each task (constant for all trials), and average rankings of the correct preference boundaries in the proposed lists (proposed method only).

Task	Proposed	Baseline	Corrections	Avg Rank
Bookshelf	**.75**	0.2	1	4
Laundry	**.75**	.56	4	1.2
Stationery	**.67**	.37	2	2.6

TABLE I
RESULTS FROM PREFERENCE BOUNDARY EVALUATION

Fig. 4. Evaluation of reward generation in toy bookshelf scenario to validate alignment with latent preference of organizing by color. The image on the top (a) was given as input, and the two on the bottom were used as test. The cost calculated using the output reward function (119) for (b), where *The Kite Runner* is placed correctly next to other red books, is indeed lower than the cost (338) for (c), where it is incorrectly placed next to the blue books.

Overall, task accuracies are higher using the proposed method ($p = 0.004$, $p = 0.003$, $p = 8.4e^{-4}$, respectively), suggesting that preference boundaries can meaningfully affect downstream tasks. Moreover, all correct latent preference boundaries were proposed within the five top candidates, validating the use of VLMs in extracting preference boundaries.

The discrepancy in performance between proposed and baseline methods is greater when the preferences deviate more from common sense. For example, in the bookshelf task, the average rank of the preference to organize by color was fourth (commonly after alphabetical by author, alphabetical by title, reverse alphabetical by author, etc.). In the baseline, the VLM struggles to organize the books by color, instead leveraging prior knowledge that books are often alphabetized by author or title even when the data contradict this (Figure 3).

These results also support the proposed updates to the bag of preference boundaries. The multiple corrections given in the laundry and stationery tasks each reflected different elements of the preference (e.g., "any decorations" includes embroidery, graphic prints, sequins, etc.), which were successfully reflected in the bag of preference boundaries, while the baseline method struggled to generalize. Moreover, corrections given in the laundry task showing an increased tolerance of fragile elements belonging in normal wash resulted in only updates to the anchoring of the correct preference boundary.

B. Keypoint-Based Alignment Validation

We evaluated whether it is possible to generate cost functions over keypoints that represent alignment to preference boundaries using the bookshelf task. The VLM was prompted with a single image manually annotated with keypoints representing each book (Figure 4 (a)) and the instruction to generate a cost function over alignment with the preference to organize by color. The outputted function, which adds the euclidean distances between the target book and the two other red books, was used to evaluate two test images, one with the book in the correct location, and one with the incorrect location as shown in Figure 4 (b) and (c), respectively. The cost for the correct placement (119) was indeed lower than the cost for incorrect placement (338), validating the preference alignment cost.

We also found that the generated cost could reflect a changing anchoring of the preference, such as if books could be approximately ordered by color. This generated function still resulted in a lower cost when the book is inserted correctly (30) in Figure 4 (b) versus incorrectly (74) in Figure 4 (c).

However, while these generated cost functions are able to reflect relative preference alignment, we found GTP-4o limited for verifying preference boundary anchoring. Specifically, the model was unable to generate a cost function where negative values represent permissible examples and positive values represent invalid examples according to anchoring. Further work is required to better evaluate preference boundary anchoring.

V. DISCUSSION

In this work, we introduced methods for abstracting a user's preferences to influence downstream tasks and validating that the policy adheres to the preferences.

A. Future Work

We plan to next explore using an open-source vision-language-action model or training a diffusion-based policy [35] in the system pipeline. To ground the preference boundaries, we are interested in investigating using a state editor to mask or augment the input image [29] or extracting a latent encoding for each preference boundary vector.

Future work also includes exploring strategies for active learning through strategic querying when verification fails, managing conflicting preference boundaries, and identifying the most effective mode(s) for interaction [10].

B. Limitations

One central limitation is this system is only as capable as the underlying pretrained policy. If the policy is incapable of handling fine-grained requirements dictated by a person's preferences (such as moving two books to insert a third), well-represented preferences cannot overcome this limitation. Additionally, visual keypoints cannot represent arbitrary preferences, and identifying that a policy does not adhere to the learned preference boundaries does not inform as to how to prevent that alignment failure in the future.

References

[1] Mohammed Alshiekh, Roderick Bloem, Rüdiger Ehlers, Bettina Könighofer, Scott Niekum, and Ufuk Topcu. Safe Reinforcement Learning via Shielding. *Proceedings of the AAAI Conference on Artificial Intelligence*, 32(1), April 2018. ISSN 2374-3468. doi: 10.1609/aaai.v32i1.11797. URL https://ojs.aaai.org/index.php/AAAI/article/view/11797. Number: 1.

[2] Shan An, Ziyu Meng, Chao Tang, Yuning Zhou, Tengyu Liu, Fangqiang Ding, Shufang Zhang, Yao Mu, Ran Song, Wei Zhang, Zeng-Guang Hou, and Hong Zhang. Dexterous Manipulation through Imitation Learning: A Survey, April 2025. URL https://arxiv.org/abs/2504.03515v3.

[3] Andreea Bobu, Marius Wiggert, Claire Tomlin, and Anca D. Dragan. Feature Expansive Reward Learning: Rethinking Human Input. In *Proceedings of the 2021 ACM/IEEE International Conference on Human-Robot Interaction*, pages 216–224, March 2021. doi: 10.1145/3434073.3444667. URL http://arxiv.org/abs/2006.13208. arXiv:2006.13208 [cs].

[4] Andreea Bobu, Marius Wiggert, Claire Tomlin, and Anca D Dragan. Inducing structure in reward learning by learning features. *The International Journal of Robotics Research*, 41(5):497–518, April 2022. ISSN 0278-3649, 1741-3176. doi: 10.1177/02783649221078031. URL https://journals.sagepub.com/doi/10.1177/02783649221078031.

[5] Andreea Bobu, Yi Liu, Rohin Shah, Daniel S. Brown, and Anca D. Dragan. SIRL: Similarity-based Implicit Representation Learning. In *Proceedings of the 2023 ACM/IEEE International Conference on Human-Robot Interaction*, HRI '23, pages 565–574, New York, NY, USA, March 2023. Association for Computing Machinery. ISBN 978-1-4503-9964-7. doi: 10.1145/3568162.3576989. URL https://dl.acm.org/doi/10.1145/3568162.3576989.

[6] Andreea Bobu, Andi Peng, Pulkit Agrawal, Julie A Shah, and Anca D. Dragan. Aligning Human and Robot Representations. In *Proceedings of the 2024 ACM/IEEE International Conference on Human-Robot Interaction*, HRI '24, pages 42–54, New York, NY, USA, March 2024. Association for Computing Machinery. ISBN 979-8-4007-0322-5. doi: 10.1145/3610977.3634987. URL https://dl.acm.org/doi/10.1145/3610977.3634987.

[7] Anthony Brohan, Noah Brown, Justice Carbajal, Yevgen Chebotar, Xi Chen, Krzysztof Choromanski, Tianli Ding, Danny Driess, Avinava Dubey, Chelsea Finn, Pete Florence, Chuyuan Fu, Montse Gonzalez Arenas, Keerthana Gopalakrishnan, Kehang Han, Karol Hausman, Alexander Herzog, Jasmine Hsu, Brian Ichter, Alex Irpan, Nikhil Joshi, Ryan Julian, Dmitry Kalashnikov, Yuheng Kuang, Isabel Leal, Lisa Lee, Tsang-Wei Edward Lee, Sergey Levine, Yao Lu, Henryk Michalewski, Igor Mordatch, Karl Pertsch, Kanishka Rao, Krista Reymann, Michael Ryoo, Grecia Salazar, Pannag Sanketi, Pierre Sermanet, Jaspiar Singh, Anikait Singh, Radu Soricut, Huong Tran, Vincent Vanhoucke, Quan Vuong, Ayzaan Wahid, Stefan Welker, Paul Wohlhart, Jialin Wu, Fei Xia, Ted Xiao, Peng Xu, Sichun Xu, Tianhe Yu, and Brianna Zitkovich. RT-2: Vision-Language-Action Models Transfer Web Knowledge to Robotic Control, July 2023. URL http://arxiv.org/abs/2307.15818. arXiv:2307.15818.

[8] Steven Carr, Nils Jansen, Sebastian Junges, and Ufuk Topcu. Safe Reinforcement Learning via Shielding under Partial Observability. January 2023. URL https://openreview.net/forum?id=f1LL39MUmc.

[9] Danny Driess, Fei Xia, Mehdi S. M. Sajjadi, Corey Lynch, Aakanksha Chowdhery, Brian Ichter, Ayzaan Wahid, Jonathan Tompson, Quan Vuong, Tianhe Yu, Wenlong Huang, Yevgen Chebotar, Pierre Sermanet, Daniel Duckworth, Sergey Levine, Vincent Vanhoucke, Karol Hausman, Marc Toussaint, Klaus Greff, Andy Zeng, Igor Mordatch, and Pete Florence. PaLM-E: An Embodied Multimodal Language Model, March 2023. URL http://arxiv.org/abs/2303.03378. arXiv:2303.03378.

[10] Tesca Fitzgerald, Pallavi Koppol, Patrick Callaghan, Russell Quinlan Jun Hei Wong, Reid Simmons, Oliver Kroemer, and Henny Admoni. INQUIRE: INteractive Querying for User-aware Informative REasoning. In *Proceedings of The 6th Conference on Robot Learning*, pages 2241–2250. PMLR, March 2023. URL https://proceedings.mlr.press/v205/fitzgerald23a.html. ISSN: 2640-3498.

[11] Letian Fu, Huang Huang, Gaurav Datta, Lawrence Yunliang Chen, William Chung-Ho Panitch, Fangchen Liu, Hui Li, and Ken Goldberg. In-Context Imitation Learning via Next-Token Prediction. October 2024. URL https://openreview.net/forum?id=2R3q4FyPlH.

[12] Tian Gao, Soroush Nasiriany, Huihan Liu, Quantao Yang, and Yuke Zhu. PRIME: Scaffolding Manipulation Tasks With Behavior Primitives for Data-Efficient Imitation Learning. *IEEE Robotics and Automation Letters*, 9(10): 8322–8329, October 2024. ISSN 2377-3766. doi: 10.1109/LRA.2024.3443610. URL https://ieeexplore.ieee.org/document/10637675/?arnumber=10637675. Conference Name: IEEE Robotics and Automation Letters.

[13] Siddhant Haldar, Jyothish Pari, Anant Rai, and Lerrel Pinto. Teach a Robot to FISH: Versatile Imitation from One Minute of Demonstrations. In *Robotics: Science and Systems XIX*. Robotics: Science and Systems Foundation, July 2023. ISBN 978-0-9923747-9-2. doi: 10.15607/RSS.2023.XIX.009. URL http://www.roboticsproceedings.org/rss19/p009.pdf.

[14] Kunal Handa, Yarin Gal, Ellie Pavlick, Noah Goodman, Jacob Andreas, Alex Tamkin, and Belinda Z. Li. Bayesian Preference Elicitation with Language Models, March 2024. URL http://arxiv.org/abs/2403.05534. arXiv:2403.05534 [cs].

[15] Wenlong Huang, Chen Wang, Yunzhu Li, Ruohan Zhang, and Li Fei-Fei. ReKep: Spatio-Temporal Reasoning of

Relational Keypoint Constraints for Robotic Manipulation. September 2024. URL https://openreview.net/forum?id=9iG3SEbMnL.

[16] Donald Joseph Hejna Iii and Dorsa Sadigh. Few-Shot Preference Learning for Human-in-the-Loop RL. August 2022. URL https://openreview.net/forum?id=IKC5TfXLuW0.

[17] Timo Kaufmann, Paul Weng, Viktor Bengs, and Eyke Hüllermeier. A Survey of Reinforcement Learning from Human Feedback. *Transactions on Machine Learning Research*, June 2024. URL https://openreview.net/forum?id=f7OkIurx4b.

[18] Moo Jin Kim, Karl Pertsch, Siddharth Karamcheti, Ted Xiao, Ashwin Balakrishna, Suraj Nair, Rafael Rafailov, Ethan Foster, Grace Lam, Pannag Sanketi, Quan Vuong, Thomas Kollar, Benjamin Burchfiel, Russ Tedrake, Dorsa Sadigh, Sergey Levine, Percy Liang, and Chelsea Finn. OpenVLA: An Open-Source Vision-Language-Action Model, September 2024. URL http://arxiv.org/abs/2406.09246. arXiv:2406.09246 [cs].

[19] Abdoulaye O. Ly and Moulay Akhloufi. Learning to Drive by Imitation: An Overview of Deep Behavior Cloning Methods. *IEEE Transactions on Intelligent Vehicles*, 6(2):195–209, June 2021. ISSN 2379-8904. doi: 10.1109/TIV.2020.3002505. URL https://ieeexplore.ieee.org/abstract/document/9117169. Conference Name: IEEE Transactions on Intelligent Vehicles.

[20] Yecheng Jason Ma, Joey Hejna, Ayzaan Wahid, Chuyuan Fu, Dhruv Shah, Jacky Liang, Zhuo Xu, Sean Kirmani, Peng Xu, Danny Driess, Ted Xiao, Jonathan Tompson, Osbert Bastani, Dinesh Jayaraman, Wenhao Yu, Tingnan Zhang, Dorsa Sadigh, and Fei Xia. Vision Language Models are In-Context Value Learners, November 2024. URL http://arxiv.org/abs/2411.04549. arXiv:2411.04549 [cs].

[21] Ajay Mandlekar, Caelan Garrett, Danfei Xu, and Dieter Fox. Human-in-the-Loop Task and Motion Planning for Imitation Learning, October 2023. URL http://arxiv.org/abs/2310.16014. arXiv:2310.16014 [cs].

[22] Vivek Myers, Andre Wang He, Kuan Fang, Homer Rich Walke, Philippe Hansen-Estruch, Ching-An Cheng, Mihai Jalobeanu, Andrey Kolobov, Anca Dragan, and Sergey Levine. Goal Representations for Instruction Following: A Semi-Supervised Language Interface to Control. August 2023. URL https://openreview.net/forum?id=0bZaUfELuW.

[23] Scott Niekum, Sachin Chitta, Andrew Barto, Bhaskara Marthi, and Sarah Osentoski. Incremental Semantically Grounded Learning from Demonstration. In *Robotics: Science and Systems IX*. Robotics: Science and Systems Foundation, June 2013. ISBN 978-981-07-3937-9. doi: 10.15607/RSS.2013.IX.048. URL http://www.roboticsproceedings.org/rss09/p48.pdf.

[24] OpenAI. GPT-4 Technical Report, March 2024.

[25] Maxime Oquab, Timothée Darcet, Théo Moutakanni, Huy Vo, Marc Szafraniec, Vasil Khalidov, Pierre Fernandez, Daniel Haziza, Francisco Massa, Alaaeldin El-Nouby, Mahmoud Assran, Nicolas Ballas, Wojciech Galuba, Russell Howes, Po-Yao Huang, Shang-Wen Li, Ishan Misra, Michael Rabbat, Vasu Sharma, Gabriel Synnaeve, Hu Xu, Hervé Jegou, Julien Mairal, Patrick Labatut, Armand Joulin, and Piotr Bojanowski. DINOv2: Learning Robust Visual Features without Supervision, February 2024. URL http://arxiv.org/abs/2304.07193. arXiv:2304.07193 [cs].

[26] Norman Palo and Edward Johns. Keypoint Action Tokens Enable In-Context Imitation Learning in Robotics. In *Robotics: Science and Systems XX*. Robotics: Science and Systems Foundation, July 2024. ISBN 979-8-9902848-0-7. doi: 10.15607/RSS.2024.XX.096. URL http://www.roboticsproceedings.org/rss20/p096.pdf.

[27] Shivansh Patel, Xinchen Yin, Wenlong Huang, Shubham Garg, Hooshang Nayyeri, Li Fei-Fei, Svetlana Lazebnik, and Yunzhu Li. A Real-to-Sim-to-Real Approach to Robotic Manipulation with VLM-Generated Iterative Keypoint Rewards, February 2025. URL http://arxiv.org/abs/2502.08643. arXiv:2502.08643 [cs].

[28] Andi Peng, Andreea Bobu, Belinda Z. Li, Theodore R. Sumers, Ilia Sucholutsky, Nishanth Kumar, Thomas L. Griffiths, and Julie A. Shah. Preference-Conditioned Language-Guided Abstraction. In *Proceedings of the 2024 ACM/IEEE International Conference on Human-Robot Interaction*, HRI '24, pages 572–581, New York, NY, USA, March 2024. Association for Computing Machinery. ISBN 979-8-4007-0322-5. doi: 10.1145/3610977.3634930. URL https://dl.acm.org/doi/10.1145/3610977.3634930.

[29] Andi Peng, Belinda Z. Li, Ilia Sucholutsky, Nishanth Kumar, Julie Shah, Jacob Andreas, and Andreea Bobu. Adaptive Language-Guided Abstraction from Contrastive Explanations. September 2024.

[30] Andi Peng, Yuying Sun, Tianmin Shu, and David Abel. Pragmatic feature preferences: learning reward-relevant preferences from human input. In *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *ICML'24*, pages 40258–40270, Vienna, Austria, July 2024. JMLR.org.

[31] Carl Orge Retzlaff, Srijita Das, Christabel Wayllace, Payam Mousavi, Mohammad Afshari, Tianpei Yang, Anna Saranti, Alessa Angerschmid, Matthew E. Taylor, and Andreas Holzinger. Human-in-the-Loop Reinforcement Learning: A Survey and Position on Requirements, Challenges, and Opportunities. *J. Artif. Int. Res.*, 79, January 2024. ISSN 1076-9757. doi: 10.1613/jair.1.15348. URL https://dl.acm.org/doi/10.1613/jair.1.15348.

[32] Mariah L. Schrum, Emily Sumner, Matthew C. Gombolay, and Andrew Best. MAVERIC: A Data-Driven Approach to Personalized Autonomous Driving. *IEEE Transactions on Robotics*, pages 1–14, 2024. ISSN 1552-3098, 1941-0468. doi: 10.1109/TRO.2024.3359543. URL https://ieeexplore.ieee.org/document/10415518/.

[33] Qiang Wang, Robert McCarthy, David Cordova Bulens,

Francisco Roldan Sanchez, Kevin McGuinness, Noel E. O'Connor, and Stephen J. Redmond. Identifying Expert Behavior in Offline Training Datasets Improves Behavioral Cloning of Robotic Manipulation Policies. *IEEE Robotics and Automation Letters*, 9(2):1294–1301, February 2024. ISSN 2377-3766. doi: 10.1109/LRA.2023.3342559. URL https://ieeexplore.ieee.org/abstract/document/10356825. Conference Name: IEEE Robotics and Automation Letters.

[34] Yanwei Wang, Lirui Wang, Yilun Du, Balakumar Sundaralingam, Xuning Yang, Yu-Wei Chao, Claudia Perez-D'Arpino, Dieter Fox, and Julie Shah. Inference-Time Policy Steering through Human Interactions, March 2025. URL http://arxiv.org/abs/2411.16627. arXiv:2411.16627 [cs].

[35] Yilin Wu, Ran Tian, Gokul Swamy, and Andrea Bajcsy. From Foresight to Forethought: VLM-In-the-Loop Policy Steering via Latent Alignment, May 2025. URL http://arxiv.org/abs/2502.01828. arXiv:2502.01828 [cs].

[36] Wen-Chi Yang, Giuseppe Marra, Gavin Rens, and Luc De Raedt. Safe Reinforcement Learning via Probabilistic Logic Shields. In *Proceedings of the Thirty-Second International Joint Conference on Artificial Intelligence*, pages 5739–5749, Macau, SAR China, August 2023. International Joint Conferences on Artificial Intelligence Organization. ISBN 978-1-956792-03-4. doi: 10.24963/ijcai.2023/637. URL https://www.ijcai.org/proceedings/2023/637.

[37] Yida Yin, Zekai Wang, Yuvan Sharma, Dantong Niu, Trevor Darrell, and Roei Herzig. In-Context Learning Enables Robot Action Prediction in LLMs, March 2025. URL http://arxiv.org/abs/2410.12782. arXiv:2410.12782 [cs].

[38] Tony Z. Zhao, Jonathan Tompson, Danny Driess, Pete Florence, Kamyar Ghasemipour, Chelsea Finn, and Ayzaan Wahid. ALOHA Unleashed: A Simple Recipe for Robot Dexterity, October 2024. URL http://arxiv.org/abs/2410.13126. arXiv:2410.13126.

APPENDIX

A. Preference Boundary Extraction and Updates

The VLM is given the following priming instruction to familiarize it with the idea of a preference boundary (concept) and anchoring (calibration):

`You are a household robot helping me to perform various tasks. As you help me with various tasks, I may have certain preferences about how they should be done and give you a correction in the form of rules that I want you to follow when you help me perform these tasks. Think of each rule as having 2 parts: there will be an underlying concept, specifying what attribute(s) of the task a preference concerns, and a calibration of the concept, specifying how much of the concept makes something preferred vs. not. For example, if I wanted you to help me select 7 avocados at the grocery store, I might have a preference that the 7 avocados should range from ripe to unripe such that they will ripen at different times and I can eat one a day for a week. The concept in this case is the variety in avocado ripeness, and the calibration is that I want it to be as varied as possible.`

The VLM is given the following priming instruction in the control evaluation case:

`You are a household robot helping me to perform various tasks. As you help me with various tasks, I may have certain preferences about how they should be done and give you a correction in the form of rules that I want you to follow when you help me perform these tasks.`

B. Keypoint-Based Alignment Evaluation

The VLM is given the following prompt to ground its code generation for evaluating alignment cost:

`Your job is to help with evaluating whether taking an action will result in what the user wants by writing code in python. The setting is given as an image of the environment overlaid with keypoints marked with their indices along with a text instruction. What the person wants is given in the text instruction as a "concept": this consists of an "idea", and "anchoring" of how much the action must comply with the idea. These keypoints are in 3D space, and are projected onto the 2D image. They are attached with the objects, and move along with them. The text instruction will also include the task that will be performed.`

`The code should be a mathematical operation on keypoints; you should decide which (or all) of the keypoints are relevant to both the task and the user's desired "concept" The input to the function is a dictionary of keypoint coordinates. So keys will be strings like "1", "2", ... and their values will be numpy arrays ([x, y, z]) representing the 3D location of the keypoint corresponding to that index. The function will return a single value representing the cost of the current state in accordance with the user's desired "concept". The lower the cost, the more the action aligns with the concept. If this action is allowable according to the person's concept anchor, the cost value should be strictly less than 0.`

`You should not forecast the cost of the future; do not attempt to decide how to solve the task or to calculate the cost after the task has been solved. Your job is only to generate a function that will be evaluated multiple times as the action is being taken to evaluate how well the user's concept is being satisfied.`

`Structure your output in a single python code block as follows:`

```
def get_concept_cost(keypoint_coordinates):
# Put your explanation here.
keypoint_indices_to_use = ?

# mathematical operation on keypoints to determine
# the cost of taking actions in accordance with
# the user's desired ``concept''
total_cost = 0
for keypoint_ind in keypoint_indices_to_use:
        # Write calculation here.
        # You may use multiple lines
        cost_ind = ?
        total_cost += cost_ind

return total_cost
```

C. VLM output for the strict organization case

```
def get_concept_cost(keypoint_coordinates):
    """
    This function evaluates how well the current
    state of book arrangement satisfies the concept
    of organizing by spine color.
    Specifically, it computes how well the book
    labeled 'A' is integrated into a local region
    with similar spine colors.

    We treat keypoints as representing individual
    books, and use spatial proximity in 3D to infer
    placement. To match the concept, we expect that
    'A' is placed adjacent to books of similar spine
    color (reddish).Based on the image, books 5, 6,
    and possibly 7 (reddish tones) are appropriate
    neighbors for keypoint 'A'.
```

```
    The cost is computed as the 3D Euclidean
    distance from 'A' to its most similar-color
    group (book 5 and 6).
    Lower distance > better placement > lower cost.
    If it's too far, the cost will be high.
    """

    # Red group neighborhood (based on spine color)
    red_group_indices = ["5", "6"]
    a_position = keypoint_coordinates["A"]

    total_cost = 0
    for keypoint_ind in red_group_indices:
        group_pos = keypoint_coordinates[
            keypoint_ind]
        cost_ind = np.linalg.norm(
            a_position - group_pos
            )
        total_cost += cost_ind

    return total_cost
```

D. VLM output for the lenient organization case (books may be within 2 books of correct placement)

```
def get_concept_cost(keypoint_coordinates):
    """
    This function evaluates how well the current
    book arrangement satisfies the concept of
    approximate color gradient organization.
    Keypoint 'A' represents the book to be
    integrated into the left grouping. The spine
    of this book is maroon/red.

    Based on the concept, the goal is for similarly
    colored books to be close together (with
    leniency: up to two books apart).

    Red-colored books are around keypoints 5 and 6.
    The ideal position for 'A' is next to these
    (either directly adjacent or within two
    positions away).

    To compute the cost, we calculate the minimum
    distance from 'A' to keypoints 4-8 (a 5-book
    window centered around the red zone),
    penalizing more for further distance.
    If 'A' is within two-book positions from 5
    or 6, cost is zero. Otherwise, cost increases
    with distance from this red cluster.
    """
    red_center_indices = [str(i) for i in range(4,
        9)]
    # range of possibly acceptable placements

    a_position = keypoint_coordinates["A"]
    min_distance = float('inf')

    for keypoint_ind in red_center_indices:
        other_pos = keypoint_coordinates[
            keypoint_ind]
        dist = np.linalg.norm(
            a_position - other_pos
            )
        if dist < min_distance:
            min_distance = dist

    # Cost is proportional to distance from the red
    # group
    # Acceptable if 'A' is within ~2-book spacing
    # (threshold set empirically as 2 book widths ~
    # 10 units in 3D)
    return min_distance - 10
    # cost < 0 means acceptable placement
```