

Abstract

**Robots Adapting to People
Without Becoming a Nuisance**

Debasmita Ghose

2026

Human-Robot Collaboration (HRC) requires robots to maintain shared understanding with people. The robot’s internal belief about a human partner’s goal must remain aligned with what the human is actually trying to accomplish, despite ambiguity, limited explicit communication, and changing objectives. In human-human collaboration, people rarely provide step-by-step instructions to their collaborators to establish a shared understanding of the task; instead, intent is conveyed implicitly through behavior and context, as well as brief, high-level language. Inspired by this idea, this dissertation develops methods that enable a robot to infer and adapt to human goals in human-robot collaboration using signals that arise naturally during interaction, while minimizing the need for frequent supervision or repeated explicit instruction.

The technical contributions are organized by increasing levels of human intervention and increasingly challenging sources of uncertainty. First, for adaptation from passive observation, we present an online contrastive representation-learning method that tailors visual object representations to the distinctions a human implicitly defines through their selections, enabling sample-efficient learning of task-relevant concepts in a shared sorting task. Second, for settings where observation alone is insufficient

due to goal ambiguity, we introduce active approaches that enable a robot to take actions that influence the human to reveal their goals through behavioral cues. We introduce Critical Decision Points as states where policies for different possible goals diverge maximally, and use receding-horizon planning to actively guide interaction toward such informative states while maintaining task progress. Third, to handle non-stationary goals, we develop a goal-change detection method and propose that the robot takes actions that support the task while actively influencing the human to take actions that reveal their updated goals after the human’s objective shifts mid-execution. Finally, we introduce BALI (Bidirectional Action–Language Inference), which treats a person’s language and their observed behavior as coupled signals about their intended goal. BALI uses high-level directives to narrow the set of plausible goal interpretations in the current context, uses the sequence of actions and task history to ground and update those interpretations as the task unfolds, and asks targeted clarification questions only when the remaining uncertainty is too high to select a supportive action.

This dissertation contributes a set of mechanisms for maintaining shared understanding during human-robot collaboration through observation, active influence, and high-level natural language directives, enabling robots to provide context-appropriate assistance without requiring humans to manage them through exhaustive instruction.

**Robots Adapting to People
Without Becoming a Nuisance**

A Dissertation

Presented to the Faculty of the Graduate School

Of

Yale University

In Candidacy for the Degree of

Doctor of Philosophy

By

Debasmita Ghose

Dissertation Directors: Brian Scassellati

May 2026

Copyright © 2026 by Debasmita Ghose
All rights reserved.

Contents

Dedication	xi
Acknowledgements	xii
List of Figures	xvi
List of Tables	xxvi
Glossary	xxvii
1 Introduction	1
1.1 Challenges for Shared Understanding with Minimal Supervision . . .	3
1.1.1 Inferring Intent from Observations of Human Behavior	3
1.1.2 Adapting under Non-Stationary Human Objectives	5
1.2 Contributions	6
1.3 Thesis Outline	8
2 Adaptation in Physical Human-Robot Collaboration: A Review	10
2.1 Introduction	11
2.1.1 Method, Scope and Organization of the Survey	12
2.1.2 Comparison to Other Surveys	14
2.2 Human Modeling Assumptions for Online Adaptation	15
2.2.1 One-Way Adaptation	16

2.2.2	Mutual Adaptation	18
2.3	Adaptation Techniques	19
2.3.1	Planning-based Adaptation Approaches	20
2.3.2	Learning-based Adaptation Approaches	23
2.3.3	Transient Behavioral Adaptation Approaches	27
2.3.4	Hybrid Adaptation Approaches	30
2.4	Human Cues and Communication for Adaptation	31
2.4.1	Human Observation Channels	32
2.4.2	Information Exchange between the Human and the Robot . .	35
2.4.3	Human Burden	37
2.5	Evaluation of Adaptation	40
2.5.1	Evaluation Tasks and Settings	40
2.5.2	Evaluation Metrics	42
2.6	Summary	50
3	Robots Inferring Human Goals through Passive Observation	51
3.1	Introduction	52
3.2	Related Work	55
3.2.1	Self-Supervised Learning	55
3.2.2	Adding Supervision to Contrastive Learning	56
3.2.3	Human Centric Robot Learning	56
3.3	Learning Visual Object Representations Tailored to Human Require- ments	57
3.4	Experimental Setup	61
3.5	Offline Evaluation	62
3.5.1	Data Collection	62
3.5.2	Evaluation Protocol	63
3.5.3	Quantitative Results	63

3.5.4	Ablation Study	64
3.6	A Case Study with a Recycling Robot	68
3.6.1	System Architecture	68
3.6.2	Experimental Protocol and Results	70
3.7	Limitations and Future Work	72
3.8	Summary	73
4	Active Goal Inference By Influencing Human Actions	75
4.1	Introduction	76
4.2	Related Work	80
4.2.1	Robots Predicting Human Goals	80
4.2.2	Robots Influencing Humans	82
4.3	Influencing Humans to Reveal their Goals at Critical Decision Points	82
4.3.1	Preliminaries	82
4.3.2	Critical Decision Points	85
4.3.3	Influencing the Human to Critical Decision Points	89
4.4	Scenario 1: Influencing the Human to Reveal their Goal in Human- Robot Collaboration	91
4.4.1	Approach	91
4.4.2	Task and Problem Representation	95
4.4.3	Experimental Setup	98
4.4.4	Comparisons	101
4.4.5	Metrics	103
4.4.6	Simulation Results	103
4.4.7	Ablation Study	109
4.4.8	Real Robot Results	110
4.4.9	Comparative Analysis of Performance of CDPs in the Collabo- rative Task	112

4.5	Scenario 2: Influencing the Human to Reveal their Goal in a Compet-	
	itive Human Robot Task	114
4.5.1	Approach	114
4.5.2	Task and Experimental Setup	117
4.5.3	Implementation Details	122
4.5.4	Comparisons	124
4.5.5	Results	125
4.5.6	Comparative Analysis of Performance of CDPs in the Compet-	
	itive Task	127
4.6	Discussion	128
4.6.1	Applicability Across Diverse Task Settings	128
4.6.2	Limitations and Future Work	128
4.7	Summary	129
5	Active Inference of Shifting Goals during Human-Robot Collabora-	
	tion	130
5.1	Introduction	131
5.2	Related Work	133
5.2.1	Passive Goal Inference	133
5.2.2	Active Human Influence	134
5.3	Method	135
5.3.1	Preliminaries	135
5.3.2	Active Selection of Robot Actions to Reveal Human Goals . .	136
5.3.3	Identifying Likely Human Goal Changes	139
5.3.4	Robot Aligning Actions with Updated Goals	140
5.4	Evaluation	142
5.4.1	Task and Problem Representation	142
5.4.2	Experiments	144

5.4.3	Implementation Details	148
5.4.4	Comparisons	149
5.4.5	Performance Measures	151
5.5	Results	151
5.5.1	Simulation	151
5.5.2	Physical Robot	152
5.6	Discussion	154
5.7	Summary	155

6 Open-Ended Goal Inference using Actions and Language for Human-

	Robot Collaboration	157
6.1	Introduction	158
6.2	Related Work	161
6.3	Preliminaries	162
6.3.1	Problem Formulation	163
6.3.2	Receding Horizon Planning (RHP)	163
6.3.3	Attractor Fields	164
6.4	Bidirectional Action-Language Inference (BALI) for Goal Prediction .	165
6.4.1	Interaction Summary	167
6.4.2	Goal Inference	167
6.4.3	Action Selection and Cost Function	168
6.4.4	Asking Informative Questions	170
6.5	Evaluation	174
6.5.1	Task and Problem Representation	174
6.5.2	Experimental Setup	175
6.5.3	Baselines	177
6.5.4	Metrics	180
6.6	Results in Simulation	180

6.6.1	Open Case	180
6.6.2	Closed Case	183
6.7	Real World Demonstration	186
6.8	Discussion	187
6.9	Summary	189
7	Conclusion	190
7.1	Summary of Contributions	191
7.2	Open Research Questions	193
7.2.1	Real-world Adaptation Considerations	194
7.2.2	Underexplored Adaptation Techniques	198
A	Additional Methodological Details For Chapter 3	203
A.1	Additional Details of the Case Study with a Recycling Robot	203
A.1.1	Custom End-Effector Attachment Design for the Stretch RE-1 robot	203
A.1.2	Detection of Picked Objects	205
A.1.3	Hit Planner	207
A.2	Properties of the Dataset	210
A.3	Implementation Details for Our Method	211
A.4	Design Considerations of Our Method	212
A.5	Additional Results	214
A.5.1	Offline Evaluation Results	214
A.5.2	Real Robot Evaluation	216
A.6	Discussion	220
A.6.1	Additional Intuitive Explanation of Our Method	220
A.6.2	Assumption about the Human Selecting One Category of Objects	221

B	Additional Methodological Details For Chapter 6	222
B.1	BALI Prompts	222
B.2	Setup, Recipes and Preferences used in the Collaborative Cooking Task	230
B.3	Implementation Details and Additional Statistics	247
	Bibliography	250

Perhaps the greatest paper of all is the walks we took along the way. This dissertation is dedicated to everyone who gave me a head pat during my time here at Yale.

Acknowledgment

First and foremost, I am deeply grateful to my advisor, Brian Scassellati (Scaz). Scaz, thank you for teaching me how to become an independent researcher — how to identify research questions that matter, think both broadly and deeply about problems, and communicate ideas clearly and effectively. I am especially thankful for the opportunity you gave me early on, and for trusting me with ambitious, unconventional ideas. You consistently encouraged me to pursue directions that initially seemed too big or too risky, allowed me the freedom to explore ideas beyond the scope of this dissertation, and supported me throughout, no matter how unconventional those ideas were. Your confidence in me and in the value of exploratory thinking shaped not only this dissertation, but how I approach research more generally.

I am also deeply thankful to Marynel Vázquez for her guidance and mentorship throughout my PhD. Marynel, your patience, steady encouragement, and unwavering optimism grounded me through the many highs and lows of research. Your practical advice — about research, career decisions, and life beyond academia has helped me develop confidence, perspective, and resilience as a researcher. You consistently offered thoughtful guidance with generosity and care, and your trust in my abilities meant more to me than I can fully express. I will always be grateful for the example you set as both a mentor and a role model.

I am grateful to my committee members, Tesca Fitzgerald and Tom Silver, for their feedback on my work and for many conversations that helped me think more

clearly about long-term research directions and professional paths. I also thank Danny Rakita for sharing his insights and experience, which were invaluable as I made key decisions about my career trajectory. I am additionally thankful to Dražen Brzčić and Chayan Sarkar for the deep research discussions during their visits and in the time since, which directly strengthened several of the papers that contributed to this dissertation.

I am especially grateful to the mentors who shaped my path before this PhD began. Rod Grupen first introduced me to robotics and to research as a way of thinking, and that early exposure changed the course of my career. Madalina Fiterau and Tauhidur Rahman gave me one of my first real opportunities to contribute to research, trusted me with meaningful work, and helped me build the confidence to see myself as a researcher. I would not be here without the opportunities and support you gave me at the very beginning.

I thank all my friends and colleagues in the Social Robotics Lab, Interactive Machines Group, and Inquisitive Robots Lab for the many conversations and collaborations over the years. I especially thank Tim Adamson, Emmanuel Adeniran, Nick Georgiou, Kayla Matheus, Fern Limprayoon, Moon Hwan Kim, Kate Candon, Nicole Salomons, Jake Brawer, Meying Qin, Nathan Tsoi, Sydney Thompson, Austin Narcomey, Qiping Zhang, Sasha Lew, Ulas Berk Karli and Anjiabei Wang for being wonderful collaborators and close friends, and for the many fun memories we share — from long research discussions to traveling and attending conferences together.

I have had the privilege of working with many dedicated undergraduate students over the years, and I thank them all for their contributions to my research. I especially thank Oz Gitelson and Michal Lewkowicz, without whose contributions much of the work in this dissertation would not have been possible. I also thank Shannon Yasuda, Kaleb Gezahegn, John Dallard, Joyce Duan, David Dong, Tom Wallenstein, Lucas Shepard, Clara Schwab, Yoony Kim, Coco Sack, Emma Adams, Andy Cheng, Julian

Lee, Tran Doan, Abigail Tramer, Natnaree “Proud” Ua-Arak, and Paula Enriquez.

I am grateful to the departmental staff for their steady support throughout my PhD. I especially thank Judi Paige for her consistent help with conference travel and reimbursements over the years. I also thank Will Tuttle and Ragini Jain for always helping me navigate registration requirements and departmental logistics. I truly appreciate how smoothly you kept things running behind the scenes.

I am deeply grateful to my friends Rishabh Mahajan, Deeksha Prabhu, Sneha Bhattacharya, Aliya Quereshi, Vijeta Deshpande, Deep Chakraborty, Sreenivas Venkabarao, Sandesh Shetty, Dipti Bhawe, Rishabh Dhoke, Maithili Deppte, Asmit Jain, Surya Teja Devarakonda, Mili Shah, Rheeya Uppal, Rohit Narasimhan, Rutuja Pai, Ajinkya Zadbuke, Suhrid Bhat, Ketaki Joshi and Ajitesh Chandak — for keeping me grounded and sane through all these years. Your presence, humor, perspective, and constant support made the difficult moments manageable and the good moments even better. Thank you for the late-night conversations, the much-needed distractions, and for reminding me that there is always more to life than deadlines and papers. I would not have made it through this journey without you.

I am grateful to my extended family, especially Sumanta Ghosh, Paramita Ghosh, Sumita Ghosh, Riley Murray, Shayan Ghosh, Grace Huang, Arnab Sinha, Shaoni Sinha, Rup Narayan Sinha, Ranjan Sinha, Chandni Sen Sinha, Rishi Roy, Rashi Roy and Sayanti Basu whom I have had the joy of visiting over the years and who have generously hosted me during vacations. I always value the time we spend together. I also want to acknowledge my family back in India for remaining an important part of my life despite the distance. To my grandmother, Snigdha Ghose, thank you for being a constant presence in my life. To my parents, Soumitra and Debjani Ghose, and to Mukesh and Ketaki Desai, thank you for your sacrifices, your steady belief in me, and for making it possible for me to pursue this path.

Finally, to my human, Shasvat Desai. Thank you for making this PhD feel like a

group project – in research and in life. You have been my sounding board, my code debugger, my editor, my reality check, and my constant support. None of this would have been possible without you.

Funding I gratefully acknowledge the funding that supported the research which appears in this dissertation. In particular, I acknowledge the generous support from the National Science Foundation and the Office of Naval Research.

List of Figures

3.1	Simplified real-world recycling setup	53
3.2	Our Approach. Data pairs are constructed using randomly applied data augmentations. The extracted features are fed into three jointly trained heads. The Instance Projection Head projects the features into a space where each row denotes an augmentation of an image and uses the instance loss to minimize the distance between two augmentations of the same image. The Cluster Projection Head projects feature vectors into a space where each column denotes each instance’s cluster assignments, and the cluster loss minimizes the distance between instances belonging to a cluster. Finally, the Human-Supervised Head minimizes the distance between each human-selected example. . . .	58
3.3	Experimental Setup. The human and the robot work together on the recycling conveyor belt to sort the recyclables.	62
3.4	Sample Images from our Offline Dataset: Ten sample images from each of our ten object categories.	64
3.5	Results of our Offline Evaluation: Comparison with Baselines and Ablation Study aggregated over 3 human-selected pools per category	66

3.6	Impact of augmenting human supervision to contrastive learning: Difference in performance between self-supervised learning augmented with human supervision and the self-supervised learning model trained independently for instance loss, cluster loss, and contrastive clustering, respectively across categories. Specifically, w.r.t. the experiment numbers in Figure 3.5, a) represents Experiment 4 - Experiment 2, b) represents Experiment 5 - Experiment 3, c) represents Experiment 7 - Experiment 6.	67
3.7	System Architecture	69
3.8	Model performance in real-time.	71
4.1	Critical Decision Point Computation. For every policy of the human in the policy bank, we compute what states in the environment produce the most divergent actions. For instance, if there were four reasonable policies the human could be following, at s_1 , all four of them produce different actions, making s_1 a Critical Decision Point (as compared to s_2 and s_3 where most policies predict the same action).	86
4.2	Robot reducing uncertainty about the Human’s Goal at a CDP: As shown by the initial state, ingredients and utensils common to salad and stew are collaboratively brought by the human and the robot in the workspace. The robot reasons about possible next actions to influence the human to reveal their goal. The robot takes an action to complete gathering all the ingredients, rather than adding them to the pan or bowl, to observe the human in a CDP. This allows the human’s next action (either adding the kale to the pan or the bowl) to reduce uncertainty about the goal. The human clarifies their goal through their action, and the robot takes action to support the human.	88

4.3	Selecting actions to infer the human’s goal at a CDP for a human-robot collaborative cooking task: A Receding Horizon Planning (RHP) tree of all possible actions the human and the robot can take in a turn-based interaction is expanded, where the human takes the first turn. The circles correspond to states, and the arrows correspond to the possible action sequences taken at a given time step. The color of the circles indicates the possible goals that can be reached from a given state. The belief distribution is shown at steps t_0 and t_2 , as the robot updates its belief over the human’s goals when a human action is observed. The green arrows show the actions that would optimize the cost function in Eq. 5.3.2. See the text for more details.	93
4.4	Setup for our Physical Robot Experiments for the Human-Robot Collaborative Cooking Task	100
4.5	Results of the Collaborative Cooking Evaluation: Comparison of performance metrics between the proposed approach, vanilla CDP [92], Information Gain Maximization [230], and Bayesian Delegation [292] in the collaborative cooking simulation. Metrics include a) the first correct guess, b) the last incorrect guess, c) the number of extra steps, and d) the percentage of correct guesses. The means and standard deviations of the metrics are calculated over 30 recipes and evaluated across three trials.	104
4.6	Ablation Study over the components of the Cost Function for the Collaborative Cooking Task	106
4.7	Ablation Study on using temporal modeling for computing CDPs for the Collaborative Cooking Task	107

4.8	Collaborative Cooking Case Study: Results of the Case Study with the Physical Robot running Bayesian Delegation, Vanilla CDP (history agnostic), Information Gain Maximization, and Our Proposed Algorithm: For Case Studies, a) and b), a group of columns represents the method. For each group, the first column depicts the agent taking action, the second column depicts the action sequences taken by the respective method, and the third column denotes the presence and position of the goal recipe in the top-3 most probable recipes from the goal prediction model (left-most being the most probable recipe).	108
4.9	Example RHP tree rollout for influencing the robot in a competitive hide and seek game: The green (seeker) and blue (hider) robots play hide-and-seek on a grid with brick-wall obstacles. The seeker uses RHP to steer the hider toward a Critical Decision Point (the red cell) so its next move will reveal its hidden strategy and speed capture. At each timestep, it simulates the hider following fixed policies out to horizon t_4 and chooses the action that best reduces goal uncertainty. At t_1 , both follow their underlying plans (shown by the black arrow). When the hider spots the seeker at t_2 , it detours evasively at t_3 , then resumes its original path at t_4 , landing exactly on the CDP unobserved and allowing the seeker to infer its strategy and catch it. The rollout identifies the optimal action sequence $a_2 \rightarrow a_1 \rightarrow a_1$ (green arrows), and by RHP the seeker executes the first move a_2 before replanning. See text for more details.	118

4.10	Test Environments and Computed CDPs for the Competitive Hide and Seek Task: (Top) Photo-realistic Simulated and real-world environments for the hide-and-seek task. (Bottom) Heatmaps of Critical Decision Points for corresponding environments. States closer to red denote the most Critical Decision Points, and states closer to blue denote the least critical points. Note: Although the figure displays Critical Decision Points for every state, typically, it is sufficient to compute them in a small region around the robot.	119
4.11	The robot seeker (S) influences the human hider (H) towards a Critical Decision Point (CDP) in a hide-and-seek game: a), b) The human-controlled hider sees the robot seeker on its way to its goal while following its high-level strategy. c) The robot seeker actively influences the human-controlled hider to escape towards a CDP (gray area). d) When the hider is at the CDP, the robot seeker optimizes for not being seen by the hider. So, the hider continues to navigate toward its goal. The robot seeker sees the hider take action toward its goal and identifies its strategy.	120
4.12	Results of Competitive Evaluation: Mean and Standard error of timesteps taken by the seeker to catch the hider for each experiment aggregated across all hider policies for a given environment and number of trials (lower is better)	126
5.1	Robot reasoning if the person has changed their mind about making the salad, after observing that the person has picked chocolate. . . .	132

5.2	Steps to Select Robot Action after Goal Change Detection	1.
	Detect Goal Change if the actions taken during the collaboration don't match any sequences in $\mathcal{P}$.	
	2. Create potential action sequences by concatenating the action at which the goal change was detected with past actions sorted by increasing generality scores.	
	3. Check if the potential action sequences are plausible.	
	4. For all plausible action sequences, expand the RHP trees.	
	5. Compute attractor field cost and select an optimal next robot action.	143
5.3	Comparison of performance measures for the proposed method, CDP [92], Information Gain Maximization [230], and Recursive Bayesian [137] in a collaborative cooking simulation. Metrics include (a) first correct guess after a goal switch, (b) last incorrect guess, (c) number of mistakes, and (d) percentage of correct guesses. Results are shown for three simulated human types: stubborn (no goal change), optimal (stochastic goal updates), and suboptimal (stochastic updates with mistakes). In the stubborn condition, all four methods are compared; in the other two conditions, our method is directly compared with Recursive Bayesian, while the red and blue lines indicate the empirical performance bounds provided by CDP and Information Gain Maximization, respectively.	147

5.4	Results of the Case Study with the Physical Robot running Recursive Bayesian [137] and Our Proposed Algorithm: A group of columns represent the method. For each group, the first column depicts the agent taking action, the second column depicts the action sequences taken by the respective method, the third column denotes the ground truth human goal, and the fourth column the presence and position of the current ground truth goal recipe in the top-3 most probable recipes from the goal prediction model (left-most being the most probable recipe).	153
6.1	Illustrative cooking task. (1) Human says “ <i>I want a quick Italian dinner</i> ” and boils water; the robot considers options that fit this preference (pasta vs. rice) and leans toward pasta. (2) Robot fetches pasta. (3) Human adds pasta to the pot, reinforcing a pasta goal. (4) Robot is uncertain about which pasta dish is intended since several variants are possible with the available ingredients, so it asks, “ <i>Do you want red or white sauce?</i> ” Human replies “ <i>red,</i> ” and the robot fetches a tomato as a result. (5) Human blends the tomato to make sauce. (6) Robot serves red-sauce pasta, completing the collaboration.	159
6.2	BALI for goal prediction: Human preferences and actions are summarized (orange) and used for Goal Inference (green) to update an estimate of plausible goals. The Ask Question module (purple) triggers clarifications when goal uncertainty exceeds a threshold. The planner (pink) expands action sequences under receding-horizon planning: at each non-leaf node, the Valid Action Filter (blue) prunes infeasible actions (shown greyed out). The cost function module (yellow) computes an attractor-field cost that links actions to goals and guides the search toward trajectories aligned with plausible human goals and preferences.	166

6.3	Interaction Summary module with an Example	168
6.4	Goal Inference module with an Example	169
6.5	Valid Action Filter module with an Example	170
6.6	Attractor Field Cost Function module with an Example	171
6.7	Experimental Setup for the Collaborative Cooking Task	174
6.8	Results for the open case. (a) shows inference timing: maroon = time until first correct guess, yellow = period of instability until the last incorrect guess, green = stable correct phase. cross = first correct guess ($\downarrow$ =better), circle = last incorrect guess($\downarrow$ =better). (b) reports Top-1 (yellow) and Top-3 (green) goal prediction accuracy ($\uparrow$ =better). (c) show average mistakes or extra steps ($\downarrow$ =better). Means and standard deviations of metrics are computed over 967 preference combinations. Visualization of standard deviations for the first correct guess and the last incorrect guess is skipped for clarity.	181
6.9	Results for the closed case. (a) shows inference timing: maroon = time until first correct guess, yellow = period of instability until the last incorrect guess, green = stable correct phase. cross = first correct guess ($\downarrow$ =better), circle = last incorrect guess($\downarrow$ =better). (b) reports Top-1 (yellow) and Top-3 (green) goal prediction accuracy ($\uparrow$ =better). (c) show average mistakes or extra steps ($\downarrow$ =better). Means and standard deviations of metrics are computed over 967 preference combinations. Visualization of standard deviations for the first correct guess and the last incorrect guess is skipped for clarity.	182
6.10	Real World Case Study	185

A.1 Interaction Flow with System Architecture:	In our interaction flow, a human selects some items from a given category from a conveyor belt. The system estimates the similarity of each remaining item on the conveyor belt. Finally, the robot selects objects most similar to the human-selected objects and sends a push/pull command to the robot. Each colored zone (and its corresponding number code) shows the system components deployed during different phases of the interaction flow.	204
A.2	Experimental Setup with Field of View of each camera highlighted . .	205
A.3	Design of our custom attachment for pushing and pulling objects off the conveyor belt	206
A.4 Visualization of Different Scenarios for the Hit Planner:	a) In Scenario 1, the removal of object 1 would inadvertently remove object 5, which has a relatively low similarity score. The hit planning algorithm would negate the score contribution of object 5 to object 1's hit score as explained in A.1.3 and determine that object 4 by itself has the highest hit score, thereby initiating a push action sequence. b) In Scenario 2, the removal of object 1 would likely remove object 2, which has a relatively high similarity score and would only increase object 1's hit score as explained in A.1.3. In this instance, the robot would target object 1 with a pull action sequence and target object 4 with a pull action sequence once again after resetting the telescopic arm to a central position.	208
A.5	Data distribution in our offline dataset	211

A.6 Comparison of t-SNE Visualization of the learned representations on the offline dataset:	
a) t-SNE plot for the self-supervised contrastive clustering [162], b) t-SNE plot for our approach. Images with yellow border denote the objects selected by the human. In our approach, clusters for each category show better separation than the contrastive clustering approach, and clusters are overlapping when an increasing number of features are common.	215
A.7 Impact of augmenting human supervision to contrastive learning (Accessible Version):	
Difference in performance between self-supervised learning augmented with human supervision and the self-supervised learning model trained independently for instance loss, cluster loss, and contrastive clustering, respectively across categories. Specifically, w.r.t. the experiment numbers in Figure 3 in the main paper, a) represents Experiment 4 - Experiment 2, b) represents Experiment 5 - Experiment 3, c) represents Experiment 7 - Experiment 6. (Best viewed on a PDF processor with zoom)	218
A.8 Sample Qualitative Results from our Robot System:	
Similarity scores of objects on the conveyor belt obtained from our method, when the human trained the system with 30 objects of type a) half-gallon milk jugs b) un-crushed cans	219
A.9 Accuracy of Our Method in Real Time	220

List of Tables

4.1	List of Recipes that the Human and the Robot could collaboratively cook in our experimental setup & the required ingredients.	96
5.1	List of Recipes that could be part of the combination that the Human and the Robot could collaboratively cook in our experimental setup & the required ingredients.	146
6.1	Paired t-tests comparing all BALI variants to all baselines across five metrics ($N = 967$). Lower is better for Mistakes, First Correct Guess, and Last Incorrect Guess; higher is better for Top-1 and Top-3 Accuracy.	184
A.1	Hyperparameters for Training Our Method	212
A.2	Impact of the number of clusters (K) on overall model performance .	214
A.3	Results of our Offline Evaluation (Accessible Version): Comparison of F1 Scores with Baselines and Ablation Study aggregated over 3 human-selected pools per category.	217
A.4	Performance Statistics of our Method on the Real Robot System . . .	219
B.1	Recipes, ingredients, and appliances/utensils.	230
B.2	Preference-to-Recipe Mapping	233
B.3	Hyperparameters for Goal-Prediction Modules.	248
B.4	Hyperparameters for MPC Tree Search and Action Filtering.	248
B.5	LLM Interface Hyperparameters and Defaults.	249

Glossary

API Application Programming Interface. 69, 205

BALI Bidirectional Action-Language Inference. 8

CC-HTNs Clique/Chain Hierarchical Task Networks. 98

CDP Critical Decision Points. 7

CLIP Contrastive Language Image Pretraining. 101

GOOD Goal Inference from Open-Ended Dialog. 178

LLM Large-Language Model. 172

MDPs Markov Decision Processes. 80

MPC Model Predictive Control. 80

MRF Material Recovery Facility. 54

NASA TLX NASA Task-Load Index. 46, 47

PDDL Planning Domain Definition Language. 97

POMDP Partially Observable Markov Decision Process. 21, 22

RHP Receding Horizon Planning. 78

SAM Segment Anything Model. 101

SEAN Social Environment for Autonomous Navigation. 121

SLAM Simultaneous Localization and Mapping. 125

SUS System Usability Scale. 47

VLA Vision-Language-Action. 200, 201

Chapter 1

Introduction

One of the central visions of robotics research is to integrate robots into everyday life, shifting from robots that excel in structured, pre-specified settings to robots that can support people in unstructured, everyday environments. In this vision, robots function ubiquitously. At work, they assist with complex unstructured tasks. In healthcare facilities, they augment patient care and clinical workflows. At home, they take on everyday chores that improve quality of life.

The status quo for real-world robotic systems stands in stark contrast to this vision. Today’s deployed robots excel in predictable, engineered environments where tasks are predefined, and operating conditions remain static. For example, in an automotive assembly line, a manipulator can weld a vehicle chassis with sub-millimeter precision because the task specification, workspace geometry, and allowable interactions are explicitly defined and kept consistent. This paradigm works when the world around the robot is constrained, meaning the workspace is structured and people are either absent or interact with the robot in tightly prescribed ways. In everyday settings, by contrast, robots must operate amid people whose goals, preferences, and constraints can vary across situations and over time, and success cannot depend on the environment, or its human partners, being engineered to match a fixed set of

assumptions.

This reliance on explicit structure highlights a key barrier to deploying robots in human-centric environments which is achieving shared understanding when working with people. When working as a team, the robot’s internal belief about the human’s current goal must match the goal the human is actually pursuing [34, 255]. When two people work together on a shared task, it is natural for them to act without repeatedly verbalizing their goals. Goals are typically left unstated and inferred from context by the partner. When working with a robot, it would be ideal if the robot could infer the human’s goal from implicit evidence in the human’s actions and the evolving task context. When that inference is correct, the robot’s assistance complements the human’s efforts. When it is wrong, the robot can perform actions that are counterproductive for the joint task. For example, consider a person and a robot collaborating to set up a workspace for an upcoming project. The person spreads several tools on a table and begins grouping them. If the robot infers the goal as “clear the surface” (or “put away loose items”), it may collect those items and store them. However, since the human’s actual goal is to sort and stage materials for the next step, this action removes tools that are intentionally laid out to support the task, interrupting progress and forcing the human to recover. The failure occurs due to a breakdown in shared understanding.

Prior work on establishing shared understanding between a human and a robot often requires the human to make the goal explicit through repeated instructions or demonstrations [57, 82, 248], which can be impractical in real-world interactions. This dissertation instead argues that the robot should assume the burden of maintaining shared understanding. To that end, we develop approaches in which the robot uses goal inference to align with the human’s intent from the signals already present in interaction, requiring minimal additional intervention. We show that accurate goal inference enables the robot to take timely, goal-directed, supportive actions that

reduce the human’s workload, rather than waiting for explicit commands, without requiring the human to continually supervise the robot.

1.1 Challenges for Shared Understanding with Minimal Supervision

When a robot attempts to establish shared understanding with people without requiring frequent supervision, two recurring challenges arise: (1) *inferring intent from observations of human behavior*, where the same observed verbal or non-verbal human behavior can admit multiple plausible interpretations depending on context, and (2) *adapting under non-stationary human objectives*, where goals and preferences can shift over the course of an interaction, requiring the robot to detect and respond to changes with or without explicit re-instruction.

1.1.1 Inferring Intent from Observations of Human Behavior

In many collaborative tasks, a person’s (both verbal and non-verbal) behavior is often not a clean “label” of a single intent for their robotic partner. People typically act and communicate in ways that are sufficient for another person who shares context, but those same signals can leave multiple interpretations open for a robot which lacks that same understanding of context. As a result, the robot must infer intent from partial evidence and decide how strongly any observed action or utterance should update its belief about the human’s goal.

The same action can mean different things depending on when it occurs in the task, what has already happened, and what constraints are currently in play. For example, if the human’s goal were to make a specific recipe, reaching toward an ingredient in a shared workspace could reflect (i) commitment to a particular goal, (ii) a preparatory step that supports many goals, (iii) a probe to check availability or

state, or (iv) a coordination action intended to guide a partner rather than directly advance the task. This creates a key inference problem. The robot must decide which observed actions should be treated as strong evidence about the human’s goal, and which are better interpreted as broadly compatible steps that do not yet disambiguate what the person is trying to do. This issue is common in longer tasks where different long-term goals share overlapping early action sequences. In a kitchen, early actions such as retrieving a bowl, clearing space, or washing produce are compatible with many recipes and therefore constrain the goal set only weakly until a more distinct action occurs, such as selecting an unusual ingredient or appliance. For the robot to provide timely support, it must recognize when observed behavior meaningfully narrows the goal set and when it does not.

Language exhibits the same property. People often communicate at a level that works for human partners because it relies on shared context and common conventions, leaving the listener to fill in missing details. Directives like “set up for the next step” specify an intended effect but not which objects should be staged, where they should go, or which continuation counts as the next step when several are plausible. “Do it like last time” refers to a prior episode but does not specify which aspects should match, such as the final arrangement, the sequence of actions, or the division of labor. Even “hand me that” can be unclear when multiple objects are present or when timing matters. The robot must determine what each utterance implies, what it leaves open, and how it should combine language with the ongoing action context to infer the human’s intent.

A further difficulty is that the robot observes only what the human chose to do or say, not the alternatives they considered or the outcomes they were trying to avoid. This missing decision context makes it hard to infer what is essential versus incidental. For example, if a person repeatedly places tools in a particular spot, the robot cannot tell whether that placement is required for the task, preferred for safety

or reachability, or simply a convenient byproduct of the current layout. The robot must therefore infer intent from behavior that is informative but incomplete, using context to interpret actions and language without requiring the human to describe every detail.

1.1.2 Adapting under Non-Stationary Human Objectives

Non-stationarity arises when the human partner’s goal changes while the task is underway. The change may be prompted by the environment (e.g., a missing resource, a new time constraint) or by the person revising what outcome they desire. In either case, a goal that was consistent with the person’s earlier behavior may no longer be the goal that best explains what they are doing now. When this happens, assistance that was helpful moments ago can quickly become unhelpful. For example, if the robot has been arranging tools or materials to support an inferred plan and the person switches to a different plan, the robot’s continued arrangement can put items in the wrong places or make the next step harder for the person.

Goal changes create a structured inference problem for the robot. The robot must first determine whether the human’s objective has changed, and it often cannot rely on an explicit announcement. Instead, it must interpret indirect evidence such as shifts in object selections, changes in the order of steps, or brief high-level directives. Even when such evidence appears, it is rarely decisive at the moment it occurs because different goals can share substantial portions of the same action sequence, so the initial actions after a change may still be compatible with both the prior and current goal. At the same time, a change in behavior does not necessarily imply a change in goal as people may reorder steps, take a shortcut, or respond to a temporary constraint while still pursuing the same objective. As a result, the robot must decide when observed differences justify updating its goal belief versus when they reflect ordinary variability within the same goal, and do so quickly enough that its assistance remains useful.

1.2 Contributions

This dissertation asks:

How can a robot maintain shared understanding with a human partner in long-horizon tasks by inferring the human’s goal and acting in a manner consistent with the human’s preferences and constraints, using the signals naturally available during interaction, without requiring frequent supervision or repeated explicit instruction?

To address this question, we tackle the two challenge dimensions described above by developing methods that enable a robot to use the signals available during interaction to infer intent and keep that inference aligned with the human over time. The first, second, and fourth contributions aim to address the challenge of inferring intent from observations of human behavior. The first contribution enables a robot to learn task-relevant visual representations from sparse object selections so that a human-defined target category becomes separable for downstream classification. The second contribution enables a robot to plan its actions that influence humans to states in their shared environment where different possible human goals predict meaningfully different next actions, so that observing the human at those states provides more informative evidence for goal inference. The fourth contribution enables a robot to jointly interpret what a person says and what they do by using language, action history, and task context to infer goals from underspecified directives, including cases where the intended goal is not restricted to a predefined set. The third contribution addresses the challenge of adapting under non-stationary human objectives by enabling a robot to detect when observed behavior is no longer consistent with its current goal prediction and to update its goal belief to recover alignment.

1. **For inferring a human-defined object category from partial behavioral evidence:** We introduce an online approach that treats a person’s object

selections during a shared task as sparse evidence of what visual distinction matters, and learns a representation that makes that distinction separable for downstream classification [93]. The key challenge is that, in realistic interaction, the robot observes only the objects the person chooses to act on, and cannot assume that non-selected objects are negatives. To address this, our method integrates human-provided positives into contrastive representation learning by adding a loss that pulls the representations of the human-selected examples together in latent space, aligning the learned representation with the human’s task-specific requirement. The resulting representation supports learning a classifier for the desired category from few examples and can be updated online as additional selections are observed.

2. For resolving goal ambiguity through informative interaction states:

We introduce Critical Decision Points (CDP) as states in the environment where the optimal policies for different goals maximally diverge. If the robot were to observe the human take an action from a CDP, it would make the human’s next action especially informative about the goal being pursued [92]. This formulation is agnostic to the interaction type, applying in both cooperative and competitive settings. To leverage these states, we use receding horizon planning [172] to look ahead and select actions that steer the interaction toward a CDP while maintaining task progress, enabling earlier disambiguation without interrupting the task for explicit queries.

3. For adapting to non-stationary goals during interaction:

We introduce an approach for goal change detection that identifies when observed behavior deviates from the robot’s predicted goal [87]. To infer the updated goal, our method selectively retains past information that remains relevant in the new context to seed a revised belief, rather than discarding the full interaction

history. It then executes differentiating actions—supportive moves designed to both advance the task and elicit behavior that helps validate the revised goal—allowing the robot to realign more quickly than approaches that assume goals remain fixed.

4. **For inferring and acting under open-ended goals from underspecified human directives coupled with behavioral cues:** We introduce Bidirectional Action-Language Inference (BALI) for goal prediction [90]. BALI uses large language models to interpret ambiguous language in conjunction with task history and observed actions to infer the human’s goal, enabling the robot to provide appropriate support even when the goal lies outside a predefined set.

1.3 Thesis Outline

This dissertation develops mechanisms that allow robots to maintain shared understanding with human partners by inferring and adapting to human goals, preferences, and constraints through natural interaction, without placing the burden of frequent supervision on the human.

We begin in Chapter 2 with a review of existing approaches for enabling robots to adapt to people online through interaction in human-robot collaboration. In this chapter we provided a broad overview of the challenges motivating our work. In Chapter 2 we examine how prior systems have attempted to learn from and adapt to human partners during collaboration, identifying key limitations that our work addresses.

After this review, Chapters 3-6 are ordered by increasing levels of human intervention required for the robot to adapt: we begin with adaptation from passive observation, move to adaptation through active influence in interaction, and then consider adaptation from ambiguous high-level verbal guidance.

Chapter 3 focuses on adaptation from passive observation. We describe a method for tailoring visual object representations to human requirements through passive observation. We show how contrastive learning can enable a robot to learn user-specific visual concepts online from sparse positive feedback, allowing it to infer goals even when they depend on arbitrary, user-defined attributes not captured in standard perceptual representations.

Chapters 4 and 5 then consider multi-step tasks where it would be impractical for the robot to wait for the human’s goal to become clear to provide adequate assistance. So, we propose that the robot use its actions to obtain more informative evidence during the task. The work in Chapter 4 introduces Critical Decision Points (CDPs), states at which the human’s actions are maximally informative because different goals induce divergent optimal behavior. We use Receding Horizon Planning to steer interactions toward CDPs through robot actions, and evaluate the approach in both long-horizon collaborative tasks and competitive scenarios, showing that goal ambiguity can often be resolved earlier without requiring explicit instruction. Chapter 5 extends this active setting to non-stationary goals. We present an approach that detects when a human partner has shifted their objective mid-task and introduce Differentiating Actions that enable the robot to efficiently determine the new goal by selectively reasoning about past information that remains relevant in the new context.

Chapter 6 tackles open-ended goal inference from people’s actions and high-level directives conveyed using natural language. We introduce BALI (Bidirectional Action-Language Inference), an approach that leverages large language models in conjunction with observed human actions and task history to infer and adapt to goals that lie outside a robot’s predefined knowledge base.

We conclude in Chapter 7 with a summary of the work presented in the dissertation, followed by limitations and directions for future work in building robots that can function in human environments.

Chapter 2

Adaptation in Physical Human-Robot Collaboration: A Review

As discussed in the previous chapter, effective human–robot collaboration requires a robot to maintain shared understanding with its human partner over extended interactions, so that it can act in ways that respect human preferences, constraints, and evolving task context. This chapter surveys prior work on *adaptation* in human–robot collaboration, focusing on how robots update their behavior during interaction rather than relying solely on pre-defined policies. We review how existing systems adjust robot behavior during an ongoing physical collaboration in response to changes in human actions, state, and task context, and organize this literature along key design dimensions that shape adaptation during execution. Through this review, we highlight recurring assumptions and limitations in prior approaches that motivate the methods developed in the remainder of this dissertation.

2.1 Introduction

This survey focuses on adaptation in physical Human–Robot Collaboration (pHRC), specifically settings in which a human and a robot must coordinate their physical efforts, operate within a shared workspace, and work toward a common task goal. We review how prior systems support such coordination by observing human behavior during interaction, interpreting these observations to model the state of the collaboration, and adapting robot behavior to better support joint task execution. Consider a collaborative assembly where the human reaches for a part before fully clearing the workspace, or changes the order of steps midstream to accommodate a momentary constraint. A robot that executes a fixed action sequence can still be “correct” with respect to its original plan, yet become an unreliable teammate. It may present parts too early, obstruct the human’s motion, apply assistance at the wrong time, or hesitate during handoffs, producing delays, awkward recoveries, or safety-critical interference. In pHRC, success is therefore not determined solely by task completion, but by whether the robot remains synchronized with a partner whose actions and intentions evolve over the course of execution [105].

As robotic systems enter industrial, medical, and domestic settings, the ability for robots to physically cooperate with humans is becoming increasingly important. Unlike autonomous operation in static environments, where a robot can execute a plan in isolation, collaborative tasks in the real world require the robot to function as a responsive teammate. Across a range of collaborative tasks, the robot’s effectiveness as a collaborator depends on its ability to coordinate with the human partner in real time [105]. Human partners introduce significant uncertainty through changing plans, evolving preferences, fluctuating physiological states, and natural variability in motion [121]. Consequently, if a collaborative robot relied on pre-scripted or rigid behaviors, it would eventually fail to coordinate effectively, leading to reduced fluency, interaction breakdowns, or safety risks.

To support seamless collaboration, the robot requires **adaptation**, the ability to adjust its behavior *during* the task to accommodate the specific and evolving needs of its human partner. Evidence from physical human–robot collaboration also supports the need for adaptation during interaction. Robots that adjust their behavior in response to human physical state, preferences, or reactions—such as reallocating roles when a human becomes fatigued [213] or *inferring a new partner’s collaboration style/preferences online (e.g., among learned user types) and selecting assistance accordingly* [197]—have been shown to improve task performance as well as perceived anticipation and comfort. More recent systems show that continuous adaptation during interaction can support trust and acceptance while maintaining safety and efficiency, especially in settings where human behavior varies or is difficult to model in advance [268, 176]. In these scenarios, the value of adaptation lies in the robot’s ability to remain responsive as the interaction unfolds and adjust its behavior based on ongoing interaction.

2.1.1 Method, Scope and Organization of the Survey

Focusing on a little over a decade of research (2015-2026) in physical human-robot collaboration, this survey examines a robot’s adaptation to human behavior in a variety of co-located physical collaborative tasks, where the robot must update its actions at execution time to advance the collaboration. To compile the corpus of literature for this survey, we searched the proceedings of the following major conferences and journals within the robotics and human-robot interaction communities: *ACM/IEEE International Conference on Human-Robot Interaction (HRI)*, *IEEE International Conference on Robotics and Automation (ICRA)*, *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, *IEEE Robotics and Automation Letters (RA-L)*, *IEEE Transactions on Robotics (T-RO)*, *IEEE International Symposium on Robot and Human Interactive Communication (RO-MAN)*, *ACM Transactions on*

Human-Robot Interaction (T-HRI), *Conference on Robot Learning (CoRL)*, *International Journal of Robotics Research (IJRR)*, *Autonomous Robots*, *Robotics and Autonomous Systems (RAS)*, *Frontiers in AI and Robotics* and *Robotics: Science and Systems (RSS)*. We then reviewed the reference lists of the identified papers to find additional relevant papers. Our search targeted paper titles and abstracts containing the terms *human-robot collaboration*, *human-robot teaming*, or *shared autonomy*, in combination with the term *adaptation*.

Given the breadth of the literature in this domain, we restrict our scope to dyads consisting of a single human and a single physical robot working in a shared workspace towards a common goal, and exclude works that fall outside the domain of co-located physical collaboration, such as social navigation and remote tele-operation settings. We prioritize approaches in which adaptation occurs during the ongoing interaction, as the human and robot act together and respond to each other’s behavior in real time. Finally, to ensure the survey addresses the challenges of real-world collaborations, we only consider methods validated on real robotic platforms with human partners, excluding works where adaptation is demonstrated solely in simulation environments.

Consistent with this focus, we exclude work that uses structured human-robot collaboration tasks solely to collect data for future adaptation, as well as long-term personalization approaches where adaptation occurs over a longer time period rather than within a single interaction. We similarly do not cover approaches that predict human intent or goals without using those predictions to alter robot actions during execution. Our survey instead centers on methods that close the loop from online observation to immediate behavioral change, reflecting the unique demands of real-time, co-located physical human-robot collaboration.

We structure this review by first analyzing the assumptions the robot makes about its human partner’s behavior in Section 2.2. Inspired by Nikolaidis et al. [195], we distinguish between One-Way Adaptation, where the robot treats the human as an

independent agent whose actions are unaffected by the robot, and Mutual Adaptation, where the robot explicitly models the interaction as a coupled system, accounting for the fact that the human partner is also adapting their behavior in response to the robot. Following these modeling assumptions, in Section 2.3 we categorize the algorithmic mechanisms used to implement responsiveness, covering the planning-based, learning-based and transient behavioral adaptation approaches to enable online adaptation in pHRC. We then examine the information flow between the human and the robot while working on a shared task, detailing how diverse cues inform specific adjustments in the robot’s behavior in Section 2.4. Section 2.5 discusses the various tasks, evaluation settings and metrics used to validate these systems.

2.1.2 Comparison to Other Surveys

In this survey, we focus on the robot’s ability to adapt its physical behavior during task execution. While prior reviews on planning, task allocation, and role assignment in physical human–robot collaboration survey coordination mechanisms broadly, they do not center on how robot actions are modified online in response to observed human behavior within an ongoing interaction [156, 244]. We instead organize the literature around within-episode adaptation, considering methods in which execution-time observations of the human and the shared task directly change the robot’s subsequent actions, from high-level task and role adjustments to continuous changes in trajectory, timing, and synchronization. Similarly, the review of interactive robot learning by Baraka et al. [25] covers learning skills from the human. In contrast, a part of this survey focus on the application of learning to support collaboration, detailing how policies are tuned online to support a human partner during collaboration. Finally, whereas the survey on intention estimation by Hoffman et al. [121] analyzes the perception of human intent, it is not concerned with how the robot uses these predictions to reason about what actions to take. We focus on this complementary action phase,

reviewing how these estimates are mapped to adaptive robot behaviors.

2.2 Human Modeling Assumptions for Online Adaptation

Different approaches make different assumptions about whether human behavior is merely observed by the robot or explicitly anticipated when choosing robot actions. Some approaches treat the human as external to the robot’s action selection process [197, 308, 129]. In these cases, the robot observes human behavior and may infer latent variables such as intent, task state, or progress. These inferences are used to reason about how the robot should adapt its actions. However, when selecting among candidate robot actions, the robot does not explicitly consider how different actions would change the human’s behavior. Human behavior informs the robot’s understanding of the current state of the interaction, but it is not modeled as something the robot actively influences through its choices. Following Nikolaidis et. al’s [195] definition, we call such approaches *One-Way Adaptation*.

Other approaches instead treat the human as responsive to the robot. Here, the robot reasons about how its own actions are likely to affect the human and uses these predictions when deciding what to do next (eg. [197, 192, 92]). For each candidate robot action, the robot predicts a corresponding human response and compares actions based on their expected impact on the joint interaction. In this formulation, the robot’s decision-making explicitly depends on how humans are expected to react, not just on the current observed state. Therefore, as proposed by Nikolaidis et. al [199], we call such approaches *Mutual Adaptation*.

In the context of physical human–robot collaboration, the key distinction is therefore not whether the robot models the human at all, but how that model is used. In one-way adaptation, human models are used only to interpret past or current behav-

ior, adapting reactively once the human’s response is observed. In contrast, mutual adaptation approaches use human models proactively, to evaluate candidate robot actions based on their predicted effects on human behavior. This commitment to action-conditional prediction enables forms of adaptation that account for how humans are likely to respond to robot choices, rather than relying solely on correction after the fact.

This distinction between One-Way and Mutual Adaptation also aligns with ideas from cognitive science on social reasoning, which explain behavior in terms of unobserved goals, beliefs, and plans [71, 96]. Computationally, such reasoning is often formalized through inverse planning, where an observer infers latent goals or beliefs from observed actions under the assumption that agents act approximately rationally with respect to those goals [21].

2.2.1 One-Way Adaptation

In One-Way Adaptation, the robot uses a model of the human to interpret observations and adjust its behavior, but it does not reason about how different robot actions would shape future human behavior [197, 308, 214]. The human model is used to estimate aspects of the interaction such as intent, progress, preferences, or capability, and robot behavior is conditioned on these estimates. Adaptation occurs by reacting to inferred human state rather than by anticipating how the robot’s own actions will influence the human. This modeling assumption appears widely across task domains and adaptation techniques. Despite the differences in task structures, the fact that the robot observes human behavior, updates internal estimates of relevant human variables, and selects actions based on those estimates, stays consistent. Human behavior may change over time, but those changes are treated as arising from the human or the environment, not as outcomes of the robot’s specific action choices.

A key modeling distinction within one-way adaptation is whether the robot treats

the human as stationary within an interaction episode or allows for non-stationary behavior. This distinction reflects whether the robot expects the human to behave in a consistent, predictable way during the episode, or whether it must be prepared for unexpected changes in human behavior that are not attributed to the robot’s own actions.

Stationary human models

Across the literature, the assumption that the human will behave in an expected and consistent way within an interaction episode was common. The robot may track interaction state such as task phase, intent, or progress, but it does not revise the underlying assumptions about how the human behaves.

This assumption is frequently used in systems that synchronize robot behavior with human activity. Many approaches that track people’s intent or task progress infer what the human is doing and adapt robot timing, assistance, or task selection accordingly, while relying on a fixed interpretation of human behavior [308, 129]. Proactive assistance approaches based on progress estimation similarly assume stable human behavior patterns that enable timely intervention [69, 113]. Related assumptions appear in shared autonomy and planning systems where the meaning of human inputs, such as language commands or low-dimensional control signals, is interpreted in a fixed manner within an episode [141, 133, 98].

Non-stationary human models

Under the assumption that the human behavior may be non-stationary, the robot does not assume that the human will continue to behave in a fixed and expected way. Instead, it models aspects of human behavior or state that may change over time and updates its internal estimates accordingly. These changes are not attributed to the robot’s actions, but to shifts in the human’s condition, preferences, or understanding.

Fatigue- and ergonomics-aware collaboration is a representative example. These systems infer changes in physical state from interaction signals and adapt assistance or role allocation as the human’s capability evolves [214, 149]. Trust-aware approaches similarly treat trust as a latent variable that changes during interaction and modulate robot behavior to avoid interference when the human’s trust in the robot is negatively affected [110, 231]. There has been other work where the robot adapts to changes in human preferences, or desired level of autonomy, updating internal models as new evidence becomes available [307, 201, 38, 88]. Learning from physical corrections or demonstrations similarly revises the representation of the human model in response to corrective inputs [18]. Error detection, explanation, and recovery systems adapt based on changes in human awareness or reaction, adjusting robot behavior without modeling those changes as consequences of robot action choices [152, 67, 251].

2.2.2 Mutual Adaptation

In Mutual Adaptation, the robot assumes that its own actions can influence how the human behaves during the task and takes this into account when selecting actions. Rather than treating human behavior as fixed or independent, the robot reasons about how different robot actions are likely to shape the human’s future behavior and chooses actions accordingly.

This assumption appears commonly in shared autonomy settings, where humans and robots act concurrently and continuously respond to one another. Prior work shows that modeling how willing a human is to adapt allows the robot to either guide the human toward a more effective strategy or comply when the human resists change, improving team performance while preserving trust [200, 195]. Related formulations use belief-based or bounded-memory models to capture how humans adjust their strategies over time and allow the robot to guide the interaction toward more efficient coordination [97].

Several works extend mutual adaptation beyond immediate interaction and considers adaptation over repeated or long-horizon interaction. In these settings, robot actions influence how humans explore, learn, or make decisions over time rather than simply responding to the current state. Examples include modeling how assistance can shape human exploration to improve long-term outcomes [47] and framing the robot’s influence as a control problem where the robot influences human behavior in multi-step tasks [233].

Mutual adaptation has also been studied in collaborative planning and task allocation, where roles are not fixed in advance. Cross-training approaches allow robots and humans to alternate roles during training so that each learns how the other adapts, leading to better alignment, trust, and fluency during execution [194]. Prior work has also proposed hierarchical planners that enable robots to select complementary actions based on observed human behavior without relying on explicit communication, allowing coordination to emerge online rather than through predefined leader–follower roles [223]. Other frameworks explicitly adapt robot behavior to individual human traits across interaction scales using belief-space reasoning [97].

Finally, modeling mutual influence does not imply that both agents are always actively adapting. A robot may reason about how its actions affect the human even when the human is not consciously adjusting their strategy. We therefore reserve claims about mutual adaptation dynamics for work that explicitly models or analyzes adaptation on both sides within the interaction, rather than cases where influence is used solely as a robot-side decision-making assumption [200, 195].

2.3 Adaptation Techniques

We organize adaptation approaches based on the mechanism through which robot behavior changes during collaboration. Across the literature, systems differ in whether

adaptation is achieved by planning actions under an explicit model, by updating internal models through learning, by temporarily modulating behavior during execution without changing the underlying model, or by a combination of these mechanisms. These choices affect what information is extracted from human behavior, how quickly can the robot adapt its behavior, and how adaptation interacts with human expectations and task structure. This section reviews planning-based approaches, learning-based approaches, transient behavioral adaptation, and hybrid approaches, highlighting how each class of approaches support adaptation in human robot collaboration and the tradeoffs they impose.

2.3.1 Planning-based Adaptation Approaches

In planning-based approaches, the robot adapts by recomputing plans under an explicit task and interaction model. Adaptation occurs through repeated planning or optimization over task structure, action sequences, or trajectories as new or inferred information about the human’s actions, goals, or internal state becomes available. Differences across methods arise from what is included in the planning model, the abstraction at which planning occurs and what sources of uncertainty are explicitly represented.

A first class of planning-based approaches adapts at the task level using hierarchical representations, where structure is the main resource for online adaptation. The key idea is that a shared task model structures the interaction into identifiable steps and choices, such as which subtask to pursue next or how work should be divided, allowing the robot to reconsider its plan at those moments without reasoning over the full space of possible actions. Ramachandrani et al. [223] instantiate this by selecting supportive actions conditioned on observed human progress within a hierarchical task model, effectively using the hierarchy to map continuous execution signals into discrete replanning updates. Roncone et al. [227] similarly use task structure to re-

evaluate role assignment and task allocation online, enabling the robot to adjust *who does what next* as execution unfolds. Mangin et al. [178] extend this paradigm to settings where progress cannot be observed directly by deriving a Partially Observable Markov Decision Process (POMDP) from the shared hierarchical representation and using an online solver to choose supportive actions and recovery behaviors under uncertainty. These works illustrate that task-level planning is well suited for adaptation when the main uncertainty lies in task progression and role allocation, and when a hierarchical structure provides a compact representation that supports repeated replanning during ongoing collaboration.

A second class integrates task planning with motion planning or trajectory optimization to adapt online when the main challenge is satisfying physical constraints during execution rather than selecting the task itself. In some systems, replanning is required because the form of physical interaction changes over time, such as when responsibility for supporting an object shifts between partners or when contact is made or broken. In others, the interaction form is fixed, but uncertainty about the human’s motion or coordination strategy requires the robot to continuously update the joint trajectory to remain safe and aligned. For example, Gottardi et al. [98] represent the case when the form of physical interaction changes over time by using receding-horizon task-and-motion planning to update both the task sequence and motion plan when execution deviates from an expected collaboration. Similarly, Gienger et al. [94] emphasize interaction mode changes by representing cooperative manipulation with discrete contact-defined interaction modes, updating the mode estimate from force interaction cues, selecting feasible mode transitions, and then generating trajectories to realize the chosen mode under current conditions. On the other hand, Yang and Mavrogiannis [296] propose an approach for keeping the interaction mode fixed while repeatedly re-optimizing trajectories under uncertainty about the joint traversal strategy, trading off task efficiency with reducing uncertainty about the partner’s

behavior.

Planning-based adaptation is useful under goal uncertainty because it lets the robot choose actions while explicitly managing two complimentary requirements of making progress under the current best goal hypothesis, and preserving safety and recoverability if that hypothesis is wrong or changes. The studies in this group mainly instantiate one of two planning structures. One structure maintains an explicit set of competing goal hypotheses and uses receding-horizon lookahead to select actions that are simultaneously supportive and informative, so the robot both helps and reduces ambiguity about what the human is trying to do [88, 91]. The other structure avoids committing to a single action sequence by synthesizing a reactive state-to-action strategy from a temporal logic specification, so behavior remains correct across modeled human behaviors while guaranteeing safety and progress constraints [95].

A further subset of planning-based approaches expands the planning state to include latent human factors that affect how the human will act, so adaptation consists of selecting robot actions that trade off immediate task progress against predicted downstream coordination effects. The common structure is: (i) posit a latent variable describing the human (e.g., preference, trust, fatigue, learning/adaptability), (ii) maintain a belief or estimate of that variable from interaction evidence, and (iii) plan actions that account for how the latent variable will shape future human behavior and may itself be influenced by the robot. Prior work using this paradigm differ mainly in the choice of latent variable and the decision model used to optimize under it which can be preferences over leading or following task allocation [201], adaptability with bounded memory in decision-theoretic planning [195], trust dynamics embedded in a POMDP [52], leader-follower reasoning with additional latent factors such as trust and fatigue in Stackelberg and stochastic game formulations [314, 167], and human learning treated as a latent evolving state that is incorporated into the planning objective [263].

Finally, a separate subset of approaches adapts the structure of the robot’s plan itself, rather than the task, motion, or human model, with the goal of making robot behavior easier for the human to anticipate and work with over long horizons. This is typically expressed as imposing reusable plan patterns to improve anticipability [168] or conditioning task planning on elicited user preferences and instruction context to produce tailored proactive suggestions [101].

2.3.2 Learning-based Adaptation Approaches

Several adaptation approaches in human–robot collaboration use learning-based methods because they allow the robot to adjust how it assists based on what it observes during the interaction, so that its behavior increasingly aligns with how the human acts and what they need as the task progresses. In shared autonomy, for example, the robot must decide when to help and when to stay out of the way, especially when the human’s intent is outside the robot’s current repertoire. Zurek et al. [318] tackle this by having the robot detect insufficiency in its intent set, return control when it is likely wrong, and then learn the new intent from its observation of human behavior during unhindered execution to grow its library of possible human intents over time. As the robot learns to support a larger set of intents, the meaning of the user’s control inputs changes, since the same input may be interpreted differently depending on what assistance the robot is now capable of providing and what the user expects it to do. Similarly, Hoegerman et al. [119] make the dependency between the distribution of the human’s control inputs and the robot’s communication of its learned assistance explicit by showing that when the robot communicates its learned assistance, humans systematically shift when they intervene versus relinquish control. The robot then uses this information to learn how to interpret human control inputs correctly under communication, rather than assuming the human behaves in a fixed manner. These shared autonomy methods focus on inferring intent and interpreting

inputs, but fluent collaboration also depends on low-level control inputs like action timing and synchronization of the collaborating humans. For example, De Lazzari et al. [69], propose an approach called PACE that focuses on timing and fluency by estimating human action completion from motion and using that progress estimate to learn a reinforcement-learning policy that synchronizes proactive robot actions to reduce waiting and improve collaboration flow.

In many collaborative settings, humans physically or verbally intervene when the robot is pursuing the wrong goal or optimizing the wrong aspect of the task. Some approaches use these interventions to update what the robot is trying to optimize, rather than assuming that the robot’s goal is known in advance and only adapting how it executes that goal. Prior work uses human interventions during the task as supervision to update that objective during the interaction, so the robot changes what it is trying to accomplish, and not only how it assists. Bajcsy et al. [19] focus on cases where a person physically intervenes because the robot is not doing the task the way the person intends, and they treat that physical interaction as a corrective signal about the parameters of the robot’s objective. The robot then updates its objective during the ongoing task so that its subsequent actions better reflect what the person is trying to communicate through the interaction, instead of returning to executing its pre-trained task policy. In practice, physical corrections can be ambiguous because a person may unintentionally change multiple aspects of the robot’s behavior while attempting to correct a specific issue. Subsequent work by Bajcsy et al. [18] address this identifiability problem by inferring which single feature the person is trying to modify and then updating one feature at a time, which reduces unintended learning and improves teaching efficiency. In some complementary work, van der Spaa et al. [276] propose an approach to jointly reason about both human intentions and human preferences during physical cooperation. They incrementally learn these models using a combination of model-based reinforcement learning and

inverse reinforcement learning, enabling step-by-step acquisition of collaborative skills from few trials.

These correction driven methods focus on updating objective parameters from physical interventions, but they typically assume that the robot already has access to task representations that capture what the human is trying to accomplish. However, learning objectives from interaction implicitly relies on a behavioral model of the human partner, since the same feedback or demonstration can be explained differently under different assumptions about how the human adapts as detailed in Section 2.2. Therefore, a class of approaches focuses on learning human and team coordination dynamics directly, and then using these learned models to plan robot behavior. Nikolaidis et al. [197] learn human models from joint-action demonstrations by clustering demonstrated sequences into human types, learning a reward function per type through inverse reinforcement learning, and using a mixed-observability Markov decision process to infer a new user’s type while selecting type-aligned robot actions. Whereas this approach captures persistent differences across users, it does not explicitly represent how a particular user may adapt within an ongoing interaction. Subsequent work by Nikolaidis et al. [192] address this by formalizing mutual adaptation with bounded memory, treating human adaptability as a latent state and allowing the robot to decide whether to lead the human toward improved strategies or adapt to maintain trust. In a further extension of this line of work, the robot adopts a game-theoretic model to trade off task reward with revealing its capabilities to shape the human’s evolving expectations [196]. Relatedly, Bansal et al. [24] model parallel-play coordination as a game and infer the human’s equilibrium strategy via Bayesian updates, using this inference to select trajectories that reduce interference and improve fluency.

Learning based approaches for adaptation often address non-stationarity in the human partner’s objectives, either because the human is still learning the task or the

robot, or because the human may switch between distinct collaboration strategies over time. Chan et al. [47] formalize the case where the human is still learning the task using by proposing an assistive multi armed bandit approach. In their approach, they model the human to be learning a reward function through experience and the robot assists based only on observed human choices, yielding conditions under which assistance can reduce long term regret. On the other hand, Nikolaidis et al. [194] treat non-stationarity in human behavior as something that can be modeled through role-switching, and propose cross training in which robot and human switch roles so the robot refines its expectations of human responses while improving alignment and team fluency. Complementing these algorithmic formulations, Van Zoelen et al. [279] provide empirical evidence that such co adaptation dynamics emerge in practice by identifying recurring interaction patterns in repeated collaboration with a reinforcement learning robot, and show that human adaptation can accelerate robot learning and improve team outcomes. Another line of work represents non-stationarity more directly by explicitly modeling a latent collaboration strategy and conditioning robot actions on an estimate of that latent state. Wang et al. [283] propose Co-GAIL, an approach to learn from human-human collaboration demonstrations and train a robot to continuously infer an unobserved latent strategy of the human partner and condition its assistance on that estimate. Complementary to this work, Xie et al. [294] use a related latent representation, but focus on learning how the human’s strategy changes in response to the robot’s behavior, enabling the robot to select actions that intentionally influence the partner toward strategies that support effective co adaptation. Finally, related work by Tabrez et al. [254] focuses on shaping human behavior during collaboration by monitoring whether the human’s actions are consistent with the task constraints encoded in the robot’s reward model. When the robot predicts that the human’s current actions will lead to failure, it intervenes with a targeted explanation that highlights the missing or misunderstood constraint, allowing the

human to adjust their behavior before failure occurs.

Taken together, these learning-based approaches span adaptation mechanisms that range from updating assistance policies and intent models, to revising objectives from corrections, to anticipating and shaping non-stationary collaborators through explicit models of human adaptation and latent strategy. In sum, a key design choice in learning for adaptation is which aspect of the interaction to treat as learnable, namely assistance, objective, representation, or human strategy, and which signals to rely on at run time.

2.3.3 Transient Behavioral Adaptation Approaches

Another family of adaptation approaches centers on transient behavior changes made during execution. In contrast to learning-based approaches, these methods do not update model parameters or a plan during the interaction. Instead, they estimate the relevant interaction state and make temporary modifications to how a pre-trained policy or pre-computed plan is applied, such as switching roles, adjusting autonomy, choosing among fixed behaviors, or adjusting motion.

One line of work focuses on estimating aspects of the human’s physical state during collaboration and adjusting robot assistance to reduce physical strain without disrupting task progress [295]. In co-manipulation, Peternel et al. [214] infer fatigue from physiological signals and adapt the interaction by shifting initiative to the robot as the human becomes tired. Related work by Suresh et al. [252] treats fatigue as a latent factor that shapes how the human behaves, infers the most likely fatigue-related type from observed actions, and selects a team policy learned from demonstrations that reflects that state. Kim et al. [149] address physical load more directly by estimating musculoskeletal strain and adjusting supportive motion online to encourage safer postures while preserving task performance. Across these approaches, adaptation is realized either through discrete role changes or through continuous motion

adjustment, which in turn determines how frequently state estimates must be updated and how precise the underlying sensing needs to be.

At the interaction level, a related set of approaches adapts robot behavior based on estimates of interaction-specific variables such as trust, willingness to accept guidance, or uncertainty about preferences. The common idea is that effective assistance depends not only on what task is being performed, but on how receptive the human is to robot initiative at a given moment. Hannum et al. [110] demonstrate this in collaborative carrying by estimating trust from motion cues, interaction history, and task outcomes, and then adjusting assistance so the robot matches the human’s pace when trust is high and reduces initiative when trust is low. Sadrfaridpour et al. [231] adopt a similar perspective in collaborative manufacturing, modeling trust as a dynamic quantity and regulating autonomy level and robot speed to maintain both productivity and user confidence. Other work focuses on preference uncertainty rather than trust. Similarly, Mahmud et al. [175] maintain a belief over whether a human prefers to lead or follow during collaborative transport and switch roles online as the robot gathers more information about the human’s preferences through interaction. Complementary experimental results by van Zoelen et al. [278] show that subtle robot behavior changes can induce humans to shift between leading and following, supporting the idea that initiative is not fixed but emerges through interaction.

Another category of work enables adaption through intent estimation, where the robot continuously interprets what the human is trying to do and uses that estimate to select actions in the moment. For example, Huang et al. [129] uses a collaborative industrial assembly task where the robot must keep pace with the human as intent evolves during execution. They estimate which collaboration stage the human is in and which subtask the human is attempting, and then use those estimates online to choose between behaviors such as yielding to avoid interference or intervening to assist when failure is likely.

A complementary line of work changes the selection and sequencing of robot behaviors at runtime based on task progress, without modifying the behaviors themselves or learning new ones. In these systems, the robot has a fixed set of skills, but it must decide which skill to execute next and when, because the human may choose different steps or complete steps earlier or later than expected. For example, Ionova and Behrens [133] propose CoBOS, which uses online constraint-based scheduling inside an execution controller to select feasible next robot actions that complement the human’s progress. Hagenow et al. [107] propose a similar approach for corrective shared autonomy, where the robot executes a default autonomous behavior and applies real-time human corrections to selected task variables during execution rather than updating the default behavior.

Finally, some approaches adapt low-level motion trajectories of the robot continuously to maintain safety and coordination as human motion. For example, Jain et al. [136] anticipate near-term human motion and update robot trajectories online using multi-objective optimization to balance safety and comfort with interaction efficiency. Ibarguren et al. [131] present a similar approach but for dual-arm co-manipulation tasks that adapts in the control loop through impedance control and force feedback, coordinating trajectories in real time and providing feedback cues so the human and robot remain coordinated under disturbances.

Overall, these transient behavioral adaptation approaches complement learning-based and planning-based techniques by enabling rapid within-interaction adjustments when uncertainty during the interaction can be handled through online state estimation and short-term modulation of assistance, rather than by updating learned policies or revising plans.

2.3.4 Hybrid Adaptation Approaches

Many hybrid approaches to adaptation in human-robot collaboration have been proposed because collaboration in complex real-world tasks often needs both anticipatory decision making and fast in-the-moment flexibility [136, 38]. On one hand, the robot benefits from learning structure from data so it can predict and coordinate with people in ways that match how teams actually behave. On the other hand, even a good learned model will sometimes lead to failures, and collaborators often prefer to correct or shape behavior with quick, lightweight interventions during execution rather than by providing many new demonstrations. This has led to two common hybrids that combine learning with different downstream mechanisms, combining learning with planning or transient behavioral adaptation approaches.

The first set of approaches combine learning with planning, where a learned model supplies the predictive or preference structure and a planner uses it online to select the robot’s next actions. For instance, Ng et al. [188] propose an approach to learn a distribution over cooperative carrying trajectories from demonstrations and then use that learned model inside a receding-horizon planner to repeatedly generate coordinated motion as the interaction unfolds. The learned model captures multimodal team strategies from data, while the online planning loop handles deviations and keeps the robot’s behavior synchronized with the human partner. Zhao et al. [307] apply a similar pattern at the task level by learning a human partner’s contribution preference during the interaction and then planning robot actions subject to that inferred constraint so task responsibility is allocated in a way that better matches how the human wants to participate. More recently, Hagenow et al. [108] propose an approach called REALM, which estimates the expected value of different forms of human input from a learned stochastic policy’s uncertainty and selects among assistance modes online while accounting for the effort imposed on the human.

The second set of approaches combine learning with transient behavioral adapta-

tion approaches, where learning produces either a policy or a control interface, and adaptation happens by giving the human an efficient handle to shape behavior in the moment without retraining or replanning. Brawer et al. [38]¹ propose Transparent Matrix Overlays, which let a collaborator impose modular constraints on top of a learned policy and update those constraints through simple commands, enabling immediate, targeted changes to how the policy behaves during execution. A related approach called LILA [141] learns a low-dimensional control interface shaped by language so that users can guide the robot online through a compact control space that reflects the instruction. More recently, Cui et al. [65] extend this by incorporating language corrections that refine the learned control interface during execution, so brief corrective phrases directly alter how user inputs translate into robot behavior in the current context. Finally, Zhou and Wachs [313] focus on coordination from a timing perspective by learning to predict turn-taking intent early from multimodal cues, which enables the robot to initiate turn-taking actions sooner based on partial evidence rather than waiting for intent to become explicit.

Taken together, these papers illustrate two distinct ways learning is paired with adaptation. Learning-with-planning uses learned structure to improve the quality of online decisions about what to do next. Learning with transient behavioral adaptation helps steer learned policies to follow human guidance better at execution time.

2.4 Human Cues and Communication for Adaptation

Human-robot collaboration depends not only on what the robot can observe, but also on how information is exchanged between partners during the interaction. Beyond passive observation of human behavior and task progress, collaborative systems must

¹Debasmita Ghose contributed to this work not covered in this dissertation

decide when, how, and how much information to communicate in order to maintain shared understanding and coordinate effectively. This section reviews how prior work structures information exchange during collaboration, distinguishing between implicit communication through action, and explicit communication through dedicated signals. We then examine how these choices affect the burden placed on the human partner and shape the tradeoffs between fluency, robustness, and responsiveness in adaptive systems.

2.4.1 Human Observation Channels

Adaptation relies on signals derived from both the human and the task that allow the robot to estimate the evolving state of the interaction and adjust its behavior during execution. Across the literature, these signals are typically categorized by the observation channels available to the robot for perceiving human behavior and task progress, which in turn support different forms of adaptation.

Physical Cues

Many adaptation approaches rely on physical cues produced by the human during task execution, including human motion, motion of task-relevant objects or contact forces used to interact with the object. These signals are typically available continuously, require no explicit communication, and arise naturally from interaction, allowing the robot to adjust its behavior smoothly and typically with low latency during execution.

In tasks involving close physical coordination, such as collaborative manipulation and transportation, robots have been shown to use physical cues like observed motion and object dynamics to adapt dynamically to maintain synchronization with the human and offer appropriate levels of assistance [110, 188]. Changes in movement speed, trajectory, or applied effort have also been shown to serve as implicit coordination signals, allowing partners to align without explicit communication [296, 175]. In some

sequential tasks, similar physical cues are often mapped to task-relevant state estimates such as progress, phase, or intent. Some approaches use action segmentation to model the interaction between people’s motion and objects in their environment to trigger timely assistance or detect deviations from expected behavior during assembly and manipulation [308, 129]. Some systems have demonstrated that human timing and motion alone can be sufficient to anticipate upcoming needs of the human while working on a shared task, such as preparing tools or components in advance [113] to speed up the collaboration.

In collaborative tasks that need contact-rich manipulation, interaction forces and contact transitions have shown to provide direct evidence about people’s intent and their preferred interaction mode. Robots have been shown to use these signals to infer task phase, adjust impedance, and coordinate roles during physical cooperation [94]. Force-based cues have also been used to adapt assistance based on inferred human physical state, such as fatigue, during sustained interaction [214]. In some settings, physical interventions have been demonstrated act as feedback, allowing humans to provide a physical correction during the interaction [19, 18].

Temporal Cues.

Some approaches use the timing of human actions as an observation signal for adaptation. Features such as brief pauses, regular pacing, and the alternation of actions between partners provide information about task progress and coordination state, allowing the robot to update its understanding of the interaction and adjust its behavior during execution.

Timing cues have been used to anticipate handovers and role transitions, allowing robots to start actions earlier and reduce waiting during collaboration [313]. Estimates of how close a person is to finishing an ongoing action or sub-task has also been used to time assistance so that it aligns with the task progress expected by

the human instead of reacting too late [69]. Because these cues depend on temporal relationships, they typically should transfer more easily across tasks. However, temporal cues provide limited information about what action the robot should take. They indicate that intervention may be appropriate, but not which specific action best supports the task. As a result, timing-based cues are most effective when combined with other observation signals that help determine what form of assistance is appropriate. They are especially well suited to settings where the robot’s role is to pace, schedule, or provide support, rather than to choose among many distinct task actions.

Language Cues.

Language provides a high-level channel through which humans can communicate goals, preferences, and constraints to a robot while working on a task together. In shared autonomy settings, short corrective utterances have been shown to constrain robot actions, allowing the human to steer execution without taking full control [65]. More broadly, language has supported online goal clarification and intent disambiguation in situated collaboration, where instructions are often partial, evolving, or under-specified [39, 91, 101, 38, 100]. Language is also commonly used to recover from failures, for example through dialogue that helps identify what went wrong and how to proceed [32].

At the same time, language use varies across people and situations. Humans differ in when they choose to speak, how detailed their instructions are, and whether they provide proactive guidance or intervene only when problems arise [5]. Because of this variability, language alone is shown to be often insufficient for stable online adaptation [171]. Systems therefore achieve more reliable behavior when linguistic input is interpreted in the context of the ongoing task and combined with other observation cues, such as motion, timing, or action history [91].

2.4.2 Information Exchange between the Human and the Robot

Collaborative systems differ in how information is exchanged between the human and the robot during the interaction. While section 2.4.1 focused on raw signals the robot can use to infer human state and task progress, this section focuses on interaction mechanisms that are intended to convey information between partners as part of the collaboration itself.

Across the literature, information exchange spans two interaction patterns. In some cases, information is conveyed implicitly through how the human and robot act in relation to one another, without either agent explicitly intending to communicate. In other cases, information is exchanged explicitly through language, gestures, or interfaces designed to convey goals, preferences, or constraints. These modes of information exchange cut across sensing modalities and task domains. Many of the cues discussed earlier can be understood in terms of how information is conveyed and interpreted during interaction.

Implicit Communication

During collaboration, the robot could communicate implicitly to the human by conveying information through the way it performs task actions, rather than through explicit communication such as speech, queries, or gestures. The robot’s movements, timing, and task choices are designed not only to accomplish the task, but also to make its intent, expectations, or next steps easier for the human to understand during collaboration.

In collaborative manipulation and transport, robots have been shown to adjust how they move shared objects so that the resulting interaction makes coordination easier without requiring explicit signaling [199, 191]. For example, the robot may initiate motion in a way that clearly indicates when it intends to lead, follow, or

transition roles, allowing the human to adapt smoothly based on how the task unfolds [296, 201]. Other work explicitly selects robot actions based on how they are expected to influence the human’s interpretation and subsequent behavior over time, rather than optimizing task performance in isolation [263, 233].

The key factor that distinguishes this line of work from passive adaptation based on observation of human behavior is that the robot is not only reacting to human behavior, but also acting in ways meant to shape the human’s understanding of the collaboration. As a result, these approaches place strong emphasis on legibility and predictability of robot behavior [75, 230]. When implicit communication is effective, coordination emerges naturally through shared action. When it fails, misunderstandings can persist because the information exchange depends entirely on how the human interprets the robot’s actions, without an explicit channel for clarification.

Explicit Communication

In approaches that rely on explicit communication, the robot conveys information to the human through dedicated signals that are separate from task execution itself, such as spoken explanations, explicit corrective gestures or visual overlays. These channels are typically used when observation and action alone are insufficient to maintain shared understanding, for example when task state is partially observable, when the robot’s internal reasoning is not obvious from its behavior, or when coordination breaks down.

Several systems use explicit communication to help align human and robot understanding during execution. For example, Tabrez et. al [254] propose a reward-coaching approach to provide targeted explanations when the robot predicts that the human’s current actions will lead to failure, guiding subsequent behavior by clarifying task-relevant constraints or objectives. Later work proposes similar visual guidance systems to support coordination by presenting descriptive information about task

progress or prescriptive suggestions about what to do next, allowing humans to diagnose errors or adjust their actions in real time [256]. In recovery scenarios, robots have been shown to generate detailed explanations of failures, unmet preconditions, or internal requirements, which helps human partners provide more effective and targeted assistance [67, 68].

In shared autonomy, brief physical interventions or control overrides have been shown to serve as clear indicators that the human uses to signal the robot that its actions should be adjusted, allowing the robot to adapt immediately [284]. Rather than treating these events as failures, robots interpret them as structured feedback about how the task should be performed and update their behavior online [19]. Corrective shared autonomy frameworks formalize this idea by designing policies that anticipate and incorporate sparse corrective events while preserving overall autonomy [107]. Some prior work has also used short corrective utterances function as discrete constraints on execution rather than as full instructions or explanations [65].

The effectiveness of explicit communication depends strongly on how it is used. Poorly timed, overly detailed, or weakly relevant messages can interrupt task flow and increase cognitive load. Prior work shows that communication must be carefully calibrated to the task context and the human’s current state in order to support fluency rather than become an additional burden [42].

2.4.3 Human Burden

Human burden refers to the additional cost placed on the human by the interaction itself, beyond the physical task they are trying to complete. During collaboration, this burden commonly shows up in three ways. First, there is attentional demand, where the human must monitor the robot’s behavior and shift focus between their own actions and the robot. Second, there is cognitive workload, which includes interpreting what the robot is doing, maintaining a shared understanding of the task,

and deciding how and when to respond. Third, there is time and effort spent on communication, such as responding to interruptions, providing confirmations, resolving misunderstandings, or correcting mistakes. This perspective is consistent with work in psycholinguistics that treats communication as a form of collaborative effort, where both speaking and understanding require work from the human partner [61]. It also aligns with established approaches in human factors that measure workload through subjective assessments like NASA-TLX, as well as through performance costs that arise from coordination and interaction overhead [112].

Reducing explicit communication can lower immediate interaction overhead, but it does not always reduce overall human burden. When the robot’s intent or internal state cannot be reliably inferred from the workspace or from ongoing motion, not having the ability to communicate their feedback shifts work back onto the human. The human must monitor the robot more closely, infer what it is doing, and remain ready to intervene if something goes wrong. As a result, mis-alignments can persist unnoticed for longer periods, and once they are detected, repairing them often requires greater effort than if they had been addressed earlier. This tradeoff is well captured by grounding theory, which predicts that limiting exchange can reduce short-term effort, but that insufficient shared understanding leads to higher costs later in the interaction through repair and re-coordination [61]. Similar conclusions appear in work on human–autonomy teaming, where a lack of transparency forces humans into a supervisory role, reducing situation awareness and increasing workload despite fewer overt communication demands [275].

Providing frequent or overly detailed communication can also increase human burden, especially when it disrupts task flow or adds unnecessary mental effort. Human attention and working memory are limited, so additional information competes with the resources needed to perform the primary task [253]. In collaborative settings, this burden often appears as interruptions or task switching, where even relevant infor-

mation could force the human to pause, reorient, and resume their work, leading to delays and frustration [179]. Findings from transparency and explanation research in human-computer interaction also reflects a similar pattern. Additional information can improve understanding and confidence when it is well timed and clearly prioritized, but beyond a certain point it becomes counterproductive. Excessive detail or poorly structured explanations can overwhelm users, reduce usability, and undermine trust [151]. Reviews on information overload further show that performance degrades when communicated content exceeds what people can reasonably process in context, particularly during ongoing tasks [13].

In contrast to both extremes, minimal explicit communication can be effective when the task is well structured and the state of the interaction is easy to infer from the shared workspace. When progress is visible, roles are predictable, and action boundaries are clear, partners can often coordinate through their actions alone, with little need for additional exchange of information. In such settings, communication can remain sparse without increasing burden, because the task itself provides sufficient information to maintain shared understanding. This pattern aligns with the principle of least collaborative effort, which suggests that teams naturally limit communication to what is needed to sustain coordination, increasing it only when uncertainty or misalignment arises [61]. Crucially, this strategy relies on the interaction remaining easy to repair. As uncertainty, novelty, or risk increases, coordination based primarily on observation becomes fragile, and the absence of a reliable way to correct misunderstandings can increase workload and reduce effectiveness [275].

Taken together, communication during collaboration should be treated as a resource that must be managed, rather than something that is always beneficial to increase. Too little communication can raise burden by forcing the human to closely supervise the robot and by making errors harder to detect and repair once they occur [61]. Too much communication can also raise burden by interrupting the human

and adding information-processing demands that compete with task execution [179]. These tradeoffs motivate adaptive approaches that remain unobtrusive when the task and workspace provide sufficient cues for coordination, and increase communication only when uncertainty, exceptions, or safety concerns make additional information necessary [281, 41].

2.5 Evaluation of Adaptation

Evaluation is key to understanding whether adaptive behavior improves human–robot collaboration in practice. Prior work assesses adaptation across a wide range of tasks, settings, and metrics, reflecting the fact that successful collaboration depends not only on task completion, but also on coordination quality, safety, human workload, and subjective experience. This section reviews how adaptation is evaluated in the literature, first summarizing the types of tasks and experimental settings commonly used, and then organizing evaluation metrics according to the aspects of interaction they are intended to capture, including task performance, fluency and efficiency, human burden, safety, trust, and subjective experience.

2.5.1 Evaluation Tasks and Settings

Across the surveyed literature, adaptation is most commonly evaluated in manipulation-centric tasks, particularly tabletop manipulation and shared object handling. These tasks include reaching, grasping and placing objects in the workspace in a specified configuration, surface cleaning, and corrective manipulation on a shared table or workspace [19, 18, 65, 107, 108, 119, 141, 318, 136]. These settings often involve close physical interaction or shared control, where the human and robot act concurrently on the same objects or control variables. As a result, adaptation in this class of tasks typically targets execution-level behavior, such as the level and timing of assistance,

or compliance, and must operate at low latency to preserve stability and responsiveness. A vast majority of these studies are conducted on real robot manipulators in controlled laboratory settings, often using fixed-base arms, with simulation used primarily for ablations or scaling analyses.

Another related and recurring task group consists of collaborative transport and co-manipulation, where humans and robots jointly carry or move large objects [94, 188, 296, 110, 175]. These tasks are characterized by continuous, tightly coordinated interaction, with both partners acting at the same time and influencing shared object motion. Adaptation in this setting is therefore primarily focused on motion, force, timing, and role balance during execution, rather than discrete task decisions. Because stability and safety depend on rapid response to human behavior, evaluations are overwhelmingly conducted with real robots in laboratory settings, often supplemented by simulation for planner comparison or robustness analysis.

The next class of evaluations use assembly tasks, where humans and robots collaborate on multi-step construction problems such as furniture building, block assembly, circuit construction, drilling, or fastening [113, 227, 178, 32, 223, 191, 195]. These tasks are typically long-horizon and sequential, with identifiable subtasks, handoffs, and decision points that structure the interaction. Accordingly, adaptation in this domain is most often applied to task-level decisions, such as subtask selection, role allocation, scheduling, or recovery choices, rather than continuous low-level control. Evaluations are typically carried out on real robots in laboratory or industrial mock-up environments, often with predefined workflows and scripted or semi-scripted human partners. Several studies complement these experiments with simulation-based evaluations to explore alternative task sequences or allocations at scale.

Manufacturing and industrial collaboration tasks form a closely related but distinct class of evaluation tasks. These include industrial assembly, disassembly, quality inspection, polishing, surface refinishing, collaborative recycling, and ergonomics-

aware collaboration [214, 149, 310, 98, 314, 129, 93]. Compared to tabletop manipulation, these tasks often involve larger workpieces, tools, sustained physical interaction, and prolonged effort, and frequently combine sequential structure with periods of concurrent execution. Adaptation targets in this domain therefore span both task-level coordination, such as scheduling and role allocation, and execution-level behavior, such as force, posture, or assistance magnitude, because small errors can directly affect safety and physical burden. Evaluations in this category frequently involve physical robots operating in lab-scale industrial setups or real-world industrial environments, with simulation used to study long-horizon behavior or alternative scheduling and monitoring strategies.

Finally, adaptation during human–robot collaboration has also been explored in household tasks. Collaborative cooking and meal preparation is a common evaluation domain [271, 272, 38, 91, 88], alongside sorting, kitting, de-cluttering, and organization tasks such as table clearing, grocery bag packing, lunchbox assembly, and general tidying [200, 52, 307, 201, 101, 84]. These tasks are typically long-horizon and semi-structured, often alternating between sequential decision-making phases and periods of parallel work. As a result, they are commonly used to study task-level adaptation, including goal inference, role allocation, and proactive planning, sometimes combined with execution-time adjustments. Evaluations are conducted using real robots in laboratory kitchen- or home-like setups, often with scripted or semi-scripted human partners, and are typically complemented by simulation to explore longer interaction horizons or alternative goal structures.

2.5.2 Evaluation Metrics

Evaluation of adaptation in human robot collaboration is typically organized around different types of metrics that capture complementary aspects of the interaction. The metrics summarized below reflect both objective and subjective measures, including

efficiency and fluency during execution, the burden placed on the human, and safety, trust, and subjective experience that evaluates if adaptive behavior is acceptable and usable in practice by people.

Task Success and Correctness

Across the literature, task level performance of the human-robot team is often treated as a baseline indicator of whether the robot’s adaptive approach is useful, but the exact definition of a successful trial differs based on the task and the collaboration dynamics. In long horizon collaborative tasks, success is often reported as the fraction of trials in which the team completes the full task sequence end to end. This choice is natural when failures arise from execution breakdowns, unmet preconditions, or unsuccessful recovery, because these methods are explicitly evaluated on their ability to detect problems and resume progress [7, 152, 32, 184]. In related failure handling work, task outcomes are also paired with measures that reflect the quality of recovery, such as time per help request and the rate of error free interventions, since completion alone can hide whether the robot asked for the right help at the right time [152].

In other domains, success is defined less by completing a scripted sequence and more by satisfying physical and safety constraints throughout the interaction. For example, in collaborative transport tasks, a trial is typically counted as successful only when the object reaches the goal while remaining stable and within workspace limits, and without collisions or unsafe contact. These definitions embed safety into the success metric itself, so success can be stricter than goal attainment alone [296, 110, 188, 175]. A similar issue appears in other co-manipulation tasks where the interaction can be completed in multiple ways, but the evaluation focuses on whether the team maintains safe forces or ergonomic conditions while achieving the task objective [214, 149, 19].

Some work reports task performance at finer granularity to make failures more

interpretable. In sequential collaboration, performance is sometimes expressed as success over subtasks or as assembly progress, which can better reflect coordination quality as opposed to a coarse measure of task completion measure [197, 223, 69]. This is also common in work that optimizes role allocation and scheduling, where a team can complete the overall goal but still incur avoidable delays if allocation decisions are poor. As a result, task completion time is frequently treated as a primary indicator of whether adaptation improves coordination efficiency [223, 133, 98, 201]. In shared autonomy settings, success is usually defined as completing the task under partial assistance, but it is rarely meaningful in isolation because assistance policies can trade off success against how much control burden is shifted to the person. These papers therefore interpret success together with measures such as total task time, amount of user input, intervention rate, or jerk and smoothness, which capture the cost of achieving the same nominal outcome [141, 65, 318, 107, 108].

Task performance is also sometimes measured with scalar return scores such as cumulative reward, percentage of optimal reward, or regret. These metrics support broad comparisons across policies and often align with the optimization target [272, 47, 307, 195]. This is particularly relevant in mutual adaptation and trust aware planning, where higher reward may reflect either better coordination or more aggressive autonomy that changes the human’s behavior. However, these metrics are not very interpretable as they can obscure which interaction constraints were violated to lead to a specific value of return, since very different behaviors can yield similar returns. For this reason, reward based evaluations are typically most informative when paired with outcome specific measures such as intervention frequency, safety stops, or success rate [52, 24].

Collaboration Fluency and Efficiency

Measures of collaboration fluency and efficiency are often inseparable in evaluation because both ask whether the interaction proceeds smoothly with minimal coordination overhead, not just whether the team finishes. Following Hoffman’s [120] framing, a common strategy is to quantify inefficiency that is induced by interaction itself, such as waiting, avoidable handoffs, and delays introduced by miscoordination. Therefore, work involving a robot offering proactive assistance typically reports human and robot waiting times and related concurrency measures, since these directly indicate whether the robot acts at moments that remove bottlenecks rather than create them [197, 69, 223]. Cross-training and mutual adaptation studies use the same behavioral signals, interpreting reduced idle time and increased concurrent activity as evidence that partners have converged on compatible role expectations and a workable division of labor [195, 194].

In many sequential collaborative tasks, efficiency is often evaluated by how much interaction overhead adaptation introduces to align the task models of the human and the robot [33], especially time spent communicating and time spent recovering from miscoordination. Unhelkar et al. [271] reports the amount of robot communication alongside task reward and time, since communicating can increase efficiency only when it prevents downstream errors or wasted work. Similarly in fault recovery systems, efficiency is often reflected in the number and duration of recovery episodes, or in the fraction of errors that can be corrected without escalating to extended interventions, because repeated human interventions increase inefficiency in long horizon tasks [152, 32, 184, 7]. Systems that rely on more targeted help requests measure efficiency through elapsed time per request and the accuracy of interventions, which distinguishes fast recovery from recovery that consumes repeated attempts [152].

In scheduling and constraint based coordination, efficiency is frequently framed as avoiding delays under uncertainty. As a result, they report improvements in com-

pletion time together with indicators of disruption such as safety stops or delays attributable to contention for shared space [133, 24]. In close proximity collaborations that involve a robot to plan a path to achieve a task, efficiency is sometimes evaluated through deviation from nominal motion or path length, to ensure that the planned path satisfies both the task constraints and is safe around the human’s close proximity [136].

Several works in shared autonomy treat efficiency as reducing the amount of human effort required to achieve the same outcome. They report task time together with the amount of user input or number of interventions [318, 107, 108]. Corrective interfaces similarly use intervention counts and correction frequency as efficiency indicators, since a method that requires fewer corrections to reach the same success level is effectively reducing coordination overhead during execution [65, 38]. Work that evaluates trust aware autonomy also sometimes use intervention rate as a proxy for collaboration efficiency, because the robot misjudges the level of autonomy the human is expecting while collaborating, it leads to frequent takeovers by either agent [52, 110].

Human Burden and Workload

In the literature, human burden is typically evaluated through intervention costs and perceived workload to the human. Many systems quantify burden by counting interventions, corrections, or help requests [38, 107, 108, 95, 52, 106]. In some work, human burden has also been measured through the number of clarifying questions and induced extra steps, explicitly trading off the cost of information gathering during interaction [88, 91].

For subjective evaluation of the perceived workload of the human, the NASA Task-Load Index (NASA TLX) survey [111] is commonly used. It is typically used as a post-hoc measure to gauge the cognitive effort or frustration of a person while collaborating

with a robot [223, 69, 276, 168, 84, 107, 108]. The advantage of using the NASA TLX survey is that it makes adaptation approaches and interaction styles comparable in terms of perceived human burden, since two approaches can yield similar task success while differing substantially in mental demand, time pressure, or frustration [223, 69, 84, 276, 168]. Some work also uses the System Usability Scale (SUS) [102] to complement the NASA TLX survey to separate workload from perceived usability, learnability, and willingness to use the interface again, which matters when a system succeeds technically but feels cumbersome or brittle [107, 108]. The main limitation of the NASA TLX and SUS surveys is that they are retrospective, self reported and aggregate over an episode. They are sensitive to framing and expectations, making it hard to attribute a high score to specific interaction events without additional task-specific metrics [69, 223, 84].

Physical effort metrics matter most in tasks where the robot is meant to make the work less physically demanding for the human. In such scenarios, prior work measures if the robot changes its role or assistance when the person shows signs of getting tired, and whether this reduces fatigue while keeping the team effective [214]. In other work, evaluations use measurable signals that reflect physical strain, such as estimated joint loading and muscle activity, to test whether the robot helps the person stay in more comfortable working postures [149].

In learning from physical interaction, papers often report how long the person had to physically guide the robot and how much physical effort was applied, alongside task cost measures, to show that learning is improving while the person has to provide fewer physical corrections over time [19]. Work on physical corrections also checks whether the robot learns the intended change without forcing the user to repeatedly correct or undo unintended changes, using measures such as forces applied and how often users had to reverse a mistaken update [18]. In industrial assembly settings, physical effort is tied to reliability by reporting interaction forces and energy like measures

together with assembly failures, so the evaluation can show whether reduced strain also coincides with fewer breakdowns during the task [129].

Safety

Safety is evaluated either by counting explicit safety critical events or by measuring how the robot moves around the person during close proximity collaboration. Many studies report discrete events such as safety stops, since these capture moments when the robot had to intervene to avoid a likely collision and therefore give a direct measure of how often interaction reached an unsafe margin [24]. Other work treats safety as a graded measure that varies over the trajectory and reports continuous motion quality measures such as how much distance the robot keeps from the human, whether the end effector stays within the human’s view, and how far the adapted motion deviates from a nominal plan [136].

In many applications, safety is not measured in isolation, but rather in combination with human burden or task success. For example, in industrial settings such as disassembly and warehouse picking, safety is often evaluated together with stress, fatigue, and task accuracy, since risky behavior can both increase physical strain and increase the likelihood of mistakes during demanding procedures [314, 303, 167]. In some collaborative transport tasks, safety is frequently built into the definition of success itself through constraints on keeping the object stable and avoiding obstacles, so safety is interpreted jointly with time and coordination outcomes [296, 175].

Trust

In the literature, trust is treated both as a metric and as a latent state that the robot should reason about. Many shared autonomy studies report trust alongside task outcomes to show that assistance policies improve performance without reducing users’ confidence or acceptance of robot behavior [200, 192]. A second line of work

models trust as time varying and the robot uses it to modulate the level of autonomy, then evaluates whether predicted trust changes align with intervention rates and team reward over an interaction [52]. Related work in manufacturing uses trust measurements together with productivity and workload to capture the relationship between the autonomy levels, human interventions, and collaboration efficiency [231].

Subjective Experience

Prior work has used subjective measures when it is useful to study aspects of interaction that are difficult to measure objectively. Measures such as ease of use, helpfulness, predictability, naturalness, and willingness to reuse have been frequently used to report peoples perception of the collaborative interaction [141, 65, 101]. Standardized surveys such as RoSAS [44] and UEQ [235] have also been used to study people’s perception of the robot along the axes of warmth, competence and discomfort [296, 131].

Other work like VADER [7] perform more customized subjective evaluations of their approach by evaluating if the robot’s help requests are timely and understandable. In similar shared autonomy settings, evaluations also ask whether users can smoothly shift between interaction modes like assisting the robot and continuing their own work [152]. When adaptation relies on communicating uncertainty or intent, subjective measures have been used to evaluate interpretability, trust, and mental load and relate these to compliance [256]. Research has also used measures such as sociability, helpfulness, and perceived intelligence to capture whether the robot’s guidance during collaboration is delivered in a manner that people prefer [254].

2.6 Summary

This chapter reviewed adaptation in physical human–robot collaboration as a closed-loop process where robots observe the human and task, reason about the evolving collaborative state, and update behavior during execution. It organized prior work by the assumptions robots make to model the human during the interaction, various state-of-the-art adaptation techniques, the cues and communication mechanisms that drive adaptation, interaction dynamics and what robots are allowed to change online and how these systems are evaluated. The remainder of this dissertation presents novel contributions that advance the state-of-the-art in adaptation for long-horizon collaboration, developing mechanisms that help robots maintain shared understanding as human goals and constraints evolve.

Chapter 3

Robots Inferring Human Goals through Passive Observation¹

Robots should be able to maintain a shared understanding while collaborating with people by inferring intent from signals present during interaction, with minimal explicit supervision. The previous chapter gave a scoping review of adaptation in human–robot collaboration. It described how adaptation is executed across the literature, spanning different assumptions about the human partner, different interaction structures, different adaptation targets, and different techniques for updating robot behavior, as well as the signals, communication mechanisms, and evaluation settings used to study these systems. In this chapter, we study how a robot can learn task-relevant perceptual distinctions from passive observation when the relevant concept is defined implicitly by a human’s choices and only a few positive examples are available. Specifically, we present an online representation-learning method that incorporates these sparse selections into contrastive learning by pulling the embeddings of human-selected objects together while preserving separability among non-selected

¹Portions of this chapter were published as: *Ghose, D., Lewkowicz, M. A., Gezahegn, K., Lee, J., Adamson, T., Vázquez, M., & Scassellati, B. (2023, March). Tailoring visual object representations to human requirements: A case study with a recycling robot. In Conference on Robot Learning (CoRL) (pp. 583-593). PMLR. [93]*

objects. Using these learned representations, the robot can identify additional objects that match the human’s selections and act on them in real time while continuing to update as it observes more examples. We instantiate this approach in a simplified recycling-sorting setting with a physical robot, demonstrating sample-efficient learning of human-defined categories without requiring repeated explicit instruction [93].

3.1 Introduction

Robots are well-suited to alleviate the burden of repetitive and tedious manipulation tasks in homes and in the public and private sectors. For example, robots might help humans with housekeeping, janitorial and custodial work, or sorting recyclables. In many of these applications, a robot may be asked to interact with a wide variety of objects, making it hard or even impossible to pre-program visual object classifiers suitable for the task of interest. For example, imagine a home robot meant to help a person clean her room. One might think of defining object categories for this task, such as “clothes”, “shoes”, “books”, etc. But what if the human wants the robot to pick up only “folded clothes” from the floor? These kinds of tasks require learning from the human what object category is relevant. Since these categories can be defined arbitrarily, the robot needs to learn the object properties that distinguish the human-selected objects from others. Importantly, another challenge in learning an object category from a human is that the robot cannot assume the objects not selected by the human do not belong in the category of interest. That is because humans are unwilling to give many examples and may only provide examples of objects they are interested in.

In this work, we study the problem of learning a classifier for visual objects based on (a few) examples provided by humans. We frame this problem from the perspective of learning a suitable visual object representation that allows the robot to distinguish

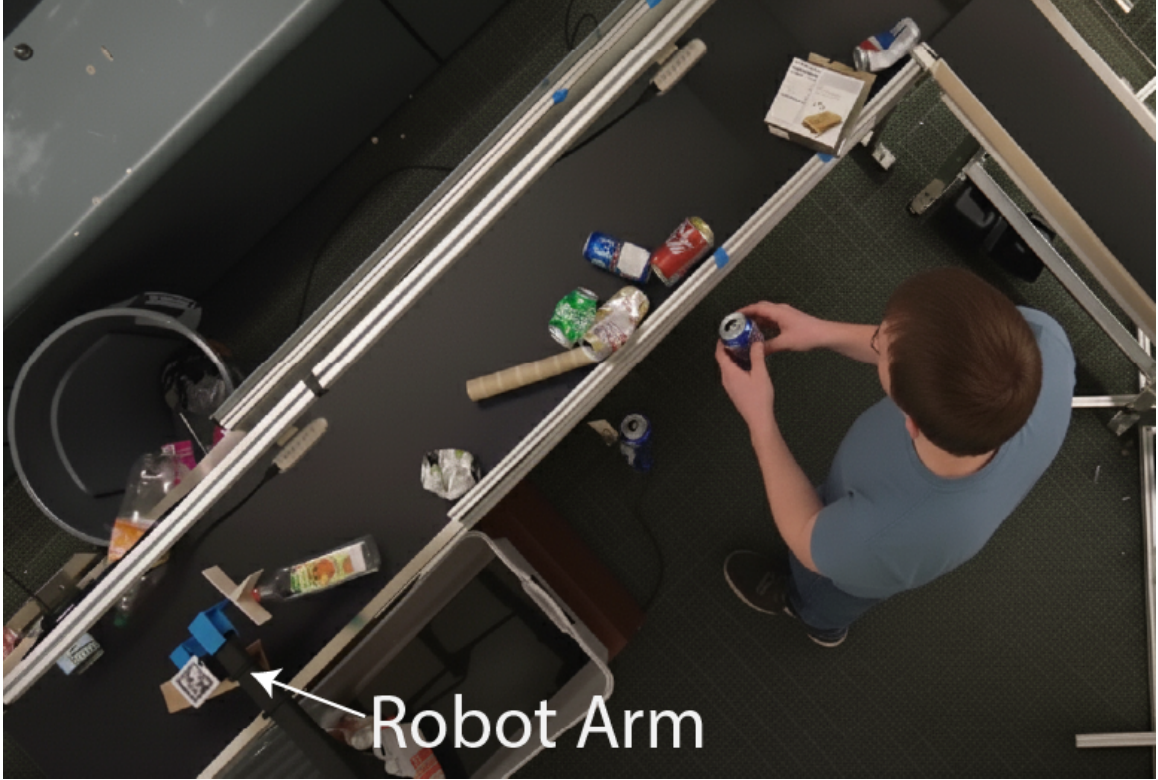


Figure 3.1: Simplified real-world recycling setup

the desired object category from others. This perspective is motivated by recent work in self-supervised learning, which has led to significant advances in learning useful representations from high-dimensional data [139]. In particular, contrastive learning [60] learns representations by pulling similar samples close to each other while pushing dissimilar ones far apart in the latent space. Recent contrastive learning methods have been successful in learning strong representations for various downstream tasks in computer vision, such as image classification, semantic segmentation, and object detection [139]. The challenge with applying existing contrastive learning techniques to the problem of learning a representation that distinguishes an arbitrary object category based on human examples is that contrastive learning will capture properties of the data, but those properties may not necessarily lead to a good object classifier according to human requirements.

We propose a new approach that integrates human supervision into the represen-

tation learning process to compute visual object representations in accordance with human requirements for the task of interest. Our approach combines an existing contrastive learning method that maximizes the agreement between different instances in the same cluster with our proposed loss function that brings the representations of objects selected by a human close to each other in the latent space. This combination results in representations that encode the human desired object characteristics.

We evaluate our approach to incorporate human supervision into the representation learning process in a simplified recycling setup derived from a standard Material Recovery Facility (MRF) as shown in Figure 3.1. We focus on this challenging application domain for three reasons. First, recycling streams can contain a wide variety of diverse objects, ranging in size, weight, color, cleanliness, and form [257]. Second, constraints on what needs to be recycled can vary dynamically because the requirements of the MRF’s customers may vary [66]. Third, because of the many objects that quickly move by, a human demonstrator cannot pick out all the items that need to be removed from the stream. This results in limited positive examples and no negative examples for the robot.

To conduct offline and online evaluations, we set up a conveyor belt that humans and robots can work on simultaneously and a feeder system to provide a steady stream of recyclables. We demonstrate that our method outperforms a fully-supervised learning method and some self-supervised learning techniques on an offline dataset. Finally, we deploy our proposed learning method in real-time on a robot working alongside a human in the simplified recycling setup. Our results show how incremental supervision from the human helps the robot learn visual object representations tailored to the human’s requirements.

3.2 Related Work

3.2.1 Self-Supervised Learning

Self-supervised learning obtains supervisory signals from the data by leveraging its underlying structure [139, 26]. One broad category of techniques in self-supervised learning is contrastive learning, which has been used in a wide range of computer vision applications [203, 240, 304]. Contrastive learning methods [60, 72] typically learn a latent space by pushing similar samples (e.g., two adjacent frames of a video [291]) close to each other while pulling dissimilar data samples (e.g., frames from two different videos) apart in the latent space. More recently, *instance discrimination* [54, 56, 76, 162], a variant of contrastive learning, has shown remarkable success in improving performance on a variety of downstream tasks. To learn latent-space representations, most instance discrimination techniques use two augmented views of the same image as a positive data pair in conjunction with a variety of contrastive learning techniques, such as the contrastive loss [54, 56, 300], triplet loss [237, 239], momentum encoding [115, 55, 103], or online clustering [43].

Recently, there have been some efforts to learn more robust latent-space representations by leveraging properties of non-trivial positive pairs that are correlated in the latent space. For example, these non-trivial positive pairs can either be selected from a support set [76, 17] or from associating multiple data instances with clusters [203, 162, 301, 312]. In this work, we leverage a clustering technique called Contrastive Clustering [162] to enable robots to learn visual object representations. Contrastive Clustering extends instance-level discrimination to distinguish between objects belonging to different clusters while maximizing agreement between objects belonging to a cluster.

3.2.2 Adding Supervision to Contrastive Learning

Prior work has guided representation learning processes by leveraging labeled data [177, 148]. Khosla et al. [148] proposed a method to train a contrastive learner in a fully-supervised fashion to improve performance on multiple downstream tasks. However, since their method is label intensive, more recent work has explored using semi-supervised [305, 8] and weakly supervised [309] learning techniques to compute strong representations. Wilber et al. [287] introduced an algorithm called SNaCK that proposes adding a loss function to visual representation learning that reflects a human’s preferences. However, SNaCK requires a human to provide a large number of examples of objects that the human thinks are similar and dissimilar, which would be infeasible in real human-robot interactions. To the best of our knowledge, our method is the first to show how human feedback can be leveraged to learn visual object representations tailored to dynamic human-defined requirements with limited examples of a single category.

3.2.3 Human Centric Robot Learning

There are many ways in which a robot can learn from a human. The three most common types of demonstrations used for robot learning are kinesthetic teaching, teleoperation, and passive observation [224]. Our work uses passive observations because of the ease at which they allow the human to teach the robot and the hardware-agnostic nature of this demonstration type. Much of the recent work in the field of learning from passive observations uses reinforcement learning to train a robot. For example, the work of Mukherjee et al. [182] uses a learned goal proximity function as a dense reward for policy training, and Karnan et al. [143] propose to learn navigation policies from a single video demonstration. Our work complements these efforts by proposing a novel approach for learning visual object representations that a robot can use for manipulation tasks, like sorting recycling streams.

Typically within the area of human preference learning, a robot learns the preferences of its human collaborators to perform a task through various reward shaping techniques [228, 158]. Our work is related to this thread of research because we want a robot to learn the preferences of its human collaborator for a sorting task. However, we want the robot to use passive observation rather than more traditional active querying techniques [289, 288, 277] because, in the recycling domain, humans are already busy picking objects from recycling streams. Preference learning in our work then entails having a robot learn the visual characteristics of the categories of objects that the human collaborator is interested in. We propose to approach this problem from the perspective of self-supervised representation learning and incorporate human supervision in a novel manner to ensure that the learned representation aligns well with the human’s preferences.

3.3 Learning Visual Object Representations Tailored to Human Requirements

We propose a method to guide the representation learning process of a contrastive learner using human supervision, as shown in Fig. 3.2. To achieve this goal, we optimize a feature extractor jointly with three "heads." These heads maximize the agreement between 1) multiple views of the same image, 2) multiple instances of the same cluster, and 3) human-selected examples.

Let $\mathcal{X}$ be the set of all objects relevant to a manipulation task and $\mathcal{C}$ be the total number of human-defined categories such that a given category $c \in \{1, \dots, \mathcal{C}\}$. As a property of our problem, a human selects a set of items $\mathcal{H}$ such that $\mathcal{H} \subset \mathcal{X}$ and all objects in $\mathcal{H}$ belong to a given human-defined category c . The paragraphs below use this notation to describe the components of our approach.

Base Encoder: The base encoder is a feature extraction network that obtains

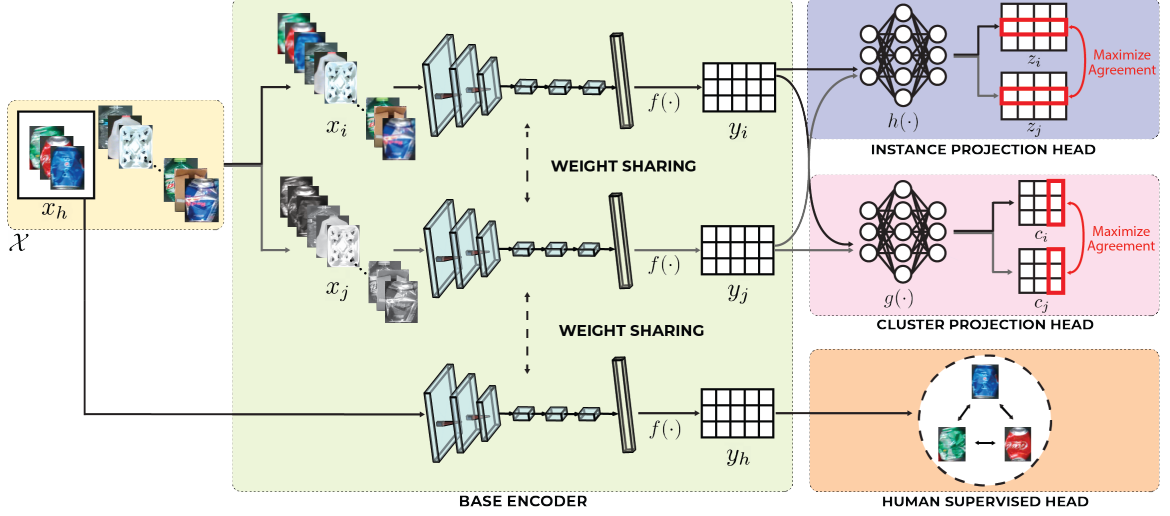


Figure 3.2: Our Approach. Data pairs are constructed using randomly applied data augmentations. The extracted features are fed into three jointly trained heads. The Instance Projection Head projects the features into a space where each row denotes an augmentation of an image and uses the instance loss to minimize the distance between two augmentations of the same image. The Cluster Projection Head projects feature vectors into a space where each column denotes each instance’s cluster assignments, and the cluster loss minimizes the distance between instances belonging to a cluster. Finally, the Human-Supervised Head minimizes the distance between each human-selected example.

representations from two stochastically augmented views of the same image [54]. For a given image x , the base encoder outputs a feature vector y , such that $y = f(x)$. Therefore, if two stochastic augmentations of an image are denoted by x_i and x_j , their feature representations can be denoted by $y_i = f(x_i)$ and $y_j = f(x_j)$. Additionally, if x_h is an image in $\mathcal{H}$, then its feature representation is $y_h = f(x_h)$. We feed the image representations output by the base encoder to the three heads described next.

1. Instance Projection Head: The Instance Projection Head is a Multi-Layer Perceptron (MLP) with one hidden layer to map the features of two augmented views of the same image to a latent space [54]. For example, for the representation of an image’s first augmentation y_i , its representation after passing through the Instance Projection Head is $z_i = h(y_i)$ (Similarly, for the second augmentation, $z_j = h(y_j)$). As in Chen et al. [54], we apply the NT-Xent loss on the representations from the Instance Projection Head. Therefore, for a positive pair of examples (i, j) constructed from two

augmented views of the same image, the instance loss w.r.t. the first augmentation is formulated as:

$$\mathbb{I}_{ins}^{i,j} = -\log \frac{\exp \left(\text{sim} \left(\mathbf{z}_i, \mathbf{z}_j \right) / \tau_{ins} \right)}{\sum_{k=1}^N \mathbb{1}_{[k \neq i]} \exp \left(\text{sim} \left(\mathbf{z}_i, \mathbf{z}_k \right) / \tau_{ins} \right)}$$

where N is the number of all positive pairs in a mini-batch, $\text{sim}(u, v) = \frac{u^T v}{\|u\| \|v\|}$ is the cosine similarity between the feature vectors, and $\mathbb{1}_{[k \neq i]} \in \{0, 1\}$, which evaluates to 1 iff $k \neq i$. We use $\mathcal{L}_{ins} = \frac{1}{2N} \sum_{i=1}^N (\mathbb{I}_{ins}^{i,j} + \mathbb{I}_{ins}^{j,i})$ to compute the instance loss over all positive pairs in a mini-batch.

2. Cluster Projection Head: The Cluster Projection Head [162] learns representations by maximizing agreement between two representations (under different augmentations) belonging to a given cluster. This head is an MLP with one hidden layer followed by a softmax operation. It projects a data instance into a latent space whose dimensionality equals the total number of pre-defined clusters ($\mathcal{K}$), which should be approximately equal to $\mathcal{C}$. Intuitively, the cluster projection head tries to partition the embedding space into a pre-specified number of clusters based on the inter-instance similarity between object features. For an encoder representation of the first augmentation of an image y_i , the latent representation after passing through the cluster projection head is $c_i = g(y_i)$ (Similarly, for a second augmentation, $c_j = g(y_j)$). Let $\mathbb{C}_i \in \mathbb{R}^{N \times K}$ be the output of the Cluster Projection Head for a mini-batch of N images under the first augmentation i . Then, the n^{th} element in the output $\mathbb{C}_i$ can be interpreted as its probability of belonging to the k^{th} cluster, where $k \in [1, K]$ can be denoted by $\mathbb{C}_i^{n,k}$. Thus, $\tilde{c}_i \in \mathbb{R}^N$ can be interpreted as the column vector of $\mathbb{C}_i$ under the first augmentation. The cluster loss can be formulated as:

$$\mathbb{I}_{clu}^{i,j} = -\log \frac{\exp \left(\text{sim} \left(\tilde{c}_i, \tilde{c}_j \right) / \tau_{clu} \right)}{\sum_{k=1}^K \mathbb{1}_{[k \neq i]} \exp \left(\text{sim} \left(\tilde{c}_i, \tilde{c}_k \right) / \tau_{clu} \right)}$$

where τ_{clu} is the cluster-level temperature parameter. Finally, the cluster loss computed by traversing all $\mathcal{K}$ clusters is

$$\mathcal{L}_{clu} = \frac{1}{2K} \sum_{k=1}^K (\mathbb{I}_{clu}^{i,j} + \mathbb{I}_{clu}^{j,i}) - \mathbb{H}(\mathbb{Y}),$$

where $\mathbb{H}(\mathbb{Y}) = \sum_{k=1}^K (P(\tilde{c}_i) \log P(\tilde{c}_i) + P(\tilde{c}_j) \log P(\tilde{c}_j))$ is the entropy of cluster assignment probabilities and $P(\tilde{c}_i) = \sum_{n=1}^N \frac{\mathbb{C}_i^n}{\|\mathbb{C}\|}$. The entropy term helps avoid the trivial solution where most instances get assigned to the same cluster as shown by [162, 312].

3. Human-Supervised Head: Our proposed human-supervised head aims to guide the representation learning process towards human requirements. Intuitively, it tries to force the representations of the objects selected by the human close to each other in the latent space to serve as a mechanism to help inform the formation of clusters according to the properties of objects important to the human. This loss is applied only to the representations of objects selected by the human y_h . Formally, let $\mathbb{Y}_h \in \mathbb{R}^{\mathcal{H} \times \mathcal{D}}$ be the representation learned by the base encoder across all objects selected by the human ($\mathcal{H} \subset \mathcal{X}$) and $\mathcal{D}$ be the latent dimension of the base encoder. Additionally, let $\mu_h \in \mathbb{R}^{\mathcal{D}}$ be the mean of the representations of all the objects selected by the human, which is considered the cluster-centroid of the human pool.

As new objects are selected by the human, we pass their image (x_h) through the base encoder to get their feature representation ($y_h = f(x_h)$). If previously there were s human-selected objects and the human selects t new objects, we re-calculate the mean of the cluster (μ'_h) to include the features of the new objects by

$$\mu'_h = \frac{\mu_h * s + \sum_t y_t}{s + t}$$

The human-supervised loss then minimizes the distance between the new centroid

of the human-selected objects and every object selected by the human:

$$\mathcal{L}_{human} = -dist(\mu'_h, \mathbb{Y}_h)$$

Theoretically, any distance metric (*dist*) can be used to maximize agreement between the centroid of the human-selected pool and all human-selected objects. However, we empirically found cosine-similarity to work better compared to L1 and L2 distances since it is bounded between -1 and 1.

Objective Function: The Instance Projection Head, Cluster Projection Head, and Human Supervised Head are combined in the final objective function that we use to train our model:

$$\mathcal{L} = \lambda_{ins}\mathcal{L}_{ins} + \lambda_{clu}\mathcal{L}_{clu} + \lambda_{human}\mathcal{L}_{human}$$

where $\lambda_{ins} > 0$, $\lambda_{clu} > 0$, and $\lambda_{human} > 0$ are hyperparameters.

3.4 Experimental Setup

We use the Stretch RE-1 robot [145] with a static push plate mounted on a telescopic arm to extract items from a moving conveyor belt (as shown in the supplementary video). The robot can extract items by either extending its telescopic arm to push objects off the conveyor belt or pulling objects towards itself. As shown in Fig. 3.3, the setup consists of a 60-gallon hopper, a small feeder belt, and the main conveyor belt (10 feet long and 1.5 feet wide). We position three cameras above the belt to monitor the recycling stream.

We use over 500 unique recyclable items of various sizes, shapes, colors, and forms in our experiments. The recyclables belong to ten categories: *crushed metal cans*, *un-crushed metal cans*, *crushed plastic bottles*, *un-crushed plastic bottles*, *un-crushed*

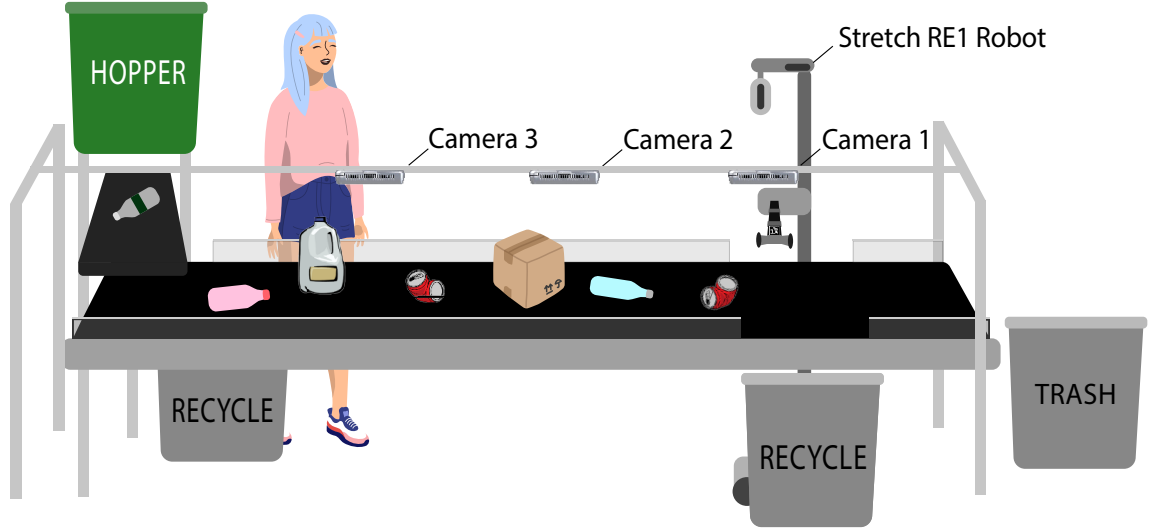


Figure 3.3: Experimental Setup. The human and the robot work together on the recycling conveyor belt to sort the recyclables.

colored plastic bottles, brown cardboard boxes, coated cardboard boxes, cardboard trays, half-gallon milk jugs, and one-gallon milk jugs. For randomizing the stream, the hopper dispenses a random selection of recyclables onto the feeder belt, which then transports the items to the main conveyor belt. A human can pick up some items of a given category by standing next to the conveyor belt. The robot is expected to sort items similar to the human-selected objects and extract them from the stream by pushing or pulling them off the conveyor belt.

3.5 Offline Evaluation

3.5.1 Data Collection

For evaluating the system offline, we created a dataset by having a selection of recyclables pass through the setup described in Section 3.4. Figure 3.4 shows a few sample images from the different categories of recyclables represented in the dataset. We used YOLACT [35], a real-time instance segmentation model with a ResNet-101 [116] backbone, to predict a rotated bounding box over each item on the conveyor

belt. We trained this instance segmentation model with 411 conveyor-belt images containing 4419 instances of recyclables, which were manually annotated with an instance mask. Using an 80% train, 20% validation split, we obtained an F1-score of 0.89 after training the YOLACT instance segmentation model for 100 epochs using the MM-Detection [51] library in PyTorch. Each predicted mask was converted to a rotated bounding box whose crops were extracted for evaluation. Finally, we created a dataset containing 1502 variable-sized crops of recyclables, where each crop belongs to one of the ten categories described in Section 3.4.

3.5.2 Evaluation Protocol

For training, we select three random sets of 40 examples from each of our 10 categories to serve as the human-selected pool for a given category (resulting in 30 human pools in total). We use each human-selected pool to train our proposed model. During inference, we compute the pairwise cosine similarity of every object not selected by the human with every object in a given human-selected pool. We then compute the average cosine similarity of the top-5 most similar objects and assign it as the similarity score of the candidate object. We compute an F1-score per category by thresholding each similarity score between $[0.1, 0.95]$ and assigning a candidate object to a given category if the similarity score is above the threshold [165, 164]. We report F1-scores for the best threshold averaged over three human-selected pools per category.

3.5.3 Quantitative Results

Experiment 7 in Fig. 3.5 shows the performance of our method on the ten categories specified in Section 3.4 using the evaluation protocol explained above. We compared our method with a supervised learning method comprised of only our human supervised head in isolation. This head effectively learns to classify instances when

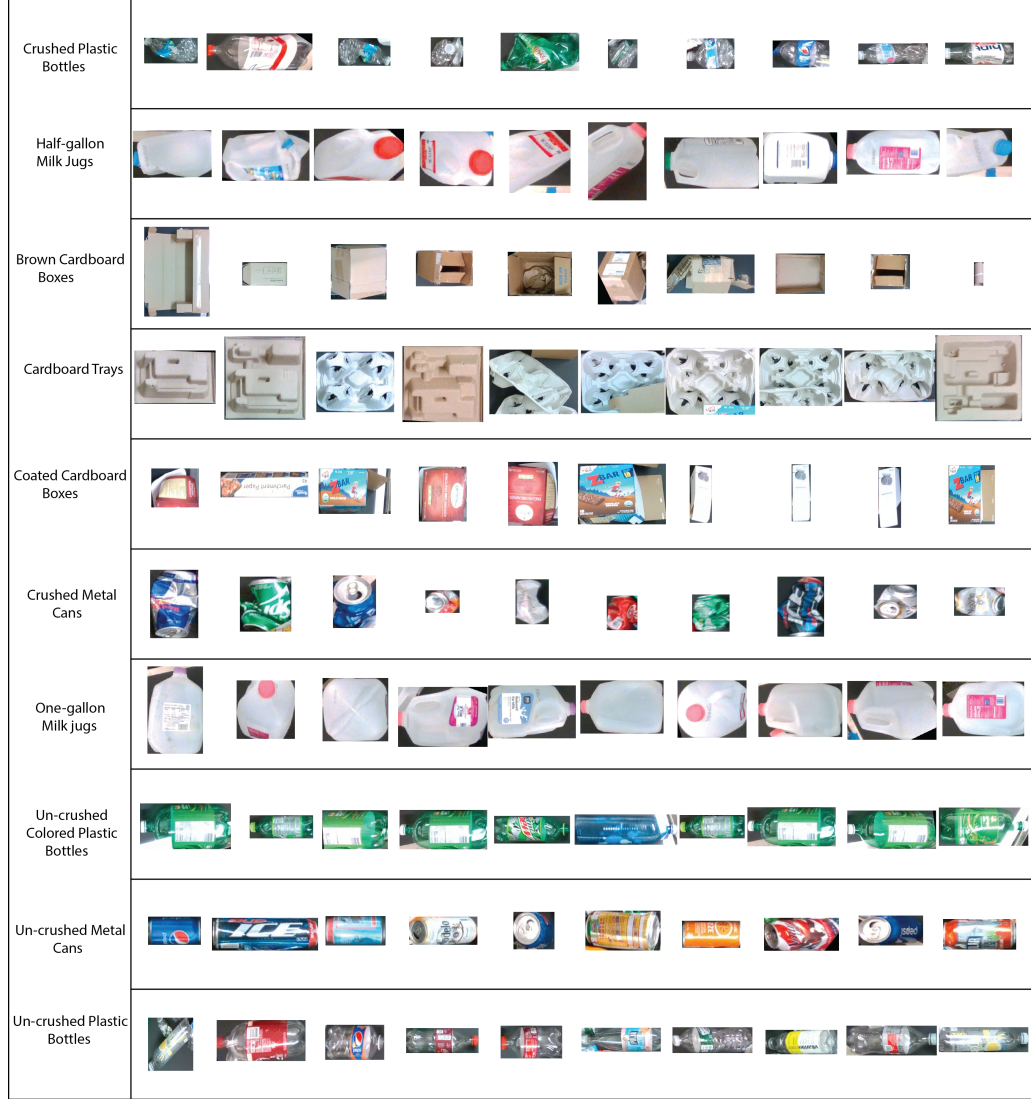


Figure 3.4: Sample Images from our Offline Dataset: Ten sample images from each of our ten object categories.

human-selected data from only one positive class is available during training [212]. Experiment 1 in Fig. 3.5 shows the F1-score of this supervised learning method to be significantly lower than our method.

3.5.4 Ablation Study

We conduct an ablation study of our method to evaluate how each component of the loss in eq. (3.3) contributes to the overall performance. Experiment 1 in Fig. 3.5

shows the F1-scores of only the human-supervised head across all categories. The human-supervised head performs poorly on its own likely because the data is heavily imbalanced between human-selected and non-human-selected objects. Experiment 2 in Fig. 3.5 shows the F1-scores of the instance projection head, which on its own is identical to SimCLR [54]. The high F1-scores across three human pools per category show that this head learns a robust visual representation for all categories. Experiment 3 in Fig. 3.5 shows that the cluster projection head performs poorer than the instance projection head.

To assess the impact of human supervision on each of our loss components, we jointly trained each of our self-supervised loss components (instance loss, cluster loss, and contrastive clustering) with our proposed loss that adds human supervision (eq. 3.3). For these experiments, we trained and evaluated models across three human pools per category (resulting in 30 human pools total). As explained in Section 3.5, each human pool consists of 40 randomly selected images belonging to a given category. Fig. 3.6a), 3.6b) and 3.6c) summarize the results. Each column in these figures denotes the human pool the method was trained with, and each row denotes the human pool the method was evaluated on. Each cell corresponds to the difference in F1-scores between the respective self-supervised method trained with human supervision and the self-supervised learning method independently.

Fig. 3.6a) and Experiment 4 in Fig. 3.5 show that the human-supervised head (eq. 3.3) does not improve the performance of the instance projection head across all categories. We attribute this result to conflicts between the two objectives. The instance loss maximizes the agreement between two augmentations of the same image while optimizing for pushing other instances away. In contrast, the human supervision head optimizes for pulling human-provided instances together. Fig. 3.6b) and Experiment 5 in Fig. 3.5 show that human supervision helps improve the performance of the model trained with the cluster loss for most categories. In this case, the objective functions

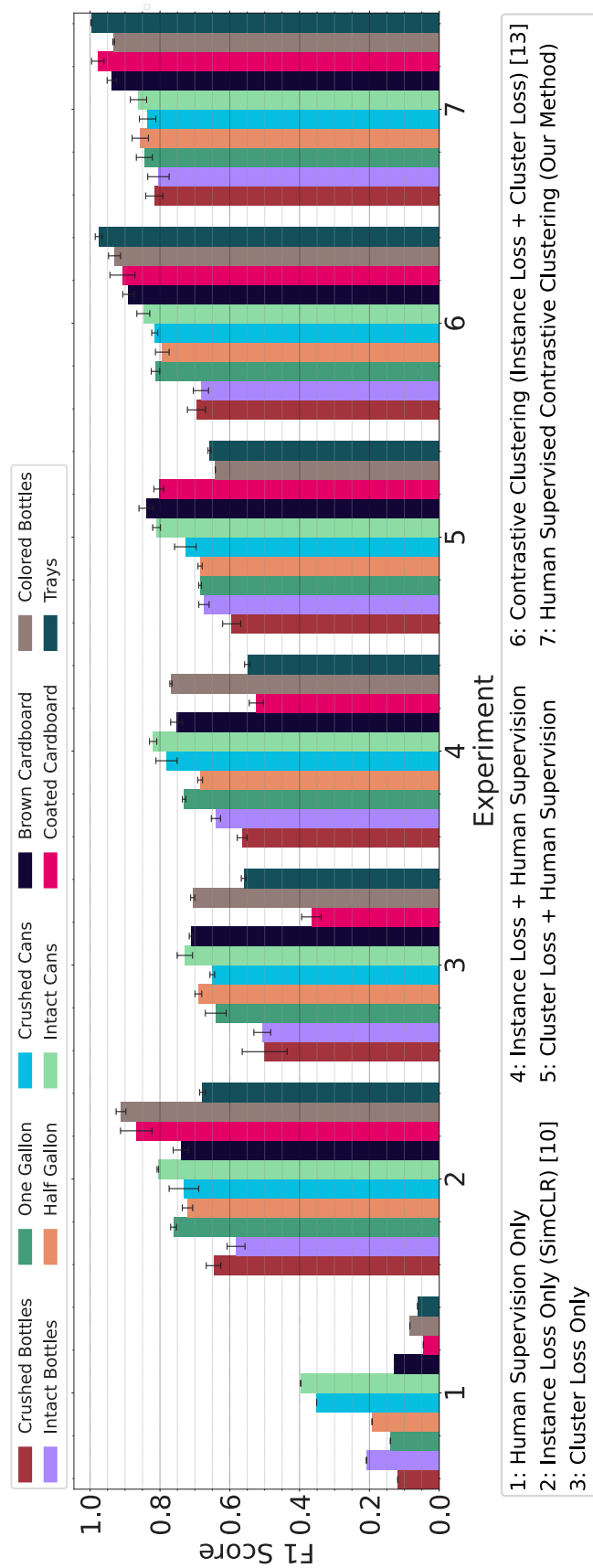


Figure 3.5: Results of our Offline Evaluation: Comparison with Baselines and Ablation Study aggregated over 3 human-selected pools per category

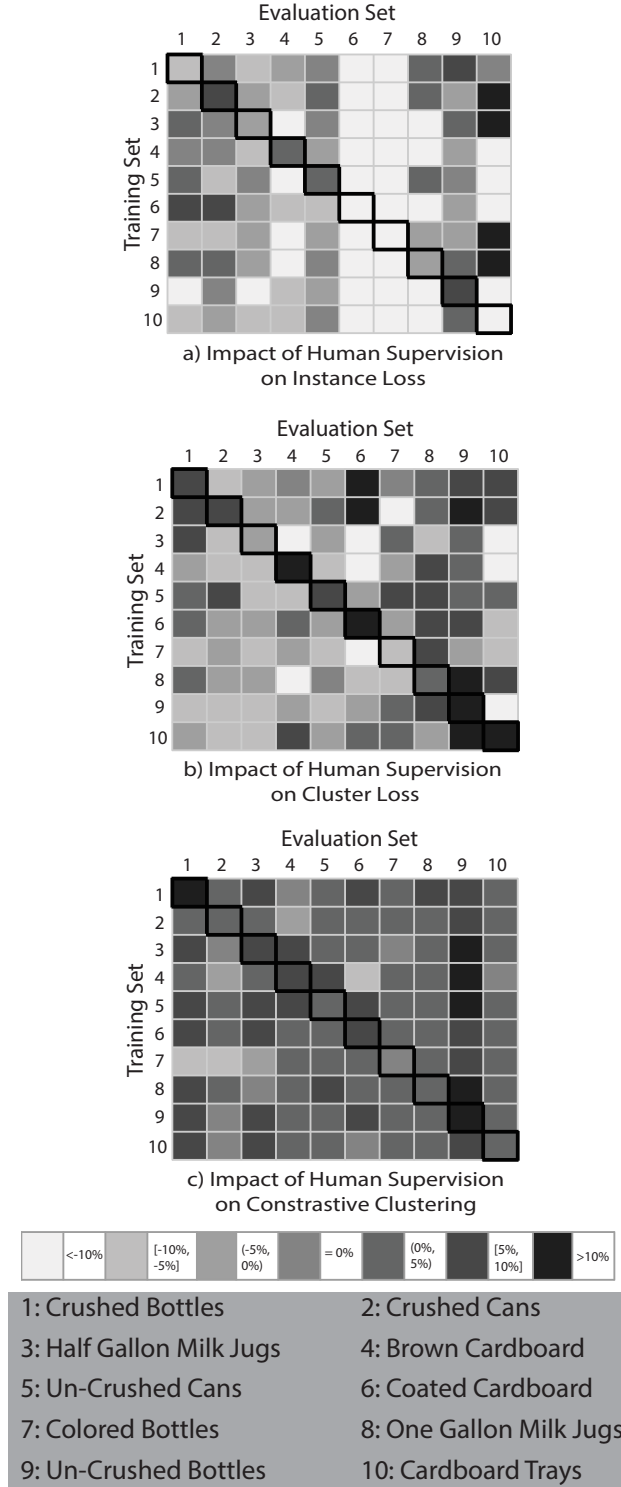


Figure 3.6: Impact of augmenting human supervision to contrastive learning: Difference in performance between self-supervised learning augmented with human supervision and the self-supervised learning model trained independently for instance loss, cluster loss, and contrastive clustering, respectively across categories. Specifically, w.r.t. the experiment numbers in Figure 3.5, a) represents Experiment 4 - Experiment 2, b) represents Experiment 5 - Experiment 3, c) represents Experiment 7 - Experiment 6.

align with human supervision supporting the formation of meaningful clusters. It is worth noting that when the human pools used for training and evaluation are the same (diagonal elements in Fig. 3.6b)), human supervision often helps improve the performance of cluster loss.

Finally, we show that our method slightly outperforms the self-supervised contrastive clustering method [162] (Experiment 6 in Fig. 3.5), showing an average improvement of 4.93% over all categories. Additionally, Fig. 3.6c) shows the performance of our method compared to contrastive clustering, which combines the instance and cluster loss. Our method performs well across categories, even when the human pools for training and evaluation belong to different categories (off-diagonal elements in Fig. 3.6c)). These results show that combining the losses that maximize agreement between both different views of the same instance (instance loss) and different instances belonging to the same cluster (cluster loss) benefit from even limited human supervision. Our human supervision loss facilitates learning a representation with more homogeneous clusters, as seen in the t-SNE plot in the supplementary material.

3.6 A Case Study with a Recycling Robot

This section describes in more detail the system we used to demonstrate our approach in a recyclable sorting task, as shown in Fig. 3.7. The robot learns the characteristics of objects that a human is sorting, working alongside this person in real-time. Afterward, we detail our experimental design, evaluation protocol, and results.

3.6.1 System Architecture

1. Object Detection: We predict instance segmentation masks over each object on the conveyor belt using the overhead cameras’ video streams as described in Section 3.5. We convert these masks to tightly-fitted rotated bounding boxes.

2. Detection of Picked Objects: This module determines which objects the

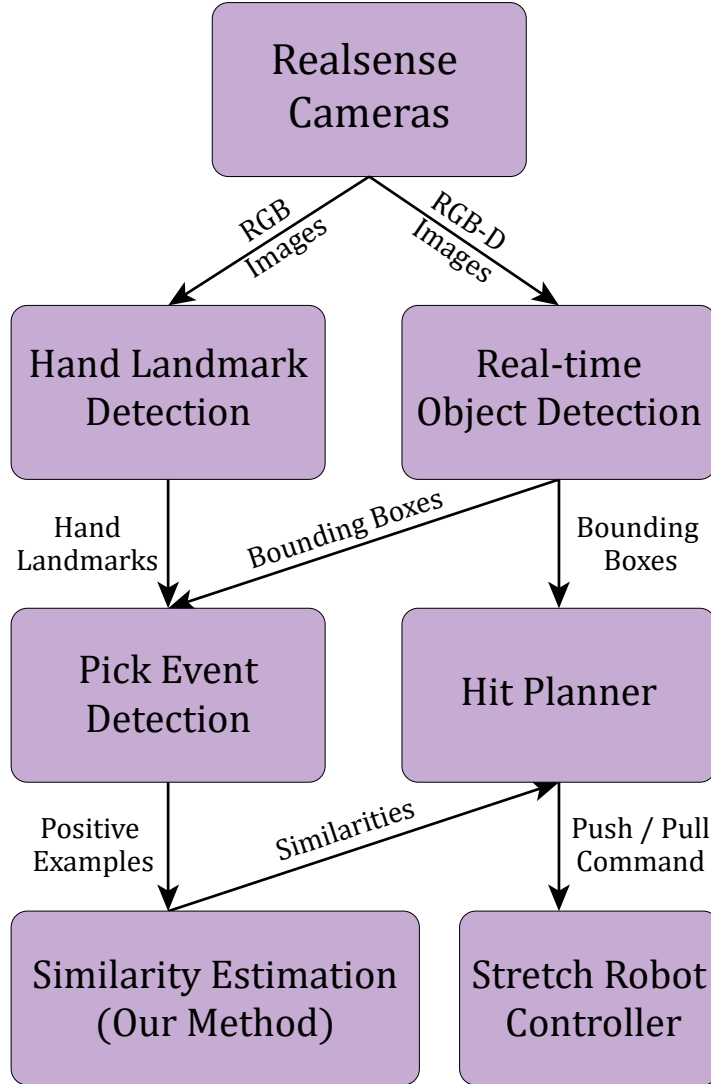


Figure 3.7: System Architecture

human is picking in real-time. This is done by identifying hand movements associated with an object being picked and then obtaining an un-occluded image of the object from previous frames in the video. The system uses the MediaPipe real-time hand tracking Application Programming Interface (API) and a ResNet-18 [116] image classifier to identify between three events: a picking event, no picking, and no hand present over the conveyor belt. To determine which item the human is picking in real-time, the model compares the placement of the person’s hand with the position of objects on the belt in a few prior frames.

3. Similarity Estimation (Our Method): Our approach allows the robot to learn the features of the objects picked by the person and to remove similar objects out of the items remaining on the conveyor belt. Our human-supervised contrastive clustering model is trained in real-time for 5 epochs on all the objects on the conveyor belt and the human-selected objects by optimizing for eq. 3.3. The robot then uses the trained learner to identify objects similar to the objects selected by the human. We run inference on every remaining object on the conveyor belt that was not picked by the human and calculate the pairwise cosine similarity between a given object and every object in the human-selected pool. We assign the average pairwise cosine similarity with the positive dataset to be the similarity score of the given object.

4. Hit Planner: A hit score is calculated for each object on the belt based on the similarity score of that object and the number of other objects that would likely be removed unintentionally should the robot try to remove the desired object from the conveyor belt. These so-called “casualties” are a consequence of our robot’s hardware only being able to push and pull objects off the belt rather than performing more complicated manipulation actions. The object with the highest hit score is targeted by the robot and removed from the belt.

3.6.2 Experimental Protocol and Results

One of the researchers removed recyclables of a given category from the moving conveyor belt. The robot system recorded the items that the human picked from an overhead camera. Once a certain number of items were removed from the stream, the robot trained on the data it had collected. The robot then began to remove items it deemed were similar to human-selected objects from the belt. The human and robot worked together on the same belt, and the robot continued to update its model as it watched the human pick more objects. When the human picked 30 objects from the recycling stream, the robot’s model was cleared, and the researcher trained the

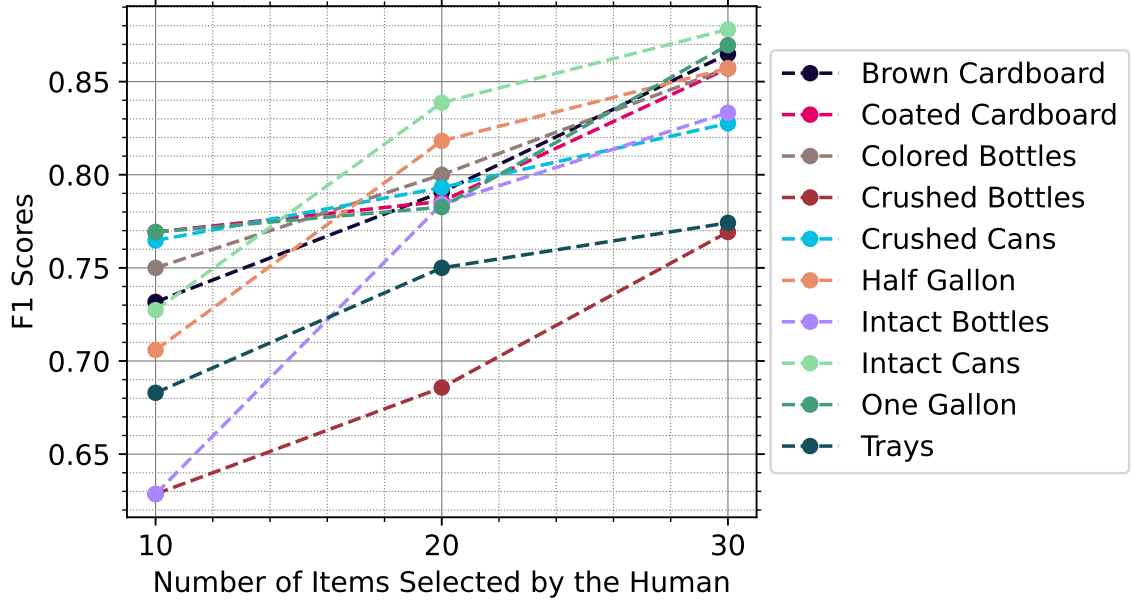


Figure 3.8: Model performance in real-time.

robot’s model for a different category of objects.

We perform inference on images of all objects on the conveyor belt (including the objects selected by the human using our trained model) to obtain their representation in the embedding space. We then compute the pairwise cosine similarity between each human-selected object and every remaining object on the conveyor belt, which are ‘candidate objects.’ For each candidate object, we average the cosine similarities between itself and the top-5 most similar human-selected objects and assign the average as the similarity score of that candidate object. These similarity scores for the candidate objects on the conveyor belt are leveraged by the hit-planner module to decide what objects are possible to remove with minimal casualties. We manually labeled the video recordings of the robot removing items of a given category to count the number of items correctly/incorrectly removed or missed to calculate the F1-score of the robot.

For sample efficiency in real-time, we initialize the robot’s model with a pre-trained contrastive clustering model trained on our offline dataset (described in Section 3.5)

with no human supervision. Fig. 3.8 shows the F1-score of the robot system across all ten categories after the researcher picks 30 items in increments of 10. On average, there were 90 ± 16 items on the belt for each round of evaluation (when the human picked ten items). For every 10 human-provided examples, the robot’s F1-score improved by 6 points on average across categories.

3.7 Limitations and Future Work

Our evaluation assumes that the human selects objects from the same category. Future experiments will include inter-category object selection and a wider participant pool outside the research team. Hardware constraints also impacted our evaluation. The robot’s limited range of motion, combined with its relatively slow speed, led to missed target items and accidental removals. Also, our system assumed the belt’s velocity was the same as every object’s velocity. When this assumption breaks, the robot miscalculates objects’ distances. These hardware constraints accounted for 36% of all missed items and 32% of all accidental removals across all rounds of evaluation. In the future, the system would benefit from using object tracking algorithms to help the robot identify the movement patterns of each object to make object removal more accurate. Additionally, we evaluated our approach on a relatively small diversity of objects compared to what is expected in a Materials Recovery Facility (MRF) on a relatively sparse and slow belt. Future work should evaluate the performance on a more diverse stream of objects on a fast-moving belt.

Aggregating human feedback at scale is an interesting extension of our work. For example, we imagine a situation in which humans observe objects near a robot via a camera feed and provide remote supervision on which objects to pick. While we don’t think this approach currently makes sense for the recycling domain due to economic constraints, future work could gather labels in this manner to accelerate

robot learning with our approach. Finally, even though our work focuses on recycling, our approach could also be applied to industrial quality control or the fruit-sorting industry. Broadly speaking, our approach could benefit applications in which humans and robots work jointly and systematically to manipulate objects. However, since we did not run explicit tests on any of these application domains, we tried not to speculate too broadly.

3.8 Summary

In this chapter, we proposed an approach to learn human-aligned visual representations by incorporating human supervision into a contrastive clustering model, aiming to better align a robot’s object representation with a human partner’s requirements. This alignment enables the robot to predict the human’s goal category of objects more accurately compared to baselines. The core idea is to treat the human’s selections during a shared task as sparse supervision: the robot updates its embedding space so that objects the human selects are pulled closer together, producing representations that capture the distinctions the human is implicitly communicating. We evaluated this approach in a simplified recycling-sorting case study where a human and a robot work together to sort recyclables, with the robot learning online from the human. In this setting, our method improved assistance performance relative to supervised and self-supervised learning baselines and showed that the robot can learn representations that are well-suited to the human’s requirements from only a few examples.

At the same time, the scope of this chapter is limited to adaptation from passive observation, where the robot can rely on informative evidence that naturally emerges from the human’s actions. This assumption becomes restrictive in multi-step, long-horizon tasks, where early behavior can be consistent with multiple goals, and it may be impractical for the robot to wait until the human’s intent becomes clear to

provide adequate assistance. The next chapter addresses this gap by moving beyond passive observation. Rather than only learning from what the human does, the robot uses active influence — selecting its own actions to obtain more informative evidence during execution, with the objective of establishing shared understanding earlier in a long-horizon sequential task.

Chapter 4

Active Goal Inference By Influencing Human Actions¹

The previous chapter focused on adaptation from passive observation, where the robot learned from the information implicit in a person’s actions. In this chapter, we study settings in which the same observed human behavior can be consistent with multiple possible human goals, so shared understanding cannot be established quickly from observation alone. We introduce Critical Decision Points (CDPs) — states in which many plausible goal-directed policies the human may be following, diverge maximally, so that the human’s next action yields highly discriminative evidence about which goal is being pursued [92]. To leverage CDPs without interrupting the interaction with explicit queries, we use receding-horizon planning to select robot actions that influence the human toward such states while optimizing for task progress. To show that this formulation is agnostic to interaction type, we evaluate it in both cooperative and competitive domains and show that our active approach to influence the human

¹Portions of this chapter were published as: *Ghose, D.* , Lewkowicz, M*., Dong, D., Cheng, A., Doan, T., Adams, E., Vázquez, M. and Scassellati, B., 2024, August. Planning with critical decision points: Robots that influence humans to infer their strategy. In 2024 33rd IEEE International Conference on Robot and Human Interactive Communication (ROMAN) (pp. 777-782). IEEE.* [92] *equal contribution

to CDPs performs better than passive baselines.

4.1 Introduction

Robots that interact with people in shared environments must understand the high-level intentions or goals that guide human behavior [121]. In both competitive and collaborative situations, a robot’s ability to anticipate what a person is trying to do or what their goal is, rather than merely reacting to their previous actions, enhances the efficiency of its actions and supports more natural interactions [73]. However, goal inference is challenging; the shared environment can be partially observable, sensing is often noisy, and many human high-level goals often share initial actions. Standard approaches that passively observe the human and update a Bayesian belief assume frequent, distinctive observations or fully observable worlds. Under sparse, ambiguous, or adversarial conditions, these methods converge too slowly, leaving the robot uncertain for a significant portion of the interaction.

To allow a robot to quickly understand a human’s goal early in the interaction, we have identified a key insight: *certain states provide much more informative cues about people’s objectives than others*. We call such states *Critical Decision Points (CDPs)*. A CDP is any state in the shared environment of the human and the robot where two or more plausible human strategies would prescribe different immediate actions, making the human’s next move highly revealing to a robot. Intuitively, at CDPs, subsequent actions of the human would clearly differentiate between competing goals.

We hypothesize that by steering the interaction toward CDPs, the robot can reduce uncertainty more rapidly than if it were to rely solely on passive observation. Thus, we propose that the robot use Receding Horizon Planning (RHP) [172] to actively influence the human toward CDPs will reveal their goal earlier in the interaction

than passive observation.

In this work, we show that the insight of identifying CDPs and influencing the human towards those CDPs is valuable in both competitive and collaborative tasks, in fully or partially observable environments, and remains effective whether the actions taken by the human and the robot are temporally independent or unfold over long action histories. We validate this idea in two scenarios: a collaborative task performed in a fully observable environment with strongly time-dependent actions, and a competitive task executed in a partially observable setting where actions are largely temporally independent.

In many real-world tasks, multiple overlapping action sequences can be used to accomplish different goals. For example, the steps for assembling a chair are similar to those for a stool. In meal preparation, making pancakes involves actions like mixing batter and pouring it onto a pan, which closely resemble those needed for muffins. This overlap can make it challenging for a robot collaborating with a human to determine a human’s specific goal, and make goal inference via passive observation unreliable. Moreover, traditional goal inference in human-robot collaboration has relied heavily on communication between humans and robots [144, 48, 198, 271, 250, 159, 206, 39, 53, 90, 174]. However, verbal communication between agents may not always be feasible due to various physical or situational constraints [93, 130, 166]. Even when verbal communication is possible, the task may be too complex to explain clearly, or humans and robots might interpret the same task description differently [5, 34, 86]. Disabilities or the impracticality of verbal communication in specific contexts, such as when it might interrupt another conversation or concurrent tasks the human might be doing, can also make it challenging for humans to convey their intentions directly [180].

To address the challenges of overlapping action sequences and the absence of verbal communication, we propose that the robot guide the human toward CDPs

which are states that could reveal their goal early in the collaboration. To guide the human towards CDPs, the robot expands a Receding Horizon Planning (RHP) tree, to imagine many possible reasonable future actions of the human and the robot, and selects an action that maximizes the amount of information gained while balancing task performance.

We evaluate our approach to influence a human toward CDPs in a collaborative cooking setup, both in simulation and with a physical robot, where a human and a robot prepare 1 of 30 meal recipes collaboratively. We chose cooking because it provides concrete ground truth for controlled experiments, supports comparison with prior work [38, 45, 277, 99, 87, 90], and includes many high-level goals that share initial action sequences. Results show that clarifying human intentions sooner enables the robot to adjust its actions more effectively, aligning them better with the human’s goal and reducing redundant or conflicting efforts. We compare our approach with prior works that have addressed the challenge of goal inference using various strategies. Information gain maximization methods, such as [230], focus on influencing the human towards taking actions that increase the robot’s information gain without fully considering whether the robot’s actions align with the human’s true goal, leading the robot to take many incorrect actions during collaboration, especially when goals have overlapping action sequences. Bayesian Delegation [292], on the other hand, continuously updates its belief over possible goals based on observed actions. However, the method tends to overemphasize these overlapping action sequences, diluting the belief in possible goals and making it harder for the robot to disambiguate between them confidently.

Next, we propose a method for the robot to influence a human to reveal their goals at CDPs using Receding Horizon Planning (RHP) [172] in a competitive task. We demonstrate our approach in a partially-observable setting through a hide-and-seek game between a human and a robot (Fig. 4.11). In this game, the seeker robot tries to

catch a human-controlled hider by estimating their strategy. Since the environment is partially observable, we demonstrate how CDPs can be precomputed over the spatial state space.

Our experiments in simulation and our demonstration in the real world show that when the robot seeker influences the hider toward CDPs, the seeker can catch the hider more quickly than when the robot attempts to estimate the human’s strategy while randomly exploring the map or minimizing the distance between itself and the hider whenever the hider is visible. Overall, our findings suggest that robots can leverage the properties of a shared environment in interactive human-robot tasks to estimate a human’s strategy under partial observability.

In summary, our contributions are:

- **Main insight.** We show that human intentions can be revealed much earlier if the robot *actively drives the interaction toward a small set of highly informative states*. We call these states Critical Decision Points (CDPs), states where competing human strategies prescribe different immediate actions.
- **CDP formulation and planner.** We provide a formal, goal-conditioned definition of CDPs and an accompanying Receding-Horizon planner that steers the human toward them while trading off task progress and information gain.
- **Comprehensive evaluation.** We evaluate the same CDP-based goal inference framework in two settings with contrasting interaction assumptions: a collaborative, fully observable cooking task with time-dependent actions, and a competitive, partially observable hide-and-seek task with largely independent actions. To our knowledge, this is the first unified goal-prediction approach shown to operate effectively across both cooperative and competitive interactions, yielding earlier and more accurate inference than baselines.

4.2 Related Work

4.2.1 Robots Predicting Human Goals

Goal Prediction in Human-Robot Collaboration

Inferring human goals and predicting their actions is crucial for effective human-robot collaboration [121]. Researchers have studied this in contexts such as predicting handover goal positions [315, 59, 157, 86], reasoning over human plans in collaborative workspaces [122, 198, 109, 77, 3, 270], inferring intentions in shared autonomy [138, 140, 218, 15, 14, 70], and predicting motion for social navigation [261, 23, 153, 262].

Several key approaches have been employed to infer human goals and intentions in human-robot collaboration. These include direct Bayesian inference, which calculates the posterior probability of each possible goal, either in a single step [211, 138, 134, 140, 79] or through dynamic processes like Markov models or Markov Decision Processes (MDPs) [122, 64, 306, 169, 23, 154, 125]. Another group of methods uses supervised learning techniques, such as neural networks, to map observations to probabilities over intentions using training data [236, 109, 273, 65, 219]. Freedman et al. [83] propose an approach for the robot to execute a weighted combination of precomputed policies for different goals based on its belief. These methods generally take a passive approach to human goal inference, relying heavily on the observation of human actions to deduce intent after the actions have been performed.

Researchers have also explored more active strategies to reveal human goals. One line of work casts the robot as a passive observer that intervenes solely to disambiguate human goals, without contributing to task execution [11, 242]. Another approach is Bayesian Delegation [292], which uses Bayesian inference to predict a human’s response to a robot’s action. Prior work has also demonstrated that robots can take information-gathering actions to better understand human internal states for autonomous driving tasks using Model Predictive Control (MPC) [230], without con-

tributing to shared task completion. In this work, we propose Critical Decision Points to enable the robot to quickly infer human goals during a sequential human-robot collaboration task in an online manner, particularly for tasks where overlapping action sequences result in multiple goals. We compare our method with an Information Gain Maximization approach [230] and Bayesian Delegation [292] for a collaborative cooking task, mimicking real-world cooking tasks where multiple goal recipes could contain many overlapping action sequences, and show that our method is more effective at inferring human goals earlier during the task.

Opponent Modeling

In competitive settings, opponent modeling is used to build a model of an opponent’s behavior using observations of their actions [186, 265, 299]. Over the years, opponent modeling has been extensively studied for applications such as playing stochastic games between agents [282], negotiations between agents [187], and for cooperative tasks [285, 297]. For human-robot interaction tasks, prior work uses Theory of Mind [202, 221, 282] to recursively reason about the human’s actions based on the robot’s behavior. Some methods use inverse reinforcement learning [1] to estimate a human’s reward model [36, 40] and then use the reward model to estimate optimal robot behavior via reinforcement learning techniques. Bisson et al. [30] treat the robot as a reactive observer in competitive games, using handcrafted provocative events to disambiguate the opponent’s goal without actively engaging in the task. Therefore, we build on these ideas by enabling the robot to participate in the game actively. Our robot uses a Bayesian approach to continuously estimate the human’s strategy while steering the interaction toward Critical Decision Points—states where human behavior is most revealing of their underlying goal.

4.2.2 Robots Influencing Humans

In recent years, there has been a growing interest in enabling robots to select actions that influence human behaviors [286, 92, 264, 161, 274, 232, 199, 205, 49, 30] in both collaborative and competitive scenarios. A common approach is to explicitly model the human’s reward [1] and then optimize for robot actions that, at least in the near future, steer the human towards states where they are more likely to act in a desired manner [229]. More recently, high-level strategies of humans have been inferred from repeated interactions using unsupervised learning [207, 293] or theory-of-mind approaches [78, 241], with the inferred strategies being used to influence their behavior. Prior work has also leveraged offline reinforcement learning to train robots to influence humans more optimally [124, 286, 204]. Building upon this line of research, we frame the problem of influencing humans to reveal their true goal as a Receding Horizon Planning problem [172, 225]. We do this by enabling both competitive and collaborative robots to select actions that drive a human towards states where their actions are more revealing about their goal.

4.3 Influencing Humans to Reveal their Goals at Critical Decision Points

4.3.1 Preliminaries

State Space

We consider a robot working alongside a person in a shared environment and aim to enable the robot to infer the person’s high-level strategy or goal. We define the full environment state to be $s = (s_w, s_h, s_r)$, where s_w is a discrete description of the world (for example, an occupancy grid or the locations of key objects), s_h is the human’s state (e.g., position or orientation), and s_r is the robot’s state. The robot

does not directly observe s_h ; instead, it receives observations o_h through its sensors (for example, an RGB-D camera or lidar), which may be noisy or partial depending on the task context. If the environment were fully observable, we assume the robot could have access to the full history $\mathcal{H}$ of all the actions taken during the interaction by the human. In a partially observable environment, since s_h is not always complete, we cannot assume access to the full history $\mathcal{H}$. Additionally, to balance participation between the human and the robot, we assume the interaction is turn-based.

Goal and Policy Bank

We assume a finite *goal set* $\mathcal{G} = \{g_1, \dots, g_n\}$, where each g denotes a distinct high-level goal the human might pursue (for example, a particular destination in a navigation task or a specific recipe in a cooking task). To each goal g we associate a distribution of policies $\mathcal{P}_g$, where each policy in $\mathcal{P}_g$ is possibly stochastic and captures “how the human would act if their true objective were g ”. Concretely, assume that $\mathcal{P}_g$ has n policies. Each policy $i \in [1, n]$ in $\mathcal{P}_g$, π_g^i , defines the human’s next action $a_h \in \mathcal{A}_h$, conditioned on the current state s and, when relevant, the interaction history H , where $\mathcal{A}_h$ signifies all possible human actions. Formally,

$$\pi_g^i(a_h \mid s, H) = P(a_h \mid s, H, \text{goal} = g).$$

In the remainder of this paper, we refer to $\mathcal{P}_g$ as the *policy bank* corresponding to goal g . For example, in competitive navigation, the policy bank $\mathcal{P}_{\text{corner}}$ might include several different evasion paths all ending at the same goal location *corner*. In collaborative cooking, $\mathcal{P}_{\text{omelette}}$ could encompass different valid ingredient-ordering strategies for making an *omelette*. Intuitively, each policy in the bank defines a mapping from a state (and optionally history) to an action distribution. The policy bank includes realistic ways a human might act to achieve the goal in the environment.

Finally, we denote *full policy bank* as the union of policies for every goal $g \in \mathcal{G}$, $\mathcal{P} = \bigcup_{g \in \mathcal{G}} \mathcal{P}_g$. We also assume the human’s actual goal-policy π_g^* remains constant throughout each interaction (i.e. they do not switch goals mid-task).

Belief over Goals

At each time step, the robot maintains a belief

$$b_{t-1}(g) = P(\text{goal} = g \mid \mathcal{H}_{t-1}),$$

where $\mathcal{H}_{t-1}$ is the sequence of observed human actions up to time $t - 1$. After the robot observes the human perform action a_h^t in state s_t , it updates this belief using a Bayes filter. We assume a uniform prior over goals, meaning all goals are equally likely before any actions are observed. This prior is folded into the belief $b_{t-1}(g)$, so subsequent updates do not explicitly show a prior term.

Each goal g can be accomplished not by a single policy but by executing a policy from a *policy bank* $\mathcal{P}_g = \{\pi_g^1, \dots, \pi_g^{n_g}\}$. Hence, the likelihood of an action must marginalise over all policies in that bank:

$$\bar{\pi}_g(a \mid s, H) = \frac{1}{|\mathcal{P}_g|} \sum_{i=1}^{n_g} \pi_g^i(a \mid s, H).$$

Intuitively, $\bar{\pi}_g$ answers the question: “*If the human’s goal were g and they could be following any valid policy in $\mathcal{P}_g$ with equal prior weight, how likely is the action I just saw?*” The belief update is therefore

$$b_t(g) = \frac{\bar{\pi}_g(a_h^t \mid s_t, \mathcal{H}_{t-1}) b_{t-1}(g)}{\sum_{g'} \bar{\pi}_{g'}(a_h^t \mid s_t, \mathcal{H}_{t-1}) b_{t-1}(g')}.$$

We can also reinterpret the Bayes update in terms of observed state transitions rather than actions. In this view, the robot asks: “*What is the likelihood of reaching*

state s_t , if the human were pursuing goal g ?". If the transition model were deterministic $T(s_t | s_{t-1}, a_h^t) = 1$, $P(s_t | g)$ would be computed as:

$$\begin{aligned} P(s_t | g) &= \sum_a T(s_t | s_{t-1}, a) \bar{\pi}_g(a | s_{t-1}, \mathcal{H}_{t-1}) \\ &= \bar{\pi}_g(a_h^t | s_{t-1}, \mathcal{H}_{t-1}), \end{aligned}$$

so the standard Bayes filter gives

$$b_t(g) \propto P(s_t | g) b_{t-1}(g).$$

Here $b_{t-1}(g)$ is the prior at time $t - 1$, and the likelihood $\bar{\pi}_g$ (or equivalently $P(s_t | g)$) is estimated empirically from state-visitation frequencies generated by all policies in $\mathcal{P}_g$.

4.3.2 Critical Decision Points

The primary insight of our work is that certain states in the environment are inherently more revealing of the human's underlying goal than others. We call these states *Critical Decision Points (CDPs)*. Different goal-conditioned policies prescribe markedly different next actions at a CDP, so observing which action the human actually takes yields maximal information about their true objective.

Mathematically, suppose our goal set is $\mathcal{G} = \{g_1, \dots, g_n\}$ and each goal g_i is paired with a policy π_g^i . At any state s , and given interaction history H if available (in fully observable environments), each goal is represented by the marginal action distribution $\bar{\pi}_g^i$ as described in eq. 4.3.1 over the human's next action. We measure the total pairwise disagreement among these distributions by

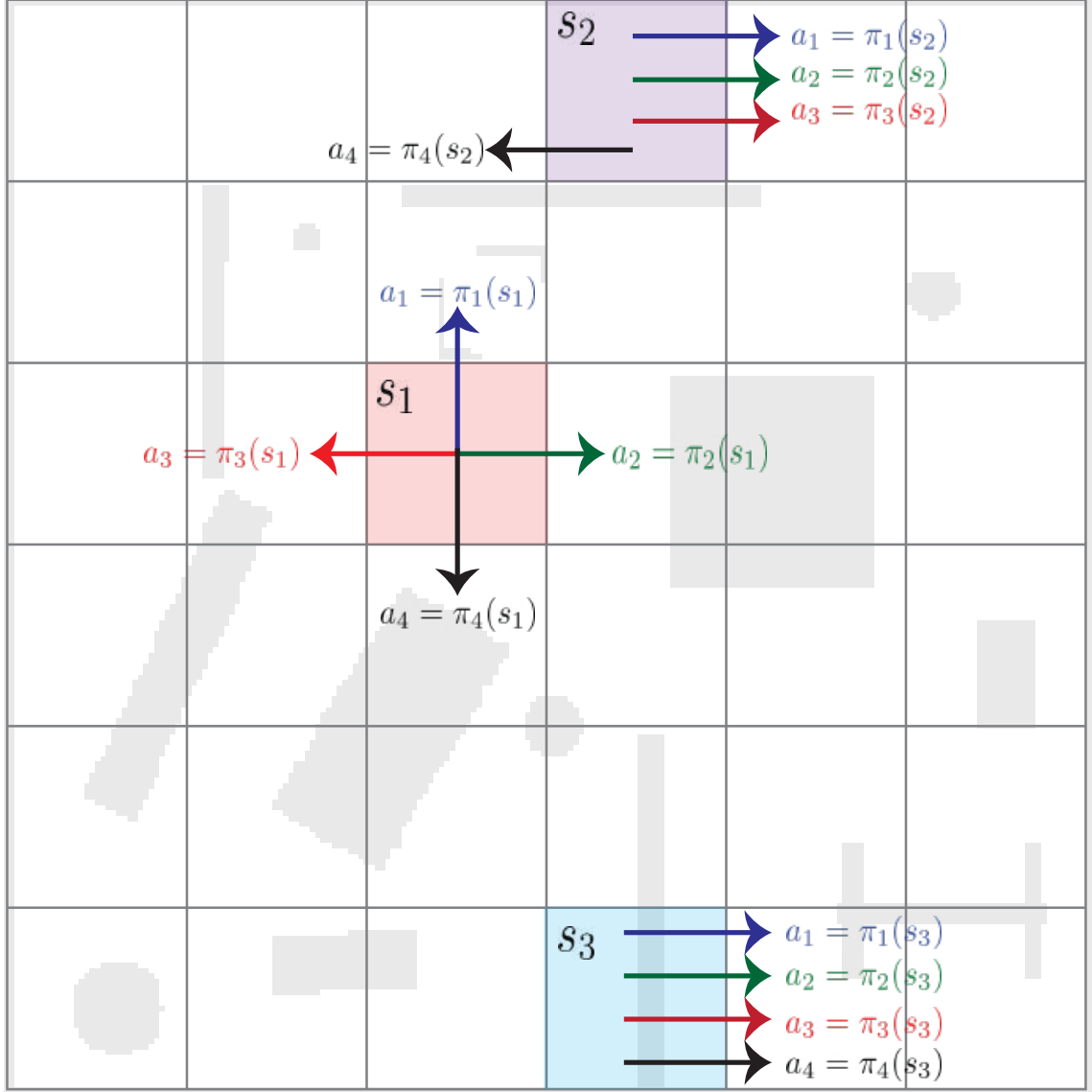


Figure 4.1: Critical Decision Point Computation. For every policy of the human in the policy bank, we compute what states in the environment produce the most divergent actions. For instance, if there were four reasonable policies the human could be following, at s_1 , all four of them produce different actions, making s_1 a Critical Decision Point (as compared to s_2 and s_3 where most policies predict the same action).

$$M(s, H) = \sum_{1 \leq i < j \leq n} D(\bar{\pi}_g^i(a_h \mid s, H), \bar{\pi}_g^j(a_h \mid s, H)),$$

where $D(p, q)$ is any symmetric divergence between two probability distributions (for example, the total-variation or ℓ_1 distance or a raw count of the number of policies diverging at a state). This total, $M(s, H)$, captures the extent of disagreement among goal-conditioned policies to identify states where multiple policies diverge meaningfully. A state s becomes a CDP when this disagreement between policies exceeds a chosen threshold θ :

$$S_{\text{CDP}}(H) = \{s \in \mathcal{S} \mid M(s, H) > \theta\},$$

Intuitively, in Fig. 4.1, there are four policies, each leading to a different goal. At state s_1 , each policy selects a distinct action—up, right, left, and down—so the pairwise divergence $M(s_1, H)$ would be 4, indicating high disagreement and making s_1 a CDP. In contrast, at s_3 , all policies choose the same action (right), yielding a divergence of 0 and making s_3 uninformative for inferring the human’s true goal.

CDPs can be precomputed offline when no access to history is available by evaluating $M(s, \emptyset)$ once for every state and storing those above the threshold, and can also be computed online during interaction by recalculating $M(s, H)$ with the current history.

Our central hypothesis is that by steering the human and the shared interaction toward these CDPs, the robot can observe the most distinguishing human actions and thus reduce its uncertainty over the human’s goal far more quickly than by passive observation alone.

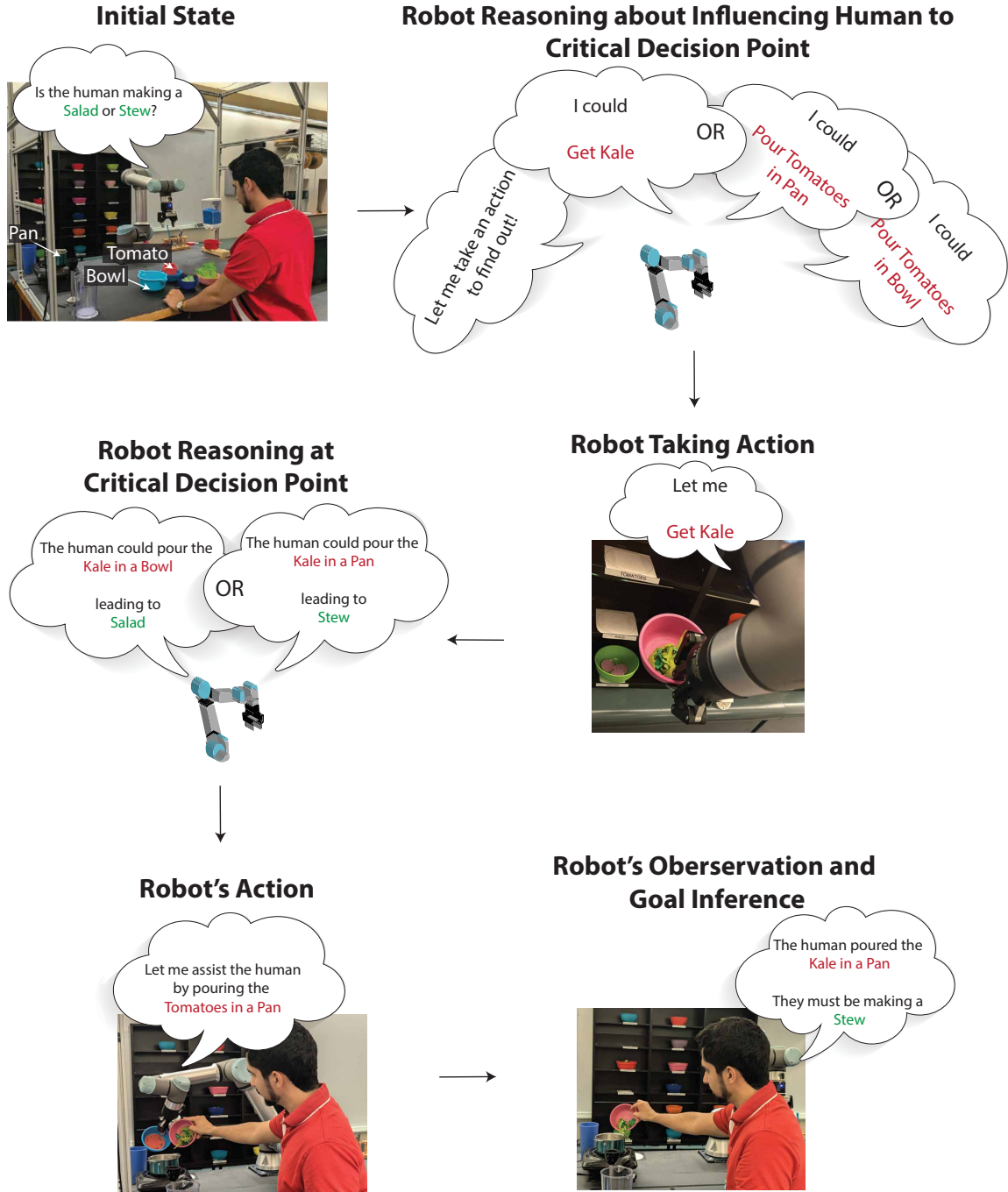


Figure 4.2: Robot reducing uncertainty about the Human's Goal at a CDP: As shown by the initial state, ingredients and utensils common to salad and stew are collaboratively brought by the human and the robot in the workspace. The robot reasons about possible next actions to influence the human to reveal their goal. The robot takes an action to complete gathering all the ingredients, rather than adding them to the pan or bowl, to observe the human in a CDP. This allows the human's next action (either adding the kale to the pan or the bowl) to reduce uncertainty about the goal. The human clarifies their goal through their action, and the robot takes action to support the human.

Algorithm 1 *Overview* - Influencing the Human toward CDPs with RHP

Input: Goal set $\mathcal{G} = \{g_1, \dots, g_n\}$; policy banks $\mathcal{P}_g = \{\pi_g^1, \dots, \pi_g^{n_g}\}$; horizon D_H ; weights $\lambda_{1,2}$; divergence D ; task cost C_{task}

Output: Next robot action a_r^t

```
1 Init:  $b_0(g) \leftarrow 1/|\mathcal{G}|$ ,  $\mathcal{H} \leftarrow \emptyset$ ,  $t \leftarrow 0$ 
2 while task not finished do
    // 1) Sense
3   observe  $s_t = (s_w, s_h, s_r)$ 
    // 2) Expand depth- $D_H$  tree
4   Alternate levels of:
      • robot branches over  $a_r \in \mathcal{A}_r$ ,
      • human samples  $a_h \sim \bar{\pi}_g(a_h | s, H)$   $\bar{\pi}_g(a | s, H) = \frac{1}{|\mathcal{P}_g|} \sum_k \pi_g^k(a | s, H)$ .
      Store every leaf  $\ell$  with  $s_\ell$  and  $\mathcal{H}_\ell$ 
    // 3) Score leaves
5   foreach leaf  $\ell$  do
6     | Informativeness Score:  $M_\ell \leftarrow \sum_{g^i < g^j} D(\bar{\pi}_g^i(a_h | s_\ell, \mathcal{H}_\ell), \bar{\pi}_g^j(a_h | s_\ell, \mathcal{H}_\ell))$  Task
6     | Score:  $C_\ell \leftarrow C_{\text{task}}(s_\ell, \mathcal{H}_\ell)$   $J_\ell \leftarrow \lambda_1 C_\ell + \lambda_2 M_\ell$ 
7   end foreach
    // 4) Act
8    $\ell^* \leftarrow \arg \min_\ell J_\ell$ ;  $a_r^t \leftarrow$  first robot step on path  $\ell^*$ ; execute  $a_r^t$ 
    // 5) Update belief
9   observe  $a_h^t$   $b_{t+1}(g) = \frac{\bar{\pi}_g(a_h^t | s_t, \mathcal{H}) b_t(g)}{\sum_{g'} \bar{\pi}_{g'}(a_h^t | s_t, \mathcal{H}) b_t(g')}$   $\mathcal{H} \leftarrow \mathcal{H} \cup \{a_h^t\}$ ,  $t \leftarrow t+1$ 
10 end while
```

4.3.3 Influencing the Human to Critical Decision Points

We develop a method for the robot to reduce the uncertainty of its belief over the human’s current goal, given the set of reasonable human goals in the goal bank $\mathcal{G}$, in both competitive and collaborative tasks. We frame the above problem as a discrete-time planning problem, which we solve using a Receding Horizon Planning (RHP) approach [172]. RHP takes inspiration from Model Predictive Control [85] but concerns discrete decision-making rather than continuous control. Generally, to perform RHP, an agent iteratively solves a planning problem over a receding horizon, considering possible future actions by expanding a search tree. After planning on a given time step, the first action leading to the best possible outcome is executed, the horizon window moves forward, and planning is performed again for the next possible

futures.

Algorithm 1 summarizes our RHP-based approach to influence the human toward CDPs. At the start, the robot has no prior belief over the human’s objectives: it initializes a uniform belief over all goals and an empty history of observed human actions. The robot first perceives the current joint state $s_t = (s_w, s_h, s_r)$, where s_w encodes the world state, s_h the human’s (possibly noisy) state, and s_r the robot’s own configuration.

Next, the robot builds a depth- D_H look-ahead tree of future states with the current state as the root. The tree consists of alternate robot and human moves. The robot action branches over every feasible action that can be taken from the current state. The potential human response to every robot action is drawn from the most likely goal-conditioned policy $\bar{\pi}_g$ the robot believes the human is following.

Each leaf ℓ is then given two scores: a *task cost* C_ℓ that quantifies how well the robot’s path advances the human’s goal, and an *informativeness cost* M_ℓ that measures how strongly the goal hypotheses disagree on the human’s next move at s_ℓ . Crucially, this same M is the divergence-based measure used to define Critical Decision Points, states where different goals prescribe markedly different actions using Eq. 4.3.2. In other words, when M_ℓ is large, s_ℓ lies in the CDP set, and observing the human there would yield maximal clarity about their true goal. The planner combines these into a single objective to favor futures that both progress the task and steer the human toward CDPs. The robot selects the leaf with the lowest combined cost, executes the first robot move along that path, and waits for the human to respond. Upon observing human actions after the robot has taken them in the real world, the robot refines its posterior over goals via Bayes’ rule and appends the new action to its history. Incrementing t , the tree is then expanded again to continually guide the interaction toward states that both support the task and most rapidly reduce uncertainty about the human’s intentions.

Algorithm 2 Scenario 1: Selecting Action at Timestep t to Influence Humans to CDPs by expanding an RHP tree for Human-Robot Collaboration

Input: Goal set $\mathcal{G} = \{g_1, \dots, g_n\}$; per-goal policy banks $\mathcal{P}_g = \{\pi_g^1, \dots, \pi_g^{n_g}\}$;

Current joint history $\mathcal{H}^{0:t-1}$ of all observed actions;

Planning horizon D_H ; weights (λ_1, λ_2) ;

Divergence $D(\cdot, \cdot)$; task-cost $C_{\text{task}}(\cdot)$

Output: Next robot action a_r^t

```

11  $b_t(g) \leftarrow 1/|\mathcal{G}|$  (no prior bias),  $\mathcal{H} \leftarrow \mathcal{H}^{0:t-1}$ 
12 while task not finished do
    // 1) Observe
13    sense  $s_t = (s_w, s_h, s_r)$ 
    // 2) Expand depth- $D_H$  look-ahead tree
14    Alternate robot branches ( $a_r \in \mathcal{A}_r$ ) with human samples  $a_h \sim \bar{\pi}_{\hat{g}}(a_h \mid s_t, \mathcal{H})$ , where
         $\bar{\pi}_g(a \mid s, H) = \frac{1}{|\mathcal{P}_g|} \sum_k \pi_g^k(a \mid s, H)$ .
        Store each leaf  $\ell$  with state  $s_\ell$ , augmented history  $\mathcal{H}_\ell$  (i.e.  $\mathcal{H}$  extended with simulated
        actions), and future sequence  $\mathcal{F}_\ell$ .
    // 3) Score leaves
15    foreach leaf  $\ell$  do
16         $M_\ell \leftarrow \text{dissimilarity}(\text{bel}(\mathcal{H}_\ell^{0:t-1}), \text{bel}(\mathcal{F}_\ell^t))$ 
17         $C_\ell \leftarrow G(\mathcal{F}_\ell^t)$ 
18         $J_\ell \leftarrow \lambda_1 C_\ell + \lambda_2 M_\ell$ 
19    end foreach
    // 4) Execute robot move
20     $\ell^* \leftarrow \arg \min_\ell J_\ell$ ;  $a_r^t \leftarrow$  first robot step on path  $\ell^*$ ; execute  $a_r^t$ 
    // 5) Observe human and update belief
21    observe  $a_h^t$   $b_{t+1}(g) = \frac{\bar{\pi}_g(a_h^t \mid s_t, \mathcal{H}) b_t(g)}{\sum_{g'} \bar{\pi}_{g'}(a_h^t \mid s_t, \mathcal{H}) b_t(g')}$   $\mathcal{H} \leftarrow \mathcal{H} \cup \{a_h^t\}$ ;  $t \leftarrow t + 1$ 
22 end while

```

4.4 Scenario 1: Influencing the Human to Reveal their Goal in Human-Robot Collaboration

4.4.1 Approach

Alg. 2 describes our Receding Horizon Planning (RHP) approach for selecting the robot's action at a given timestep, such that the robot could influence the human to take more distinguishing actions at CDPs during human-robot collaboration (see Fig. 4.2 for an example scenario in a collaborative cooking task).

First, we expand an RHP tree with the current interaction sequence (i.e., the

history of actions leading up to the last action taken by the human) as its root. To do this recursively, we iterate through every leaf node in the tree, which is a sequence of possible future actions both agents can take from the root node of the tree. Then, for each valid action from the corresponding state, we execute that action and add the resultant state as a child node to the selected leaf node (lines 5-6). We continue this process until we reach depth $\mathcal{D}_H$. Then, we observe the history of actions taken only by the human, and use this to calculate the cost for every leaf node according to Eq. 5.3.2 (lines 7-11). Finally, we return the first action leading to the leaf node with the lowest cost (lines 12-13), observe the human’s next action (line 15), and update the robot’s belief (line 16).

We steer the robot in the collaboration towards taking actions that are 1) unlikely to be detrimental to achieving the human’s goal and 2) likely to significantly alter the robot’s belief about human goals.

For the first consideration, we induce the robot to take general actions that are common for reaching multiple goals. Intuitively, as the task progresses, the robot’s tendency to act generically reduces the availability of general actions for the human, effectively pushing them to take specific actions that could clarify their true goal. We formalize this via a generality score $G(\mathcal{F}^t)$, where $\mathcal{F}^t$ is a sequence of future actions taken from the state s^t . For a given sequence $\mathcal{F}^t$, the score is computed by estimating how often the policies in the bank $\mathcal{P}$ would result in the same sequence of actions as in $\mathcal{F}^t$.

For the second consideration regarding the belief about human goals, we calculate a dissimilarity measure that enables the robot to identify actions that maximize changes to this belief. We compute this dissimilarity measure as the disagreement of actions available from each state in the state space, as given in Eq. 4.3.2. This dissimilarity measure takes into account the action history of an interaction up to state s^t , which we refer to as $\mathcal{H}^{0:t-1}$, and a possible sequence of future actions $\mathcal{F}^t$

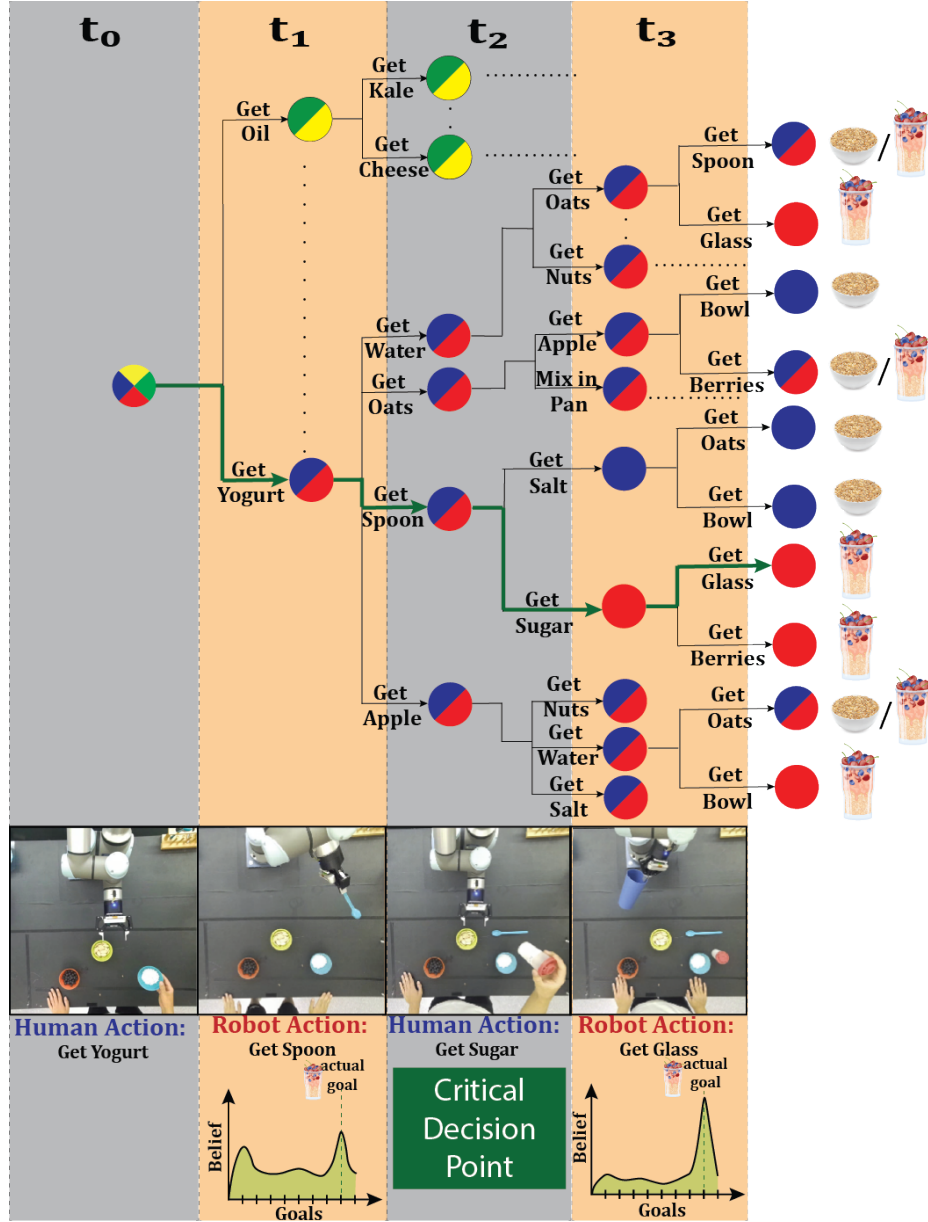


Figure 4.3: Selecting actions to infer the human’s goal at a CDP for a human-robot collaborative cooking task: A Receding Horizon Planning (RHP) tree of all possible actions the human and the robot can take in a turn-based interaction is expanded, where the human takes the first turn. The circles correspond to states, and the arrows correspond to the possible action sequences taken at a given time step. The color of the circles indicates the possible goals that can be reached from a given state. The belief distribution is shown at steps t_0 and t_2 , as the robot updates its belief over the human’s goals when a human action is observed. The green arrows show the actions that would optimize the cost function in Eq. 5.3.2. See the text for more details.

from s^t :

$$divergence := dissimilarity(bel(\mathcal{H}^{0:t-1}), bel(\mathcal{F}^t))$$

where $bel(\cdot)$ corresponds to the likelihood over possible human goals induced by the policies in $\mathcal{P}$ given the input sequence, either past or future actions, and is computed using the Bayes filter from Eq. 4.3.1. In this work, we calculate dissimilarity using cosine similarity; however, other measures, such as Wasserstein distance, could also be used as an alternative. Notably, this dissimilarity measure can be calculated online as the interaction progresses, if the state space is prohibitively large.

Taking both considerations together, the robot ends up acting by choosing a branch of the RHP tree that minimizes the weighted combination of task and informativeness costs:

$$C = \underbrace{\lambda_1 * G(\mathcal{F}^t)}_{\text{task cost}} + \underbrace{\lambda_2 * dissimilarity(bel(\mathcal{H}^{0:t-1}), bel(\mathcal{F}^t))}_{\text{informativeness cost}}$$

where, λ_1 and λ_2 are empirically determined hyperparameters. This cost is implemented in lines 16-17 of Alg. 2, where the input sequence of past actions is computed in line 15, and the future possible actions are computed in lines 3 to 14.

Fig. 4.3 illustrates how our proposed approach works. A tree of all possible actions the human and the robot can take in a turn-based interaction is expanded, where the human takes the first turn. After t_0 , the robot is already biased towards either the parfait or oatmeal being the human’s goal, so it chooses a more general overlapping action (gathering a spoon) to influence the human into a CDP, which would be either gathering the salt or the sugar. At the CDP (t_2), the human would gather sugar, which disambiguates between the two most likely goals, leading the robot to update its belief and confirm the parfait as the goal. The robot can then align its actions with the actual objective, which in t_3 is gathering the glass. The green arrow shows

the path that minimizes the cost. The robot would take the first action on this path, observe the environment, and reconstruct the tree at the next step.

4.4.2 Task and Problem Representation

Collaborative cooking is a commonly used testbed for evaluating human-robot collaboration algorithms [38, 45, 277, 99, 87, 90]. It is particularly interesting to demonstrate the efficacy of our approach because many overlapping sequences of actions can result in different meals being prepared. For example, chopping vegetables, boiling water, or mixing ingredients are common steps in various recipes, making it challenging to accurately infer the specific dish being made without additional distinguishing actions.

To evaluate our approach, we conducted experiments in both a simulation and a real-world setup where a human and a robot could collaboratively cook 30 possible recipes, taking turns performing one action each. We assume that the human and the robot can perform any required preparatory step, with each step taking the same time, regardless of who completes it. Our experimental setup consists of a robot and a user in a simplified kitchen environment (refer to Fig. ??) preparing a meal. The environment consists of a blender, a sink containing water, a pan on a stove, a serving bowl, a glass, a serving spoon, an eating spoon, a measuring cup, and a storage shelf containing ingredients for making the meals. The meals could be a type of pasta, stew, salad, oatmeal, smoothie or parfait. The details of recipes are given in Table 5.1.

1. **Pasta:** The pasta sauce is prepared by blending tomatoes, pouring them into a pan with oil, and cooking it on the stove. The pasta is added to the stove and cooked. Variations include adding cheese or vegetables to the pasta.
2. **Stew:** The stew is prepared by filling a pan with water and adding salt, onion,

Table 4.1: List of Recipes that the Human and the Robot could collaboratively cook in our experimental setup & the required ingredients.

Food Type	Basic Ingredients	Ingredients	Variants	Additional Ingredients
Oatmeal	water, oats, salt		Plain, Chocolate, Chocolate	Fruit, strawberries, blueberries, chocolate chips
Parfait	yogurt, oats, sugar		Plain, Fruit Nut, Chocolate, Chocolate, Nut Chocolate	Fruit, strawberries, blueberries, nuts, chocolate chips
Pasta	tomato, oil, pasta		Tomato Veggie, Cheese	cheese, carrot, onion, broccoli
Salad	cucumber, lettuce, tomato, onion, kale, salt, oil		Veggie, Pasta, Avocado, Nut Avocado, Fruit Avocado, Fruit Nut, Pasta Fruit Nut	pasta, avocado, nuts, bananas, apple
Smoothie	water, yogurt, sugar		Berry, Chocolate, Chocolate, Fruit, Tomato, Berry	strawberries, blueberries, bananas, apple, chocolate chips, kale, tomato
Stew	water, salt, onion, tomato, lettuce, cucumber, carrot, broccoli		Veggie, Avocado, Cheese, Avocado, Cheese	avocado, cheese

tomato, and various vegetables to the pan. Variations include adding avocado, cheese, or both. The stew is cooked on the stove, mixed, and then served.

3. **Salad:** The salad is prepared by adding ingredients like cucumber, lettuce, tomato, and onion into a bowl and mixing. Variations contain pasta, avocado, nuts, or fruits.
4. **Oatmeal:** The oatmeal is prepared by filling a pan with water, adding oats and salt to the pan, and cooking. Variations include adding toppings like strawberries, blueberries, or chocolate chips.
5. **Smoothie:** The smoothie is prepared by adding water, yogurt, and sugar to the blender, followed by specific fruits or chocolate, depending on the recipe. The smoothie is blended, and the final product is poured into a glass.
6. **Parfait:** The parfait is prepared by layering ingredients such as yogurt, oatmeal, and sugar in a glass. Variations may include adding fruits, nuts, or chocolate chips.

The robot’s state and action space are defined using the Planning Domain Definition Language (PDDL) [4]. We used PDDLgym [243] to dynamically update the environment’s state based on actions taken by either the robot or the human. Both the human and the robot can take actions

$$A = \{\text{gather}(i,a), \text{pour}(i,d,a), \text{mix}(i,a), \text{cook}(i,a), \text{turn_on}(i,a), \\ \text{collect_water}(t,a), \text{blend}(i,a), \text{reduce_heat}(i,a), \text{serve}(i,a)\}$$

where i is an item available in the interaction, d is a container such as a bowl, glass, or pan, and $a \in \{HUMAN, ROBOT\}$ denotes the agent acting. The state space is described as a vector of PDDL literals where each element corresponds to

a feature of the environment, including the presence of an item in the workspace or storage area, whether an item is in a container or appliance; the power state of the stove and blender; the state of an item (simmering, blended, cooked). We assume a deterministic environment and a fully observable state space.

4.4.3 Experimental Setup

The objective of both the physical and simulated collaborations is for a human and a robot to work together to prepare a meal, which could be a type of pasta, stew, salad, oatmeal, smoothie, or parfait (see Table 4.1). While only humans know the specific meal being made, robots attempt to infer the intended meal by collaborating with humans and observing the sequence of actions they take in response to their own actions. We assume that the human always takes the first action in the interaction. Below, we detail our experimental setup for both the physical robot and simulation experiments.

Simulation

Modeling Human Behavior: In the simulation, the human’s policy bank is represented by ground-truth sequences of desired actions derived from corresponding Clique/Chain Hierarchical Task Networks (CC-HTNs) [114]. These CC-HTNs model tasks that must be performed sequentially (e.g., the strawberry must be added to the blender before the blender is turned on) or can be done in an unordered fashion (e.g., the stew toppings can be retrieved in any order).

Approximating Belief Distribution: Because our recipes are designed with many steps that can be performed in any order, the number of action sequences to complete a given recipe can become prohibitively large. This makes it challenging to calculate the belief distribution over possible goals after observing an action, as the lookup would need to account for all possible CC-HTN combinations across all recipes. To

address this, we approximate the belief distribution over goals using a transformer-based model to approximate $bel(\mathcal{H}^{0:t-1})$ and $bel(\mathcal{F}^t)$ in Eq. 4.4.1. To train this goal prediction model, we sampled 10,000 valid action sequences for each of the 30 recipes (300,000 total action sequences across all recipes). The model employed a transformer architecture with multi-head attention [280], comprising 4 layers and 8 attention heads, and was trained using cross-entropy loss to capture dependencies between actions within the sequence. The model produced a probability distribution over possible goals based on either a sequence of proposed actions ($bel(\mathcal{F}^t)$) or a sequence of actions performed by the human ($bel(\mathcal{H}^{0:t-1})$) in Eq. 5.3.2, with temperature scaling applied to control prediction confidence. The model was evaluated on a held-out validation set comprising 20% of the dataset and produced a goal prediction accuracy of 83.86%. This slightly low accuracy reflects the inherent difficulty of the task, as many potential goals share overlapping action sequences, making early goal prediction challenging.

Expanding the RHP Tree: Due to our large recipe set, as shown in Table 5.1, we have a large number of possible actions valid at any given time step, making it computationally intractable to expand the RHP tree beyond a single timestep fully. To get around this limitation, we calculate the probabilities for each action, given a certain action history. We do this offline with an n-gram-based method, observing how often each action occurred with a certain prefix of n actions. Then, we only explore branches of the RHP tree corresponding to the top 5 most likely actions, which are returned by *get_actions* in Alg. 2 on line 7. This makes it computationally tractable for us to expand the tree to a depth of 4 ($\mathcal{D} = 4$). However, the n-gram-based approach also fails at a certain point, as the exponentially increasing number of unique action sequences for a given length means that we will eventually encounter a sequence not in our dataset. In this worst case, we fall back on our model’s belief of the goal probability distribution and explore branches corresponding to actions that

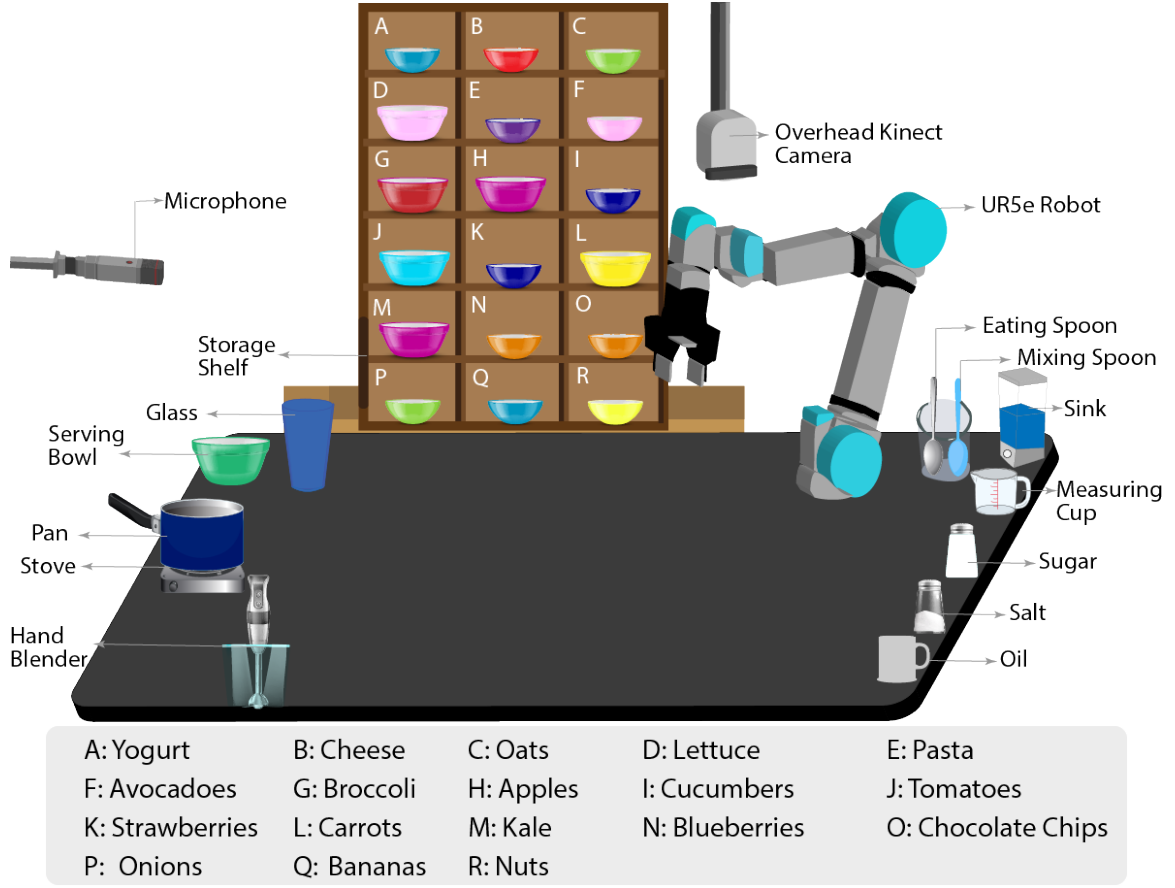


Figure 4.4: Setup for our Physical Robot Experiments for the Human-Robot Collaborative Cooking Task

would help reach the most likely goals. In our cost function described in Eq. 4.4.1, we selected λ_1 to be 1 and λ_2 to be 10.

Physical Robot

As shown in Figure ??, our physical setup consisted of a UR-5e robot, an overhead Microsoft Azure Kinect camera [246] that monitored the position of ingredients in the workspace, and a microphone. To avoid having to visually perceive the actions being performed by humans while they are doing them, humans were instructed to describe the actions they are doing in natural language. We then developed a simple language model that used fuzzy matching to map natural language to particular actions represented as PDDL literals. Realistic-looking artificial ingredients were used

to ensure equipment safety, and liquids were replaced with colored beads. Similarly, all appliances were modified to turn on but not emit heat or have any sharp blades. The workspace in front of the robot contained a blender and a pan on a stove, both of which had been modified to enable the robot to manipulate them easily. To the robot’s left, the water-dispensing sink, a stand containing the serving and the eating spoon, and a shelf containing the condiments, salt, sugar, and oil were placed. The serving bowl, the glass, and the measuring cup were located to the robot’s right. The storage shelf containing the ingredients required to make the meals was behind the robot. Each ingredient was placed in a distinct colored container, which the robot could bring to the workspace whenever needed. Similar to Tilekbay et al. [266], we used the Contrastive Language Image Pretraining (CLIP)[222] model to localize objects in the robot’s workspace based on predefined descriptive language prompts for each possible ingredient, to identify the region of interest containing the object and within that region of interest, we employed the Segment Anything Model (SAM) [150] to predict a segmentation mask over the object of interest. Finally, we applied a heuristic grasp point prediction method using the segmentation mask, enabling the robot to manipulate the items in its workspace appropriately.

4.4.4 Comparisons

We compare the performance of our approach against the following two algorithms:

Bayesian Delegation

Bayesian Delegation [292] uses Bayesian Inference to predict and coordinate actions in human-robot collaborative tasks. This approach first determines the most likely task allocation from the action history, indicating which task each agent should work on. Then, it chooses each agent’s action based on the goal it believes that the agent is trying to accomplish. Bayesian Delegation allows both agents to work on the

same sub-task, which is not applicable in our setup. Moreover, it assumes that humans and robots act in parallel, which does not happen in our setup. We adapt the concurrent action execution in the original method to our turn-based setup by having the robot alternate actions with the human and expanding a tree of actions of depth 2 to simulate potential human responses. We consider the robot’s actions and the human’s expected responses to calculate the Q values by giving a positive reward for each proposed action that aligns with the most likely recipe.

Information Gain Maximization

We use the information-gathering framework proposed in [230] as another baseline in our setup. This approach enables a robot to actively probe a human’s internal state by planning actions that maximize expected information gain. Following their method, we adapt it to our task by having the robot expand a receding horizon planning (RHP) tree of depth 4. Based on the original formulation of this approach, at each step, the robot optimizes a cost function that includes two terms: the first term maximizes the information gain of goal probabilities between the root node and the tree’s leaf nodes, reducing ambiguity about the human’s goal. The goal probabilities at the root and the leaf are calculated using the method described in Sec. 4.4.3. The second term rewards the robot for taking actions aligned with the most likely goal and penalizes actions that deviate from it. The robot executes the first action that minimizes the combined cost in our implementation. It then allows the human to respond by taking the next valid action defined by the appropriate HTN, expands the tree again from the new state, and iteratively updates its goal probabilities.

In contrast to our CDP approach, this baseline intuitively selects actions expected to maximally shift the robot’s belief about human goals. CDP, by design, identifies informative states based on how much goal-conditioned policies disagree, regardless of the robot’s current belief, by computing informativeness directly from divergence

in action choices. This baseline, on the other hand, defines informativeness as the expected change in the robot’s belief distribution. Unlike CDP, which assigns a fixed informativeness to each state, IGM’s measure varies with the robot’s current uncertainty, potentially causing the robot to overexplore states it is currently uncertain about, even if those states are unlikely to reveal the human’s true goal.

4.4.5 Metrics

We compare the performance of our method against each baseline comparison for our proposed collaborative cooking task using four key metrics.

1. **First Correct Guess:** The first timestep in an interaction when the robot correctly identifies the human’s goal.
2. **Last Incorrect Guess:** The last timestep in an interaction when the robot wrongly identifies the human’s goal.
3. **Number of Extra Steps:** Number of extra steps taken by both agents to complete the interaction compared to the number of ground-truth steps from the corresponding HTN.
4. **Percentage of Correct Guesses:** Percentage of timesteps during an interaction when the robot correctly identifies the human’s goals.

We report each method’s top-1 and top-3 accuracy for the first correct guess, the last incorrect guess, and the percentage of correct guesses.

4.4.6 Simulation Results

Fig. 4.5 compares our proposed approach with the Bayesian Delegation [292] and Information Maximization [230] approaches across several key metrics in a simulated collaborative cooking task. The means and standard deviations of the metrics are

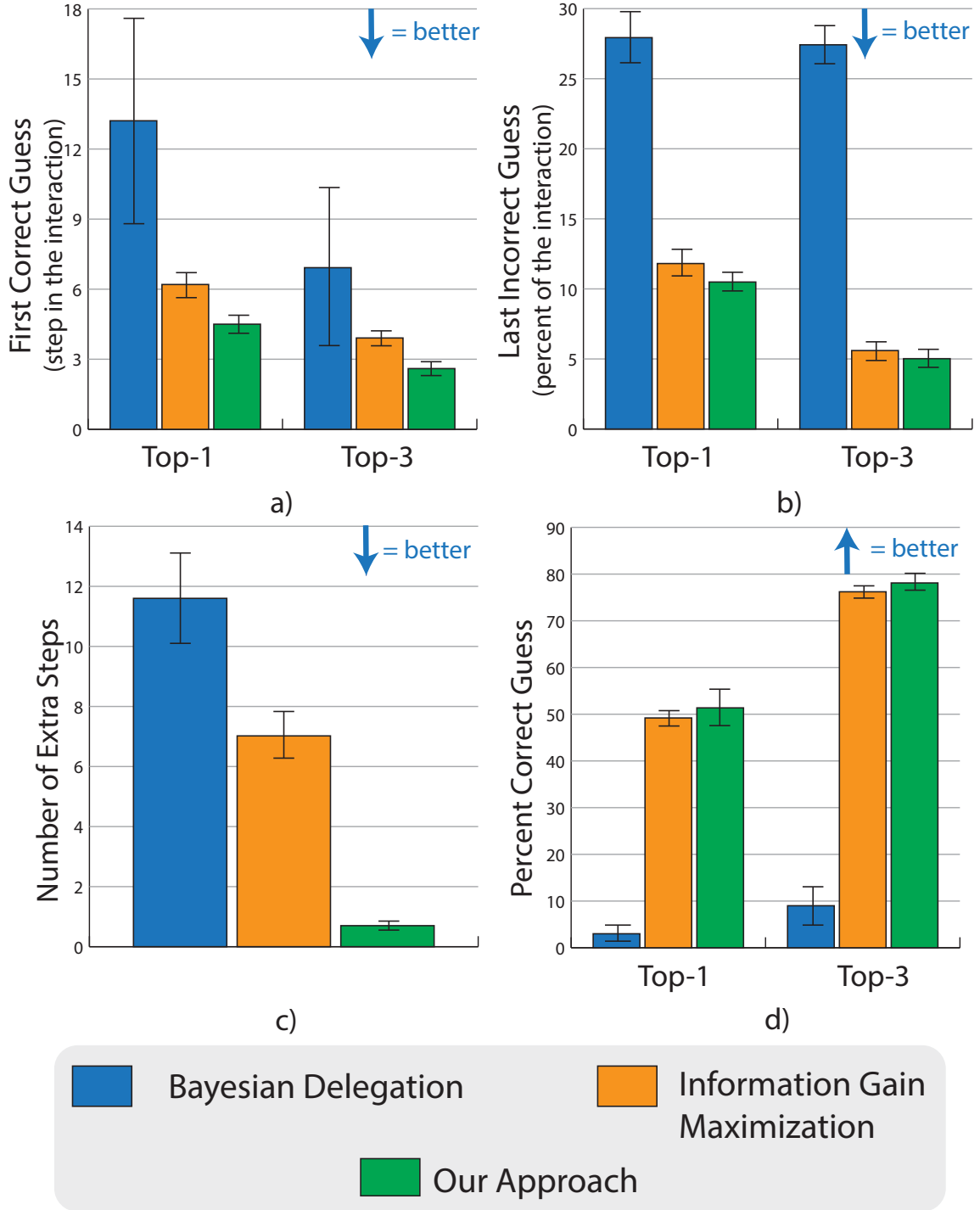


Figure 4.5: Results of the Collaborative Cooking Evaluation: Comparison of performance metrics between the proposed approach, vanilla CDP [92], Information Gain Maximization [230], and Bayesian Delegation [292] in the collaborative cooking simulation. Metrics include a) the first correct guess, b) the last incorrect guess, c) the number of extra steps, and d) the percentage of correct guesses. The means and standard deviations of the metrics are calculated over 30 recipes and evaluated across three trials.

calculated over 30 recipes and evaluated across three trials. When following the ground truth, each interaction sequence, on average, involves 18.7 steps, setting a baseline to assess the robot’s performance.

Fig. 4.5 a) shows that our approach consistently identifies the human’s goal earlier in the interaction than all baseline approaches. The first correct guess typically occurs in the first third of the sequence, well before the interaction’s midpoint. However, this metric can be somewhat noisy, as all methods might occasionally predict the correct goal by chance early in the interaction without resolving uncertainty about the goal.

Therefore, we compute the last incorrect guess metric, as it marks the point in the interaction where the robot no longer makes erroneous predictions about the human’s goal. Fig. 4.5 b) shows that our approach generally achieves this around the midpoint of the sequence, earlier than all baselines. This earlier resolution of uncertainty indicates that by the time half the sequence is completed, our method has effectively understood the human’s intentions, leading to a higher percentage of correct guesses as shown by Fig. 4.5 d).

Fig. 4.5 c) shows the total number of extra steps both agents took to complete the collaboration. If the robot performs every action incorrectly, the interaction could take up to twice as long, as the human would need to perform every correct step towards task completion. Conversely, if the robot correctly assists the human, the task can be completed in the baseline of 18.7 steps, the average length of a ground truth sequence. The number of extra steps in our approach is minimal, averaging 0.70 steps, compared to 7.02 steps for the information gain maximization approach, 9.46 steps for the vanilla CDP approach (a “history-agnostic” CDP baseline, in which each state’s informativeness is computed in isolation from any prior actions called “Vanilla CDP” in the rest of the Section).), and 11.0 steps for the Bayesian Delegation approach, showing that our approach can lead to efficient human-robot collaboration.

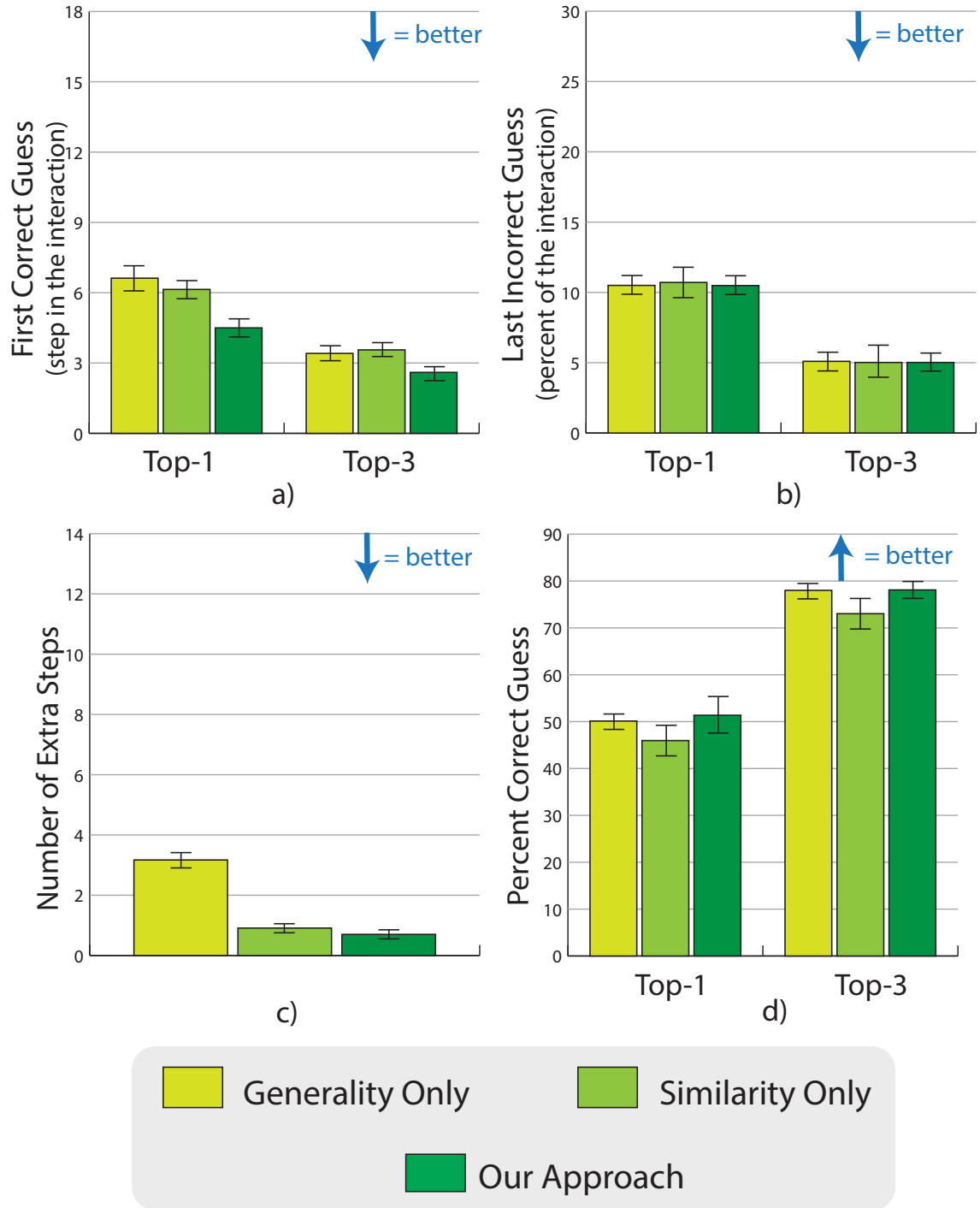


Figure 4.6: Ablation Study over the components of the Cost Function for the Collaborative Cooking Task

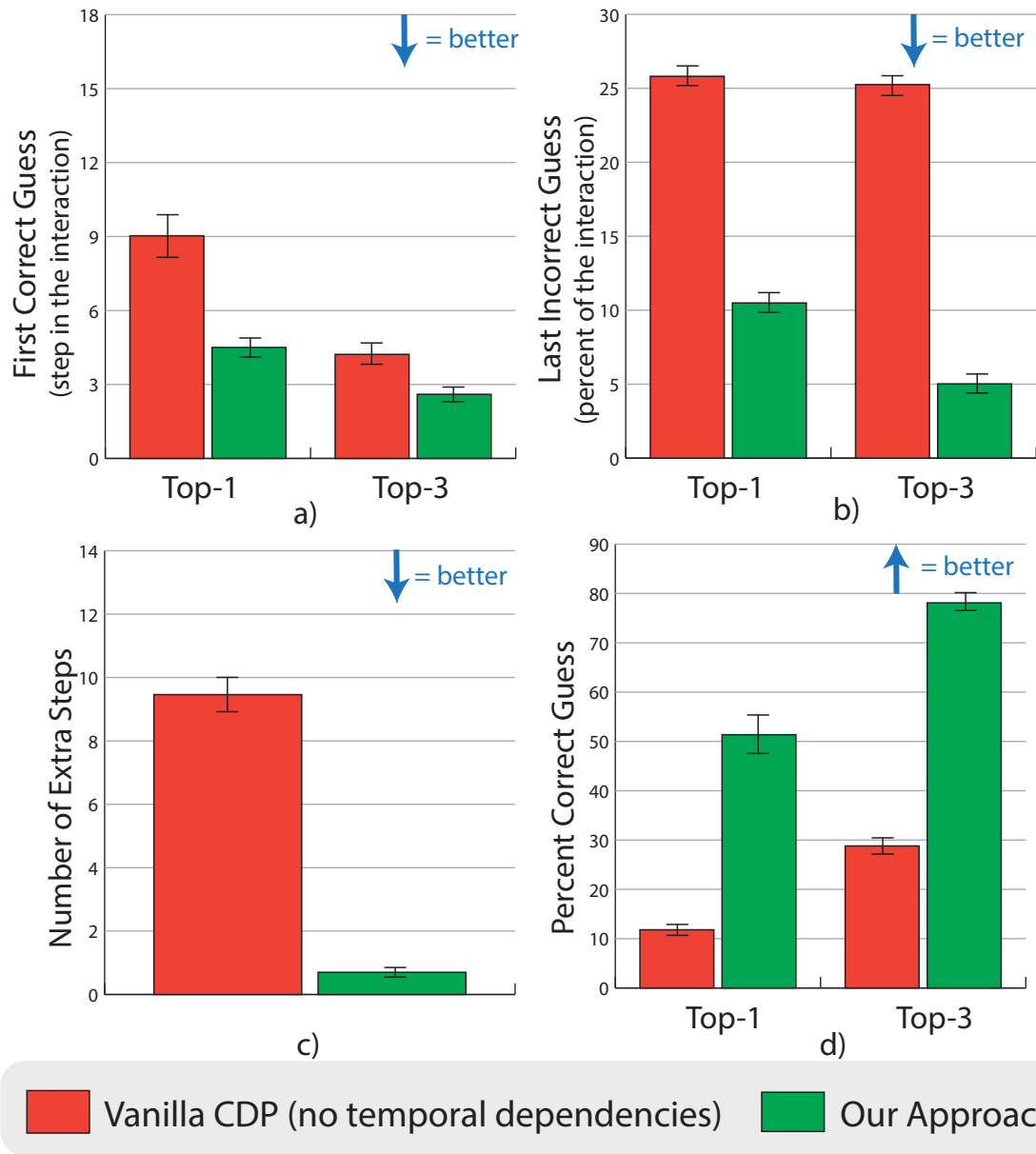


Figure 4.7: Ablation Study on using temporal modeling for computing CDPs for the Collaborative Cooking Task

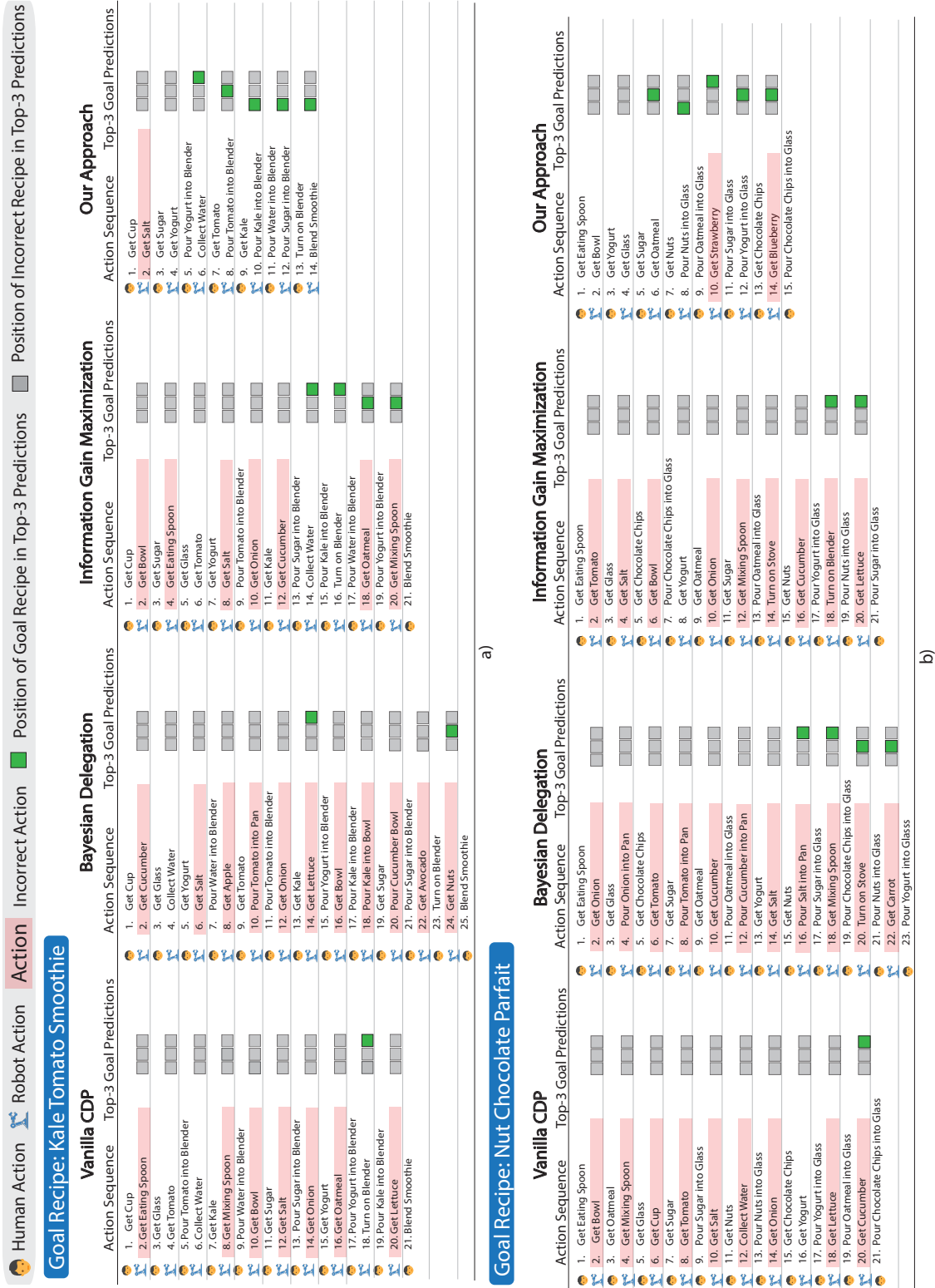


Figure 4.8: Collaborative Cooking Case Study: Results of the Case Study with the Physical Robot running Bayesian Delegation, Vanilla CDP (history agnostic), Information Gain Maximization, and Our Proposed Algorithm: For Case Studies, a) and b), a group of columns represents the method. For each group, the first column depicts the agent taking action, the second column depicts the action sequences taken by the respective method, and the third column denotes the presence and position of the goal recipe in the top-3 most probable recipes from the goal prediction model (left-most being the most probable recipe).

4.4.7 Ablation Study

Cost Function Ablation

Fig. 4.6 shows the results of the ablation study we conducted to determine the impact of the generality and similarity terms in the cost function given in Eq. 4.4.1. Overall, we observed that our method performed better on all metrics over individual components of the loss functions.

When generality is considered individually, we observed that the robot tended to take very general actions, often avoiding task-specific actions. This behavior caused the robot to deviate from the task at hand, leading to unnecessary steps to complete the task, as illustrated in Figure 4.6c). Notably, since the robot is inclined to undertake actions that correspond with a broader range of goals by prioritizing generality, it performed well in accurately predicting the correct goal, as shown in Figure 4.6d).

On the other hand, when the similarity term is considered in isolation, we found that the robot effectively selects the most beneficial actions for the human in the moment. However, it struggles to guide the human toward CDPs, which are essential for quickly clarifying their goals. This limitation is evident in Figure 4.6d), where the robot performs poorly in goal prediction. Conversely, it takes fewer unnecessary steps, as depicted in Figure 4.6c).

Time Dependency Ablation

In this ablation (Fig. 4.7), we isolate the effect of incorporating interaction history into the informativeness score calculation (Eq: 4.4.1) to establish a “history-agnostic” CDP baseline, in which each state’s informativeness is computed in isolation from any prior actions (this baseline is called “Vanilla CDP” in the rest of the Section). At every time step, this approach evaluates the divergence among goal-conditioned policies at the current state, chooses the move that leads to the highest immediate informa-

tion gain, observes the human’s response, and updates its belief, never allowing past observations to influence which states are deemed most revealing. By contrast, our approach conditions the informativeness calculation on the entire interaction history $\mathcal{H}$. As Fig.4.7 shows, this temporal reasoning enables the robot to correctly identify the human’s goal significantly earlier in the interaction and to complete the task in far fewer extra steps than the history-agnostic CDP baseline.

4.4.8 Real Robot Results

We conducted two proof-of-concept case studies using a real robot and a human collaborator (with one of the researchers acting as the collaborator using the setup shown in Figs. 4.2 and 4.4) to compare the performance of Bayesian Delegation [292], information-gain maximization [230] and vanilla CDP [92] methods with our proposed approach in collaborative cooking tasks as shown in Fig. 4.8. For each step the robot took in all four methods, we evaluated the predictions of our goal predictor model to determine whether the correct goal recipe is among the top three most probable options. To ensure a fair comparison of the methods, humans always took the same first action, preventing any revealing actions at the outset.

We found that our method strategically selected general actions common to most recipes to influence the human collaborator to perform distinguishing actions early, reducing uncertainty about the human’s goal. This allowed the robot to become certain sooner and avoid incorrect actions. In contrast, Bayesian Delegation relied on noisy initial goal estimates, leading to early mistakes that compounded and made it difficult for the goal predictor to identify the correct recipe by the end. Vanilla CDP identifies Critical Decision Points (CDPs) based solely on the immediate information gain at a given state, without considering the action history or the long-term impact of prior actions, which is shown to be problematic given the sequential nature of the task. Finally, we find that Information Gain Maximization fails to balance exploration

(gathering information to clarify the goal) with exploitation (taking actions that align with the most likely goal). Consequently, for each of our three baseline approaches, the human performed most steps in the task.

Fig. 4.8a) shows our first case study where our goal recipe is the Kale Tomato Smoothie; our approach influenced humans to add distinctive ingredients early by performing general actions like getting the yogurt and collecting water. The robot became certain of the goal by step 10 by just taking one extra step at the beginning of the interaction. Bayesian Delegation, relying on uncertain estimates, led the robot to make mistakes such as pouring tomato into the pan instead of the blender, adding unnecessary ingredients like lettuce, and taking ten extra steps. Vanilla CDP fails to consider the entire action history, leading the robot to use combinations of actions that do not contribute to any recipe in the Goal Bank, e.g., Get Glass and Get Mixing Spoon, which never occur together in any recipe. This led the robot to take seven extra steps. Information Gain Maximization tries to narrow down the goal distribution without considering whether the goal it is converging to is correct or not, which leads it to take overly specific actions that are not helpful to the recipe, such as Get Onion, leading the robot to take seven extra steps.

Fig. 4.8b) presents our second case study, where the goal recipe is Nut Chocolate Parfait. Our approach began with general actions like retrieving the bowl and glass and influencing the human to add yogurt, nuts, and chocolate chips early, key steps in making the parfait. As a result, the robot quickly became certain of the goal, with the correct recipe consistently appearing in the top 3 predictions from step 6 onward. Although our method took a few extra steps due to confusion between the Fruit Nut Chocolate Parfait and the Nut Chocolate Parfait because the robot erroneously fetched strawberries and blueberries, it only required three extra steps. In contrast, Bayesian Delegation caused the robot to make irrelevant actions like retrieving tomato, cucumber, and onion, leading to ten extra steps and fewer correct

guesses. Like in Fig. 4.8a), Vanilla CDP takes combinations of actions that can never plausibly occur together, such as Get Glass and Get Salt. This leads the robot to take eight additional steps. Information Gain tries to acquire as certain an estimate of the human’s goal as soon as possible, which leads to it taking incorrect actions with relatively low generality very early (such as Get Tomato), which leads the robot to take eight extra steps.

4.4.9 Comparative Analysis of Performance of CDPs in the Collaborative Task

Bayesian Delegation

Bayesian Delegation [292] struggles when dealing with tasks involving overlapping action sequences that lead to different goals. When different goals share many of the same actions, this approach struggles to differentiate between them because the observed actions do not provide enough discriminatory power, causing the belief to remain spread across multiple possible goals. This results in delayed disambiguation, as the Bayesian approach accumulates evidence slowly, making it harder for the robot to infer the correct goal promptly and confidently. Moreover, the probability mass tends to be diluted across several goals rather than concentrating on the correct one, making it difficult for the system to favor the correct goal even after multiple observations. As observed in our experiments, these limitations lead to slower and less accurate goal inference.

Information Gain Maximization

Information Gain Maximization [230] focuses on reducing uncertainty at each step by selecting actions that maximize immediate information gain. However, this approach is shortsighted for tasks where multiple goals share the same action sequences because

the robot prioritizes actions that provide immediate information. Intuitively, this method fails to balance exploration (gathering information to clarify the goal) with exploitation (taking actions that align with the most likely goal). By always taking the action that provides the most information immediately, it may be pursuing the wrong goal, when it should be taking more general actions that provide less information but help support the human’s goal.

Vanilla CDP

To isolate the effect of incorporating interaction history into our approach, we compare our method against vanilla CDP [92]. Vanilla CDP considers only the current state without past actions. This leads the robot to take implausible actions, as it ignores the context in which human actions were performed. The issue is especially problematic when multiple action sequences overlap, making context crucial for distinguishing between goals. Without an action history, the robot’s mistakes compound, making it harder to infer the human’s goal. Also, since whether a state is informative about a human’s goal depends on when it occurs, ignoring temporal context weakens the robot’s ability to identify key moments that reveal human intent.

Our Approach

In contrast to the above baseline methods, our approach influences human actions actively to reveal their goal and addresses the challenges faced by other methods.

We address the issue of overlapping action sequences faced by Bayesian Delegation by guiding the human towards more actions that reduce the robot’s uncertainty about the human’s true goal. This strategy reduces the impact of overlapping sequences by incorporating generality in the cost function, which steers the human towards taking distinguishing actions while also increasing the likelihood that, even if the robot’s belief about the human’s goal is incorrect, the chosen action remains on policy,

ensuring that the robot’s actions are robust to uncertainties in goal prediction.

Our approach addresses the issues faced by Information Gain Maximization by incorporating similarity into our cost function. By doing this, we encourage the robot to take actions that are likely to be relevant to the human’s goal, rather than the action that maximally narrows down the belief distribution over the human’s goals, regardless of whether the goal it is converging to is correct.

Finally, unlike Vanilla CDP, our approach considers the entire action history when selecting new actions, which prevents us from taking implausible combinations of actions, which are actions that are unlikely to be taken together in the context of the policy bank $\mathcal{P}$. Using our approach, the robot aims to steer the collaboration toward taking actions that are unlikely to be detrimental to achieving the human’s goal or significantly alter the robot’s belief over human’s goals. In effect, it enables humans to take actions that reduce the robot’s uncertainty about its true goal earlier in the interaction than vanilla CDP.

4.5 Scenario 2: Influencing the Human to Reveal their Goal in a Competitive Human Robot Task

4.5.1 Approach

In this scenario, we consider a different instantiation of the problem: a competitive environment in which the robot seeks to infer the human’s goal under partial observability. Unlike the collaborative setting where the robot can condition its strategy on the complete interaction history $\mathcal{H}$, here the history may be unavailable or unreliable due to limited sensing or occlusion. This changes the nature of the inference problem

Algorithm 3 Scenario 2: Selecting Action at Timestep t to Influence Humans to CDPs by expanding an RHP tree for Human-Robot Competition

Input: Goal set $\mathcal{G} = \{g_1, \dots, g_n\}$ with policy banks $\mathcal{P}_g = \{\pi_g^1, \dots, \pi_g^{n_g}\}$; Pre-computed CDP set $S_{\text{CDP}}(\emptyset)$; planning horizon D_H ; weights (λ_1, λ_2) ; robot action space $\mathcal{A}_r$

Output: Next seeker action a_r^t

```

23  $b_0(g) \leftarrow 1/|\mathcal{G}|$ ,  $t \leftarrow 0$ 
24 while hider not caught do
    // 1) Sense current state
25 observe  $s_t = (s_w, s_h, s_r)$ 
    // 2) Build depth- $D_H$  tree
26 Alternate levels of
    • Seeker branch: enumerate  $a_r \in \mathcal{A}_r$ 
    • Hider branch: sample  $a_h \sim \bar{\pi}_{\hat{g}}(a_h | s_t)$  where  $\bar{\pi}_g(a | s) = \frac{1}{|\mathcal{P}_g|} \sum_k \pi_g^k(a | s)$ 

    Store leaves  $\ell$  with state  $s_\ell$  and future sequence  $\mathcal{F}_\ell$ 
    // 3) Score each leaf
27 foreach leaf  $\ell$  generated under goal  $g$  do
28    $C_{\text{CDP}}(\ell) = \min_{s_{\text{crit}} \in S_{\text{CDP}}} \text{dist}(s_{h,\ell}, s_{\text{crit}})$ 
29    $C_{\text{task}}(\ell) = C_{\text{task}}(s_\ell)$ 
30    $J_\ell = \lambda_1 C_{\text{task}}(\ell) + \lambda_2 C_{\text{CDP}}(\ell) \frac{1}{b_t(g)}$ 
31 end foreach
    // 4) Execute best seeker move
32  $\ell^* = \arg \min_\ell J_\ell$ ;  $a_r^t \leftarrow$  first seeker action on path  $\ell^*$ ; execute  $a_r^t$ 
    // 5) Observe hider and update belief
33 observe  $a_h^t$   $b_{t+1}(g) = \frac{\bar{\pi}_g(a_h^t | s_t) b_t(g)}{\sum_{g'} \bar{\pi}_{g'}(a_h^t | s_t) b_t(g')}$   $t \leftarrow t + 1$ 
34 end while

```

– requiring the robot to reason from immediate state information alone and adapt its strategy accordingly. However, this enables the robot to pre-compute CDPs if the state space is not prohibitively large.

Alg. 3 summarizes our approach for a robot to influence a human towards Critical Decision Points in a competitive task under partial observability. Similar to the human-robot collaboration scenario described in Sec. 4.4.1, at a given time t , the robot rolls out all possible future actions that can be taken by the human per the policy bank $\mathcal{P}$ along with its own responses to the human’s possible actions. These steps are repeated to grow the search tree to a certain horizon D_H . For every leaf

node in the tree that results from the human acting according to a policy π_i , the robot computes a cost $\mathcal{C}$ that describes how good that specific future is based on 1) the distance between the projected human position at time $t + \mathcal{D}$ and the closest precomputed CDP (s_{crit}) as a measure of informativeness similar to Eq. 4.4.1,

$$divergence := C_{CDP} = dist(s_h^{t+\mathcal{D}_H}, s_{crit})$$

and 2) other task-related costs that support the robot's competitive objectives, like the distance between the human and the robot:

$$\mathcal{C} = \left(\underbrace{\lambda_1 C_{task}}_{\text{task cost}} + \underbrace{\lambda_2 C_{CDP}}_{\text{informativeness cost}} \right) \frac{1}{b_t(g)}$$

where $dist(s_h^{t+\mathcal{D}}, s_{crit})$ is the shortest path between the human and the closest Critical Decision Point $\mathcal{D}$ steps in the future when the human acts according to π_i and the robot takes corresponding actions. The informativeness cost C_{CDP} serves the same function as CDP computation in the collaborative scenario, but is pre-computed rather than being dynamically inferred. λ_1, λ_2 are weights to balance the task and informativeness costs. The cost $\mathcal{C}$ is scaled by the likelihood of the human policy $b_t(g)$, where $b_t(g)$ is the robot's belief that the human's hidden goal is g (updated via the Bayes filter in Sec. 4.3.1 with the same marginal likelihood $\bar{\pi}_g$). such that policies that are less likely will result in higher costs. Finally, the robot selects its best response action a_r^* at time t as the action that minimizes the cost function $\mathcal{C}$ for all possible $\pi_i \in \mathcal{P}$ over the horizon $\mathcal{D}_H$. This allows it to influence the human toward Critical Decision Points, because the cost depends on where the human ends up after $\mathcal{D}_H$ steps in the future.

Fig. 4.9 illustrates how our proposed approach works. The green and the blue agents are playing a hide-and-seek game, where the green agent (the seeker) is trying to influence the blue agent (the hider) to a Critical Decision Point (the red cell)

using RHP to infer its policy and therefore catch it faster. The brick walls represent obstacles. The green agent imagines all possible future actions, assuming the blue agent follows a fixed policy, and selects the most optimal action in the next timestep to lead to the most favorable outcome at horizon $\mathcal{D}_H = t_4$. The black arrow at t_1 represents the underlying strategy. Both agents’ fields of view are restricted to all the cells in the direction they face. At t_2 , when the blue agent detects the green agent, it abandons its underlying strategy at t_3 to perform an evasive maneuver, then resumes its original path once out of the green’s field of view at t_4 . This allows the green agent to observe the blue agent’s underlying policy at a critical decision point at t_3 without being observed, and ultimately catch it in t_4 . Note that the green arrow denotes the optimal next action for the green agent. After rolling out all possible futures, the green agent determines its optimal sequence of actions to be $a_2 \rightarrow a_1 \rightarrow a_1$ (depicted by the green bold arrows). Since the green agent performs RHP, it takes the first action a_2 from the sequence and constructs the tree from the next state.

4.5.2 Task and Experimental Setup

Task

We use the game of hide and seek as our experimental setup, as shown in Fig. 4.11. The game allows the construction of arbitrarily complex environments, requires the agents to take a large variety of environment-dependent strategies, and requires that agents not communicate about their strategies [20]. Moreover, we believe it is a good testbed to evaluate our approach in a partially observable, discretizable environment, in a reasonably small state space, to showcase how CDPs can be pre-computed, and how humans can be influenced by CDPs to reveal their strategies.

We developed the hide-and-seek task in a photo-realistic simulation and the real world. In both settings, we embodied the agents with Turtlebot3 robots, so there is no advantage due to the different agent morphology in the game. In other words,

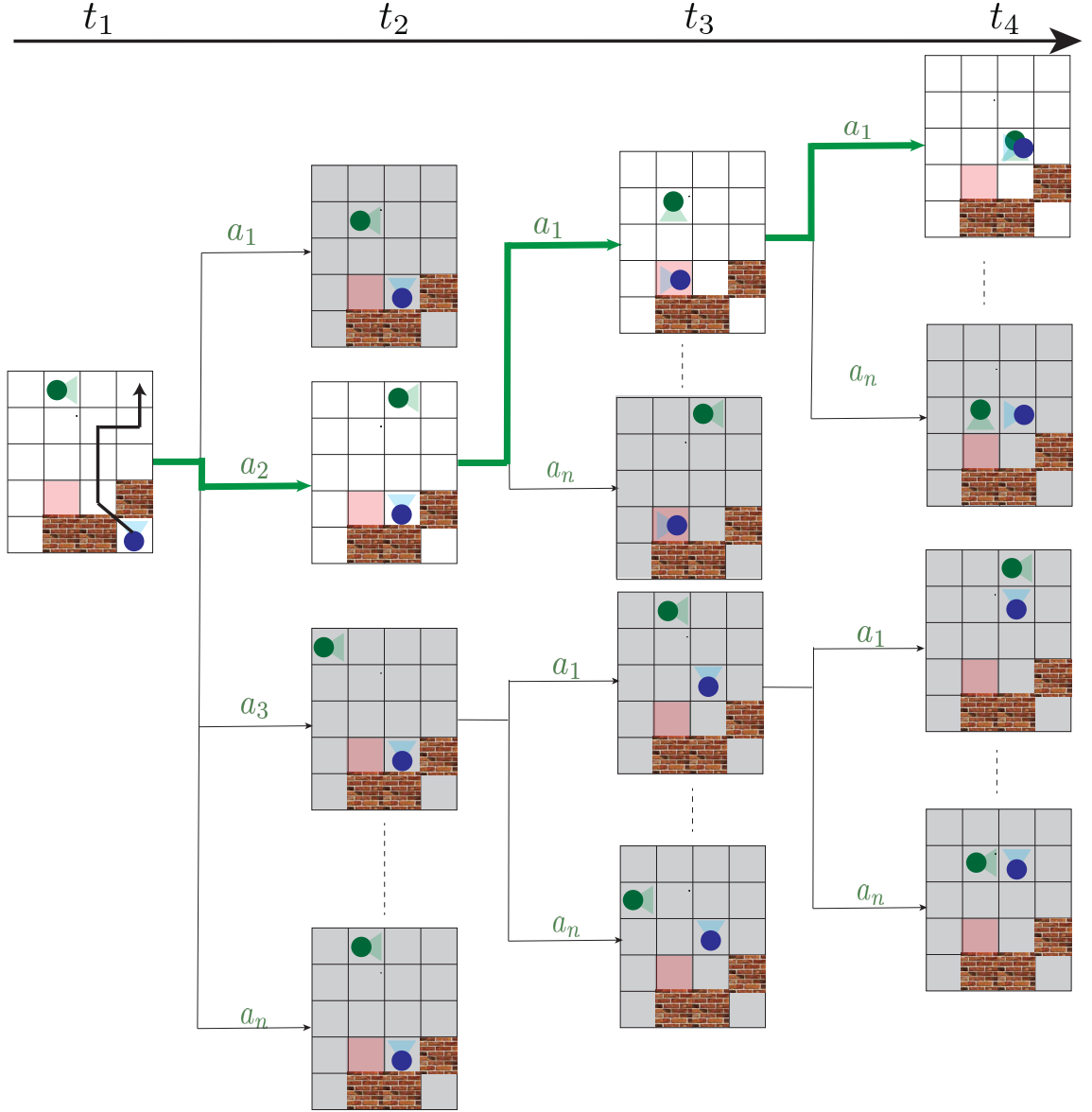


Figure 4.9: Example RHP tree rollout for influencing the robot in a competitive hide and seek game: The green (seeker) and blue (hider) robots play hide-and-seek on a grid with brick-wall obstacles. The seeker uses RHP to steer the hider toward a Critical Decision Point (the red cell) so its next move will reveal its hidden strategy and speed capture. At each timestep, it simulates the hider following fixed policies out to horizon t_4 and chooses the action that best reduces goal uncertainty. At t_1 , both follow their underlying plans (shown by the black arrow). When the hider spots the seeker at t_2 , it detours evasively at t_3 , then resumes its original path at t_4 , landing exactly on the CDP unobserved and allowing the seeker to infer its strategy and catch it. The rollout identifies the optimal action sequence $a_2 \rightarrow a_1 \rightarrow a_1$ (green arrows), and by RHP the seeker executes the first move a_2 before replanning. See text for more details.

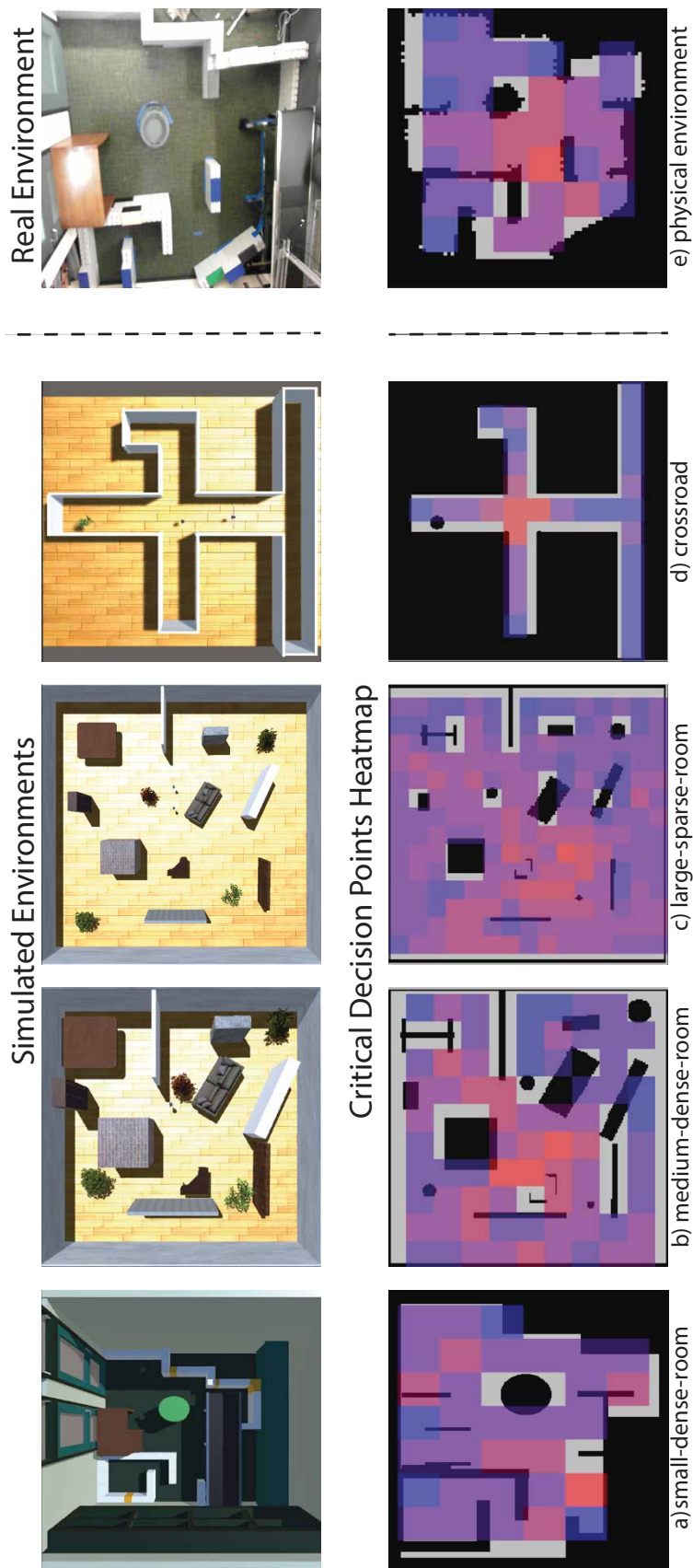
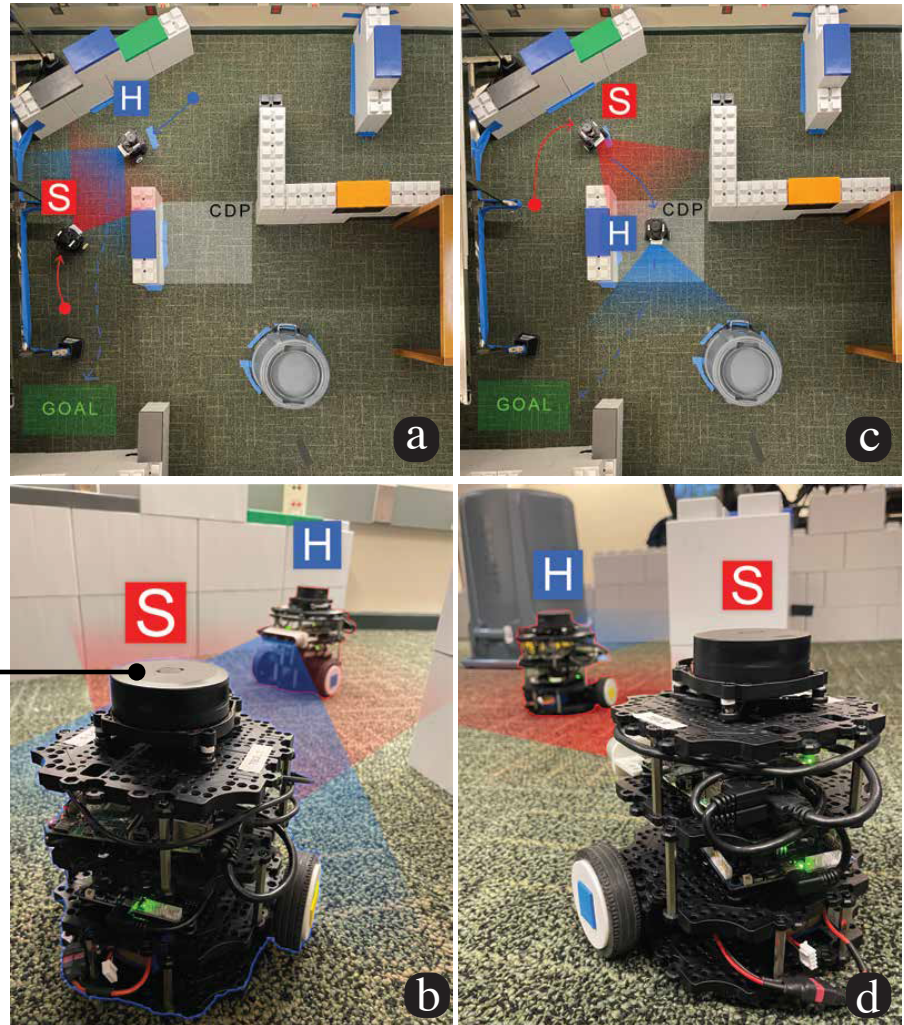


Figure 4.10: Test Environments and Computed CDPs for the Competitive Hide and Seek Task: (Top) Photo-realistic Simulated and real-world environments for the hide-and-seek task. (Bottom) Heatmaps of Critical Decision Points for corresponding environments. States closer to red denote the most Critical Decision Points, and states closer to blue denote the least critical points. Note: Although the figure displays Critical Decision Points for every state, typically, it is sufficient to compute them in a small region around the robot.

S Robot Seeker
H Human Hider



Turtlebot robot

Figure 4.11: The robot seeker (S) influences the human hider (H) towards a Critical Decision Point (CDP) in a hide-and-seek game: a), b) The human-controlled hider sees the robot seeker on its way to its goal while following its high-level strategy. c) The robot seeker actively influences the human-controlled hider to escape towards a CDP (gray area). d) When the hider is at the CDP, the robot seeker optimizes for not being seen by the hider. So, the hider continues to navigate toward its goal. The robot seeker sees the hider take action toward its goal and identifies its strategy.

when humans played the game, they did so by controlling the TurtleBot3 robot using a joystick, and its perception and control abilities thus limited them. The robots have a forward-facing camera, providing a first-person view of the environment. The placement of objects in the environment and agents’ access to visual information through their cameras make the hide-and-seek game partially observable for both agents.

The photo-realistic simulation environments were created by modifying the Social Environment for Autonomous Navigation (SEAN) 2.0 simulator [269] to support multiple robots. In simulation, an autonomous agent assumes the role of the seeker, while a simulated human controls the hider. In all experiments, both agents access similar sensor data and have the same form factor. For real-world experiments, we set up a lab environment similar to one of the simulated worlds. One of the researchers controlled the hider robot, which had access to only the front-facing camera feed and map of the environment, while the seeker robot made decisions autonomously.

Environments

We construct four environments in simulation, as shown in the top half of Fig. 4.10. The first environment is a **small-dense-room** with several objects densely placed in a small room. The second environment is a **medium-dense-room** with objects spread across the map. The third is a **large-sparse-room**, which is sparse and has the same objects as **medium-dense-room** but wider open spaces. The last one is the **crossroad** environment, with four hallways emanating from a single crossroad and distinct objects or environmental geometries at the end of the hallways. The laboratory environment for physical experiments is similar to the **small-dense-room**.

Policy Bank

We model the various strategies of the human-controlled hider using behavior trees [62, 160]. Each behavior tree takes the hider to various hiding locations within a given map, determined by the presence of particular objects in those locations or locations with different environmental occlusions. If the hider were to spot the seeker while navigating to a pre-defined location, the behavior tree would enable the hider to perform a fixed evasive maneuver to escape the seeker’s field of view. Each behavior tree was executed multiple times with different starting locations on the map for both the hider and the seeker to collect a dataset of states and actions for each behavior of the hider. To construct the hider’s policy bank, each policy was trained using the state-action trajectories in the corresponding dataset with a Long Short-Term Memory network [118] and behavioral cloning [1].

4.5.3 Implementation Details

The map of the shared environment was discretized into an $n \times n$ grid, and each map cell was represented as a node in an undirected graph. Each node in the environment graph would contain information about the center position of the cell, the objects present in the cell, and a visibility score of that cell, computed from every other cell in the map using ray casting in a raster grid. The state space S_H of the hider agent was 6-dimensional, containing $(x_h, y_h, \theta_h, \hat{x}_s, \hat{y}_s, \hat{\theta}_s)$, where x_h, y_h denoted the position of the hider, θ_h is their heading, $\hat{x}_s, \hat{y}_s$ is the estimated position of the seeker by the hider, and $\hat{\theta}_s$ was the estimated heading of the seeker by the hider. To obtain the perceived position and heading of the hider, we first fine-tuned the Yolact [35] real-time object detection model to predict a bounding box and a heading estimate for a given robot when it is visible in the first-person view of another robot. Then, based on the robot’s known physical size, the predicted bounding box in the image, and the camera parameters, we use a pinhole camera model to estimate the relative

node position of the hider w.r.t. the seeker. The action space of the seeker consists of transitions to each adjacent cell from an inhabited cell, where diagonal transitions are weighted proportionately in path computations.

To compute Critical Decision Points, we predict the actions the hider would take per a given policy for every node in an environment, $\pi_i \in \mathcal{P}$, assuming that the hider cannot see the seeker. This results in a map of Critical Decision Points, as shown in the bottom half of Fig. 4.10. The map shows that the environments and the policy bank created for the hide-and-seek task have afforded the existence of sparsely located Critical Decision Points.

The seeker jointly optimizes to drive the hider toward the closest Critical Decision Point (with $\theta \geq 4$ in Eq. 4.3.2) in a fixed radius around the hider’s current position while minimizing its distance to the hider (C_{task} in eq. 34). Additionally, the seeker optimizes for staying out of the hider’s field of view when it approaches a Critical Decision Point using RHP.

Whenever the hider is visible, the seeker expands a tree of all possible future actions it could take in response to a hider following a given policy up to a certain horizon ($\mathcal{D}_H = 5$). Concurrently, the seeker performs a Bayesian update on the belief of the hider’s policies, given observations of the hider, to compute the likelihood of each policy being executed from the policy bank. Finally, this belief is used to evaluate the cost function for each branch in the tree. Fig. 4.9 shows an example of tree expansion. Note that if the hider has been seen in the past 4 ($\mathcal{D}_{H-1}$) timesteps and is not currently visible, the seeker uses the stored action queue from the RHP tree in anticipation of spotting the hider in the future if its belief over the policy currently being followed by the hider is correct. Once no more stored actions are in the queue, the autonomous seeker reverts to a default exploration strategy where the seeker chooses an object on the map according to the policy it is following and navigates toward it.

4.5.4 Comparisons

We evaluate our method in simulation and demonstrate our approach in a real-world environment.

Simulation

We measure the effectiveness of our approach by comparing the time it takes the autonomous seeker to catch the human-controlled hider against two baselines:

1. **Baseline 1:** The seeker randomly explores the environment and performs a Bayesian update of the belief over the hider’s policies whenever it is visible (per eq. 4.3.1).
2. **Baseline 2:** The seeker uses RHP to optimize for minimizing its distance between itself and the estimated position of the hider whenever it is visible while computing the Bayesian update (per eq. 4.3.1).
3. **Our Method:** The seeker selects actions by expanding the RHP tree to drive the hider towards the closest Critical Decision Point per eq. (34), and estimates their strategy using the Bayesian update (per eq. 4.3.1).

For our simulated experiments, we assume that a simulated human controls the hider and follows a randomly selected policy from the policy bank during an interaction episode. Depending on the environment, the hider’s policy bank contains between 6 and 8 policies. We run each experiment for 20 episodes or trials, each of which ends when the seeker lands on the same node as the hider (indicating it has caught the hider) or after 100 time steps. To compare our method’s performance with the baselines, we compute the average number of time steps needed for the seeker to catch the hider in each environment, aggregated along all ground truth policies from the policy bank, followed by the human hider for all trials.

Real World

We demonstrate the applicability of our method in the real-world environment (shown in Fig. 4.11). We used Simultaneous Localization and Mapping (SLAM) ² to generate a map of the physical environment. Like the simulation environments, we discretized the map into an $n \times n$ grid to create an undirected graph with the states as nodes and valid actions from the states as edges. Depending on the size of the environment, n varied between 7 and 15. Then, we developed the policy bank of the hider by collecting trajectories of the robot moving between certain pre-defined locations of the map using the behavior trees and generating policies with behavioral cloning. We computed the Critical Decision Points for the real-world environment by computing the divergence of actions at every cell in the discretized grid. One of the researchers controlled the hider robot through velocity commands using a PlayStation controller while following a given policy from the policy bank. The seeker robot selected actions optimized for driving the opponent toward the Critical Decision Points following our RHP-based approach.

4.5.5 Results

Simulation

Fig. 4.12 shows that our method outperformed the baselines across all four simulation environments. When the seeker influenced the hider toward Critical Decision Points, it caught the hider faster than when it randomly explored the map (Baseline 1) or minimized its distance to the hider whenever the hider was visible (Baseline 2).

Qualitatively, we observed that with Baseline 1, when the seeker randomly traversed the map and updated its belief of the hider's policy with each sighting, the seeker could not accurately estimate the hider's strategy. In Baseline 2, when the

²<https://wiki.ros.org/gmapping>

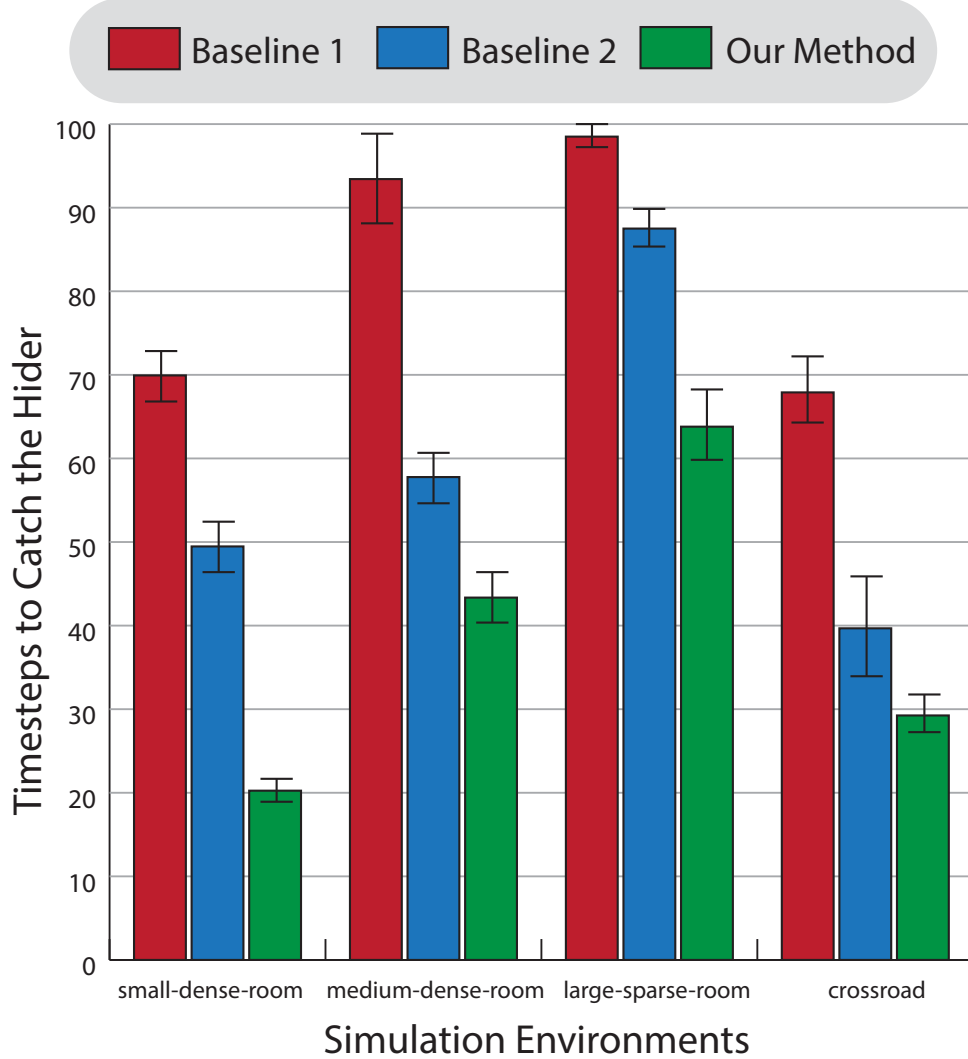


Figure 4.12: Results of Competitive Evaluation: Mean and Standard error of timesteps taken by the seeker to catch the hider for each experiment aggregated across all hider policies for a given environment and number of trials (lower is better)

seeker used RHP to minimize its distance to the perceived position of the hider, it frequently entered the hider’s field of view, making the hider trigger a fixed evasive maneuver. The seeker’s estimate of the hider’s strategy then became inaccurate. With our method, the seeker entered the hider’s field of view to trigger an evasion and then exited it to observe the hider at the Critical Decision Point continue to its original goal. It then selected actions to intercept the hider quickly once it learned the hider’s strategy.

Real World

We demonstrated our method on the real robot across five trials each for six policies, as illustrated in Fig. 4.11. A researcher controlled the hider robot, with access to only the map of the environment and the first-person view from the robot’s camera. Our method took the seeker 27 ± 3 timesteps to catch the hider. Baselines 1 and 2 took the seeker 39 ± 2 timesteps and 24 ± 1 timesteps longer than our approach to catch the hider, respectively.

4.5.6 Comparative Analysis of Performance of CDPs in the Competitive Task

In this work, we developed a novel method for a robot to identify a human’s strategy with limited observability of human actions. Assuming the robot has access to reasonable policies a human might follow, the robot examines the actions these policies predict at all states in the environment. Then, by observing humans at the states where most policies diverge (which are CDPs), the robot can more quickly predict which of the strategies the human is using in the environment. However, the number and spatial distribution of CDPs depend on both the layout of the shared environment and the range of plausible human strategies under consideration. In some environments, CDPs are abundant, and passive observation suffices. In others, they are rare, and the robot must actively optimize its behavior to increase the chance of observing the human at a CDP.

We demonstrated our approach in a game of hide-and-seek in simulation and the real world, assuming that the human’s strategy remained unchanged throughout a single game. Our results suggest that influencing a human-controlled hider toward Critical Decision Points can help an autonomous robot infer the human’s strategy more quickly in the game.

4.6 Discussion

4.6.1 Applicability Across Diverse Task Settings

This work evaluates our method in two domains that represent opposite ends of several key axes. The collaborative cooking domain is fully observable, features a large state space, and involves a cooperative human partner who shares the robot’s objective. In contrast, the competitive navigation domain is partially observable, has a relatively small state space, and models an adversarial human whose goal is intentionally hidden from the robot. Despite these differences, our approach of leveraging policy divergence to identify CDPs and influence humans toward those CDPs proves effective for identifying the human’s goal in both settings. This highlights the flexibility of our framework, which does not assume a fixed interaction mode, observability level, or environment size. As such, it may be adapted to domains that combine properties along these axes, for example, semi-collaborative settings, mixed observability, or medium-sized domains, provided task-specific models and policy banks can be defined.

4.6.2 Limitations and Future Work

We make some limiting assumptions about our experimental setup that may not apply in the real world. We demonstrate our approach in a deterministic environment. We assume that humans always act rationally, make no mistakes, and do not change the task during the interaction. We also assume that the interaction is turn-based, with neither agent able to take a “*wait*” action, and actions once taken by either agent cannot be repeated in the collaborative case. Our approach also requires an explicit policy bank, which may not be readily available in many real-world scenarios. It assumes that the task can be represented in a discrete space. Future work will involve relaxing these assumptions to support suboptimal human decision-making and more

varied action execution.

4.7 Summary

In this chapter, we addressed goal ambiguity in interaction settings where the same observed human behavior can be consistent with multiple possible human goals, making passive observation insufficient for establishing shared understanding early in a task. We introduced **Critical Decision Points (CDPs)** — states where competing goal-conditioned strategies prescribe different immediate actions, and formalized them using a policy-divergence measure that quantifies how strongly goal hypotheses disagree about the human’s next action. Building on this formulation, we proposed optimizing a receding-horizon planner that trades off task progress with information gain, selecting robot actions that steer the interaction toward highly informative states without interrupting the task with explicit queries. Across evaluations in both a collaborative, fully observable cooking task and a competitive, partially observable hide-and-seek task, in simulation and on real robots, our approach inferred goals earlier and more accurately than baselines. In collaboration, earlier clarification allowed the robot to better align its actions with the human’s goal and reduce the number of incorrect actions; in competition, influencing the human-controlled hider toward CDPs enabled the seeker to catch the hider more quickly than baselines.

The work presented in this chapter assumes that the human’s goal remains stationary throughout an interaction. The next chapter addresses this gap by extending our active approach to influence people to reveal their goals when their goals can change as the interaction progresses. Our approach detects when a human partner shifts objectives mid-task and efficiently identifies the new goal by selectively leveraging past information that remains relevant even when the human’s goals change, and demonstrates its effectiveness in a collaborative cooking task.

Chapter 5

Active Inference of Shifting Goals during Human-Robot Collaboration¹

The previous chapter showed how a robot can reduce goal ambiguity by planning its actions to guide an interaction toward states that make different possible human goals easier to distinguish, assuming the human’s goal remains fixed throughout the interaction. In this chapter, we study a more realistic setting in which a human may change goals mid-task, creating two challenges for the robot — detecting when a goal switch has occurred, and rapidly realigning to the updated goal to provide adequate support to the human without requiring explicit communication [87]. We introduce an approach for goal change detection that reasons about when observed behavior becomes inconsistent with the robot’s current goal hypothesis. Following detection, the robot revises its goal belief by retaining only the subset of past observations that remain informative under the new possible goals, rather than treating the full inter-

¹Portions of this chapter were published as: Ghose, D.*, Gitelson, O.*, Jin, R., Abawe, G., Vázquez, M., & Scassellati, B. (2025, November). *I’ve Changed My Mind: Robots Adapting to Changing Human Goals during Collaboration*. *Robotics and Automation Letters (RA-L)*. *equal contribution [87]

action history as equally relevant. The robot then executes *Differentiating Actions* — supportive actions chosen to both advance the task and elicit behavior that efficiently confirms the updated goal, allowing it to recover shared understanding faster than approaches that assume goals are stationary.

5.1 Introduction

In real-world scenarios, humans often change their goals in response to evolving circumstances, new information, or spontaneous decisions. Previous work often addresses changing human goals by relying on explicit communication [216, 171, 38]. While effective, relying on communication assumes humans can and will communicate with the robot, which is often impractical due to physical, situational, or cognitive constraints [130, 93, 166, 86, 3]. Moreover, in complex tasks, humans may struggle to articulate their goals clearly, leading to misinterpretation. On a busy construction site, for example, a mason and a robot may start with a brick-laying task, the robot assisting by mixing cement and bringing sand, water, and buckets. When a sudden rainstorm halts exterior work, the crew pivots indoors to install drywall panels instead. The materials the robot has already delivered remain useful for rinsing tools and cleaning surfaces, but mixing additional mortar and staging bricks is now counter-productive. Because the noise of heavy machinery and hearing protection prevents verbal updates, the mason cannot notify the robot; only by observing that the worker now reaches for drywall screws rather than bricks can the robot infer the new objective and adapt. Such goal changes are challenging for robots, especially when there is an overlap in the sequence of actions required to achieve different goals. Without explicitly modeling these goal changes, the robot cannot understand the updated goal effectively or determine which past actions should be retained as relevant context.

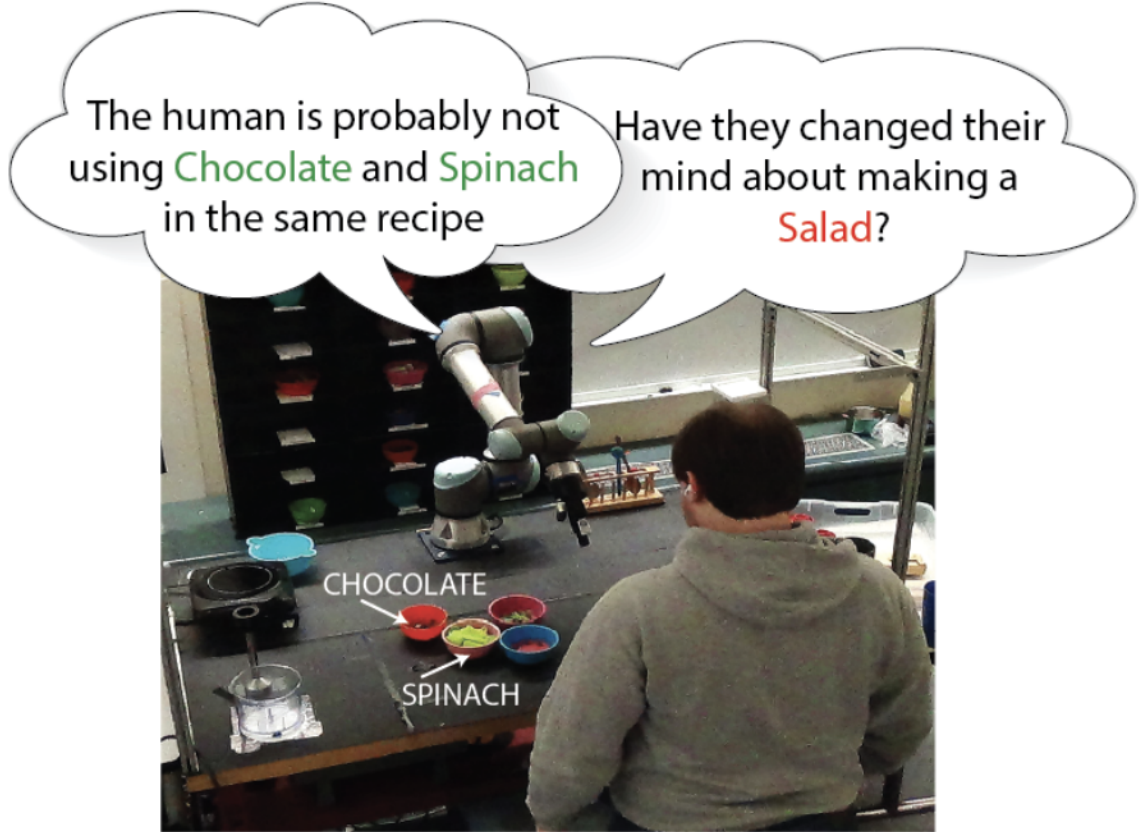


Figure 5.1: Robot reasoning if the person has changed their mind about making the salad, after observing that the person has picked chocolate.

To that end, we propose a method to explicitly model goal changes without relying on explicit communication. Our approach identifies when likely goal changes occur by tracking multiple candidate action sequences from the history of the collaboration and verifying their plausibility against a policy bank. Once a goal change is detected, the robot refines the set of plausible past actions and constructs Receding Horizon Planning (RHP) trees (similar to Chapter 4) for each sequence, performing actions that assist the human while encouraging them to take *Differentiating Actions*, which are actions that reveal crucial information about their updated goal.

We evaluate our approach in a collaborative cooking environment (as shown in Fig. 5.1 and described in Chapter 4), both in simulation and on a physical robot. Cooking provides a structured yet realistic setting where goal changes are easily observable. A human and a robot can cook up to 30 unique recipes in our setup. Our primary point

of comparison is the *Recursive Bayesian* [137] approach that passively models likely goal changes based on observation of human actions. We dynamically manage the relevance of past actions while taking an active approach to human goal prediction, which allows us to outperform the Recursive Bayesian approach across three plausible types of human behavior. Additionally, to illustrate the importance of explicitly modeling goal switches and highlight the magnitude of performance improvement that considering goal switches enables, we benchmark these approaches against two active goal prediction techniques designed for cases where human goals are static: 1) *Critical Decision Points (CDP)*[92], where a robot actively selects actions to support human goals while influencing them to reveal critical information about their goal as described in Chapter 4, and 2) *Information Gain Maximization* [230], where a robot actively maximizes its information gain about a human’s goal at every state without necessarily helping them achieve it.

5.2 Related Work

In human-robot collaboration, a significant body of work focuses on robots inferring human goals by observing human actions and inferring their intent (for a survey, refer to [121] and Chapter 2). Most of this work assumes that human intent remains consistent throughout the collaboration [193, 93, 48, 121, 92, 157, 109, 204, 86, 215, 29]. This assumption does not hold in the real world, as humans can change their minds spontaneously and frequently in response to evolving circumstances and new information. Recent methods for handling changing human goals fall into two categories:

5.2.1 Passive Goal Inference

Passive approaches rely solely on observing human actions to infer goals [10]. Jain and Argall [137] use recursive Bayesian filters to update goal beliefs, while Murata

et al. [183] use gradient descent for dynamic goal adjustments. Both rely on short action histories, making them unsuitable for tasks with overlapping actions. For example, chopping vegetables applies to both a salad and a stew, and these methods lack mechanisms to determine which past actions remain relevant after a goal change. Hiramatsu et al. [117] improve on these by explicitly modeling goal transitions.

5.2.2 Active Human Influence

Active approaches aim to influence humans to modify their goals, particularly under uncertainty. Pandya et al. [204] use reach-avoid dynamic games to guide humans toward safer goals, while Pandya et al. [205] extend this line of work by influencing humans to adopt more precise goals. These methods assume humans are uncertain and receptive to influence [249], which is impractical when humans are confident in their objectives. Moreover, they neglect the action history before the goal change and do not explicitly model goal changes, limiting their utility in tasks with overlapping actions or abrupt changes.

Existing methods for goal inference are limited in handling goal changes. Passive approaches struggle with abrupt changes or overlapping tasks because they rely on incremental updates and short-term action histories. Active approaches fail to address scenarios where humans are certain about their goals and cannot dynamically manage action relevance, either retaining all past actions, causing confusion, or discarding history entirely, losing valuable context.

To address these limitations, we explicitly model goal changes by identifying their precise timing and dynamically managing the relevance of past actions. We introduce *Differentiating Actions*, key actions that clarify human intent after a goal change. Using Receding Horizon Planning (RHP) [172], we guide humans to reveal updated goals without relying on explicit communication, enabling robust adaptation to complex, real-world collaborative tasks. To validate our approach in a collaborative cooking

task, we benchmarked it against several goal-prediction methods.

5.3 Method

We present an algorithm that lets a robot both predict and assist a human partner whose intentions may change mid-task. Consider urban way-finding: a guide robot escorts a pedestrian whose destination is unknown. At each major intersection, the pedestrian’s selected street eliminates most remaining destinations; we call such pivotal turns *Differentiating Actions*. The robot plans its trajectory to influence the human to these informative intersections while staying on routes that best serve the currently most probable destination. Suppose the pedestrian deviates, for example, by turning toward an unexpected café. In that case, the robot interprets this as a goal switch, updates its belief about possible destinations, and adjusts its assistance accordingly.

5.3.1 Preliminaries

The environment’s state is represented as s , with a shared action space $\mathcal{A}$. The state and the action spaces are fully observable to both the human and robot at all times. The action space contains high-level actions (e.g., in a cooking task, the action space can include boiling water, chopping vegetables, etc.). Both agents have access to a shared action history $\mathcal{H}^{0:t-1}$ up to (but not including) the current time step, and a complete goal bank $\mathcal{G}$, listing all possible human task goals. We also keep track of the individual action histories of the human and the robot, $\mathcal{H}_h^{0:t-1}$ and $\mathcal{H}_r^{0:t-1}$. Additionally, the robot has a policy bank $\mathcal{P}$, which contains many (but not all) potential policies to achieve the goals in $\mathcal{G}$. Each policy ($\pi \in \mathcal{P}$) maps states to actions to progress toward completing a task (e.g., a recipe within a cooking task) for a given goal. We can derive a bi-directional mapping from the policy bank specifying

which actions help accomplish which goals. We assume actions cannot be repeated in an interaction, and collaboration occurs in a turn-taking manner.

The robot aims to reduce uncertainty about a human’s current goal and perform actions that support it. At each timestep, the robot infers g_{pred} , a probability distribution over the human’s possible goals, and at alternate timesteps, selects a robot action a_r .

5.3.2 Active Selection of Robot Actions to Reveal Human Goals

Like prior work on active goal inference [92, 230], our robot acts to reduce its uncertainty about the human’s goal. It does so by selecting actions that steer the partner to perform *Differentiating Actions* – human actions that are informative because they advance only a small subset of feasible goals. Actions that aid nearly every remaining goal, or none of them, reveal little; actions that help just a few narrow the robot’s belief and accelerate correct identification of the human’s goals early in the interaction.

Building on Chapter 4, we formulate the problem of the robot influencing the human to take Differentiating Actions as a discrete-time planning problem, which we solve using Receding-Horizon Planning (RHP) [172].

Expanding the RHP Tree

The robot expands an RHP tree to reason about how its future actions could influence the human’s subsequent actions, considering the past actions taken by the human. The RHP tree is rooted in the history of past actions taken by the human $\mathcal{H}_h^{0:t-1}$. Each branch of the tree represents a sequence of potential actions that both agents can take from the root node in a turn-taking manner.

Choosing Robot Actions

The robot selects differentiating actions using attractor fields – a concept initially developed for motion planning [147]. The core idea is that certain states (e.g., areas in a map for motion planning) exert attractive or repulsive forces that influence an agent’s decision. Attractive forces pull the agent toward desirable states (e.g., navigating to a goal), while repulsive forces push it away from undesirable states (e.g., avoiding obstacles).

Based on Jain and Argall [137], we adapt this concept from physical space to action space, where certain actions exert an attractive or repulsive force depending on whether they align with a human’s goals. We model each goal as such an attractor field over the action space. Intuitively, an attractor field over actions represents which actions are relevant for its corresponding goal. Mathematically, in an action space with n possible actions, a uniform attractor field for a given goal g is represented as $\{0, 1\}^n$, where position i is 1 if action i supports g and 0 otherwise. If attractor fields are summed element-wise over a given set of goals, an action will be more attractive if it is relevant to more goals and less attractive if it helps accomplish fewer goals. Crucially, attractor field values are strictly non-negative in our formulation of this concept.

In the context of attractor fields, a differentiating action possesses *low, non-zero attraction*. Actions with zero attraction are irrelevant to any goals under consideration and should be avoided by both the robot and the human. Conversely, if an action is very attractive, it pertains to numerous goals, making it hard for the robot to identify the human’s specific goal if such an action is taken by the human.

We define the plausible goals for the history of human actions $\mathcal{H}_h^{0:t-1}$ as the set of goals for which every action in $\mathcal{H}_h^{0:t-1}$ helps accomplish every goal. Intuitively, by aggregating attractor fields across all plausible goals, we effectively create a measure of how helpful each action will be on average, given the action history. This allows the

robot to identify and perform actions that are supportive for many plausible goals. As actions cannot be repeated, the robot taking many broadly helpful actions effectively steers the human to take actions that are supportive of fewer goals, which reveals more information about their actual goal.

For $\mathcal{H}_h^{0:t-1}$, we compute the summed attractor field over its plausible goals similar to [137], referred to as *attractor_field*. The initial cost at the root node ($d = 0$) of a branch b for an RHP tree is then calculated as:

$$branch_costs[b, 0] = -1 \cdot attractor_field[b[0]],$$

where $b[0]$ is the first action on branch b . As the tree expands, the cost of each branch is progressively updated at increasing depths d by subtracting the *attractor_field* value for $b[d]$, the action at depth d using:

$$branch_costs[b, d] = branch_costs[b, d - 1] - attractor_field[b[d]]$$

Notably, the cost is calculated only at timesteps when the robot would act. If the cost were updated on human timesteps, it would steer the robot towards influencing the human to take high-attraction actions, which would not be differentiating. Conversely, including robot timesteps influences high-attraction actions that are likely to be relevant (satisfying the action-to-goal mapping from $\mathcal{P}$), regardless of the human's goal. Also, we only update the cost for a branch if it was one of the lowest-cost branches at the previous depth, to prevent the robot from performing an irrelevant action earlier in favor of a potentially relevant future action.

5.3.3 Identifying Likely Human Goal Changes

Assume the human starts by following a policy π_{h1} , which maps states to actions to achieve the goal g_{h1} . At some point in the interaction, the human might decide to switch to a different goal g_{h2} , adopting a new policy π_{h2} , where both g_{h1} and g_{h2} belong to the goal bank $\mathcal{G}$.

Tracking Candidate Human Action Sequences

Since the human's goal may change at any time, the robot does not assume that every past action remains relevant to the current goal. Instead, the robot tracks candidate human action sequences, denoted as $\mathcal{C}_h$. These sequences are drawn from the complete history of human actions, represented as: $\mathcal{H}_h^{0:t-1} = (a_h^{(0)}, a_h^{(2)}, \dots, a_h^{(t-1)})$ where each $a_h^{(i)}$ represents an action taken by the human at timestep i . Each candidate sequence $c_h \in \mathcal{C}_h$ is a subset of actions in $\mathcal{H}_h^{0:t-1}$, meaning it consists of only some of the human's past actions. These sequences represent possible actions that might still be relevant to the human's current most likely goal. The candidate sequences are chosen so that all actions can be taken while pursuing the given goal. Because different goals may require different subsets of actions, and the robot is uncertain about the human's true goal, the robot needs to track multiple candidate sequences simultaneously. Each sequence reflects a different hypothesis about which actions still matter. Whenever the human takes a new action $a_h^{(t)}$, the robot updates all candidate sequences by adding this action to each of them.

Detecting a Goal Change

To determine whether the human has changed their goal, the robot checks whether the observed human actions still align with any goal reachable by following a policy in the policy bank $\mathcal{P}$. For each candidate action sequence $c_h \in \mathcal{C}_h$, the robot verifies whether it is plausible that a human following a given policy from $\mathcal{P}$ could have

taken those actions. This means checking if there exists a policy in $\mathcal{P}$ that could have generated the observed sequence or any of the permutations of actions in the observed sequence for that given goal. If at least one candidate sequence matches a policy in $\mathcal{P}$, the robot assumes the human is still following the given goal from the previous timestep. If all sequences become implausible, it suggests the human’s actions may no longer align with any goals in the goal bank $\mathcal{G}$, indicating a likely change to a different goal. In other words, if no tracked subset of human actions ($\mathcal{C}_h$) is consistent with any goal, the robot infers a goal switch; if at least one subset remains consistent with one goal, it cannot be sure a switch occurred. Here, we define a sequence as *consistent* if there is a policy $\pi \in \mathcal{P}$ such that the observed sequence is a subsequence of π .

5.3.4 Robot Aligning Actions with Updated Goals

Once a goal change is detected (as described in Sec. 5.3.3, Fig. 5.2, Step 1), the robot must reassess which past actions are still relevant and determine the best way to assist the human while reducing uncertainty about the new goal.

Filtering Relevant Human Actions

The robot aims to identify past actions relevant to the human’s likely new goal. A goal change may render some previous actions useless, prompting the robot to clear its list of relevant robot actions ($\mathcal{H}_r^{0:t-1}$) and find subsets of human actions likely to contribute to the new goal. However, simply considering all possible subsets of past human actions could be computationally intractable, as the number of possible sequences grows exponentially with the number of actions taken. The robot ranks actions $a_h^{(i)} \in \mathcal{H}_h^{0:t-1}$ based on *Generality*. Generality measures how commonly an action appears across different goals in the policy bank $\mathcal{P}$. More general actions (i.e., those that occur in many policies) are more likely to be relevant, even after a goal

shift. We progressively eliminate the least general action in an action sequence from $\mathcal{H}_h^{0:t-1}$. This effectively leads the robot to perform a ranking and enumeration over all possible subsets of human actions, prioritizing longer subsets composed of more general actions. When the robot encounters a subset of actions that could have all been taken while following the same goal, the robot selects this subset, appends the latest human action to it (Fig. 5.2, Steps 2 and 3), and adds it to $\mathcal{C}_h$. The exploration continues until $\mathcal{C}_h$ reaches length j (or till there are no more subsets to explore). j is a hyperparameter specifying how many candidate sequences to track.

Expanding RHP Trees for Each Plausible Action Sequence

Since there can be multiple plausible action sequences ($\mathcal{C}_h$), the robot cannot assume any one sequence is correct since it is uncertain about the human’s goal. Instead, it considers multiple plausible interaction histories ($c_h \in \mathcal{C}_h$) and evaluates potential robot actions under each hypothesis. To do this, the robot constructs an RHP tree for each sequence c_h (following the procedure described in Sec. 5.3.2). Each tree begins at a root node representing one of the plausible candidate sequences c_h and expands by simulating future actions for the human and the robot (Fig. 5.2, Step 4).

To decide which action to take that minimizes uncertainty over a human’s goals, the robot computes a cost for each branch in every tree, using the attractor field-based cost function from Eqs. 5.3.2 and 5.3.2 (Fig. 5.2, Step 5). As explained in Sec. 5.3.2, this cost function aims to steer the human to take differentiating actions early in the interaction. The cost update rule from Eq. 5.3.2 is applied only if the branch b had the minimum cost at the previous depth $d - 1$. This prevents the robot from prioritizing branches with suboptimal early actions in favor of better actions later, as the belief in the human’s goal may shift before reaching those later actions. Finally, once the RHP tree is expanded for all plausible action sequences $\mathcal{C}_h$, the branch with the minimum cost summed across all RHP trees is selected. If the same

action sequence occurs in multiple trees, we sum their costs, because they are more likely to be relevant regardless of which candidate action sequence supports the true human goal. The robot then takes the first action in that branch and reconstructs the trees after each human response.

5.4 Evaluation

As described in Chapter 4, collaborative cooking scenarios are a standard benchmark for human-robot collaboration algorithms [38, 45, 277, 99]. These tasks often involve overlapping goals, for example, preparing a smoothie or a parfait shares actions like chopping fruits or fetching yogurt, and mirror real-world situations where humans frequently change goals mid-task [38]. This makes cooking scenarios ideal for evaluating our approach via simulation and physical setups. For such evaluation, we use 30 unique recipes, each of which can serve as the initial goal. After the switch, the human could adopt any of the remaining 29 recipes as the new goal (we disallow “switching” to the same recipe). Thus, for every first-goal choice there are 29 distinct second-goal options, giving $30 \times 29 = 870$ pairs of (first goal, second goal).

5.4.1 Task and Problem Representation

We assume that both the human and the robot can perform any step necessary to prepare a recipe and each step takes the same amount of time, regardless of the task or who performs it. We also assume a deterministic environment with a fully observable state space. Our experimental setup features a robot and a user in a simplified kitchen environment (Fig. 5.1). The kitchen includes a blender, a sink with water, a pan, a stove, a serving bowl, a glass, a serving spoon, an eating spoon, a measuring cup, and a storage shelf containing the necessary ingredients. Ingredients are considered non-depletable, meaning a single ingredient can be used in multiple containers. Meals

1. Detect Goal Change

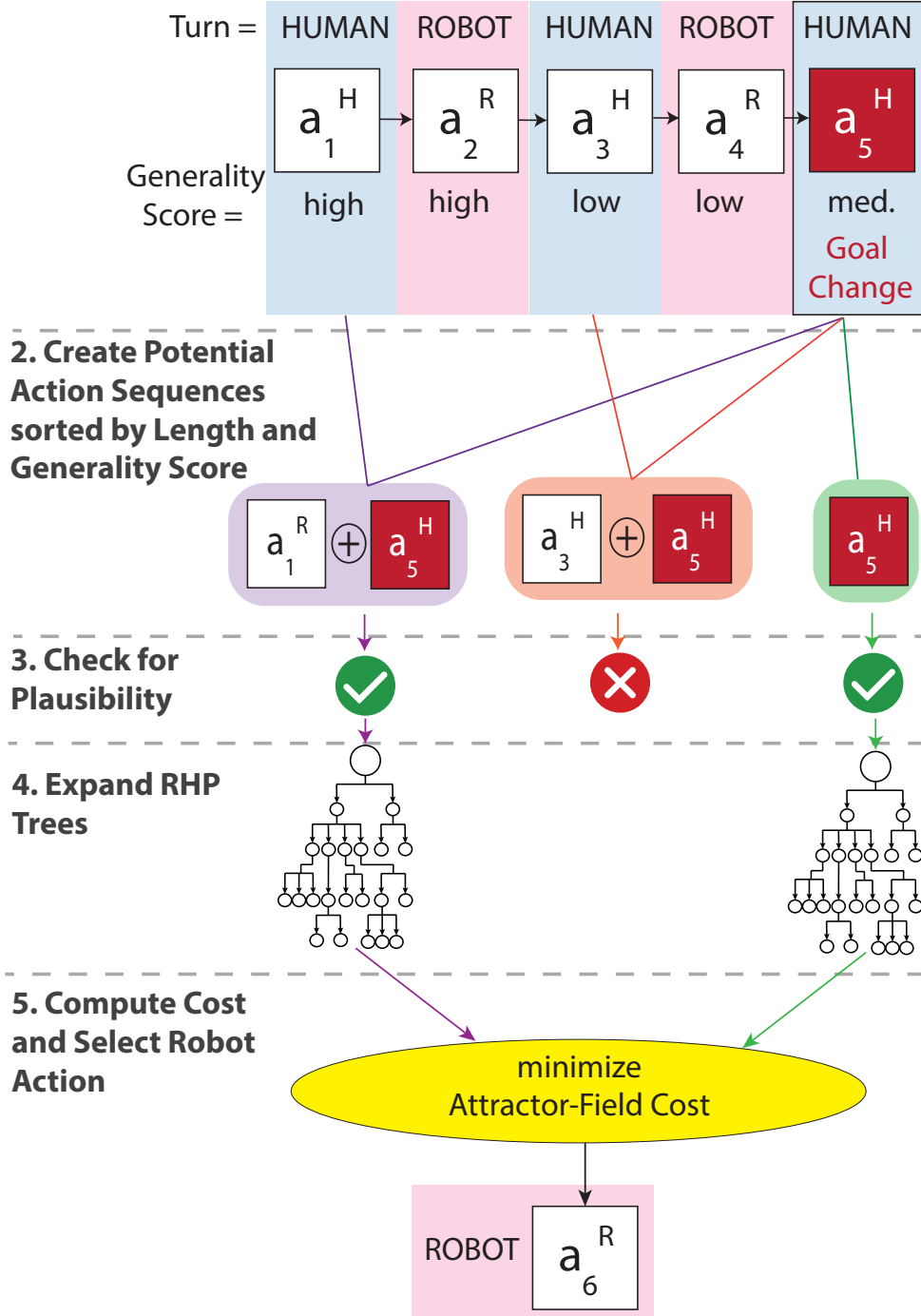


Figure 5.2: Steps to Select Robot Action after Goal Change Detection 1. Detect Goal Change if the actions taken during the collaboration don't match any sequences in $\mathcal{P}$. 2. Create potential action sequences by concatenating the action at which the goal change was detected with past actions sorted by increasing generality scores. 3. Check if the potential action sequences are plausible. 4. For all plausible action sequences, expand the RHP trees. 5. Compute attractor field cost and select an optimal next robot action.

in the experiment could be one of six types: Pasta, Stew, Salad, Oatmeal, Smoothie, or Parfait.

The robot’s state and action space are defined using the Planning Domain Definition Language (PDDL) [4]. We used the PDDLgym library [243] to update the environment’s state based on actions taken by both the human and the robot. Both the human and the robot could perform the following actions:

$$A = \{\text{get}(i), \text{pour}(i, d), \text{mix}(i), \text{cook}(i), \text{blend}(i), \text{turn_on}(a), \\ \text{collect_water}(d), \text{reduce_heat}(a), \text{serve}()\}$$

where i is an ingredient, d is a dish such as a bowl, glass, or pan, and a is an appliance. The state space is represented as a set of PDDL literals, with each literal denoting a specific environment feature, like an ingredient’s location (workspace or storage), the state of a container or appliance (e.g., whether the stove is on or off), or the status of a meal (e.g., blended or cooked). Such symbolic abstractions are standard in planning and reasoning [37], as they allow tractable inference over long-horizon tasks. In our cooking experiments, the human verbalizes each action, which we map to the appropriate discrete symbol in the policy bank using fuzzy matching. Other non-intrusive mappings are equally feasible, such as visually detecting state changes (e.g., an ingredient placed in the workspace) or using human action recognition models.

5.4.2 Experiments

In our experiments, the goal for both the human and the robot is to work collaboratively on a combination of two recipes. By combination, we refer to a setup in which the human could start making one recipe and, partway through the interaction, change to another recipe that they complete. In our experiments, only humans know

their current goal, and the robot must infer their goal by observing their actions.

We model our simulated human using Clique/Chain Hierarchical Task Networks (CC-HTNs) [114], which can represent both actions that must be performed in sequence and those that can be executed in any order. This makes CC-HTNs ideal for modeling recipes. Each recipe is represented by its own CC-HTN, which the simulated human uses to select actions. We assume that the human always takes the first action, alternating with the robot thereafter. We evaluate our method using three human models:

1. **Human with Fixed Goal:** The human follows a single, unchanging goal throughout the interaction (common assumption on human behavior used in literature [92, 199]).
2. **Optimal Human with Goal Changes:** The human may change their goal randomly at any point during the interaction. All actions taken by the human are optimal and correctly aligned with their current goal (a common paradigm used in literature [137, 204])
3. **Suboptimal Human with Goal Changes:** The human may also change their goal randomly during the interaction. However, the human’s actions may include mistakes, where the probability of making mistakes at any step is $p = 0.1$. This mode of behavior is studied in [46, 126, 155].

To manage computational constraints and conduct controlled experiments, we ensure human models 2 and 3 experience only one goal change after a certain number of steps by fixing the goal post-change. This restriction to a single switch is not a requirement of our method, but an experimental control to ensure comparability across trials.

Table 5.1: List of Recipes that could be part of the combination that the Human and the Robot could collaboratively cook in our experimental setup & the required ingredients.

cct

Food Type	Basic Ingredients	Variants	Additional Ingredients
Oatmeal	water, oats, salt	Plain, Fruit, Chocolate, Fruit Chocolate	strawberries, blueberries, chocolate chips
Parfait	yogurt, sugar	oats, Plain, Fruit, Fruit Nut, Fruit Chocolate, Nut Chocolate, Fruit Nut Chocolate	strawberries, blueberries, nuts, chocolate chips
Pasta	tomato, pasta	oil, Tomato Cheese, Veggie, Veggie Cheese	cheese, carrot, onion, broccoli
Salad	cucumber, lettuce, onion, kale, salt, oil	lettuce, tomato, avocado, Fruit Avocado, Fruit Nut, Pasta Fruit Nut	pasta, avocado, nuts, bananas, apple
Smoothie	water, yogurt, sugar	Berry, Fruit, Chocolate, Chocolate Fruit, Kale Tomato, Kale Berry	strawberries, blueberries, bananas, apple, chocolate chips, kale, tomato
Stew	water, onion, lettuce, cucumber, carrot, broccoli	salt, tomato, cucumber, carrot, Veggie, Cheese, Cheese	avocado, cheese

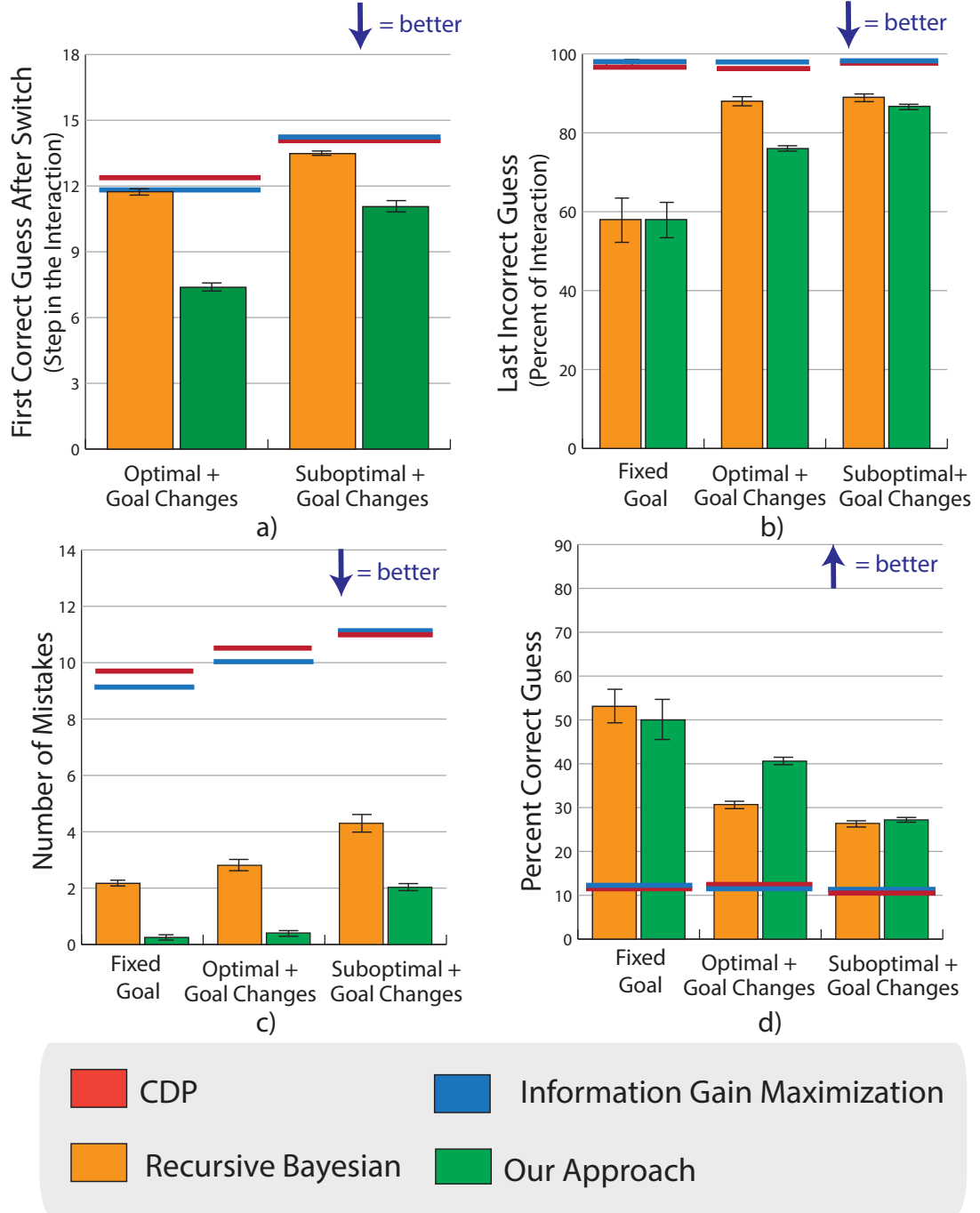


Figure 5.3: Comparison of performance measures for the proposed method, CDP [92], Information Gain Maximization [230], and Recursive Bayesian [137] in a collaborative cooking simulation. Metrics include (a) first correct guess after a goal switch, (b) last incorrect guess, (c) number of mistakes, and (d) percentage of correct guesses. Results are shown for three simulated human types: stubborn (no goal change), optimal (stochastic goal updates), and suboptimal (stochastic updates with mistakes). In the stubborn condition, all four methods are compared; in the other two conditions, our method is directly compared with Recursive Bayesian, while the red and blue lines indicate the empirical performance bounds provided by CDP and Information Gain Maximization, respectively.

5.4.3 Implementation Details

Approximating the Belief Distribution

Our policy bank $\mathcal{P}$ includes various permutations of action sequences for preparing each goal meal in the goal bank $\mathcal{G}$. Since many recipe steps can be performed in different orders, the number of possible sequences is prohibitively large, making exact computation of the goal distribution infeasible. Instead, we approximate this distribution with a machine learning model. Specifically, we trained a Random Forest classifier on 300,000 sampled sequences (10,000 per recipe across 30 recipes), where each sequence was encoded as a binary presence vector over the action space. The classifier used 200 trees and standard regularization (minimum two samples per leaf, minimum ten samples to split an internal node), yielding 86.8% accuracy on a held-out 20% test set. We also experimented with a transformer-based sequence model, which performed comparably but incurred significantly higher latency. Given the real-time demands of our setting, we selected the Random Forest for tractability. Our approach, however, is agnostic to the choice of classifier.

Pruning the RHP Tree

Due to the large number of valid actions at each state, expanding our full RHP tree to arbitrary depth is computationally prohibitive. We prune the tree by computing the probability of each action based on its preceding sequence using an offline 2-gram approach, which counts how often an action follows a given past action. We then restrict expansion to the $b = 5$ most likely actions (branching factor), allowing the tree to reach a depth of $d = 3$. To construct candidate human action sequences (Sec. 5.3.4), we set $j = 3$, which specifies the number of candidate sequences tracked. When an action sequence is absent from the n-gram dataset due to the combinatorial growth of unique sequences, we rely on the belief distribution from our random forest model

to select branches leading to the most probable goals. In the worst case, expansion of the RHP tree grows as $O(j \times b^d)$. With these pruning parameters ($b = 5, d = 3, j = 3$), runtimes remain tractable, requiring 2–5 seconds per robot action on an NVIDIA RTX 4090.

5.4.4 Comparisons

To benchmark our approach, we implemented three algorithms in simulation by adapting their original implementations to the collaborative cooking task. Our primary point of comparison is Recursive Bayesian [137], designed to handle changing human goals. Additionally, we benchmarked two active goal detection algorithms, Critical Decision Points [92] (as described in Chapter 4) and Information Gain Maximization [230], known to perform well for static goals but not for dynamic ones. This comparison establishes empirical performance bounds for our method and Recursive Bayesian, highlighting the value of designing algorithms that explicitly model goal changes.

Recursive Bayesian

The Recursive Bayesian algorithm [137] employs Bayesian filtering and a Hidden Markov Model (HMM) to compute goal probabilities. The method assumes that humans can change their goals stochastically during interactions while still taking optimal actions. For a given goal g , it calculates the likelihood of the most recent action under g and multiplies it by the HMM state probability, $\sum_{g_{t-1} \in G} P(g_t | g_{t-1}) \cdot b(g_{t-1})$. For action selection, actions are modeled as attractor-repeller fields; an action a attracts g if it aids in achieving g , and repels otherwise. They represent a as a vector with an entry of 1 for an attractor and -1 for a repeller for each goal. Fields are generated from the most recent human action and candidate robot actions, and the robot selects the action most similar to the human action based on cosine similarity.

This differs from our approach, as they construct attractor fields per-action, whereas we construct an attractor field for each goal. Additionally, we directly use attractor field values for action selection, whereas they use them to compute similarity between the potential next robot actions and the most recent human action.

Critical Decision Points

As described in Chapter 4, in the Critical Decision Points (CDP) approach [92], the robot identifies states where observing human actions yields insight into their goals, treating each state independently and ignoring temporal dependencies. It evaluates the information gained in each state and selects actions to influence humans to reach that state by expanding an RHP tree. The robot observes the human’s response and updates its belief about its goal by executing the first action that minimizes the cost. It then lets the human respond with a valid HTN action, expands the tree from the new state, and iteratively updates its goal probabilities.

Information Gain Maximization

The Information Gain Maximization approach [230] enables a robot to explore a human’s internal state by planning actions that maximize expected information gain. We adapt this by expanding an RHP tree to depth 3, where the robot optimizes a cost function with two terms: one maximizing the magnitude of entropy loss between the root and leaf nodes to reduce ambiguity, and the other rewarding actions aligned with the most likely goal while penalizing deviations. Goal probabilities are computed as detailed in Sec. 5.4.3 and the RHP tree is expanded for the combined cost as explained for Critical Decision Points.

5.4.5 Performance Measures

We compare the performance of our method against the baselines for our proposed task using the following measures.

1. **First Correct Guess After Switch:** The number of timesteps taken to correctly identify the human’s goal after a goal change (irrelevant to the stubborn human model).
2. **Last Incorrect Guess:** Percentage of the interaction that had elapsed when the robot incorrectly guessed the human’s goal for the last time. While this measure considers both goals for human models 2 and 3, the last incorrect guess almost always occurs during the second goal.
3. **Number of mistakes:** The total extra steps taken by both agents to finish the interaction, compared to the ground-truth steps from the respective HTN.
4. **Percentage of Correct Guesses:** Percentage of timesteps when the robot correctly identifies the human’s goals.

5.5 Results

5.5.1 Simulation

Fig. 5.3 quantitatively compares our proposed method with three baseline approaches across several key measures and three human models in the simulated collaborative cooking task (as described in Sec. 5.4). Mean and standard errors were calculated across 870 recipe combinations and averaged over three trials. For clarity, we don’t report standard error bars for the empirical performance bounds. When the human makes no mistakes, the recipe combinations had an average length of 28.7 steps.

Fig. 5.3 a) demonstrates that our approach accurately identifies the human’s goal relatively quickly after detecting a goal switch, especially when compared to other algorithms across both relevant human models. However, this metric can be somewhat noisy, as all methods might occasionally predict the correct goal by chance after a goal switch without resolving uncertainty about the goal.

Therefore, we compute the last incorrect guess metric, as it marks the point in the interaction where the robot no longer makes erroneous predictions about the human’s goal. Fig. 5.3 b) shows that our approach and Recursive Bayesian approach generally achieve this around the midpoint of the sequence when the human does not switch goals earlier than the other two baselines. However, our approach outperforms all baselines when the human switches goals stochastically. This shows that our method can resolve uncertainty about the human’s goals faster than other algorithms, leading to a higher percentage of correct guesses, as shown by Fig. 5.3 d) when the human switches goals stochastically.

Fig. 5.3 c) shows the total extra steps to complete the collaboration. If the robot performs every action incorrectly, the interaction could take twice as long, as the human would need to perform every correct step towards task completion. Conversely, if the robot correctly assists the human, the task can be completed in 28.7 steps, the average length of a ground truth sequence. The number of extra steps in our approach is minimal across all three human models, showing that our approach can help maximize collaboration efficiency.

5.5.2 Physical Robot

We conducted a proof-of-concept study using a real robot and a human collaborator (one of the researchers) to compare our method with Recursive Bayesian [137] in a collaborative cooking task (Fig. 5.4). For each robot step, we checked whether the correct recipe goal appeared among the top three predictions. To ensure fairness,

	Human Action		Robot Action	Action	Incorrect Robot Action
	Position of Goal Recipe 1 in Top-3 Predictions		Position of Goal Recipe 2 in Top-3 Predictions		Position of Incorrect in Top-3 Predictions
Goal 1: Berry Smoothie		Goal 2: Fruit Parfait			
Recursive Bayesian			Our Approach		
Action Sequence	Ground Truth Human Goal	Top-3 Goal Predictions	Action Sequence	Ground Truth Human Goal	Top-3 Goal Predictions
 1. Get Cup	Goal 1		 1. Get Cup	Goal 1	
 2. Collect Water		  	 2. Collect Water		  
 3. Get Blueberry			 3. Get Blueberry		
 4. Get Strawberry		  	 4. Get Strawberry		  
 5. Switch On Blender			 5. Switch on Blender		
 6. Pour Water into Blender		  	 6. Get Glass		  
 7. Get Yogurt			 7. Get Sugar		
 8. Get Glass		  	 8. Get Yogurt		  
 9. Pour Yogurt into Glass	Goal 2		 9. Pour Yogurt into Glass	Goal 2	
 10. Pour Blueberries into Glass		  	 10. Get Oatmeal		  
 11. Get Oatmeal			 11. Pour Strawberries into Glass		
 12. Pour Oatmeal into Glass		  	 12. Pour Oatmeal into Glass		  
 13. Get Sugar			 13. Pour Blueberries into Glass		
 14. Pour Yogurt into Blender		  	 14. Pour Sugar into Glass		  
 15. Pour Sugar into Glass			 15. Get Eating Spoon		
 16. Pour Strawberries into Glass		  			
 17. Get Eating Spoon					

Figure 5.4: Results of the Case Study with the Physical Robot running Recursive Bayesian [137] and Our Proposed Algorithm: A group of columns represent the method. For each group, the first column depicts the agent taking action, the second column depicts the action sequences taken by the respective method, the third column denotes the ground truth human goal, and the fourth column the presence and position of the current ground truth goal recipe in the top-3 most probable recipes from the goal prediction model (left-most being the most probable recipe).

the human performed the same initial actions in both methods and changed goals at the same timestep by taking an identical revealing action without mistakes. In the study, the human initially planned a Berry Smoothie but switched to a Fruit Parfait. In our approach, after berries were fetched and the blender was switched on, the robot supported the initial goal by fetching yogurt and a glass, then quickly detected the switch when the human added yogurt to the glass instead of the blender, and subsequently assisted by fetching oatmeal and adding fruits. Recursive Bayesian correctly identified the initial goal but struggled to update after the change until the

final step.

5.6 Discussion

Our approach relaxes many prior assumptions to reflect real-world human-robot collaboration better. A large body of research work assumes fixed human goals [121], optimal human behavior [86], and non-overlapping actions between goals [292]. In contrast, our approach can handle changing human goals, suboptimal actions, and many overlapping actions between different goals. We achieve this by explicitly modeling when a goal shift occurs by tracking multiple different sets of actions that can be plausibly taken towards a given goal. Our approach expands multiple RHP trees to track potential futures for all plausible sequences simultaneously by utilizing our novel attractor-field-based cost function, which models the relevance of future actions related to past human actions. This effectively selects actions that reduce the robot’s uncertainty about the human’s goals while concurrently supporting them in achieving the most likely outcomes, regardless of whether the human acts optimally.

Two of our baselines, CDP [92], and Information Gain Maximization [230], struggle with dynamic goal changes despite actively influencing the humans to reveal their goals as evident in our results (Fig. 5.3). They determine the most likely goal at each timestep based solely on current observations, ignoring how human actions evolve, often missing goal changes. Therefore, we don’t use these methods as a direct comparison to our work, but instead to serve as empirical performance bounds for our method. On the contrary, Recursive Bayesian [137] captures the progression of actions under changing goals but relies on a short action history. This works well when the human’s goals remain unchanged, but it overlooks valuable context that could help the robot track past relevant human actions. Moreover, this approach relies on passive observation of human behavior instead of actively influencing human

behavior to help the robot reduce uncertainty over human goals. While this is not a fundamental limitation of Bayesian inference, it means that Bayesian approaches must wait for distinguishing actions to occur naturally, which can take significantly longer. Our method differs by enabling the robot to take differentiating actions early, actively disambiguating overlapping goals and thereby accelerating recognition after a change (see Fig. 5.3a)).

We acknowledge several limiting assumptions that may not apply in the real world. Our approach requires an explicit goal and policy bank, assumes tasks are represented in a discrete space, and uses a turn-based interaction model where neither agent can “*wait*” and actions, once taken, cannot be repeated. Future work will relax these constraints to enable human goal prediction without a predefined goal bank.

5.7 Summary

In this chapter, we addressed interaction settings in which a human’s goal is non-stationary, i.e., settings in which the human can shift objectives as the task unfolds. We presented a unified approach that (i) detects goal changes by identifying when the human’s observed behavior becomes inconsistent with the robot’s current goal hypothesis, and (ii) supports rapid re-alignment by updating the robot’s goal belief [87]. Our method selectively retains past observations that remain informative in the new context to seed a revised belief over goals. To efficiently validate and refine this revised belief during execution, we introduced Differentiating Actions — robot actions chosen to simultaneously make progress on the task and elicit human behavior that separates remaining goal hypotheses. Across our evaluations in a simulated and real robot collaborative cooking task, this approach enabled the robot to recover shared understanding more quickly and reliably than baselines.

While this chapter focuses on detecting and adapting to changes that are expressed

primarily through behavior during collaboration, people do not rely on behavior alone to signal evolving goals, constraints, or priorities. In practice, humans frequently communicate changes directly, but they do so at a high level, using short, underspecified utterances that convey intent without detailing how the task should be executed [5]. This creates a gap between what is communicated and how a robot must act, especially when behavioral cues alone are insufficient to disambiguate the human’s intent in a timely way.

Moreover, across Chapters 3–5, the approaches assume that the human’s goal lies within a fixed, known set, even if that goal must be inferred from sparse demonstrations, corrected through interaction, or refined as the task unfolds. Adaptation in these settings operates by revising beliefs, policies, or assistance strategies over this predefined goal space. While effective for many collaborative tasks, this assumption limits the robot’s ability to respond when humans articulate goals or preferences that fall outside what the robot has been designed to recognize.

The next chapter relaxes both these assumptions by addressing open-ended goal inference. It introduces BALI (Bidirectional Action–Language Inference), which combines high-level language with evidence from human actions and task history to infer goals that are not pre-specified in the robot’s model. By jointly reasoning over what people say and what they do, BALI enables the robot to expand its understanding of the task on the fly and provide appropriate support even when the human’s objective was not anticipated at design time [90].

Chapter 6

Open-Ended Goal Inference using Actions and Language for Human-Robot Collaboration¹

The previous chapters developed mechanisms for maintaining shared understanding during interaction, expecting minimal explicit instructions from humans. Chapters 3-5 focused on goal inference from behavior, first by learning task-relevant perceptual distinctions from passive observation, then by actively planning robot actions that make different possible human goals easier to disambiguate, and finally by handling settings where the human’s objective can change over time.

In this chapter, we build upon the ideas presented in the prior chapters by considering how people communicate intent through both what they do and what they say. The robot must interpret these signals together. Language is often underspecified, and action sequences can be ambiguous without the context that language provides. Therefore, to reason about the complementary information that actions and lan-

¹Portions of this chapter were published as: Ghose, D., Gitelson, O., Vázquez, M., & Scassellati, B. (2026). *Open-Ended Goal Inference using Actions and Language for Human-Robot Collaboration*. In *Proceedings of the 2026 ACM/IEEE International Conference on Human-Robot Interaction (HRI)* [90].

guage provides while performing a shared task, we introduce BALI (Bidirectional Action–Language Inference) [90]. BALI jointly reasons over natural-language input, observed actions, and task history to infer the human’s goal as the task unfolds. BALI treats these signals as coupled: language provides constraints and context for interpreting behavior, while actions ground and refine the interpretation of language as the task unfolds. Unlike prior chapters that assume the goal lies in a fixed, pre-defined set, BALI is designed for open-ended goals, leveraging large language models to represent and update goal hypotheses and to select supportive actions under uncertainty. When the uncertainty around open-ended goals is too high, BALI can request targeted clarification; otherwise, it selects actions consistent with its current goal belief.

6.1 Introduction

For robots to be effective partners in real-world human–robot collaboration, they must be able to anticipate and support the human’s end goals [121]. Yet these goals are often unknown, difficult to articulate, or challenging for the robot to interpret, especially in long-horizon tasks where many intermediate actions overlap across different possible outcomes [5]. Further, in many everyday settings, the space of potential goals is effectively unbounded: the same ingredients in a kitchen could be combined into countless recipes depending on preferences; the same tools and parts in an assembly task could produce different products; or a household reorganization could yield multiple equally plausible end states. To collaborate effectively in such settings, a robot must reason over this open-ended goal space while minimizing the burden on its partner, asking clarifying questions only when needed rather than demanding step-by-step instructions or demonstrations [180, 271, 163, 95].

To address the above challenges, we propose ***Bidirectional Action Language***

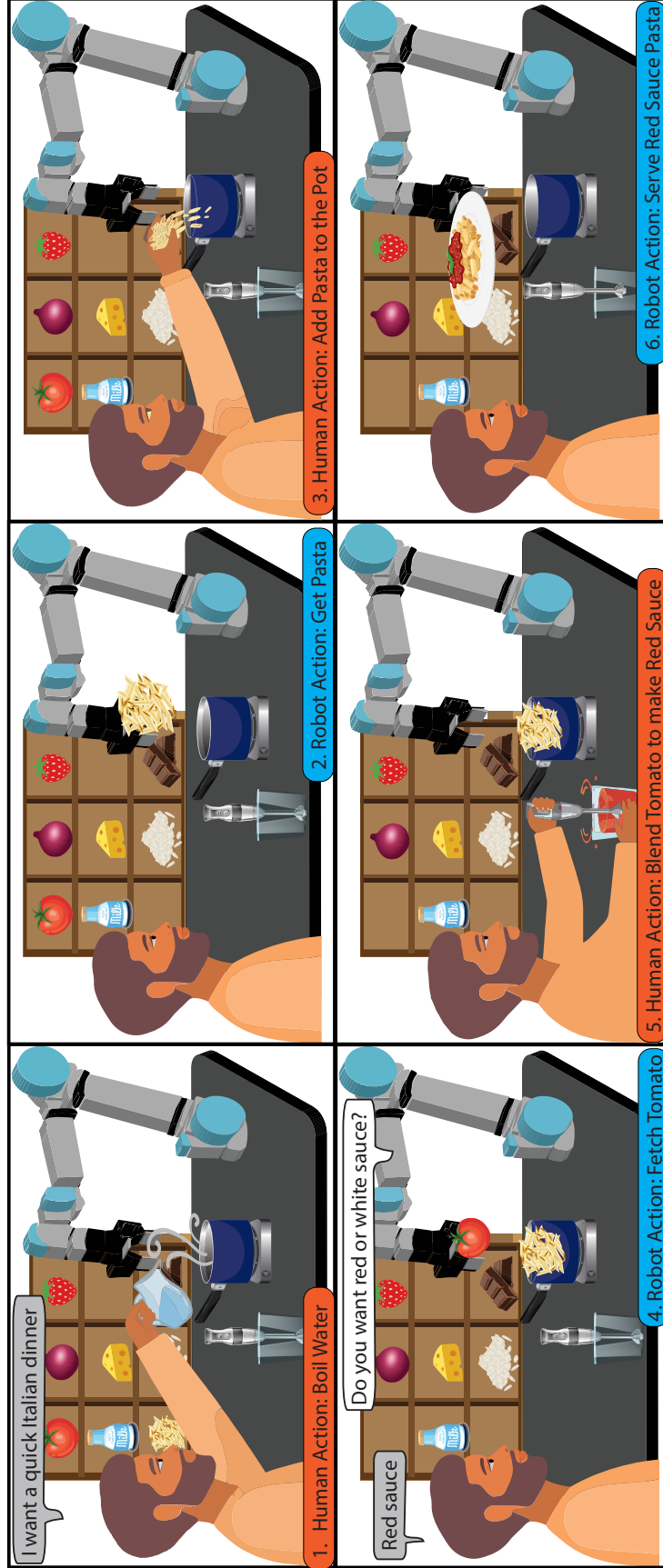


Figure 6.1: Illustrative cooking task. (1) Human says “I want a quick Italian dinner” and leans toward pasta; the robot considers options that fit this preference (pasta vs. rice) and leans toward pasta. (2) Robot fetches pasta. (3) Human adds pasta to the pot, reinforcing a pasta goal. (4) Robot is uncertain about which pasta dish is intended since several variants are possible with the available ingredients, so it asks, “Do you want red or white sauce?” Human replies “red,” and the robot fetches a tomato as a result. (5) Human blends the tomato to make sauce. (6) Robot serves red-sauce pasta, completing the collaboration.

Inference (BALI) for goal prediction, an approach for inferring human goals in collaborative tasks from both language and action. For example, consider a collaborative cooking task. When a person says “*I want a sweet, healthy breakfast*” and then begins fetching oats and fruit, the robot must reason that the likely goals include preparing a *fruit oatmeal* or a *fruit parfait*. BALI leverages ambiguous verbal instructions and observed human actions to infer likely goals and guide robot behavior. Building on prior work, we propose to achieve this with Receding Horizon Planning (RHP), where the robot expands a tree of possible joint future actions toward plausible goals. But with open goal spaces, the tree can grow prohibitively large. Thus, BALI prunes unlikely branches by conditioning the tree expansion on the human’s instructions and observed actions, focusing on those most likely to support the human’s objectives. In effect, our approach enables the robot to use human actions to disambiguate the human’s natural language description of their preferences. Similarly, it allows the robot to use explicit human preferences to guide its choice of supportive actions, establishing the bidirectional coupling between action and language. Furthermore, when uncertainty about the human’s goal is high, such as distinguishing between *fruit oatmeal* and a *fruit parfait*, BALI allows the robot to selectively ask clarifying questions like “*Do you want something warm or chilled?*”. BALI incorporates the human’s response to improve its inferences, as in Figure 5.1.

We evaluate BALI in simulated and proof-of-concept real-world collaborative cooking tasks. We compare against several approaches: using language alone, using actions alone without natural language instructions from the human, and using GOOD [171], a Bayesian reasoning method for open-ended goal prediction using dialog. BALI performs significantly better than these approaches when goals are left open-ended, making fewer mistakes and maintaining strong goal prediction accuracy. Thus, our main contributions are:

1. We introduce BALI, the first approach – to our knowledge – that combines natural

language preferences and observed human actions for goal inference, where goal spaces can be open-ended.

2. BALI integrates a question-asking module that weighs goal uncertainty against interruption cost. Our experiments show that this enables the robot to keep interventions minimal, asking clarifying questions only when needed during goal inference.
3. We evaluate BALI in simulated and proof-of-concept real-world collaborative cooking tasks under open- and closed-world assumptions. In the open case, the goal set is unknown to the robot and potentially of infinite size. In the closed case, the robot has access to the full set of human goals, which total 30 different options. BALI outperforms baselines with fewer mistakes, faster convergence, and higher goal-prediction accuracy.

6.2 Related Work

In human-robot collaboration (HRC), a significant body of work addresses how robots infer human goals from observed actions (see Hoffman et al. [121] and Chapter 2 for a survey). Most approaches assume a discrete hypothesis set and update a posterior over candidate goals via plan or intent recognition. This includes goal inference in shared workspaces [122, 77, 3], in shared-autonomy [140, 218], and navigation with fixed destinations [23, 153]. In these settings, inference is typically short-horizon, estimating instantaneous intent in shared autonomy from recent inputs [138, 15, 14, 70], predicting the next step in workspace collaboration [198, 109, 270], or forecasting navigation sub-goals [262]. Extensions expand the temporal scope through hierarchical or dynamic models and active disambiguation, though still within finite goal sets, as in Bayesian Delegation [292], informative control for self-driving [230], and workspace re-configuration [270]. Other work explores alternative inference mechanisms over fixed goal banks, such as framing goal recognition as reinforcement learning [9], combining

action sequences with spoken language via Bayesian inverse planning with LLM-based likelihoods [298], or augmenting plan recognition with dialog to refine beliefs [132].

A complementary line of research investigates intrinsic motivation, where robots autonomously generate and pursue open-ended goals. Prior work exploits intrinsic signals such as curiosity, empowerment, or uncertainty reduction [220, 226, 185, 74], or relies on demonstrations and tutoring to acquire parameterized policies for parameterized goals [189, 190, 260, 234]. While this work targets lifelong goal discovery rather than inference of a collaborator’s current task goal, it motivates open-ended goal reasoning.

More recently, several approaches have inferred goals without a predefined goal set [170, 171, 311]. Zhi et al. [311] cast goal inference as a Particle Filter over open-ended goal spaces, proposing goals from subgoal statistics and reweighting via inverse planning on observed actions. Ma et al. introduce GOOD [171], which maintains a posterior over unbounded natural-language goals using an LLM to score dialog. Our approach lies at this intersection: unlike methods that rely only on action traces [311] or only on dialog [171], we jointly reason over both observed actions and natural-language signals. This enables maintaining an open-ended goal posterior, predicting a human’s *long-term* goal, and asking clarifying questions when the expected information gain outweighs the interruption cost.

6.3 Preliminaries

Before we explain how BALI works, we describe our problem setup and foundational ideas that led to our approach.

6.3.1 Problem Formulation

A robot aims to support a human in a collaboration, where they take action one after the other. Let the environment’s state be s , and a shared action space be $\mathcal{A}$. We assume that the state and the action spaces are fully observable. The action space contains high-level actions (e.g., in a cooking task, the action space can include boiling water, chopping vegetables, etc.). Both agents have access to a shared action history $\mathcal{H}^{0:t-1}$ up to (but not including) the current time step. We also keep track of the individual action histories of the human and the robot, denoted by $\mathcal{H}_h^{0:t-1}$ and $\mathcal{H}_r^{0:t-1}$. Further, the human expresses a set of preferences for the task in natural language ($\mathcal{L} = l_1 \dots l_n$) that can be open-ended and ambiguous, like *"I want a savory dinner meal."*

The robot seeks to reduce uncertainty about the human’s current goal so that it can take actions that support them. At each timestep, the robot infers $p(g)$, a probability distribution over the human’s possible goals $\mathcal{G}$, and at alternate timesteps, selects a robot action a_r . The set of all possible human goals $\mathcal{G}$ is potentially unbounded, so we propose for the robot to continuously estimate which goals are plausible given observations of the human’s actions and speech.

6.3.2 Receding Horizon Planning (RHP)

Following Sadigh et al. [230] and Chapter 4, the robot chooses actions using Receding Horizon Planning (RHP) [172]. Model Predictive Control (MPC) inspires RHP [85], but RHP applies to discrete decision-making rather than continuous control. At each timestep, the robot expands a search tree, where each node is a state s over a finite planning horizon. It evaluates possible sequences of human and robot actions conditioned on the robot’s current belief about the goal. Then, the robot executes the first action of the sequence that maximizes expected support for the human’s

goal. Finally, the horizon window shifts forward, the robot updates its belief over goals using the newly observed human action, and planning repeats. As explained later in Sec. 6.4, BALI expands this formulation to also account for verbal human preferences.

6.3.3 Attractor Fields

Per Jain and Argall [137] and as described in Chapter 5, *an attractor field expresses the extent to which one element in the first set exerts a pull toward certain elements in the second set*. Formally, let a target set Y have n elements (e.g., n actions). An attractor field for a source element x (e.g., a goal) is a vector:

$$F_x = [F_x(1), F_x(2), \dots, F_x(n)], \quad F_x(i) \geq 0,$$

where $F_x(i)$ indicates how strongly x "pulls" on the i -th element in Y . If multiple sources are present, their fields are summed element-wise. As a result, targets that are relevant to more sources accumulate higher attraction scores, while those relevant to fewer sources accumulate weaker scores. For example, in a collaborative cooking task, a goal such as "*fruit oatmeal*" exerts an attractor field over actions like *fetching oats*, *cutting fruit*, and *boiling water*. A high-level preference like "*healthy*" can also be modeled as an attractor field, pulling the robot toward ingredients such as *fruit* and away from high-sugar toppings like *chocolate chips*. Similarly, appliances can act as attractors: a *blender* exerts a strong pull toward goals like "*smoothie*" but little or no pull toward goals like "*salad*."

Attractor fields can be computed using known statistical mappings between the source and target elements (as shown in Chapter 5) or the world knowledge captured by large language models (LLMs). More specifically, given a source element (e.g., a goal like "*fruit parfait*" or a preference like "*sweet*"), we propose to use an LLM-as-

Algorithm 4 BALI Summary

Input: state s , action space $\mathcal{A}$, language pref. $\mathcal{L}$, horizon H , prior belief over goals $p(g)$, human history $\mathcal{H}_h$, robot history $\mathcal{H}_r$

Output: next robot action a_r , updated belief over goals $p(g)$

```
35 Observe human action; update  $\mathcal{H}_h$  Generate interaction summary Predict plausible human
    goals ( $g_{pred}$ ) and goal uncertainty  $p(g)$  if goal uncertainty is high and interruption cost is
    low then
36 |   Ask clarifying question; update  $p(g)$  and attractor fields  $\mathcal{L} = \mathcal{L} +$  human's answer to
    |   question
37 end if
38 Initialize RHP tree with root  $s$  for  $depth = 1$  to  $H$  do
39 |   foreach node at this depth do
40 |   |   Enumerate valid robot actions  $\mathcal{A}_{valid}$ 
    |   |   Score each  $a \in \mathcal{A}_{valid}$  using goal- and preference-based attractor fields
    |   |   Filter to most likely actions based on scores
    |   |   Expand node with filtered actions
41 |   end foreach
42 end for
43 Evaluate branch costs; select minimum-cost branch; execute its first action  $a_r$ ; update belief
    over goals  $p(g)$  Return: next robot action ( $a_r$ ), updated belief over goals ( $p(g)$ )
```

a-judge [104] to score the relevance of possible targets (e.g., assigning higher scores to “cut fruit” than to “fry bacon”, or to “blueberries” rather than “lettuce”). We use these scores to populate the attractor field vector, allowing the robot to construct attractor fields even when the source is open-ended, ambiguous, or novel.

6.4 Bidirectional Action-Language Inference (BALI) for Goal Prediction

Our approach, BALI, is designed for turn-based human-robot collaborations (as explained in Sec. 6.3) and is summarized in Algorithm 4. At each time step, the robot observes the human’s latest action and updates its belief over possible goals using attractor fields that link actions to preferences expressed in natural language and plausible goals. If uncertainty is high and the cost of interruption is low, the robot asks a clarifying question, incorporating the human’s response to refine its beliefs

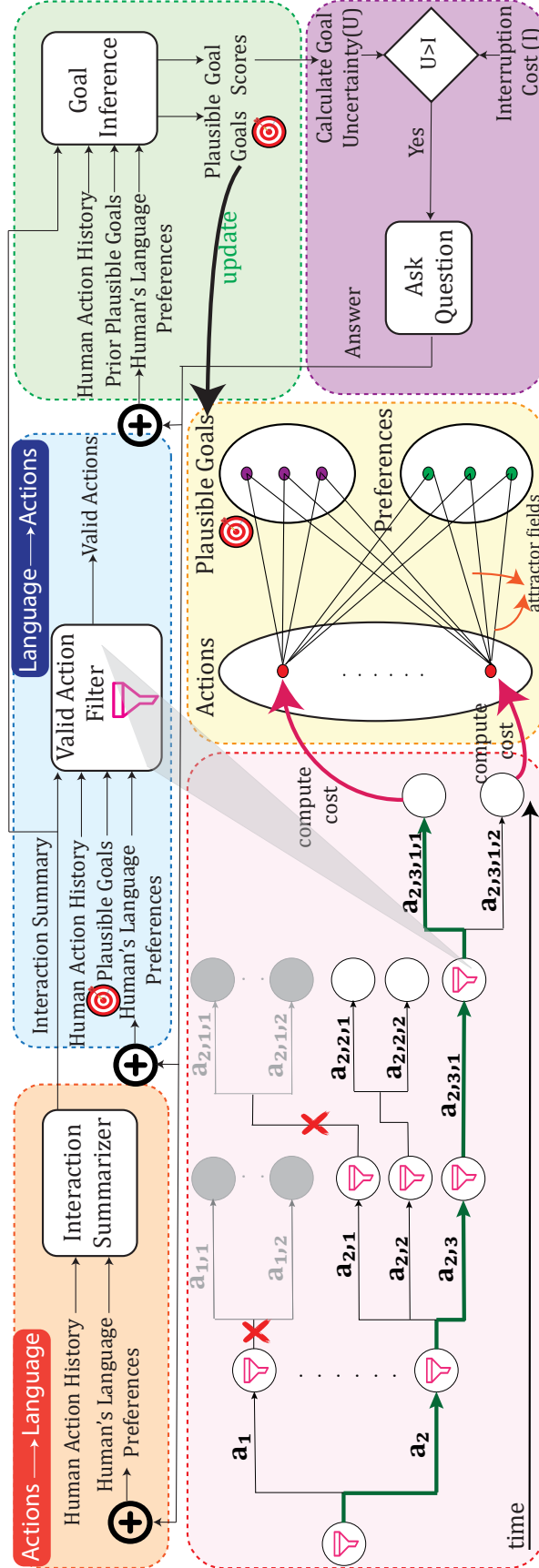


Figure 6.2: BALI for goal prediction: Human preferences and actions are summarized (orange) and used for Goal Inference (green) to update an estimate of plausible goals. The Ask Question module (purple) triggers clarifications when goal uncertainty exceeds a threshold. The planner (pink) expands action sequences under receding-horizon planning: at each non-leaf node, the Valid Action Filter (blue) prunes infeasible actions (shown greyed out). The cost function module (yellow) computes an attractor-field cost that links actions to goals and guides the search toward trajectories aligned with plausible human goals and preferences.

(lines 4-7). The robot then performs receding horizon planning (lines 9-16): it expands a search tree of alternating human and robot actions up to a fixed horizon (H), where at each node valid robot actions are enumerated, scored with respect to goals and preferences, filtered to retain the most likely, and expanded. The resulting tree is evaluated, and the branch with the minimum cost is identified. The robot executes the first action along this branch as its next contribution to the task (line 17). The rest of the section explains each key module in BALI in more detail.

6.4.1 Interaction Summary

We maintain a concise interaction summary that condenses a human’s preferences and actions into a single natural-language sentence, inspired by the concept of Dialog State Tracking (DST) in conversational systems [135] as illustrated in Figure 6.3. For example, in a cooking task, the summary describes the likely dish, the current phase (e.g., gathering, assembling, finishing), the items in play, and sensible next steps (orange box in Fig. 6.2). We encode the history in natural language because our goal inference leverages large language models, which inherently reason over text-based representations. The summary provides the planner with a consistent reference across timesteps, preventing the robot from overreacting to isolated ambiguous actions, enforcing coherence as the interaction unfolds, and preserving long-range context. Moreover, this module helps the robot use human actions to disambiguate and ground high-level instructions.

6.4.2 Goal Inference

The Goal Inference module (green box in Fig. 6.2) enables a robot to infer goals by analyzing the action history and updating a ranked list of candidate goals (g_{pred}) as new actions are taken as illustrated in Figure 6.4. Conditioned on the human’s preferences, action history, the interaction summary, and environmental constraints,

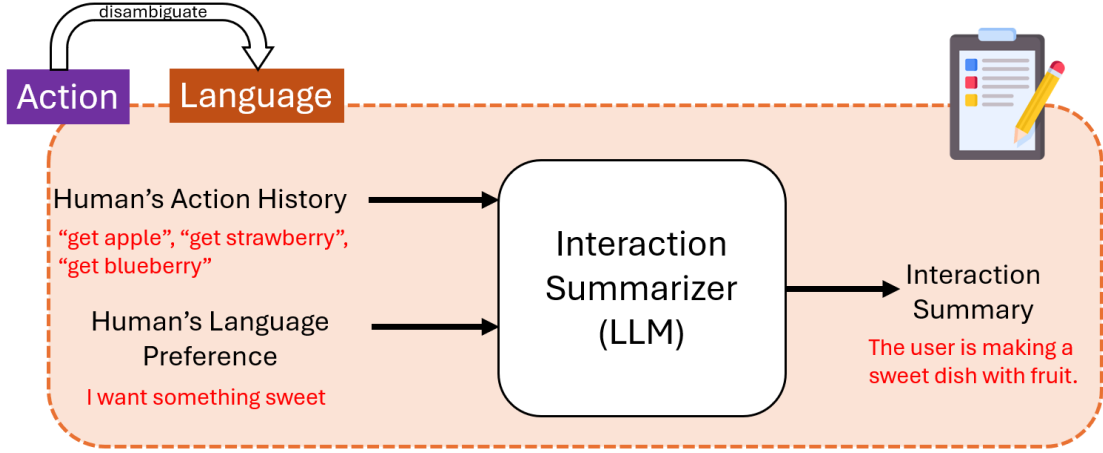


Figure 6.3: Interaction Summary module with an Example

the robot queries an LLM that leverages commonsense knowledge to generate and continually update a set of plausible goals, inspired by Ma et al. [171]. Each goal is also assigned a score, using LLM-as-a-judge, that indicates how aligned the goal is with the human’s action history and preferences. The system handles removing and adding goals by continuously re-evaluating the candidate set as new human actions and preferences are observed. When new actions or preferences indicate that specific goals are no longer supported, they are removed. Conversely, they are added if the observed behavior support new feasible goals.

6.4.3 Action Selection and Cost Function

BALI chooses robot actions within the receding horizon planning (RHP) framework described in Sec. 6.3 (pink box in Fig. 6.2). Action selection proceeds by expanding a search tree of possible future action sequences, alternating between human and robot actions, up to a fixed planning horizon H . At each node, a valid action filter (blue box in Fig. 6.2 and Figure 6.5) uses the interaction summary (Sec. 6.4.1), goal predictions (Sec. 6.4.2), and action history to select actions consistent with the current goals and preferences ($\mathcal{A}_{valid}$) using an LLM. The filtering repeats at each level of the RHP

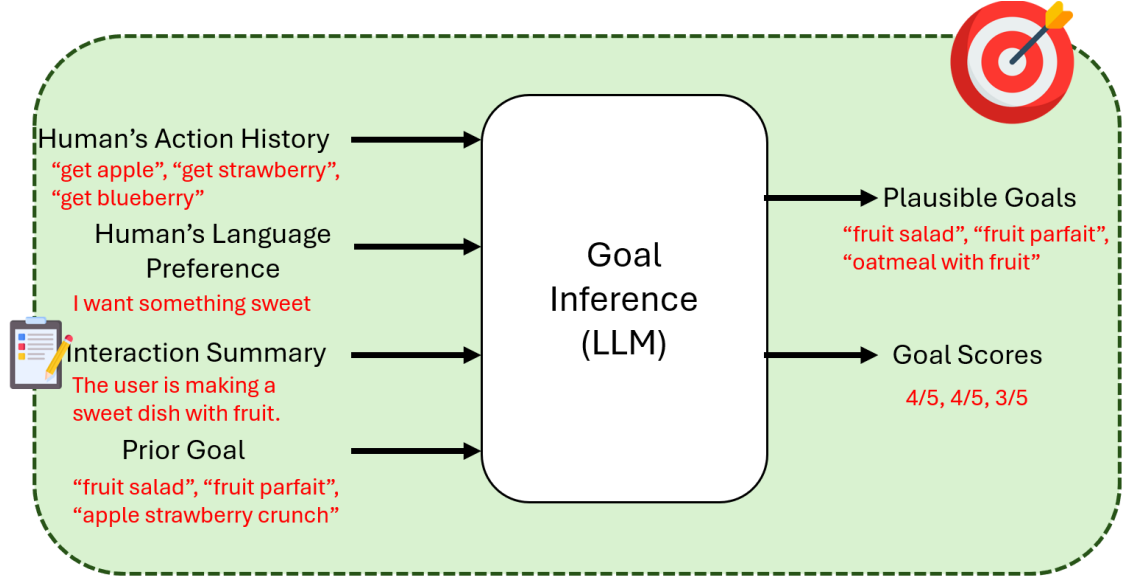


Figure 6.4: Goal Inference module with an Example

tree until depth H , generating candidate action sequences. After the sequences are generated, attractor fields are computed per Eq. 6.3.3 (yellow box in Fig. 6.2 and as illustrated in Figure 6.6), producing relevance scores for each candidate action with respect to every plausible goal ($\forall g \in g_{\text{pred}}$) and preference ($\forall l \in \mathcal{L}$).

The robot aggregates the attraction scores from all plausible goals and preferences to assign a single scalar value to each candidate action along every branch of the RHP tree. Formally, for each $a \in \mathcal{A}_{\text{valid}}$, the aggregate attractor score is:

$$S(a) = \sum_{g \in g_{\text{pred}}} w_g F_g(a) + \sum_{l \in \mathcal{L}} w_l F_l(a),$$

where $F_g(a)$ and $F_l(a)$ are the attractor field scores for goals (g) and language preferences (l), and w_g , w_l are tunable weights reflecting the importance of attractor fields for g and l , respectively over a .

The cost of a branch in the RHP tree, which encodes an action sequence $\mathbf{a} =$

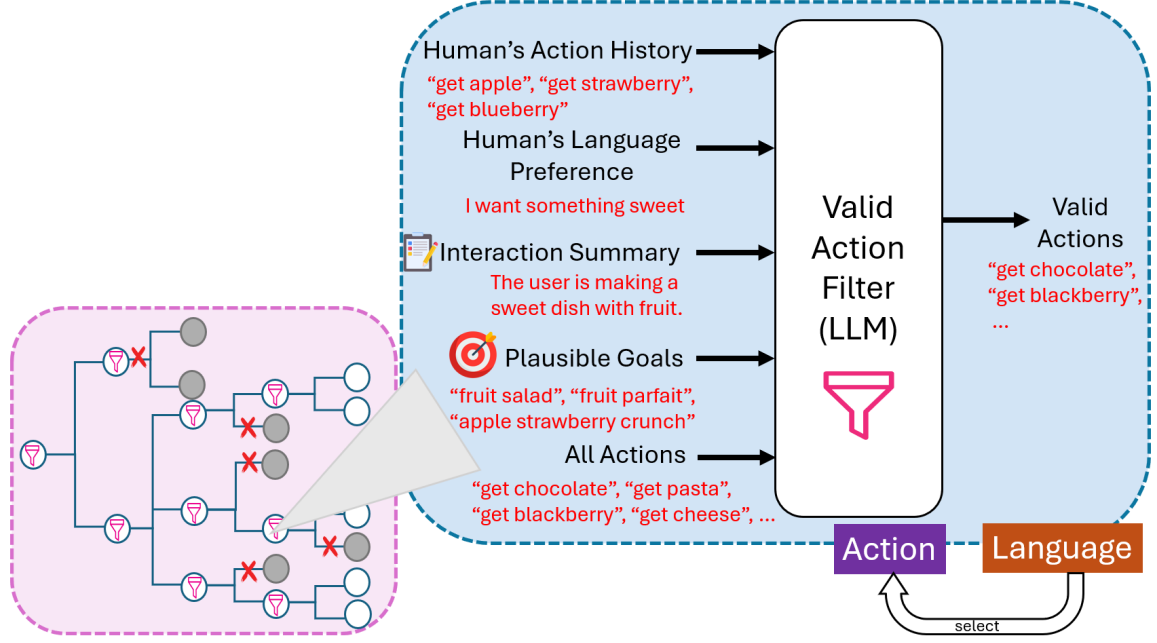


Figure 6.5: Valid Action Filter module with an Example

$(a_1, \dots, a_H)$ for the human and robot, is

$$C(\mathbf{a}) = -\sum_{t=1}^H S(a_t)$$

with S as in eq. (6.4.3). The robot identifies the branch with minimum cost (the green line in the pink box of Fig. 6.2), and executes the first action (a_r) along that branch (a_2 in Fig. 6.2).

6.4.4 Asking Informative Questions

The robot must decide *when* to ask a question, *what* question to ask and *how* to use the answer to reduce uncertainty over goals, as shown in the purple box in Fig. 6.2.

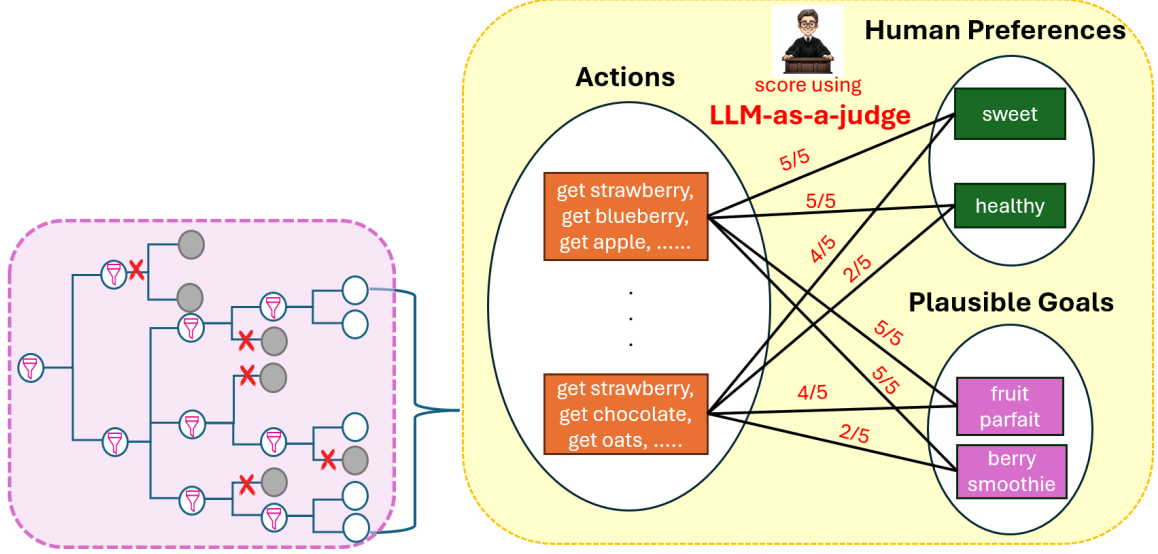


Figure 6.6: Attractor Field Cost Function module with an Example

When to Ask

The robot asks a question only if the expected benefit (information gained about the human’s goal) outweighs the cost of interrupting. Inspired by [271], we model the cost as time-dependent to discourage repeated interruptions: immediately after asking, the interruption cost is high and then decreases linearly toward a minimum. Let Δt be the time since the last question, T_q the interval after which the minimum cost is reached, and $\Delta C = C_{\max} - C_{\min}$ the range of costs. The cost of asking at time t is:

$$C_q(t) = \begin{cases} C_{\min}, & \Delta t \geq T_q, \\ C_{\max} - \Delta C \cdot (\Delta t / T_q), & \Delta t < T_q, \end{cases}$$

The robot compares this interruption cost to its current uncertainty or entropy over goals $p(g)$: $H(g) = -\sum_g p(g) \log p(g)$. Since the value of resolving uncertainty increases with the number of plausible goals (n_{goals}), we scale the interruption cost by

$\log(n_{\text{goals}})$: $C_q^{\text{scaled}} = C_q(t) \log(n_{\text{goals}})$. A question is asked whenever $H(g) > C_q^{\text{scaled}}$, i.e., when the robot’s uncertainty is high enough that reducing it is worth the interruption.

What to Ask

Once the robot decides to ask, it must choose the most informative question. Candidate questions are drawn from categories relevant to the task (e.g., in cooking: high-level food preferences, ingredients, or appliances). These categories can be predefined or dynamically generated by an Large-Language Model (LLM) based on the domain. Each candidate question is then hypothesized to have several possible answers from the human.

The usefulness of a question is measured by how much it reduces uncertainty about the human’s goal. We quantify uncertainty by the entropy of the current belief distribution $p(g)$ over goals:

$$H = - \sum_g p(g) \log p(g).$$

Inspired by [31, 80, 217], when the robot asks a question, each possible answer (ans) changes its belief over goals, yielding $p(g \mid ans)$ whose entropy reflects how uncertain the robot would be if that answer were given. We compute this posterior probability $p(g \mid ans)$ using Bayes’ rule:

$$p(g \mid ans) = \frac{p(ans \mid g)p(g)}{p(ans)}.$$

The likelihood of each answer given a goal, $p(ans \mid g)$, is approximated using attractor fields, which encode how strongly each answer is associated with the goal. For example, if the question is “*Should the dish be warm or chilled?*”, then goals like “*oatmeal*” have high attraction to the “*warm*” answer and low attraction to the

“*chilled*” answer, while “*parfait*” has the reverse pattern. $p(g)$ denotes the probability distribution over possible human goals estimated by the Goal Inference module (Sec. 6.4.2). For simplicity, we assume that each possible human answer is equally likely, i.e., $p(ans)$ is uniform. Therefore, the expected uncertainty after asking is the weighted average entropy for all possible answers:

$$\mathbb{E}[H \mid ans] = \sum_{ans} p(ans) H(p(g \mid ans)),$$

and the question’s value is its expected reduction in entropy:

$$\Delta H = H - \mathbb{E}[H \mid ans].$$

Intuitively, a good question partitions the plausible goals into groups depending on the answer, so that the robot’s uncertainty decreases regardless of how the human responds. For instance, like the earlier example, asking “*Should the dish be warm or chilled?*” separates “*oatmeal*” from “*parfait*” and removes ambiguity.

Processing Human Answers

Human answers are incorporated into BALI in three ways (see the $\oplus$ symbol in Fig. 6.2): (i) the Interaction Summary module (Sec. 6.4.1) integrates the answers into the natural-language summary of preferences and actions, (ii) the Goal Inference module (Sec. 6.4.2) uses the answers as additional evidence when predicting the human’s goal, and (iii) the Action Selection module (Sec. 6.4.3) conditions the RHP tree expansion on the updated goals, producing actions that align with new preferences.

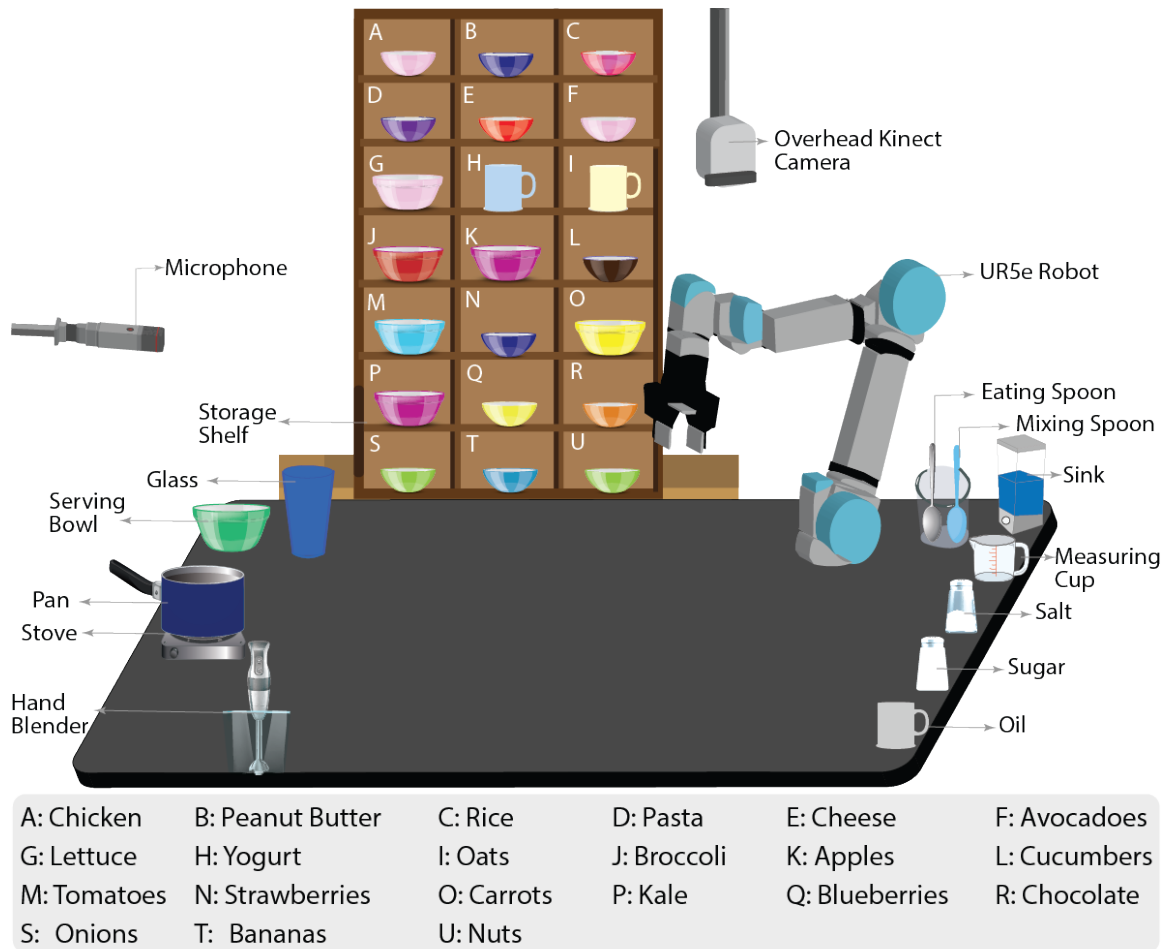


Figure 6.7: Experimental Setup for the Collaborative Cooking Task

6.5 Evaluation

6.5.1 Task and Problem Representation

We evaluate BALI on collaborative cooking, a testbed widely used in human–robot collaboration [38, 45, 277, 99] similar to Chapters 4 and 5. Cooking provides concrete ground truth for controlled experiments while also reflecting the subjective and often unique preferences that make goals both explicitly nameable and inherently emergent. Our experiments span both a simulation and a real-world setup (as shown in Fig. 6.7), which is expanded from Chapters 4 and 5 to include more ingredients for improved variability. The human can prepare any meal using the available ingredients and

appliances. We assume either partner may perform any preparatory step, with each step taking the same time regardless of who executes it.

The robot’s state and action space are defined in the Planning Domain Definition Language (PDDL) [4], with environment updates handled by PDDLgym [243]. Both agents can perform actions $A = \{\text{gather}(i,a), \text{pour}(i,d,a), \text{mix}(i,a), \text{cook}(i,a), \text{turn_on}(i,a), \text{collect_water}(a), \text{blend}(i,a), \text{reduce_heat}(i,a), \text{serve}(i,a)\}$ where i is an item, d a container (e.g., a bowl), and $a \in \{\text{human}, \text{robot}\}$ denotes the acting agent. The state is represented as a vector of PDDL literals describing item locations and states (simmering, blended, cooked). We assume a deterministic, fully observable environment. For simplicity, the human initiates the interaction by taking the first action and providing high-level meal preferences (e.g., “*I want something sweet and healthy*”).

6.5.2 Experimental Setup

We primarily assess BALI in an open-ended goal inference setting, which we refer to as the ***open case*** going forward. In this setting, we conduct experiments both in simulation and in the real-world. Additionally, for completeness, we investigate performance in simulation in a ***closed case*** where the goal set is known. This closed setting includes additional baselines not possible in the open case.

Simulation

To evaluate our approach in a controlled setting, the human prepares one of 30 possible goal recipes similar to Chapters 4 and 5, where each recipe could be one of six types: *Pasta*, *Stew*, *Salad*, *Oatmeal*, *Smoothie*, or *Parfait*, in both the open and closed cases. The human’s ground-truth action sequences are derived from Clique/Chain Hierarchical Task Networks (CC-HTNs) [114], capturing ordered steps (e.g., add strawberry before turning on the blender) and steps that can be performed unordered (e.g.,

retrieve stew toppings in any order). To construct the simulated preferences, we used GPT-o3 to propose 50 diverse food preferences given the available ingredients, appliances, and exemplar recipes (the 30 goal recipes the simulated human could be preparing). For example, the preferences include “*sweet*”, “*savory*”, “*warm*”, “*breakfast*”, etc. We manually mapped each of the 30 recipes to all applicable preferences. Next, we generated 967 preference pairs by taking all unordered pairs of preferences that shared at least one goal. For example, the overlap of “*sweet*” and “*breakfast*” includes “*fruit parfait*”. From each pair, we uniformly sampled one overlapping goal to execute, resulting in 967 experiments. The length of simulated human recipes ranges from 9 to 31 steps, with an average length of 22.7 ± 5.5 steps.

Real World

Similar to Fig. 5.1, our real-world setup comprises a UR-5e, an overhead Microsoft Azure Kinect [246] for workspace monitoring, and a microphone as shown in Fig. 6.7. To avoid online action recognition, humans were instructed to verbalize their actions. A custom lightweight parser used fuzzy matching to map utterances to PDDL literals. For safety, we used realistic mock ingredients (liquids replaced with colored beads) and modified appliances to actuate without heat or blades. A blender and a stove with a pan were placed in front of the robot. A sink, utensils, and condiments were to the left, serving vessels were to the right, and a storage shelf with ingredients was behind. For perception, we used SEEM [317], a promptable segmentation model, to localize and segment ingredients in different colored containers from text prompts. Then, we applied a heuristic grasp-point selector on the resulting mask to enable manipulation.

Implementation Details

When comparing approaches that rely on LLM models, we use a *larger* general-purpose LLM (Gemini 2.5 Flash [63]) and a *smaller* instruction-tuned model (Phi-4-mini-instruct [2]), with identical prompts and decoding settings. Furthermore, when our method asks questions, it uses the clarifying question module introduced in Sec. 6.4.4. In the open case, we use the FlowJudge [81] model as the LLM-as-a-judge for attractor field scoring. In the closed case, we leverage information about the goals as explained further in the next section.

6.5.3 Baselines

We consider five baselines in our evaluation. Two baselines (LLM Only, and GOOD) are completely different methods to BALI. The other three have some similar components to BALI, but they either use less information from the human, or leverage access to a goal bank (which is generally not available to BALI in the open case).

LLM Only

This baseline examines a purely language-driven approach: Can a large language model, given a user’s stated preferences and the valid actions at each step, select sensible next actions for the robot without explicit planning or an internal belief over goals? At each timestep, the system lists the valid robot actions from the current state and prompts an LLM with the user’s active preferences (and brief interaction context) plus the valid action list. The model is instructed to return exactly one action from that list that best supports the human’s preferences, leveraging the model’s general cooking knowledge. In the *open case*, the prompt contains only preferences and the action list. In the *closed case*, it additionally includes the set of possible goals (the 30 recipes) so the model can use this information to guide its action choice. Decisions are made one step at a time, without considering possible futures.

Goal Inference from Open-Ended Dialog (GOOD)

Goal Inference from Open-Ended Dialog (GOOD) [171] maintains a posterior over open-ended, natural-language goals that is updated from dialog alone. At each turn, it prompts an LLM to role-play the user under each candidate goal, uses the resulting likelihoods to perform Bayesian updates, and adds/prunes goal hypotheses as the conversation evolves. In our setup, we run this dialog→inference→goal-management loop and map the current most probable goal to a next action from the valid action set. In the *open case*, GOOD maintains a posterior over unbounded natural-language goals. In the *closed case*, the posterior is restricted to the known recipes. Unlike our method, GOOD neither looks ahead to predict future action sequences nor conditions action selection on past human actions.

Actions Only

This baseline aims to evaluate goal inference based on the human’s action history, ignoring natural-language preferences. In the *closed case*, we implement this baseline using Critical Decision Points (CDP) [92], as this approach uses RHP and infers a human’s long-horizon goals considering past human actions. The CDP approach identifies states where candidate goals diverge most in their next-step predictions, and chooses the robot action that steers the interaction toward such informative states. Because CDP relies on a goal-driven policy bank, in the *open case* we expand an RHP tree, but use a language model to score admissible next actions based on perplexity [12] given the current interaction sequence, treating lower perplexity as more likely. In this context, perplexity measures how surprising the model finds future actions in the sequence; actions that yield lower perplexity are considered more consistent with the interaction so far. We keep the top-K actions, look a few steps ahead with the model to score candidate paths, and execute the first action from the best-scoring path.

Known Goals (BALI-KG):

This baseline is designed only for the *closed case*. We apply our method but restrict the Goal Management module to a fixed goal bank $\mathcal{G}$ of known recipes. All other components—interaction summarization (Sec. 6.4.1), RHP tree expansion, and attractor-field cost computation (Sec. 6.4.3) remain unchanged. This isolates the effect of assuming access to the complete goal set and removes the need to propose new goals online.

Known Goals and Policy Bank (BALI - KG+PB):

This baseline is designed for the *closed case* only, like BALI-KG, but assumes the robot has access to both a complete goal bank $\mathcal{G}$ and a policy bank $\mathcal{P}_g$ containing many ways of achieving each goal. For example, if $\mathcal{G}$ includes the recipe *oatmeal*, $\mathcal{P}_g$ may contain permutations of the same preparation steps, such as *get bowl* $\rightarrow$ *pour oats* $\rightarrow$ *add milk* $\rightarrow$ *stir*, *get bowl* $\rightarrow$ *add milk* $\rightarrow$ *pour oats* $\rightarrow$ *stir*, etc. From $\mathcal{P}_g$, we derive mappings between goals to actions, and preferences to actions. These mappings support the construction of precomputed attractor fields: for each goal or preference (source), attraction scores are assigned to actions (targets) according to their frequency in associated policies. Actions consistently appearing in policies for a source receive high attraction; otherwise, they have low attraction.

Goal inference in BALI - KG+PB is handled by a Random Forest trained on 300,000 action sequences (10,000 per recipe across 30 recipes in our specific simulation setup), which achieved 86.8% accuracy on a 20% held-out test set. During interaction, the model maps partial human action sequences to candidate goals in $\mathcal{G}$ and updates the ranking as new actions are observed. Action selection is further constrained at each node of the RHP tree by n-gram co-occurrence statistics and policy frequencies from $\mathcal{P}_g$, with candidates refined using the same LLM prompting strategy as in our method but with their costs determined entirely by precomputed attractor fields.

6.5.4 Metrics

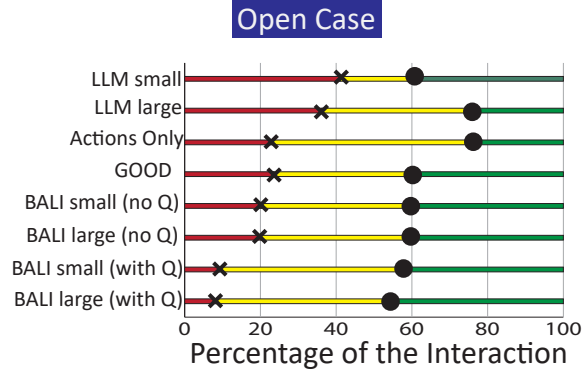
We compare BALI against each baseline using five metrics:

1. **First Correct Guess:** The first timestep in an interaction when the robot correctly identifies the human’s goal.
2. **Last Incorrect Guess:** The final timestep when the robot’s predicted goal is wrong, indicating when its belief stabilizes on the correct goal. This complements First Correct Guess, which may be noisy since a method can predict correctly by chance early on without resolving uncertainty.
3. **Goal Prediction Accuracy:** Percentage of timesteps when the robot correctly identifies the human’s goal. We report both the top-1 and top-3 accuracy.
4. **Number of Questions Asked:** Total clarifying questions the robot asks in an interaction.
5. **Number of Extra Steps:** Number of extra steps taken by both agents to complete the interaction. In simulation, the metric measures extra steps in comparison to the number of ground-truth steps from the corresponding CC-HTN. Because the simulated humans make no mistakes, all extra steps are attributed to robots. In the real-world evaluation, we ask the human to flag any steps they think did not belong in the recipe they were trying to make.

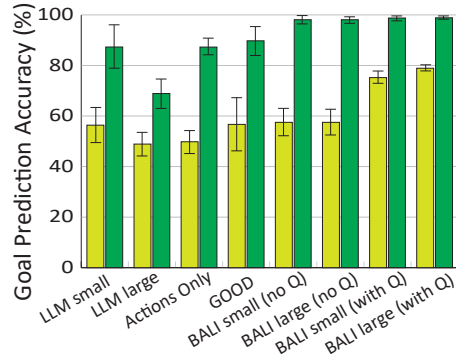
6.6 Results in Simulation

6.6.1 Open Case

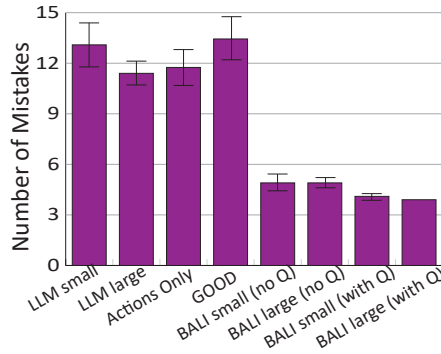
Fig. 6.8a–6.8c shows the results in simulation when the robot does not know the set of possible human goals.



a)



b)



c)

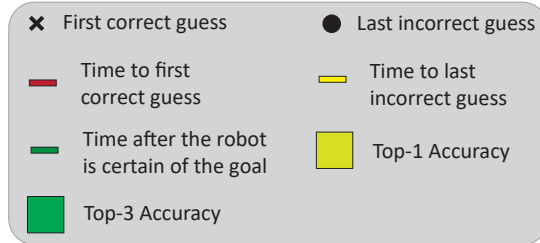


Figure 6.8: Results for the open case. (a) shows inference timing: maroon = time until first correct guess, yellow = period of instability until the last incorrect guess, green = stable correct phase. cross = first correct guess ($\downarrow$ =better), circle = last incorrect guess ($\downarrow$ =better). (b) reports Top-1 (yellow) and Top-3 (green) goal prediction accuracy ($\uparrow$ =better). (c) show average mistakes or extra steps ($\downarrow$ =better). Means and standard deviations of metrics are computed over 967 preference combinations. Visualization of standard deviations for the first correct guess and the last incorrect guess is skipped for clarity.

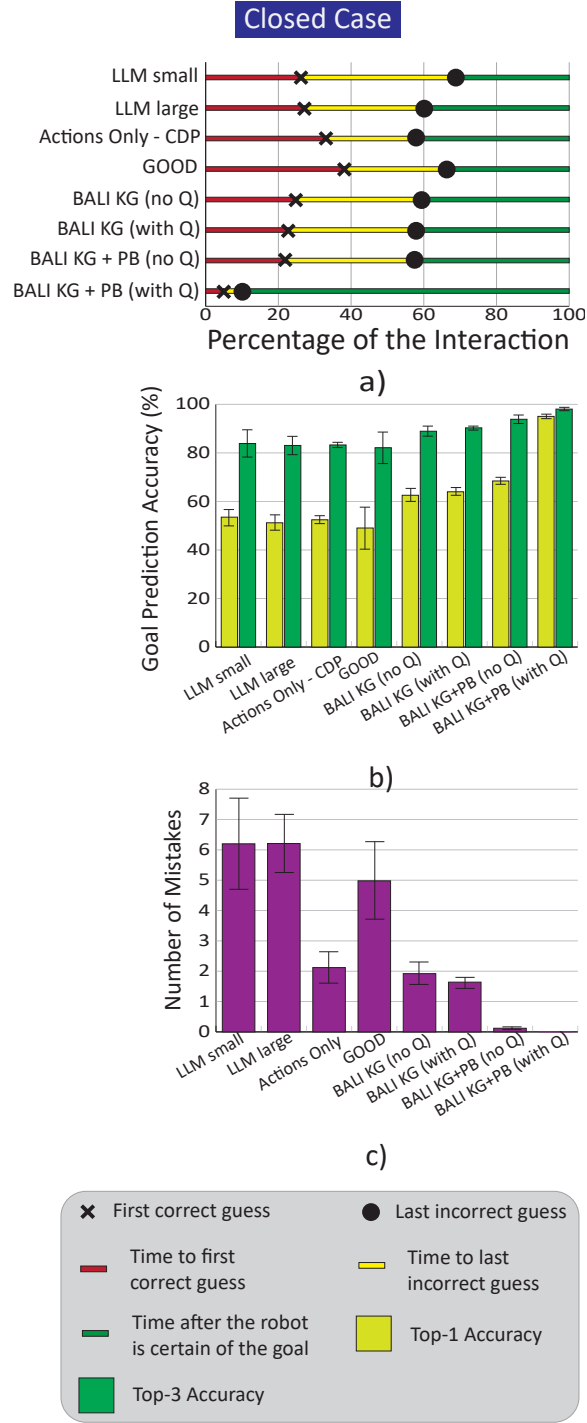


Figure 6.9: Results for the closed case. (a) shows inference timing: maroon = time until first correct guess, yellow = period of instability until the last incorrect guess, green = stable correct phase. cross = first correct guess ($\downarrow$ =better), circle = last incorrect guess ($\downarrow$ =better). (b) reports Top-1 (yellow) and Top-3 (green) goal prediction accuracy ($\uparrow$ =better). (c) show average mistakes or extra steps ($\downarrow$ =better). Means and standard deviations of metrics are computed over 967 preference combinations. Visualization of standard deviations for the first correct guess and the last incorrect guess is skipped for clarity.

Baselines: Pure small LLMs made on average 13.09 mistakes per interaction, with 56.38% top-1 and 87.30% top-3 goal prediction accuracy. Their first correct guess occurred at 41% of the interaction, and the last incorrect guess at 59%, showing that predictions remained unstable well into the task. The larger LLM reduced mistakes slightly to 11.40, but top-1 accuracy dropped to 45.40% and top-3 to 74.87%. Their first correct guess occurred at 35% of the interaction, and the last incorrect guess at 69%. Our action-only baseline made 11.75 mistakes, with 49.82% top-1 and 87.27% top-3 accuracy, producing its first correct guess at 22% and last incorrect guess at 76%. GOOD [171], which adds Bayesian reasoning over language-defined goals, achieved 56.69% top-1 and 89.75% top-3 accuracy, but still made 13.44 mistakes, with the first correct guess at 23% and the last incorrect guess not until 60%.

BALI: Without asking questions, BALI reduced mistakes to 4.90, with 57.50% top-1 and 98.09% top-3 accuracy. The system produced its first correct guess earlier than baselines, at 20% of the interaction, and its last incorrect guess by 60%. With questions, performance improved further: mistakes dropped substantially to 4.10 (small model) and 3.12 (large), top-1 accuracy rose to 75.20% and 78.90%, and top-3 accuracy reached 98.77% and 98.85%. The first correct guess occurred within 10% of the interaction, and the last incorrect guess within 6–7%. On average, the robot asked 1.94 ± 0.92 questions per interaction. The LLM within BALI further improved goal prediction accuracy and reduced mistakes relative to the small model, with the gains especially pronounced in this open setting where generalization is critical.

We conducted paired t-tests across all 967 trials, confirming that all performance gaps between BALI and the baselines are statistically significant as shown in Table 6.1.

6.6.2 Closed Case

Fig. 6.9a–6.9c show the results when the robot accessed the goal bank.

Table 6.1: Paired t-tests comparing all BALI variants to all baselines across five metrics ($N = 967$). Lower is better for Mistakes, First Correct Guess, and Last Incorrect Guess; higher is better for Top-1 and Top-3 Accuracy.

BALI Variant	Baseline	Mistakes			Top-1 Acc.			Top-3 Acc.			First Correct			Last Incorrect		
		t	p	d	t	p	d	t	p	d	t	p	d	t	p	d
BALI-small-noQ	LLM-small	-21.73	$< 10^{-6}$	-1.88	0.70	0.482	0.07	29.94	$< 10^{-6}$	3.01	-13.07	$< 10^{-6}$	-1.31	0.12	0.905	0.01
	LLM-large	-17.25	$< 10^{-6}$	-1.49	7.59	$< 10^{-6}$	0.76	64.33	$< 10^{-6}$	6.46	-9.39	$< 10^{-6}$	-0.94	-6.10	$< 10^{-6}$	-0.61
	Actions-Only	-18.19	$< 10^{-6}$	-1.57	4.82	0.000002	0.48	30.03	$< 10^{-6}$	3.02	-1.60	0.110	-0.16	-10.61	$< 10^{-6}$	-1.07
	GOOD	-22.67	$< 10^{-6}$	-1.95	0.51	0.613	0.05	23.41	$< 10^{-6}$	2.35	-2.06	0.039	-0.21	-0.10	0.920	-0.01
BALI-small-Q	LLM-small	-23.87	$< 10^{-6}$	-2.06	11.80	$< 10^{-6}$	1.19	32.61	$< 10^{-6}$	3.28	-19.47	$< 10^{-6}$	-1.96	-33.47	$< 10^{-6}$	-3.36
	LLM-large	-19.38	$< 10^{-6}$	-1.67	18.68	$< 10^{-6}$	1.88	66.91	$< 10^{-6}$	6.72	-15.79	$< 10^{-6}$	-1.59	-39.72	$< 10^{-6}$	-3.99
	Actions-Only	-20.32	$< 10^{-6}$	-1.75	15.91	$< 10^{-6}$	1.60	32.70	$< 10^{-6}$	3.29	-7.96	$< 10^{-6}$	-0.80	-44.25	$< 10^{-6}$	-4.44
	GOOD	-24.81	$< 10^{-6}$	-2.14	11.60	$< 10^{-6}$	1.17	26.01	$< 10^{-6}$	2.61	-8.43	$< 10^{-6}$	-0.85	-33.69	$< 10^{-6}$	-3.38
BALI-large-noQ	LLM-small	-25.46	$< 10^{-6}$	-2.20	13.25	$< 10^{-6}$	1.33	32.73	$< 10^{-6}$	3.29	-19.85	$< 10^{-6}$	-2.00	-33.03	$< 10^{-6}$	-3.32
	LLM-large	-20.97	$< 10^{-6}$	-1.81	20.09	$< 10^{-6}$	2.02	67.00	$< 10^{-6}$	6.73	-16.17	$< 10^{-6}$	-1.62	-39.28	$< 10^{-6}$	-3.94
	Actions-Only	-21.91	$< 10^{-6}$	-1.89	17.33	$< 10^{-6}$	1.74	32.83	$< 10^{-6}$	3.30	-8.34	$< 10^{-6}$	-0.84	-43.81	$< 10^{-6}$	-4.40
	GOOD	-26.40	$< 10^{-6}$	-2.27	13.05	$< 10^{-6}$	1.31	26.13	$< 10^{-6}$	2.63	-8.81	$< 10^{-6}$	-0.88	-33.25	$< 10^{-6}$	-3.34
BALI-large-Q	LLM-small	-26.47	$< 10^{-6}$	-2.28	14.13	$< 10^{-6}$	1.42	32.93	$< 10^{-6}$	3.31	-20.31	$< 10^{-6}$	-2.04	-33.28	$< 10^{-6}$	-3.34
	LLM-large	-21.98	$< 10^{-6}$	-1.90	20.97	$< 10^{-6}$	2.11	67.20	$< 10^{-6}$	6.75	-16.62	$< 10^{-6}$	-1.67	-39.53	$< 10^{-6}$	-3.97
	Actions-Only	-22.92	$< 10^{-6}$	-1.98	18.21	$< 10^{-6}$	1.83	33.03	$< 10^{-6}$	3.32	-8.78	$< 10^{-6}$	-0.88	-44.06	$< 10^{-6}$	-4.43
	GOOD	-27.41	$< 10^{-6}$	-2.36	13.93	$< 10^{-6}$	1.40	26.33	$< 10^{-6}$	2.65	-9.26	$< 10^{-6}$	-0.93	-33.50	$< 10^{-6}$	-3.36

 Human Action
  Robot Action
  Mistake

 Question
  Answer

Initial Preference: I want something **quick**

Human's Actual Recipe: Smoothie

Action Sequence	Top 2 Predicted Goals
 1. Get Strawberry	
 2. Get Banana	Fruit Salad, Fruit Smoothie
 3. Get Yogurt	
 4. Pour Yogurt into Blender	Fruit Smoothie, Fruit Yogurt Parfait
 5. Get Apple	
 6. Pour Banana into Blender	Fruit Smoothie, Berry Smoothie
 7. Pour Strawberry into Blender	
 8. Pour Apple into Blender	Fruit Smoothie, Berry Smoothie
 9. Blend Smoothie	
 10. Serve Smoothie	Fruit Smoothie, Berry Smoothie

a)

Initial Preference: Can we make something **healthy**?

Human's Actual Recipe: Veggie Salad

Action Sequence	Top 2 Predicted Goals
 1. Get Salt	
 2. Get Onion	Healthy soup, Salad
 3. Get Carrot	
 4. Get Cucumber	Carrot soup, Vegetable Stew
 5. Get Lettuce	
 6. Get Bowl	Vegetable Stew, Salad
 7. Get Eating Spoon	
 8. Pour Cucumber into Bowl	Salad, Salad with Chicken
 9. Pour Broccoli into Bowl	
 10. Pour Lettuce into Bowl	Salad, Broccoli Chicken Salad
 11. Pour Salt into Bowl	
 12. Pour Oil into Bowl	Salad, Vegetable Salad with Chicken
 13. Pour Onion into Bowl	
 14. Pour Carrot into Bowl	Salad, Vegetable Salad with Chicken
 15. Get Tomato	
 16. Pour Tomato into Bowl	Salad, Vegetable Salad with Chicken
 17. Get Kale	
 18. Pour Kale into Bowl	Salad, Vegetable Salad with Chicken
 19. Serve Salad	

Q 1 Would you use the **pan** or the **bowl** to assemble the dish?

A 1 Bowl

b)

Figure 6.10: Real World Case Study

Baselines: Similar to the open case, we see the same patterns. Pure small LLMs made 6.20 mistakes, with 53.52% top-1 and 83.87% top-3 goal prediction accuracy. Their first correct guess came at 26% of the interaction and the last incorrect at 62%, showing late convergence. Large LLMs reduced mistakes slightly to 6.21, but accuracy dropped slightly to 51.20% top-1 and 83.06% top-3, with first correct at 26% and last incorrect at 62%. GOOD [171] reached 49.05% top-1 and 82.14% top-3 with 4.98 mistakes, but converged late (first correct 38%, last incorrect 66%). Our actions-only baseline, CDP [92] performed better than the other baselines, with 2.12 mistakes, but still performed worse than BALI.

BALI: BALI-KG, the same method as in the open case but evaluated with a fixed goal set, made 1.92 mistakes, with 62.54% top-1 and 88.92% top-3 accuracy. The first correct guess occurred at 25% and the last incorrect at 59%. With questions, mistakes dropped to 1.64, with 63.98% top-1 and 90.32% top-3 goal prediction accuracy, first correct at 23% and last incorrect at 41%. Adding the policy bank (BALI-KG+PB) yielded the strongest results, as expected: mistakes nearly vanished (0.12), with 68.43% top-1 and 93.85% top-3 accuracy, first correct at 22% and last incorrect at 58%. Mistakes disappeared entirely when the robot was allowed to ask questions; top-1 accuracy reached 95.04%, and top-3 reached 98.04%. The first correct guess occurred at 5% and the last incorrect at 9%. The system asked 1.23 ± 0.56 questions per interaction. This shows that the question-asking module partitions the goal space effectively, so answers remove ambiguity about goals well. The large model was not tested here since the small one had already achieved near-perfect performance.

6.7 Real World Demonstration

We analyzed BALI in 2 proof-of-concept real-world trials, involving participants from the research group who were not involved in the work. The robot prompted par-

ticipants to take an action to start the collaboration and provide a food preference in natural language, allowing them to add more preferences at any time. When uncertain, it asked clarifying questions to adapt based on feedback. At the end of the interaction, participants reported their goal recipe and annotated robot actions that did not align with their goal.

Fig. 6.10 shows the two case studies. In the first case (Fig. 6.10a), the participant asked for “*something quick*” and began preparing a smoothie. Based on this preference, the robot predicted fruit-based recipes as likely goals. It started processing ingredients early in the interaction by blending instead of continuing to fetch additional items, which the participant later noted. In the second case (Fig. 6.10b), the participant asked for “*something healthy*” and began preparing a veggie salad. The vegetables fetched were consistent with multiple possible goals, so the robot asked a clarifying question about the food preparation container (“*Would you use the pan or the bowl to assemble the dish?*”) to disambiguate the intended recipe. Based on the answer (“*The bowl*”), it aligned its actions with their goal of making a salad, though one step of *pouring oil into bowl* was flagged as a mistake.

In real-world experiments, our implementation achieved an average inference latency of 12–14 seconds per timestep on an NVIDIA RTX 4090 GPU, using an RHP tree of depth 2 with parallelized low-latency LLM calls with the small model.

6.8 Discussion

Our results demonstrate that language and actions complement one another in revealing human goals. Explicit preferences expressed in language help interpret the intent behind actions, while observed actions clarify ambiguous or open-ended instructions. This bidirectional coupling allows the robot to refine human language with actions and ground its supportive actions in human preferences. More concretely, BALI integrates

language and action in a receding-horizon planner that imagines possible futures of the collaboration, yielding earlier and more stable convergence than language-only or action-only approaches. One unique capability of our method is integrating clarifying questions into the inference loop, balancing interruption cost against expected information gain. This allows the robot to resolve ambiguity efficiently without burdening the human by asking for instructions to improve collaboration fluency.

Our work bridges two paradigms of goal inference that have traditionally been studied in isolation: closed-set recognition, where robots classify among a finite set of goals, and open-ended reasoning, where goals may be unbounded and novel. Research on goal prediction has primarily remained in the closed case [93, 92, 86, 292, 138], with the open case largely avoided by constraining the problem to predefined goals. The few open-ended approaches, such as GOOD [171], rely exclusively on dialogue. By contrast, BALI fuses observed actions with preferences, grounding open-ended goals in the collaboration where intent is expressed through behavior and language.

Our work has limitations, which are opportunities for future work. First, the experiments assumed fixed turn-taking to induce equal participation between the human and the robot, but this is not a fundamental requirement of our method. Second, BALI was evaluated assuming the human’s end goal remains fixed throughout the interaction. Evaluating BALI when human goals change over time or when human behavior is suboptimal (like [87]) would be interesting. Third, in real-world deployments, BALI can incur noticeable latency from LLM-based scoring and tree search; this could be reduced by more aggressive pruning of candidate actions, caching attractor-field computations, or using smaller or quantized fine-tuned LLMs. Finally, BALI relies on LLMs as judges to score attractor fields in the open case, introducing variability and opacity. Future work includes addressing these issues through more interpretable scoring methods, studying human perceptions of clarifying questions in real deployments, and extending BALI with algorithms for open-world planning and

goal reasoning [50, 16].

6.9 Summary

In this chapter, we presented BALI (Bidirectional Action–Language Inference), an approach for maintaining shared understanding by inferring a person’s goal from two signals that naturally co-occur in collaboration: what the person says at a high level and what the person does. This work contributes a bidirectional formulation that uses high-level language to narrow the space of plausible actions the robot should take to support the task, and uses observed human actions and task history to ground and revise that interpretation of the natural-language instructions — supporting goal inference even when the intended goal is open-ended rather than drawn from a fixed set. We evaluated BALI in collaborative cooking across simulation and real-robot tasks, comparing it to baselines that rely solely on language and actions. Across these evaluations, BALI produced more reliable goal inference and more appropriate assistance than any single-modality baseline, while requiring only limited, targeted clarification when goal-uncertainty is high.

The final chapter concludes the dissertation by summarizing the thesis’s contributions and outlining additional avenues for future research.

Chapter 7

Conclusion

This dissertation aims to enable robots to adapt to people’s requirements to work with them fluently, where tasks are long-horizon and goals are rarely stated explicitly. Prior work has largely addressed this problem by relying on explicit step-by-step instructions or structured demonstrations [57, 82, 248]. In contrast, this work studies how robots can adapt on the job by maintaining shared understanding — inferring human goals from naturally occurring interaction signals and using that inference to provide timely assistance with minimal supervision.

This problem is difficult for two related reasons. First, goals must be inferred from human behavior, both actions and language that is inherently ambiguous [3, 22, 75, 5, 181, 208, 258]. The same actions or instructions can be consistent with multiple goals depending on context. Second, this ambiguity is compounded by non-stationary human goals, where changes in goals are rarely announced in natural interactions and can be difficult to distinguish from normal variability in behavior [175, 195]. Together, these challenges make it tricky for robots to reason under uncertainty about both what the human intends and whether their goals are changing, while acting in a manner aligned with the human’s expectations [33].

7.1 Summary of Contributions

To address the challenges reiterated above, we summarize the contributions we made throughout this thesis below.

- Chapter 2 conducted a survey of prior work on adaptation in physical human-robot collaboration, focusing on how robots updated behavior during an ongoing physical interaction during a shared task. Specifically, it organized the literature around: (i) the assumptions robots make about the human during interaction, (ii) adaptation techniques used in practice, (iii) the cues and communication channels the robot uses to reason about human behavior, and (iv) how these systems are evaluated.
- Chapter 3 proposed a method for tailoring visual object representations to human requirements using passive observation and sparse positive feedback. This chapter studied how a robot could learn task-relevant perceptual distinctions when the relevant concept was defined implicitly by a person’s choices and only a small number of positive examples were available, demonstrated in a collaborative recycling task. The core idea was to incorporate a human-guided learning objective into representation learning so that objects selected by the human became close in the embedding space, even when the distinguishing attributes were arbitrary and not captured by standard semantic categories. The method paired contrastive clustering with a human-supervised loss that avoided treating unselected objects as negatives, reflecting realistic interaction settings where the robot observes what the person chooses but cannot reliably draw inferences from what they reject. The results showed that this approach learned user-aligned visual concepts online from limited feedback and improved goal inference in cases where intent depended on user-defined attributes rather than fixed labels.

- Chapter 4 addresses goal inference in long-horizon tasks where the same observed human behavior could be consistent with multiple possible goals, which could make passive observation insufficient for establishing shared understanding early in a task. This chapter introduced Critical Decision Points (CDPs) as states where competing goal-conditioned strategies prescribed different immediate actions, and it formalized these points using a policy-divergence measure that quantified how strongly goal hypotheses disagreed about the human’s next action. Building on this formulation, the chapter proposed a receding-horizon planner that traded off task progress with information gain, selecting robot actions that steered the interaction toward highly informative states. The approach was evaluated in both a collaborative, fully observable cooking task and a competitive, partially observable hide-and-seek task, in simulation and on real robots, and it inferred goals earlier and more accurately than baselines. In collaboration, earlier clarification allowed the robot to better align its actions with the human’s goal and reduce incorrect actions, while in competition, influencing the human-controlled hider toward CDPs enabled the seeker to catch the hider more quickly than baseline strategies.
- Chapter 5 studied a more realistic interaction setting in which a human could change goals mid-task, which created two challenges for the robot — detecting when a goal switch had occurred and rapidly realigning to the updated goal so it could continue providing useful support without requiring explicit communication. Therefore, the chapter introduced a goal-change detection approach that reasons about when observed human behavior becomes inconsistent with the robot’s current goal hypothesis and the policy bank. After detecting a change, the robot revises its goal belief by retaining only the subset of past observations that remains informative under the newly plausible goals, rather than treating the entire interaction history as equally relevant. The robot then executed Dif-

ferentiating Actions, which were supportive actions selected to both advance the task and elicit human behavior that efficiently confirmed the updated goal. Together, these components enabled the robot to recover shared understanding quickly and provide better assistance than approaches that assumed human goals were stationary.

- Chapter 6 extended the work done in the prior chapters to settings where the human’s objective was not restricted to a predefined set and was communicated through a combination of language and action. It introduced BALI (Bidirectional Action–Language Inference) for goal prediction, which maintained an explicit set of evolving goal hypotheses and updated them online as new behavioral and natural language evidence became available from the human. The approach integrated this inference process with task-level planning so that assistance remained grounded in the physical state of the task. When the robot’s uncertainty about goals is higher than the cost to interrupt the human, the robot generated targeted clarification questions to the human to reduce its uncertainty about the human’s goals. Otherwise, it took supportive actions aligned with its current belief about the human’s goals. Evaluations in collaborative cooking, in simulation and on real robots, showed that BALI improved goal inference and reduced the number of incorrect actions compared to baselines.

7.2 Open Research Questions

This dissertation proposes several adaptation approaches for humans and robots working on a shared task, beginning with how a robot can learn task-relevant perceptual distinctions from passive observation of human behavior, then developing mechanisms for resolving goal ambiguity through action, reasoning about mid-task goal changes, and finally extending inference to open-ended goals communicated through both lan-

guage and action.

As adaptive robots get deployed in more complex and real-world environments, additional challenges arise that were not directly addressed in the preceding chapters. These include reasoning about human state when people are multi-tasking, adapting in the presence of multiple collaborators. Adapting in these more complex and realistic situations would require us to use more sophisticated adaptation techniques like causal reasoning and generative end-to-end policies. The remainder of this section outlines these open research questions.

7.2.1 Real-world Adaptation Considerations

Realistic Assumptions about Humans during Collaboration

Prior work on adaptation in controlled laboratory settings, where tasks are isolated, attention is sustained, and interaction proceeds in a structured and predictable manner. One challenge for adaptation in unstructured real-world collaborative tasks is modeling the human’s internal state while they are engaged in multiple, interleaved activities. People are not always focused on a single task or consistently available to interact with a robot. Attention, physical effort, and time pressure fluctuate as people manage competing demands. For example, during everyday household collaboration, a person may alternate between loading the dishwasher, handwashing dishes, and periodically stirring a pot of food, shifting focus based on urgency, timing constraints, or fatigue rather than completing tasks sequentially. This pattern of behavior is well documented in cognitive science and human–computer interaction. Research on multitasking and interruption shows that people frequently suspend tasks mid-execution and later resume them, often relying on environmental cues rather than explicit plans to manage progress [267]. Work on rationality and attention further demonstrates that responsiveness and availability vary with cognitive load [127, 128]. Together, these findings show that human state is dynamic and context-dependent, but they

have primarily informed interface design and interruption management rather than robot decision-making during physical collaboration.

For adaptation in HRC, this creates an interesting challenge. The robot must adapt its behavior while the human’s state or situational constraints, such as fatigue, urgency, or attentional load, varies across concurrently active subtasks. When multiple subtasks are interleaved, the same human state can have different implications depending on which subtask the human is currently prioritizing. Prior work in HRI has modeled aspects of human state such as fatigue, trust, or engagement and used these estimates to modulate robot behavior online [213]. However, these models generally assume that the human is focused on a single task and do not address how state estimation should interact with multitasking behavior, where the relevance of a particular state depends on the subtask being performed. For instance, physical fatigue may be critical while the human is unloading the dishwasher or scrubbing the dishes but largely irrelevant when briefly checking on the food, while time pressure may dominate while checking on the food. Existing approaches therefore struggle to adapt when effort, attention, and availability are distributed unevenly across multiple ongoing activities, motivating the need to reason about human state as both time-varying and subtask-dependent.

These challenges directly affect communication between the agents during the collaboration. Human–computer interaction research shows that communicative interruptions should be sensitive to workload and timing [123], yet many collaborative robot systems treat communication as possible and preferable by the human at every stage of the interaction. In multitasking settings, a human who is busy, rushed, or tired may be unwilling or unable to respond to clarification questions, even when those questions would be informative from the robot’s perspective. At the same time, the same human may welcome communication when it is clearly grounded in the subtask they are currently pursuing or expect the robot to assist with. For example, a brief

clarification about dish placement may be appropriate while loading the dishwasher but disruptive during a time-critical cooking step.

These challenges raise a broader set of open questions that are key to adaptation during collaboration.

- How should a robot estimate human state when people are working on multiple tasks concurrently where attention, fatigue, and urgency fluctuate across sub-tasks, and where these factors may have different relevance depending on the current activity?
- How should a robot infer which tasks the human is actively working on, which are temporarily suspended, and which are likely to be resumed?
- Given multiple ongoing tasks, how should the robot decide which tasks to assist with and which to avoid interfering in, particularly when responsibility has not been explicitly delegated?
- Over longer-term interaction, how should a robot learn which tasks a person is comfortable handing off and which they prefer to retain control over, especially when such preferences are implicit and may change over time?

These questions also complicate assumptions about communication.

- How should communication be conditioned jointly on the human's estimated state, such as being busy, in a hurry, or tired, and the current or anticipated sub-task?
- How much communication should the robot expect during the collaboration, and how should it adapt when feedback is sparse, delayed, or absent altogether?
- How should a robot update its internal models of human state, task priorities, and preferences when explicit corrections or confirmations are unavailable or are very sparse, relying instead on partial observation over extended interaction?

Adapting to Multiple People

In many real-world settings, like homes and hospital rooms, robots can be expected to collaborate with, more than one person [238]. In such unstructured environments, people typically coordinate implicitly and task responsibilities evolve over time. For a robot to become an effective collaborator in such settings, it must operate without a fixed notion of who it is collaborating with or how work is divided, and must dynamically adjust its behavior during the interaction.

When a shared task involves multiple humans collaborating, the robot’s adaptation is no longer solely about adjusting its behavior to a specific individual’s needs. It must decide who that adjustment should be oriented toward at a given moment. Demands for assistance may arise concurrently, differ in urgency, or conflict with one another. Therefore, the relevance of adapting to a particular person can change rapidly based on recent actions, interruptions, or emerging task bottlenecks. At the same time, collaborators often differ in skill, experience, physical capability, fatigue, situational awareness, and preferred level of autonomy, with no single individual necessarily possessing complete knowledge or capability for the task. Effective collaboration may therefore require the robot to infer and aggregate information distributed across multiple people while deciding where and how to adapt [3]. These human states must be inferred and updated for several individuals simultaneously under partial observability and time pressure, and may evolve at different rates in response to different aspects of the interaction [89]. As a result, it is insufficient to rely on a single shared model of human behavior or a fixed assignment of roles. Instead, the robot must continuously allocate its limited adaptive capacity across people based on ongoing interaction.

There has been very limited work about robots reasoning about adaptation in such contexts. While task allocation and coordination have been widely studied in multi-agent systems, these approaches typically assume structured task representa-

tions, centralized coordination, or agents with known objectives [146, 290]. Such assumptions do not hold in unstructured real-world collaborative settings involving multiple people, where task allocation is often implicit and situational. At the same time, most work on human–robot collaboration focuses on adaptation to a single human or assumes fixed roles or autonomy levels [244]. As a result, current adaptive systems lack mechanisms for dynamically selecting which human to adapt to based on observed need, context, or interaction history.

These challenges raise a number of interesting open questions.

- How should a robot infer, from ongoing interaction, which human currently requires assistance and what form that assistance should take?
- How should adaptation be prioritized when multiple people stand to benefit from different interventions at the same time?
- How can a robot maintain and update separate internal models for multiple human collaborators that capture differences in skill, preferences, fatigue, autonomy expectations, and communication style, while preserving distinctions between collaborators during ongoing interaction?

7.2.2 Underexplored Adaptation Techniques

Causality and Causal Models

Despite progress in adaptation, explicitly causal approaches remain relatively uncommon in physical human–robot collaboration. Most existing systems map observed cues to robot actions without representing why a human believes a step will succeed or how task components depend on one another. In complex real-world tasks, however, reasoning about the underlying cause–effect structure of a task can be critical. Humans naturally reason in causal terms, constructing internal models of how actions generate outcomes and how sub-tasks constrain one another, and they often

act by mentally simulating these structures [210, 245, 259]. Collaborative robots, in contrast, typically respond to surface behavior without explicitly representing this underlying structure.

Prior work shows that hand-crafted causal models provided by the human can be incorporated into robot decision making under uncertainty, for example by using causal structure as an approximate prior in POMDP formulations of object assembly and troubleshooting, leading to improved planning performance in experimental studies [28]. Related work further demonstrates that a small set of human-generated causal models can support transfer to new but structurally related tasks, reducing the need for task-specific data collection [27]. However, these approaches have largely been studied outside the context of real-time physical collaboration.

Therefore, one open question is whether and how robots can infer, maintain, or update a representation of a human partner’s causal model during an ongoing interaction, and how such a representation could be used to support adaptation. If available, causal models could help disambiguate human actions that are observationally similar but causally distinct, such as whether a correction reflects a preference change, an error recovery strategy, or a misconception about the task. Causal reasoning could also support more targeted assistance and communication by allowing the robot to intervene at the level of task structure rather than surface behavior, and by supporting explanation and repair when coordination breaks down.

At the same time, making causal reasoning effective during physical collaboration remains challenging. Inferring or updating causal structure, whether about the physical task or the human’s internal causal model, during live interaction must be performed under strict real-time constraints, limited opportunities for intervention, and safety-critical requirements. Moreover, causal representations are often most useful when they support communication about task structure or failure, yet explicit causal queries risk interrupting task flow, while purely action-based implicit communi-

cation may be difficult to convey causal intent. Understanding when causal reasoning improves collaboration, and how lightweight causal models can be integrated into adaptation to improve action selection, recovery, and selective communication without increasing human burden or reducing fluency, remains an open research direction.

Generative Approaches: Vision-Language-Action (VLA) Models and Diffusion Policies

While recent end-to-end approaches to robot action generation like VLA models and diffusion-based policies have enabled impressive generalization across robotic tasks, their role in adaptation during human-robot collaboration remains largely unexplored. VLAs provide a unified representation that combines perception, language, and action, reducing reliance on explicit task decomposition or hand-engineered state representations [173]. Diffusion policies similarly offer strong capabilities for generating continuous action sequences under uncertainty and have shown impressive performance in manipulation and control [247, 58]. Together, these approaches aim for the robot to learn flexible, general-purpose robotic policies.

In their current form, however, these approaches largely exclude the human from the decision-making loop. Most work treats the robot as operating autonomously without human intervention, with tasks specified through demonstrations or instructions and executed without modeling a human collaborator as an adaptive agent. When humans are present, they are typically sources of offline data or static commands rather than partners whose actions, preferences, and expectations evolve during interaction [209]. As a result, current end-to-end models are difficult to apply directly to human-robot collaboration, where adapting to human behavior is central.

There has been some very early work exploring how human feedback might be incorporated into vision-language-action models. For example, recent work has begun to investigate how robots can use model uncertainty from VLA models to ask humans

for assistance [142]. However, these efforts remain preliminary and do not yet address the broader challenges of collaborative interaction, such as reasoning about human intent, handling ambiguous or partial feedback, or adapting behavior over extended interaction. More broadly, widely used VLA models such as SayCan [6] and RT-2 [316] demonstrate strong language-grounded control, but still assume fixed policies at deployment time and do not adapt to a human collaborator.

This omission of humans from the decision making loop of such generative approaches poses a key challenge for adaptation in HRC. End-to-end policies collapse perception, reasoning, and action into a unified representation, without explicitly modeling human actions, intent, or preferences. Without modeling the human, it is tricky for a robot to distinguish between environmental variation and changes driven by a human collaborator, or how it can adapt in response to human corrections or evolving expectations. In this sense, the absence of explicit human modeling makes existing end-to-end approaches poorly aligned with the requirements of collaboration. At the same time, this limitation highlights why end-to-end models remain a compelling direction for adaptation in HRC. A unified model that integrates perception, language, and action could enable adaptation that is grounded directly in the same sensory and linguistic signals that humans naturally use during collaboration, without relying on brittle task abstractions or manually specified interaction rules. If extended to explicitly account for humans as adaptive agents, such models could offer a principled way to couple perception, communication, and action in long-horizon interaction.

These observations raise several open questions.

- How can end-to-end policies be extended to model humans as adaptive collaborators rather than a part of the environment?
- What internal structure or auxiliary representations are required to support inference over human intent, preferences, or expectations?

- Where should adaptation occur within an end-to-end system, whether through conditioning, internal state, objectives, or policy parameters, while maintaining stability and safety during interaction?

Communication further complicates these issues. Although language-conditioned models can respond to instructions, it remains unclear how communication should function as an adaptive signal during collaboration rather than a one-time task specification. Therefore, several open questions remain how communication between the human and the robot should be modeled while using these generative models. For example:

- How should an end-to-end policy interpret corrective or ambiguous human utterances?
- How should it decide when to request clarification, when to act autonomously, and when to defer in the absence of explicit feedback?

Appendix A

Additional Methodological Details For Chapter 3

A.1 Additional Details of the Case Study with a Recycling Robot

This section explains in detail components of our system architecture. Figure A.2 shows our experimental setup highlighting the field of view of our three cameras. Figure A.1 depicts the interaction flow of our experimental setup in conjunction with the components of our system.

A.1.1 Custom End-Effector Attachment Design for the Stretch RE-1 robot

The Stretch RE-1 robot [145] extracts items from the conveyor belt by either extending its telescopic arm to push objects off the conveyor belt or pulling objects towards itself. The default gripper of the robot was not appropriate for removing recyclables from the conveyor belt because of the slow nature of the servo motors used for en-

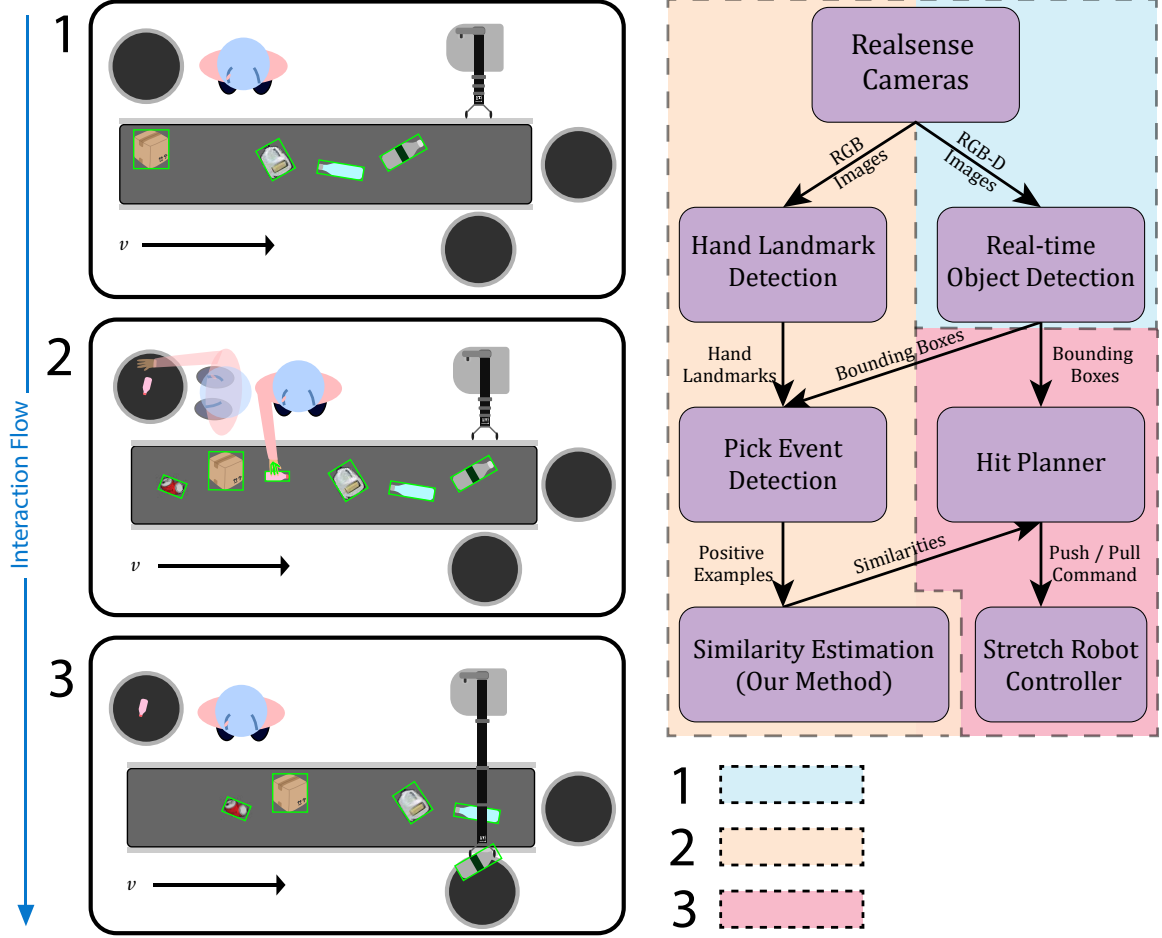


Figure A.1: Interaction Flow with System Architecture: In our interaction flow, a human selects some items from a given category from a conveyor belt. The system estimates the similarity of each remaining item on the conveyor belt. Finally, the robot selects objects most similar to the human-selected objects and sends a push/pull command to the robot. Each colored zone (and its corresponding number code) shows the system components deployed during different phases of the interaction flow.

gaging the grippers. Therefore, we designed and built a custom static attachment to the end of the telescoping arm of the Stretch RE-1 robot (as shown in Figure A.3) that allowed the robot to push or pull items from the conveyor belt. Our custom end-effector attachment consists of a base plate with a divider across the middle of two concave halves to help guide targets to the pull or push action and is mounted onto the threaded shaft for the removed gripper. It is supported by a cover that slides onto the servo motor on the arm and also clips onto the edge of the telescoping bar for stability. We coat this attachment with sandpaper on all possible contact areas

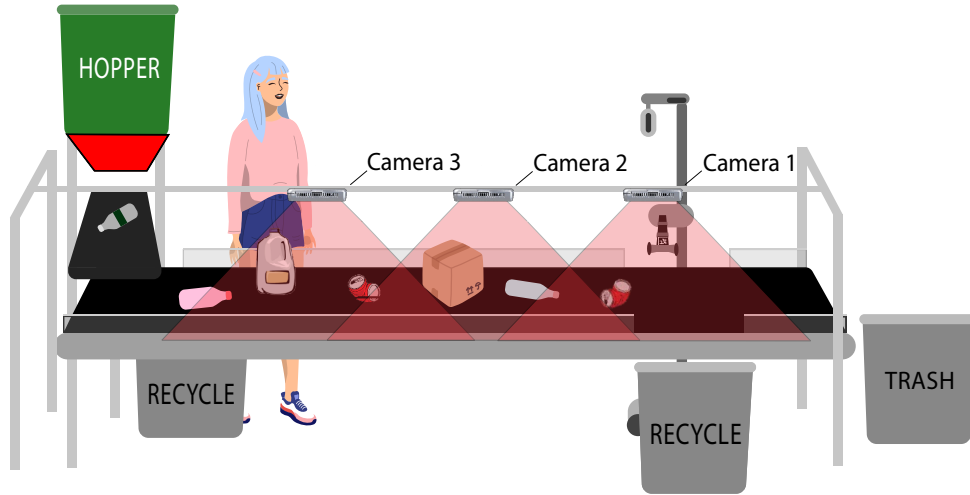


Figure A.2: Experimental Setup with Field of View of each camera highlighted

for enhanced grip.

Empirically, we found this attachment to be highly effective at removing most kinds of objects. As shown in the supplementary video, the attachment would gently guide the object to the nearest edge if the target objects were primarily on either half of the belt. However, for highly deformed objects (like some heavily crushed cans or bottles) or objects with a lot of protrusions (like trays), the objects could sometimes get stuck on the attachment’s divider. Occasionally, this caused the robot to fail in removing an object even when it attempted to remove the desired object.

A.1.2 Detection of Picked Objects

This module determines which objects are picked by humans in real-time. This module has the following two parts:

1. **Pick Event Detection:** The system uses the MediaPipe real-time hand tracking API [302] to detect 21 2D landmarks on the user’s palm. These landmarks were superimposed on the images. These images were then used to train a

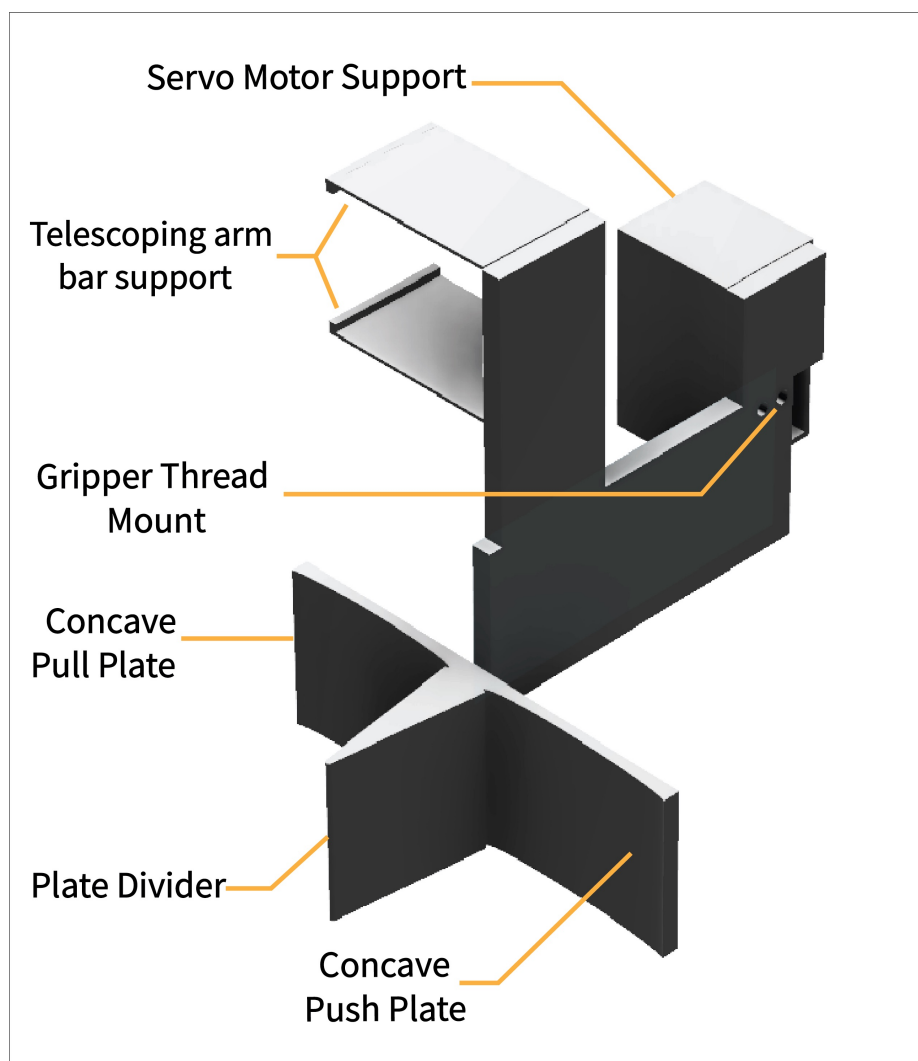


Figure A.3: Design of our custom attachment for pushing and pulling objects off the conveyor belt

ResNet-18 [116] image classifier offline. The images are classified as either a picking event, not a picking event, or none if there is no hand present in the image. To train this model, we collected videos of 13 people picking items off the moving conveyor belt to capture the different techniques people use to pick up different objects. We labeled each frame manually as either a pick event, not a pick event, or no hand present in the frame. The trained model predicts a pick-event onset with 96% accuracy on the leave-one-out cross-validation set.

2. Identification of Picked Objects: To detect which object was picked by

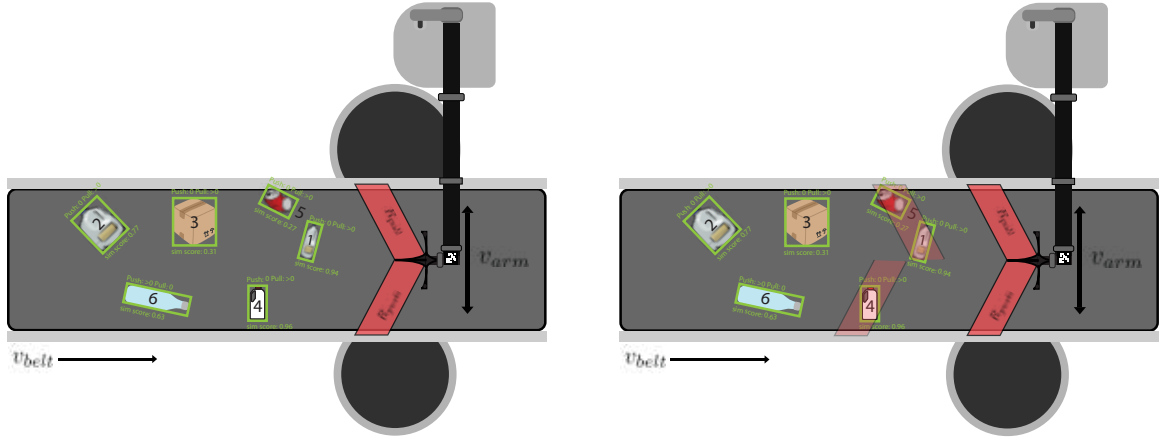
the person, we first ran inference on the belt images using the trained instance segmentation model (YOLACT [35]) to identify each object’s position. We then used the predictions of the trained pick onset detector described above to identify the precise time when the person picks up an object. We use the 21 landmarks predicted on the person’s palm to draw a tightly fitted rotated bounding box on their palm. Finally, we find an overlap between the rotated bounding box over the person’s palm and each of the predicted object bounding boxes in a few frames preceding the onset of *Pick* event to obtain an un-occluded image of the object that was picked.

A.1.3 Hit Planner

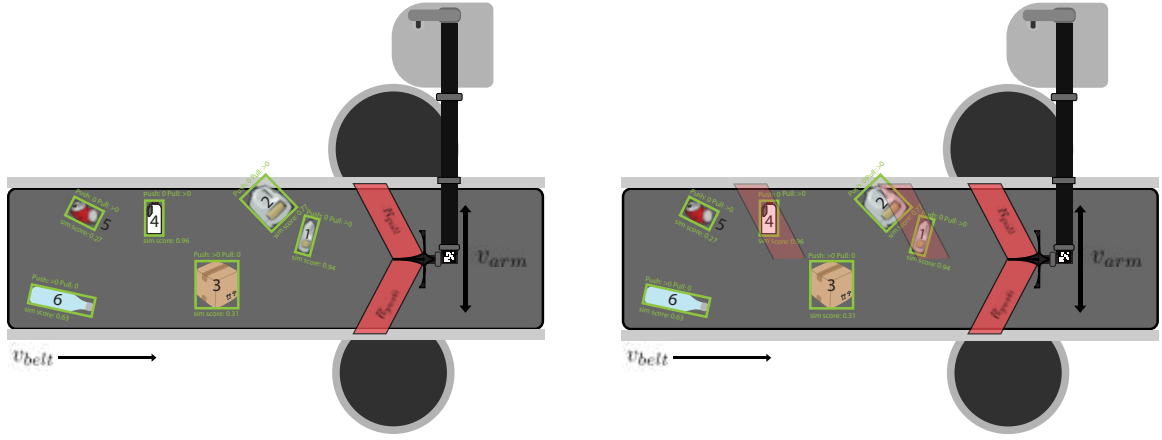
We used our method to obtain similarity scores for all objects on the conveyor belt within the camera’s frame situated between the robot and the human (Camera 2 in Figure A.2). These similarity scores were then passed to the object removal algorithm within the hit planner module to assign each object an associated “hit score” using a multi-variable function. This score was maximized across all objects in the frame, S , and one item was selected for removal by the robot.

For each object, our object removal action policy first determines the action sequence that would be most successful at removal given the object’s positioning relative to the plate mounted to the end of the robot’s telescopic arm, as seen in Figure A.3. The action space of the robot is comprised of either a pushing, pulling or avoiding action sequence.

Given our removal actuation constraints, removing a target object may remove other unintended objects, referred to as “casualties.” To account for the possibility of casualties, for every object, we approximated a region of collision using the speed of the belt, v_{belt} , and the robot’s arm extension/retraction speed, v_{arm} . The regions of collision for every corresponding action sequence were two dimensional parallelo-



(a) Scenario 1



(b) Scenario 2

Figure A.4: Visualization of Different Scenarios for the Hit Planner: a) In Scenario 1, the removal of object 1 would inadvertently remove object 5, which has a relatively low similarity score. The hit planning algorithm would negate the score contribution of object 5 to object 1's hit score as explained in A.1.3 and determine that object 4 by itself has the highest hit score, thereby initiating a push action sequence. b) In Scenario 2, the removal of object 1 would likely remove object 2, which has a relatively high similarity score and would only increase object 1's hit score as explained in A.1.3. In this instance, the robot would target object 1 with a pull action sequence and target object 4 with a pull action sequence once again after resetting the telescopic arm to a central position.

grams extending from the push plate with inclination $\tan^{-1} \left(\frac{v_{arm}}{v_{belt}} \right)$. All objects whose bounding boxes overlap with this region have a removal probability proportional to their percent overlap with the collision region.

Given a frame containing a set of objects S , we perform optimizations that yield the object with the highest push score, i_{push} , and similarly the object with the highest pull score, i_{pull} . If there are multiple objects in S it is possible for i_{push} and i_{pull} to be distinct. Therefore

$$i_r = \max\{i_{push}, i_{pull}\}$$

returns the single object i_r to be removed by the robot as well as the corresponding action sequence request that should be sent to the robot's onboard computer. For the push action sequence, we optimized for

$$i_{push} = \max_{i \in S} \mathbb{1}_{push}(i) * \left[\alpha_1 sim_i + \sum_{j \in S, i \neq j} C(j) * \alpha_1 sim_j * \alpha_2 \left(\frac{|R_{push} \cap R_j|}{|R_j|} \right) \right]$$

Similarly for the pull action sequence, we optimized for

$$i_{pull} = \max_{i \in S} \mathbb{1}_{pull}(i) * \left[\alpha_1 sim_i + \sum_{j \in S, i \neq j} C(j) * \alpha_1 sim_j * \alpha_2 \left(\frac{|R_{pull} \cap R_j|}{|R_j|} \right) \right]$$

where α_1 and α_2 are tuning constants, sim_i represents the similarity score for object i , and $\frac{|R_{pull} \cap R_i|}{|R_i|}$ represents the percent overlap of object's bounding box, R_i , with the collision region R_{pull} . $\mathbb{1}_{push}$ and $\mathbb{1}_{pull}$ are indicator functions for assessing whether a pull or push is necessary for given object i . If the position of an object i is above the push plate of the robot relative to the belt, $\mathbb{1}_{push} = 1$ and $\mathbb{1}_{pull} = 0$ and vice versa for when i is situated below the push plate. We include a mapping function $C(i)$ that either subtracts or adds the score contribution of a casualty i depending on

whether or not the similarity score is above some predefined threshold T_{sim} .

$$C(i) := \begin{cases} 1 & \text{if } sim_i \geq T_{sim} \\ -1 & \text{if } sim_i < T_{sim} \end{cases}$$

This is to ensure that we prioritize removal of objects that have a high probability of removing objects that are similar to the human-selected objects.

A.2 Properties of the Dataset

It is challenging to develop a dataset that includes the enormous variety of recyclables that a real Materials Recovery Facility (MRF) processes. For our research, we toured such facilities and watched numerous videos documenting the composition of streams at MRFs. Then, we worked to create a collection of objects representing a characteristic sampling of recyclables typically seen in these facilities. More specifically, we selected our set of over 500 unique objects that were diverse in category, material, size, color, shape, deformability, reflectiveness, dirtiness, opacity, and density. We deliberately excluded paper-based and glass-based recyclables because air pumps and optical sorters typically remove these items reasonably well in a MRF. For our experiments, we captured images of the selected objects under different lighting conditions, orientations, and levels of motion blur on a moving conveyor belt, adding to the diversity of our dataset.

The dataset was collected using three Intel RealSense D435i RGB-D cameras mounted over the conveyor belt. We used our trained instance segmentation model to run inference on the overhead camera stream at 1fps. The predicted instance segmentation masks were converted to rotated bounding boxes, then cropped and saved offline. The dataset was manually curated to obtain 1502 images of objects across the ten categories whose distribution is shown in Figure A.5. Figure 3.4 shows

some sample images from each of our categories in the dataset.

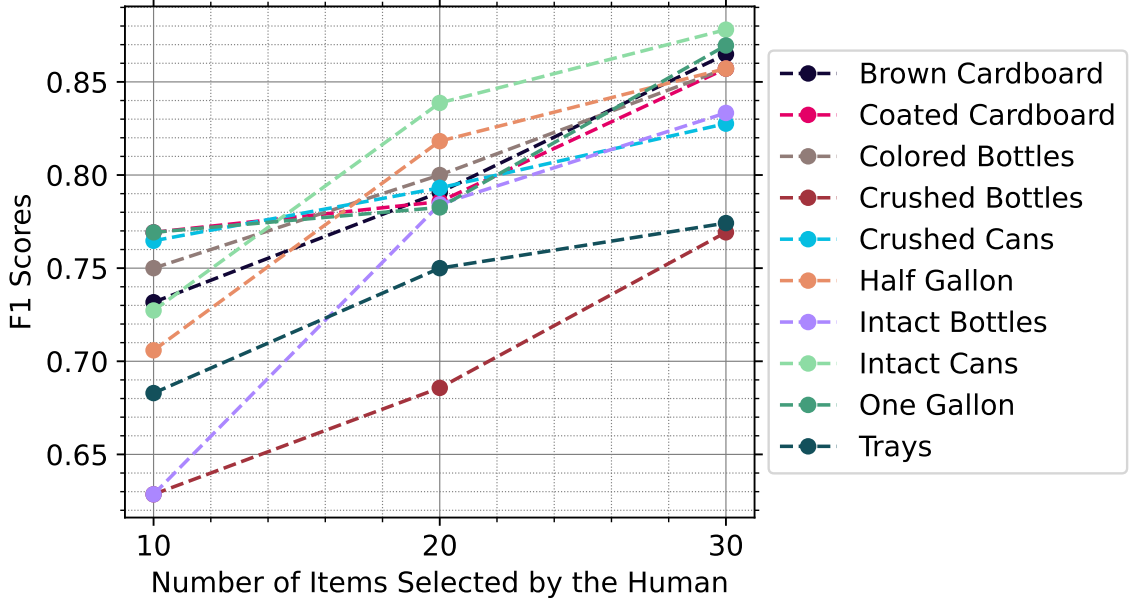


Figure A.5: Data distribution in our offline dataset

A.3 Implementation Details for Our Method

To deal with memory and computational constraints, we resize each image to 128 x 128. We then stochastically apply three augmentations in the following sequence: *random cropping* followed by resizing to original size, *random color distortions*, and *random gaussian blur*. Finally, we train and evaluate our approach end-to-end on a single NVIDIA GTX-3080ti GPU with the hyperparameters shown in Table A.1. We use the same set of hyperparameters for the ablation studies as we did not notice any significant variation in the results with changes in the learning rate, projection dimension, instance temperature, or cluster temperature. We also studied the importance of loss weighting of the instance and the cluster losses (λ_{ins} and λ_{clu}). Similar to Li et.al.[162], we obtained optimal performance when the losses were weighted equally. We ran a hyperparameter search between 0.1 and 1 to determine the optimal

weighting for the human-supervised loss (λ_{human}). We obtained the best performance across all categories when λ_{human} was 1.

Parameters	Values
Learning Rate	0.01
Batch Size	64
Feature Extractor	ResNet-18
Number of Epochs	100
Projection Dimension	128
Number of Clusters	10
Instance Temperature (τ_{inst})	0.5
Cluster Temperature (τ_{clu})	1.0
Instance Loss Weight (λ_{ins})	1.0
Cluster Loss Weight (λ_{clu})	1.0
Human Supervised Loss Weight (λ_{human})	1.0

Table A.1: Hyperparameters for Training Our Method

A.4 Design Considerations of Our Method

We wanted to design a system that enables a human to teach a robot partner the kinds of objects that they are sorting with a few examples and that could be demonstrated in the context of recycling in Materials Recovery Facilities (MRFs). In such facilities, the human workers typically sort out one type of recyclables while adapting to variations in task specifications and seasonal changes in the stream composition. Because of how objects are handled in a MRF and the economic reality of the recycling industry, one of our primary problem constraints was that the robot could quickly learn from a limited number of examples of an object category provided by a human. The next paragraphs explain each of our design considerations in detail:

1. **Large object variety:** In the recycling setting, the objects are not guaranteed to look constant through time because of the way they are handled, their deformable properties, and the frequent introduction of new objects. For a standard supervised learning method to perform well at classifying these objects, it

would require training models with large datasets to keep up with the performance requirements. Therefore, we chose to approach this problem from a self-supervised contrastive learning perspective, which could adapt to the changing stream composition by constantly re-training in a more active learning fashion with just a few new human labels.

We explored various state-of-the-art contrastive learning techniques for our work and found that these techniques typically pass two stochastically augmented views of the same image through an image classification network and a multi-layer perceptron (MLP) to bring them close to each other in the embedding space. These methods have been shown to learn very strong representations for initializing large neural networks that perform a wide variety of downstream supervised learning tasks like image classification, semantic segmentation, and object detection [139]. However, since we wanted to learn a representation of objects such that similar objects were grouped in the embedding space, we needed to take into account inter-instance similarity. We found contrastive clustering [162] to be the best self-supervised method for learning object representations.

2. **Humans are unable to select all examples of a given category:** In a real-world recycling setting, where the conveyor belts are typically very crowded, the human sorters cannot pick out all the objects that belong to a given category. So, there could be objects left on the conveyor belt that belong to the category of interest. This means that the robot would have to gain an understanding of the properties of the objects of interest to their human partner with a limited number of positive examples but no negative examples. Therefore, we designed the proposed human-supervised loss to force the features of the human-selected examples to be close to each other in the embedding space. When the human-supervised loss is jointly trained with the contrastive clustering objective function, the human-supervised loss guides the formation of clusters in the

K	Average F1 scores across all categories
2	0.820 ± 0.016
8	0.876 ± 0.02
9	0.882 ± 0.02
10	0.886 ± 0.017
11	0.883 ± 0.019
12	0.877 ± 0.02
15	0.867 ± 0.021
20	0.847 ± 0.018
50	0.828 ± 0.022

Table A.2: Impact of the number of clusters (K) on overall model performance

embedding space to be better aligned with human choices.

3. **Real time training considerations:** Theoretically, any image classification network could have been used as a backbone for the contrastive learner. However, since we intended to use this model in real-time, we opted for ResNet-18 [116], which is lightweight and has been shown in the literature to be powerful enough to learn good visual representations [54, 56].

A.5 Additional Results

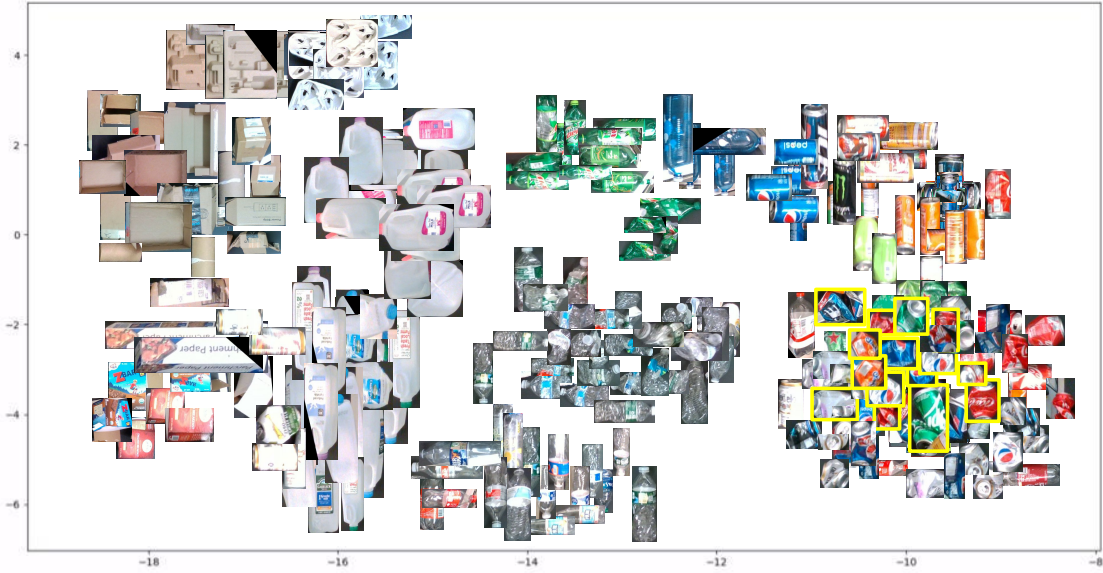
A.5.1 Offline Evaluation Results

Impact of the Number of Clusters (K) in the Cluster Projection Head

We base our work on a self-supervised clustering algorithm, Contrastive Clustering [162]. Thus, we are required to provide the number of clusters K as a hyperparameter of the model. In the results presented in the paper, we defined the number of clusters K to be the number of categories C to keep our evaluation consistent with the original Contrastive Clustering paper [162]. However, our method does not need to know precisely how many pre-defined object categories are being observed; instead, a ballpark amount suffices. To demonstrate this idea, we conducted an experiment showing that



(a) Contrastive Clustering



(b) Our Approach

Figure A.6: Comparison of t-SNE Visualization of the learned representations on the offline dataset: a) t-SNE plot for the self-supervised contrastive clustering [162], b) t-SNE plot for our approach. Images with yellow border denote the objects selected by the human. In our approach, clusters for each category show better separation than the contrastive clustering approach, and clusters are overlapping when an increasing number of features are common.

the number of clusters (K) need not be precisely equal to the number of categories (C). As indicated in the table A.2, if the number of clusters is significantly lower or higher than the number of categories in the downstream task, there is a significant decrease in performance. Otherwise, the performance of our approach is robust to the value of K .

Qualitative Results

Figure A.6 shows plots of 10 clusters obtained by projecting the embedding vectors for the points (128-dimensions) onto 2-dimensions using t-SNE for contrastive clustering (Figure A.6a) and for our approach (Figure A.6b). In Figure A.6b, the images with a yellow border show the human-selected objects. This shows that the clusters formed as a result of our approach are more homogeneous and show good separation across classes, overlapping only when an increasing number of features are common. For example, in contrastive clustering (Figure A.6a), many crushed cans lie in the same cluster as un-crushed cans if they are of the same color or with crushed bottles. However, in our approach, most crushed cans get pulled into the same cluster when the human provides examples of crushed cans.

A.5.2 Real Robot Evaluation

Figure A.8 shows some qualitative examples of similarity scores for every object on the conveyor belt when the human trained our approach in real-time with 30 examples of different types of objects. These results show that the similarity scores are closer to 1 for the objects belonging to the category of interest, and the similarity scores are lower for those not belonging to the category of interest.

Table A.4 shows the average number of True Positives, False Positives, False Negatives, and True Negatives across all categories as our approach was trained with 10, 20, and 30 human-selected examples. These results show that the average number

Experiment	1	2	3	4	5	6	7
Crushed Bottles	0.12	0.65	0.50	0.56	0.59	0.70	0.82
	± 0.00	± 0.02	± 0.06	± 0.01	± 0.03	± 0.03	± 0.02
Intact Bottles	0.21	0.58	0.51	0.64	0.67	0.68	0.80
	± 0.00	± 0.03	± 0.02	± 0.01	± 0.01	± 0.02	± 0.03
One Gallon	0.14	0.76	0.64	0.73	0.68	0.81	0.84
	± 0.00	± 0.01	± 0.03	± 0.01	± 0.01	± 0.01	± 0.02
Half Gallon	0.19	0.72	0.69	0.68	0.69	0.79	0.86
	± 0.00	± 0.02	± 0.01	± 0.01	± 0.01	± 0.02	± 0.02
Crushed Cans	0.35	0.73	0.65	0.78	0.73	0.82	0.84
	± 0.00	± 0.04	± 0.01	± 0.03	± 0.03	± 0.01	± 0.02
Intact Cans	0.40	0.81	0.73	0.82	0.81	0.85	0.86
	± 0.00	± 0.00	± 0.02	± 0.01	± 0.01	± 0.02	± 0.02
Brown Cardboard	0.13	0.74	0.71	0.75	0.84	0.89	0.94
	± 0.00	± 0.02	± 0.01	± 0.02	± 0.02	± 0.02	± 0.01
Coated Cardboard	0.05	0.87	0.37	0.52	0.80	0.91	0.98
	± 0.00	± 0.05	± 0.03	± 0.02	± 0.01	± 0.04	± 0.02
Colored Bottles	0.08	0.91	0.71	0.77	0.64	0.93	0.93
	± 0.00	± 0.01	± 0.01	± 0.00	± 0.00	± 0.02	± 0.00
Trays	0.06	0.68	0.56	0.55	0.66	0.98	1.00
	± 0.00	± 0.01	± 0.01	± 0.01	± 0.01	± 0.01	± 0.00

Human Supervision Only

Instance Loss Only (SimCLR)

Cluster Loss Only

Instance Loss + Human Supervision

Cluster Loss + Human Supervision

Contrastive Clustering (Instance Loss + Cluster Loss)

Human Supervised Contrastive Clustering (Our Method)

Table A.3: Results of our Offline Evaluation (Accessible Version): Comparison of F1 Scores with Baselines and Ablation Study aggregated over 3 human-selected pools per category.

		Evaluation Set									
		1	2	3	4	5	6	7	8	9	10
Training Set	1	-0.08	0	-0.06	-0.02	0	-0.21	-0.26	0.03	0.05	0
	2	-0.02	0.05	-0.02	-0.07	0.02	-0.38	-0.42	0.02	-0.01	0.25
	3	0.02	0	-0.04	-0.33	0	-0.44	-0.48	-0.17	0.04	0.13
	4	0	0	-0.06	0.01	-0.02	-0.33	-0.38	-0.1	-0.01	-0.15
	5	0.01	-0.09	0	-0.3	0.01	-0.64	-0.68	0.03	0	-0.16
	6	0.07	0.08	-0.02	-0.05	-0.07	-0.34	-0.39	-0.15	-0.01	-0.22
	7	-0.09	-0.05	-0.04	-0.13	-0.03	-0.29	-0.34	-0.01	-0.01	0.11
	8	0.04	0.02	-0.04	-0.1	0	-0.29	-0.33	-0.03	0.03	0.12
	9	-0.15	0	-0.13	-0.08	-0.04	-0.24	-0.28	-0.12	0.06	-0.14
	10	-0.05	-0.01	-0.07	-0.07	0	-0.36	-0.41	-0.16	0.02	-0.13

a) Impact of Human Supervision
on Instance Loss

		1	2	3	4	5	6	7	8	9	10
Training Set	1	0.09	-0.09	-0.04	0	-0.02	0.13	0	0.01	0.08	0.07
	2	0.06	0.08	-0.02	-0.03	0.04	0.13	-0.1	0.04	0.13	0.08
	3	0.05	-0.05	-0.01	-0.28	-0.04	-0.15	0.02	-0.08	0.04	-0.14
	4	-0.04	-0.05	-0.09	0.13	-0.08	-0.1	-0.01	0.05	0.02	-0.17
	5	0.02	0.08	-0.05	-0.07	0.08	-0.01	0.07	0.08	0.02	0.01
	6	0.03	-0.01	-0.02	0.04	-0.01	0.44	-0.01	0.08	0.05	-0.07
	7	-0.07	-0.03	-0.08	-0.03	-0.07	-0.14	-0.06	0.05	-0.04	-0.08
	8	0.02	-0.01	-0.04	-0.25	0	-0.08	-0.08	0.04	0.1	0.05
	9	-0.06	-0.05	-0.07	-0.02	-0.08	-0.02	0.03	0.06	0.17	-0.11
	10	-0.04	-0.06	-0.09	0.05	-0.02	0.01	0.01	-0.01	0.1	0.1

b) Impact of Human Supervision
on Cluster Loss

		1	2	3	4	5	6	7	8	9	10
Training Set	1	0.12	0.01	0.07	0	0.04	0.05	0.02	0.05	0.09	0.02
	2	0.04	0.02	0.01	-0.02	0.03	0.03	0.02	0.02	0.08	0.01
	3	0.05	0	0.06	0.07	0.04	0.03	0	0.02	0.1	0.02
	4	0.03	-0.01	0.04	0.05	0.05	-0.06	0.01	0.03	0.1	0
	5	0.07	0.01	0.05	0.05	0.01	0.06	0.01	0.01	0.1	0.02
	6	0.07	0.02	0.05	0.01	0.03	0.07	0.01	0.02	0.09	0.02
	7	-0.09	-0.05	-0.01	0.03	0.03	0.03	0	0.01	0.08	0.02
	8	0.08	0.01	0	0.02	0.05	0.04	0.02	0.03	0.11	0.02
	9	0.08	0	0.05	0.02	0.02	0.05	0.01	0.03	0.12	0.01
	10	0.06	0	0.06	0.01	0.03	0	0.01	0.04	0.09	0.02

c) Impact of Human Supervision
on Contrastive Clustering

1: Crushed Bottles	3: Half Gallon Milk Jugs	5: Un-crushed Cans	7: Colored Bottles	9: Un-crushed Bottles
2: Crushed Cans	4: Brown Cardboard	6: Coated Cardboard	8: One Gallon Milk Jugs	10: Cardboard Trays

Figure A.7: Impact of augmenting human supervision to contrastive learning (Accessible Version): Difference in performance between self-supervised learning augmented with human supervision and the self-supervised learning model trained independently for instance loss, cluster loss, and contrastive clustering, respectively across categories. Specifically, w.r.t. the experiment numbers in Figure 3 in the main paper, a) represents Experiment 4 - Experiment 2, b) represents Experiment 5 - Experiment 3, c) represents Experiment 7 - Experiment 6. (Best viewed on a PDF processor with zoom)



Figure A.8: Sample Qualitative Results from our Robot System: Similarity scores of objects on the conveyor belt obtained from our method, when the human trained the system with 30 objects of type a) half-gallon milk jugs b) un-crushed cans

of True Positives and True Negatives increases with more human-provided examples. In contrast, the average number of False Positives and False Negatives decreases as the number of human-provided examples increases.

The main paper demonstrates the performance of the real robot by showing the F1-scores of the robot system as the researcher picks 30 items in increment of 10. Figure A.9 shows the accuracy of the robot system when evaluated in a similar fashion. The robot’s accuracy improved by 3 points on average across categories for every ten human-provided examples.

Number of Training Items	10	20	30
True Positives	13.4 ± 4.94	15.2 ± 4.31	16.8 ± 4.73
False Positives	4.7 ± 1.88	3.5 ± 1.71	2.3 ± 1.33
False Negatives	5.8 ± 2.93	4.9 ± 2.23	4.2 ± 2.20
True Negatives	60.6 ± 16.61	69 ± 13.07	69.4 ± 14.30
Precision	0.739 ± 0.07	0.814 ± 0.074	0.885 ± 0.058
Recall	0.706 ± 0.10	0.761 ± 0.074	0.806 ± 0.08
F1-score	0.715 ± 0.05	0.782 ± 0.04	0.838 ± 0.03
Accuracy	0.87 ± 0.04	0.908 ± 0.02	0.929 ± 0.02

Table A.4: Performance Statistics of our Method on the Real Robot System

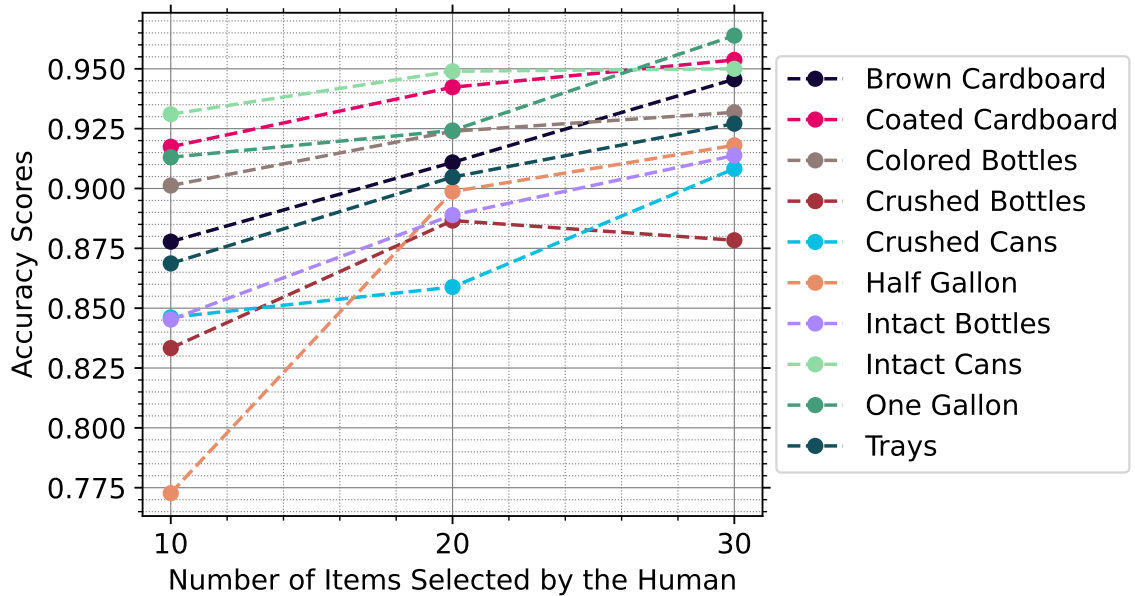


Figure A.9: Accuracy of Our Method in Real Time

A.6 Discussion

A.6.1 Additional Intuitive Explanation of Our Method

It is important to note that inductive bias is key for a machine learning model to learn from just a few samples. In our paper, this bias is provided through the cluster loss, which partitions the embedding space into a pre-specified number of clusters that we hope represent object categories – though our approach does not enforce this. Adding the human-supervised loss to the representation learning process helps guide the formation of the clusters according to the human’s requirements. Worth noting, that knowing the exact number of object categories that the robot needs to handle in practice is not a requirement of our approach.

A.6.2 Assumption about the Human Selecting One Category of Objects

The assumption that a human would manipulate objects from a single object category holds across many real-world Materials Recovery Facilities (MRF), which our setup models. When we started this project, we visited a recycling facility to understand how each category of recyclables is sorted sequentially using a combination of automated sorters and humans. At every stage in the sorting process, a particular category of recyclable was extracted from the stream using an automatic sorter. The items of a given category missed by the automatic sorter were sorted by human sorters who tried manually extracting the remaining objects on a fast-moving and crowded conveyor belt. The human sorters are trained to extract only one category of interest at a given time because some objects in the recycling stream are more valuable than others. For example, after an automatic sorter has extracted most of the paper from the stream, the human sorters would be tasked with extracting all the remaining paper-based products. This would make the stream more uniform for subsequent automatic sorters to remove a certain kind of plastic. In practice, though, the human sorters may not be able to extract all the objects of the desired category, which could lead to quality issues in the target end-product. Therefore, if a robot were to assist a human sorter in sorting through a particular kind of object, the robot would have to learn the properties of the objects that its human collaborator is interested in. Our work proposes a solution to this problem whereby the robot observes the human picking up a limited number of objects from a single category and quickly learns to distinguish these types of objects.

Appendix B

Additional Methodological Details For Chapter 6

B.1 BALI Prompts

Intent Summary Prompt

You are DishGen, a cooking assistant. Analyze ingredient-fetching actions and group them by category (fruits, vegetables, dairy, grains, tools). Think about what food the human could be making based on their actions and preferences.

Preferences: {prefs_all}

Action History: {acts_all}

Think step-by-step and close with `</think>`. Inside `<answer></answer>` provide one concise sentence summarizing: (a) the dish being made, (b) stage of completion, (c) containers and ingredients “in play,” (d) next logical steps.

Valid Action Filter Prompt

System Prompt

You are DishGen, an English-only collaborative cooking assistant. Your job is to answer YES or NO to whether a specific action is relevant to the current cooking

task.

Consider:

- The human's implicit intent summary, explicit preferences, and potential recipe they could be making
- The action history already taken by the human
- Whether this action would logically advance their intended recipe by using the ingredients, containers, and appliances the human has been using
- Cooking commonsense and logical next steps
- Container consistency — stick to what the human is already using
- Recipe-specific relevance — avoid actions that don't fit the specific dish

Answer with ONLY "YES" or "NO" — no other text.

Optional context lines add explicit preferences, dislikes, and possible goals when present.

Few-shot Examples

1. Human assembling in GLASS:

Context:

- Intent Summary: cold, layered snack
- Possible Goals: layered yogurt parfait, layered strawberry parfait
- Action History: get glass, get yogurt, get strawberry, pour yogurt glass

Q: Is "pour strawberry glass" relevant? A: YES

Q: Is "pour strawberry bowl" relevant? A: NO

Q: Is "pour strawberry pan" relevant? A: NO

Q: Is "pour strawberry blender" relevant? A: NO

Q: Is "get yogurt" relevant? A: NO

Q: Is "get apple" relevant? A: YES

Q: Is "get mixingspoon" relevant? A: NO

Q: Is "get eatingspoon" relevant? A: YES

2. Human assembling in BLENDER:

Context:

- Intent Summary: fruity blended dish
- Possible Goals: berry smoothie, chocolate smoothie, fruit juice
- Action History: get blender, get banana, pour banana blender

Q: Is "pour banana blender" relevant? A: YES

Q: Is "pour banana glass" relevant? A: NO

Q: Is "get blueberry" relevant? A: YES

Q: Is "get eatingspoon" relevant? A: NO

Q: Is "get mixingspoon" relevant? A: NO

3. Human assembling in BOWL:

Context:

- Intent Summary: warm breakfast with grains
- Possible Goals: fruit oatmeal, porridge
- Action History: get bowl, get oats, get banana, pour oats bowl

Q: Is "pour banana bowl" relevant? A: YES

Q: Is "pour banana glass" relevant? A: NO

Q: Is "get yogurt" relevant? A: YES

Q: Is "get mixingspoon" relevant? A: YES

Q: Is "get eatingspoon" relevant? A: YES

Q: Is "get blueberry" relevant? A: YES

Q: Is "get rice" relevant? A: NO

Goal Management Prompt

System prompt

You are DishGen, a cooking assistant.

TASK:

Given:

1. Human's verbal preferences.
2. Human's explicit dislikes and what they don't want.
3. Human's past cooking actions.
4. Actions and methods the human dislikes.
5. A summary of their intent.
6. Current proposed recipes.
7. Available ingredients: *ingredients_str*
8. Available containers/appliances: *containers_appliances_str*

You must suggest NEW possible recipes they could be making.

RULES:

9. Use ONLY the listed ingredients, containers, and appliances.
10. Do not repeat or make very similar recipes to current ones.
11. Keep each recipe short (2–5 words).
12. Suggest only realistic, common recipes, that are not very eclectic or obscure,

that people would actually make.

13. CRITICAL: Avoid recipes that use ingredients, methods, or actions the human has explicitly said they don't want.

14. CRITICAL: ONLY suggest recipes that are directly supported by the human's actual actions.

15. If no new valid recipes, return an empty list.

16. Output ONLY SINGLE LIST inside <answer> tags.

User prompt

Preferences: *prefs_all*.

DOES NOT WANT: *negative_prefs_all*.

Action History: *acts_all*.

DISLIKED ACTIONS: *negative_actions_all*.

Intent Summary: *intent_summary*.

Current Proposed Recipes: *goals_str*.

Think step-by-step, then close your thought with </think>.

Within the <answer></answer> tags, produce the following:

Goals: Each goal recipe should be a concrete food item and not a description.

Ideally, a goal recipe should be several simple words, eg: fruit parfait, vegetable stew, chocolate banana smoothie. Output a python list of strings, where each string depicts a goal recipe the person could be currently making based on the preference and action history that is not currently in the list of proposed recipes. Do not suggest goals that are overly similar to existing goals. If no new goals are suggested, output an empty list.

CRITICAL VALIDATION: Before suggesting any recipe, verify that the human's actions actually support making it.

IMPORTANT: You MUST output your response in this exact format: <answer>["recipe name 1", "recipe name 2"]</answer>

Few Shot Examples

Example A (Parfait/Fruit Salad):

Preferences: dairy-friendly

DOES NOT WANT: spicy food, hot dishes

Actions: *get_bowl*, *get_strawberry*, *get_yogurt*

DISLIKED ACTIONS: *turn_on_stove*, *use_spicy_ingredients*

Intent: quick cold snack

Current Recipes: fruit salad

Available: Ingredients=[strawberry, banana, yogurt, granola, nuts]; Containers=[glass, bowl]; Appliances=[blender, pot, fridge]

<answer>["fruit parfait", "strawberry yogurt bowl"]</answer>

Example B (Pasta/Stew):

Preferences: vegetarian

DOES NOT WANT: meat, fish, dairy

Actions: *get_pot*, *add_tomato*, *add_pasta*, *add_cheese*

DISLIKED ACTIONS: *get_meat*, *use_dairy_products*

Intent: warm savory dish

Current Recipes: veggie stew

Available: Ingredients=[pasta, tomato, onion, kale, cheese]; Containers=[bowl]; Appliances=[pot, blender]

<answer>["tomato cheese pasta", "veggie stew"]</answer>

Example C (Negative/Correction):

Preferences: sweet breakfast

Actions: *get_bowl*, *get_milk*, *get_egg*

Intent: morning meal

Current Recipes: fruit parfait, oatmeal

Available: Ingredients=[milk, egg, strawberry, flour]; Containers=[bowl]; Appliances=[blender, pot]

<invalid>["pancakes", "waffles", "banana bread", "cookie", "cake", "muffin", "chocolate brownie"]</invalid>

Reason: Requires oven or skillet, which are unavailable.

<answer>["banana smoothie", "oatmeal with fruit"]</answer>

Goal Likelihood Prompt

GOAL

Evaluate how likely each potential goal is based on the action and utterance history.

HISTORY

Actions:

{action_entries}

Utterances:

{utterance_entries}

POTENTIAL GOALS

{goal_entries}

EVALUATION CRITERIA AND SCORING RUBRIC

Criteria: Goal Likelihood, Evidence from Actions, Evidence from Utterances

Rubric: 1: Very Unlikely, 2: Unlikely, 3: Possible, 4: Likely, 5: Very Likely

INSTRUCTIONS

For each goal, assign a numeric score between 1–5 based on the evidence from actions and utterances.

Be conservative while scoring.

Output in the format:

Goal: <goal> : <score>

One goal per line, no additional text.

Attractor Field Based Scoring: Action→Recipe Scoring Prompt

GOAL

Evaluate each cooking action mapped to its recipe according to the criteria and rubric.

Consider the interaction history when making your evaluation.

{history_section}

MAPPINGS TO EVALUATE (YOU MUST RATE ALL OF THESE)

{entries_text}

#EVALUATION CRITERIA AND SCORING RUBRIC

Criteria: {evaluation_criteria}

Rubric: {rubric}

#CRITICAL INSTRUCTIONS

You MUST rate EVERY single mapping listed above. Do not skip any pairs.

For each mapping, assign a numeric score between the rubric bounds. Be

conservative while scoring.

Evaluate each pair independently based on the evaluation criteria and the interaction history.

Consider how well each action fits with the previous actions taken by the human.

REQUIRED OUTPUT FORMAT

Output EXACTLY one line for each mapping in this format:

Action: <action> -> Recipe: <recipe> : <score>

IMPORTANT: You must output $\{(pairs)\}$ lines total, one for each mapping above.

Do not add any explanations, comments, or additional text.

B.2 Setup, Recipes and Preferences used in the Collaborative Cooking Task

Table B.1: Recipes, ingredients, and appliances/utensils.

Recipe	Ingredients Used	Appliances & Utensils
Pastas		
tomato cheese pasta	tomato, pasta, cheese, oil	blender, stove, pan, bowl, mixing spoon, eating spoon
veggie red sauce pasta	broccoli, carrot, cheese, oil, onion, pasta, tomato	blender, bowl, eating spoon, mixing spoon, pan, stove
chicken red sauce pasta	tomato, pasta, cheese, oil, chicken	blender, bowl, eating spoon, mixing spoon, pan, stove

Recipe	Ingredients Used	Appliances & Utensils
chicken red sauce veggie pasta	broccoli, carrot, cheese, oil, onion, pasta, tomato, chicken	blender, bowl, eating spoon, mixing spoon, pan, stove
veggie white sauce pasta	broccoli, carrot, cheese, oil, onion, pasta	bowl, eating spoon, mixing spoon, pan, stove
chicken white sauce pasta	pasta, cheese, oil, chicken	bowl, eating spoon, mixing spoon, pan, stove
chicken white sauce veggie pasta	broccoli, carrot, cheese, oil, onion, pasta, chicken	bowl, eating spoon, mixing spoon, pan, stove
Stews		
avocado cheese stew	avocado, tomato, cucumber, lettuce, onion, cheese, salt, water	stove, pan, bowl, cup, mixing spoon, eating spoon
chicken stew	chicken, cheese, salt, water	stove, pan, bowl, cup, mixing spoon, eating spoon
chicken veggie stew	chicken, avocado, tomato, cucumber, lettuce, onion, cheese, salt, water	stove, pan, bowl, cup, mixing spoon, eating spoon
Oatmeals		
fruity chocolate oatmeal	oatmeal, strawberry, blueberry, banana, chocolate chips, salt, water	stove, pan, bowl, cup, mixing spoon, eating spoon
chocolate oatmeal	oatmeal, chocolate chips, salt, water	stove, pan, bowl, cup, mixing spoon, eating spoon
fruity oatmeal	oatmeal, strawberry, blueberry, banana, salt, water	stove, pan, bowl, cup, mixing spoon, eating spoon

Recipe	Ingredients Used	Appliances & Utensils
plain oatmeal	oatmeal, salt, water	stove, pan, bowl, cup, mixing spoon, eating spoon
peanut butter chocolate oatmeal	oatmeal, peanut butter, chocolate chips, salt, water	stove, pan, bowl, cup, mixing spoon, eating spoon
Smoothies		
chocolate fruit smoothie	apple, banana, blueberry, strawberry, chocolate chips, sugar, water, yogurt	blender, cup, glass
fruit smoothie	apple, banana, blueberry, strawberry, sugar, water, yogurt	blender, cup, glass
peanut butter oatmeal smoothie	oatmeal, peanut butter, sugar, water, yogurt	blender, cup, glass
Parfaits		
fruit parfait	blueberry, apple, banana, oatmeal, strawberry, sugar, yogurt	eating spoon, glass
fruit chocolate parfait	blueberry, apple, banana, chocolate chips, oatmeal, strawberry, sugar, yogurt	eating spoon, glass
fruit nut chocolate parfait	blueberry, chocolate chips, apple, banana, nuts, oatmeal, strawberry, sugar, yogurt	eating spoon, glass
yogurt parfait	oatmeal, sugar, yogurt	eating spoon, glass
Salads		

Recipe	Ingredients Used	Appliances & Utensils
avocado salad	avocado, cucumber, kale, lettuce, onion, salt, tomato	bowl, mixing spoon, eating spoon
pasta salad	cucumber, kale, lettuce, onion, pasta, salt, tomato	bowl, mixing spoon, eating spoon
veggie salad	cucumber, kale, lettuce, onion, salt, tomato	bowl, mixing spoon, eating spoon
fruit salad	apple, banana, blueberry, strawberry	bowl, mixing spoon, eating spoon
grain salad	cucumber, kale, lettuce, onion, salt, tomato, rice, nuts	bowl, mixing spoon, eating spoon
avocado grain salad	cucumber, kale, lettuce, onion, salt, tomato, rice, avocado, nuts	bowl, mixing spoon, eating spoon
chicken salad	cucumber, kale, lettuce, onion, salt, tomato, rice, avocado, nuts, chicken	bowl, mixing spoon, eating spoon

Table B.2: Preference-to-Recipe Mapping

Preference	Recipes that Match	Reasoning
avocado-centric	avocado cheese stew; avocado grain salad; avocado salad	contains avocado

Preference	Recipes that Match	Reasoning
blended (smooth)	chocolate fruit smoothie; fruit smoothie; peanut butter oatmeal smoothie	prepared in blender and consumed as a smooth drink
breakfast	chocolate fruit smoothie; chocolate oat- meal; fruit chocolate parfait; fruit nut chocolate parfait; fruit parfait; fruit salad; fruit smoothie; fruity chocolate oatmeal; fruity oatmeal; peanut butter chocolate oatmeal; peanut butter oatmeal smoothie; yogurt parfait	common breakfast for- mats (oatmeal, smoothie, parfait or fruit salad)
carb-dense	avocado grain salad; cheese pasta; chicken red sauce pasta; chicken red sauce veggie pasta; chicken white sauce pasta; chicken white sauce veggie pasta; fruit chocolate parfait; fruit nut chocolate parfait; fruit parfait; fruity chocolate oatmeal; fruity oatmeal; grain salad; pasta salad; peanut butter chocolate oatmeal; peanut butter oatmeal smoothie; tomato cheese pasta; veggie red sauce pasta; veggie white sauce pasta	includes pasta, rice, oats or added sugar

Preference	Recipes that Match	Reasoning
cheese-heavy	avocado cheese stew; cheese pasta; chicken red sauce pasta; chicken red sauce veggie pasta; chicken stew; chicken veggie stew; chicken white sauce pasta; chicken white sauce veggie pasta; tomato cheese pasta; veggie red sauce pasta; veggie white sauce pasta	cheese is a primary ingredient
chilled	avocado grain salad; avocado salad; chicken salad; chocolate fruit smoothie; fruit chocolate parfait; fruit nut chocolate parfait; fruit parfait; fruit salad; fruit smoothie; grain salad; pasta salad; peanut butter oatmeal smoothie; veggie salad	served cold and does not require stove heating
chocolatey	chocolate fruit smoothie; chocolate oatmeal; fruit chocolate parfait; fruit nut chocolate parfait; fruity chocolate oatmeal; peanut butter chocolate oatmeal	contains chocolate chips
creamy	avocado cheese stew; chicken white sauce pasta; chicken white sauce veggie pasta; veggie white sauce pasta	thickened with cheese/cream sauce on stove
crisp	avocado grain salad; avocado salad; chicken salad; fruit salad; grain salad; pasta salad; veggie salad	raw salads with crunchy vegetables or fruit

Preference	Recipes that Match	Reasoning
dairy-free	avocado grain salad; avocado salad; chicken salad; grain salad; pasta salad; veggie salad	contains no cheese or yogurt
dairy-rich	avocado cheese stew; cheese pasta; chicken red sauce pasta; chicken red sauce veggie pasta; chicken white sauce pasta; chicken white sauce veggie pasta; fruit chocolate parfait; fruit nut chocolate parfait; fruit parfait; tomato cheese pasta; veggie white sauce pasta	includes cheese and/or yogurt
fiber-rich	avocado grain salad; avocado salad; chicken veggie stew; fruit chocolate parfait; fruit nut chocolate parfait; fruit parfait; fruit salad; fruit smoothie; fruity chocolate oatmeal; fruity oatmeal; grain salad; pasta salad; peanut butter chocolate oatmeal; peanut butter oatmeal smoothie; veggie salad; veggie red sauce pasta	provides high-fiber foods (oats, berries, leafy greens, broccoli, etc.)
fruity	chocolate fruit smoothie; fruit chocolate parfait; fruit nut chocolate parfait; fruit parfait; fruit salad; fruit smoothie; fruity chocolate oatmeal; fruity oatmeal	contains apple, banana, strawberry or blueberry

Preference	Recipes that Match	Reasoning
gluten-free	avocado cheese stew; avocado grain salad; avocado salad; chicken salad; chicken stew; chicken veggie stew; chocolate fruit smoothie; chocolate oatmeal; fruit chocolate parfait; fruit nut chocolate parfait; fruit parfait; fruit salad; fruit smoothie; fruity chocolate oatmeal; fruity oatmeal; grain salad; peanut butter chocolate oatmeal; peanut butter oatmeal smoothie	free of wheat pasta
healthy-fats	avocado cheese stew; avocado grain salad; avocado salad; chicken salad; peanut butter chocolate oatmeal; peanut butter oatmeal smoothie	features avocado, peanut butter or nuts
high-protein	chicken red sauce pasta; chicken red sauce veggie pasta; chicken stew; chicken veggie stew; chicken white sauce pasta; chicken white sauce veggie pasta; chicken salad; peanut butter chocolate oatmeal; peanut butter oatmeal smoothie	provides chicken, peanut butter, yogurt or abundant cheese
keto-leaning	avocado cheese stew; avocado salad; chicken stew; chicken veggie stew; veggie salad	no grains, rice, oats, fruit, sugar or chocolate chips

Preference	Recipes that Match	Reasoning
layered	fruit chocolate parfait; fruit nut chocolate parfait; fruit parfait; yogurt parfait	built in visible layers
low-carb	avocado cheese stew; avocado grain salad; avocado salad; chicken salad; chicken stew; chicken veggie stew; veggie salad	excludes grains, rice, oats, fruit, sugar, chocolate chips
low-fat	fruit salad; fruity chocolate oatmeal; fruity oatmeal; pasta salad; veggie salad	little or no cheese, yogurt, avocado, peanut butter, nuts or added oil
low-sodium	chocolate fruit smoothie; fruit chocolate parfait; fruit nut chocolate parfait; fruit parfait; fruit salad; fruit smoothie	salt not listed among ingredients
mineral-rich	avocado grain salad; avocado salad; chicken salad; fruit salad; grain salad; veggie salad	contains kale, lettuce, avocado or nuts
nutty	avocado grain salad; chicken salad; fruit nut chocolate parfait; grain salad; peanut butter chocolate oatmeal; peanut butter oatmeal smoothie	includes nuts or peanut butter
probiotic-rich	fruit chocolate parfait; fruit nut chocolate parfait; fruit parfait; fruity chocolate oatmeal; fruity oatmeal; fruit smoothie; yogurt parfait	contains yogurt (live cultures)

Preference	Recipes that Match	Reasoning
raw-vegetables	avocado grain salad; avocado salad; chicken salad; grain salad; pasta salad; veggie salad	all vegetables are un-cooked
vegetable-loaded	avocado cheese stew; avocado grain salad; avocado salad; chicken salad; chicken red sauce veggie pasta; chicken veggie stew; grain salad; pasta salad; veggie salad; veggie red sauce pasta; veggie white sauce pasta	at least three different vegetables
vitamin-rich	avocado grain salad; avocado salad; chicken red sauce veggie pasta; fruit chocolate parfait; fruit nut chocolate parfait; fruit parfait; fruit salad; fruit smoothie; grain salad; veggie salad; veggie red sauce pasta	includes leafy greens or fruit
wheat-based	cheese pasta; chicken red sauce pasta; chicken red sauce veggie pasta; chicken white sauce pasta; chicken white sauce veggie pasta; pasta salad; tomato cheese pasta; veggie red sauce pasta; veggie white sauce pasta	uses wheat pasta

Preference	Recipes that Match	Reasoning
vegetarian	avocado cheese stew; avocado grain salad; avocado salad; cheese pasta; chocolate fruit smoothie; chocolate oatmeal; fruit chocolate parfait; fruit nut chocolate parfait; fruit parfait; fruit salad; fruit smoothie; fruity chocolate oatmeal; fruity oatmeal; grain salad; pasta salad; peanut butter chocolate oatmeal; peanut butter oatmeal smoothie; tomato cheese pasta; veggie red sauce pasta; veggie white sauce pasta; veggie salad	contains no chicken
non-vegetarian	chicken red sauce pasta; chicken red sauce veggie pasta; chicken stew; chicken veggie stew; chicken white sauce pasta; chicken white sauce veggie pasta; chicken salad	contains chicken
sweet	chocolate fruit smoothie; chocolate oatmeal; fruit chocolate parfait; fruit nut chocolate parfait; fruit parfait; fruit salad; fruit smoothie; fruity chocolate oatmeal; fruity oatmeal; peanut butter chocolate oatmeal; peanut butter oatmeal smoothie	dominated by sweet ingredients (sugar, chocolate chips or fruit)

Preference	Recipes that Match	Reasoning
savory	avocado cheese stew; avocado grain salad; avocado salad; cheese pasta; chicken red sauce pasta; chicken red sauce veggie pasta; chicken stew; chicken veggie stew; chicken white sauce pasta; chicken white sauce veggie pasta; chicken salad; grain salad; pasta salad; tomato cheese pasta; veggie red sauce pasta; veggie white sauce pasta; veggie salad	primary flavour is savory (cheese/veg/protein)
dinner meal	avocado cheese stew; avocado grain salad; avocado salad; cheese pasta; chicken red sauce pasta; chicken red sauce veggie pasta; chicken stew; chicken veggie stew; chicken white sauce pasta; chicken white sauce veggie pasta; chicken salad; grain salad; pasta salad; tomato cheese pasta; veggie red sauce pasta; veggie white sauce pasta; veggie salad	hearty savory dishes suitable for dinner

Preference	Recipes that Match	Reasoning
quick meal	avocado grain salad; avocado salad; chicken salad; chocolate fruit smoothie; fruit chocolate parfait; fruit nut chocolate parfait; fruit parfait; fruit salad; fruit smoothie; grain salad; pasta salad; peanut butter oatmeal smoothie; veggie salad; yogurt parfait	ready fast, no stove cooking
vegetable-averse	cheese pasta; chicken stew; chicken white sauce pasta; chocolate fruit smoothie; chocolate oatmeal; fruit chocolate parfait; fruit nut chocolate parfait; fruit parfait; fruit salad; fruit smoothie; fruity chocolate oatmeal; fruity oatmeal; peanut butter chocolate oatmeal; peanut butter oatmeal smoothie	contains none of the main vegetables (broccoli, carrot, onion, cucumber, lettuce, kale, tomato)

Preference	Recipes that Match	Reasoning
kid-friendly	avocado cheese stew; cheese pasta; chicken red sauce pasta; chicken red sauce veggie pasta; chicken stew; chicken veggie stew; chicken white sauce pasta; chicken white sauce veggie pasta; chocolate fruit smoothie; chocolate oatmeal; fruit chocolate parfait; fruit nut chocolate parfait; fruit parfait; fruit salad; fruit smoothie; fruity chocolate oatmeal; fruity oatmeal; peanut butter chocolate oatmeal; peanut butter oatmeal smoothie; tomato cheese pasta; veggie red sauce pasta; veggie white sauce pasta; yogurt parfait	mild, cheesy or sweet, no strong veg chunks
nut-free	avocado cheese stew; avocado salad; cheese pasta; chicken red sauce pasta; chicken red sauce veggie pasta; chicken stew; chicken veggie stew; chicken white sauce pasta; chicken white sauce veggie pasta; chocolate fruit smoothie; chocolate oatmeal; fruit chocolate parfait; fruit parfait; fruit salad; fruit smoothie; fruity chocolate oatmeal; fruity oatmeal; pasta salad; tomato cheese pasta; veggie red sauce pasta; veggie white sauce pasta; veggie salad	free of nuts and peanut butter

Preference	Recipes that Match	Reasoning
plain-flavor	cheese pasta; chicken red sauce pasta; chicken stew; chicken white sauce pasta; tomato cheese pasta	5 or fewer ingredients, mild seasoning
soft-texture	avocado cheese stew; chicken stew; chicken veggie stew; chocolate fruit smoothie; fruit chocolate parfait; fruit nut chocolate parfait; fruit parfait; fruity chocolate oatmeal; fruity oatmeal; peanut butter chocolate oatmeal; peanut butter oatmeal smoothie; plain oatmeal; fruit smoothie	spoon-soft or smooth
fruit-only	chocolate fruit smoothie; fruit chocolate parfait; fruit parfait; fruit salad; fruit smoothie; fruity chocolate oatmeal; fruity oatmeal	includes fruit but no vegetables or nuts
no-cook	avocado grain salad; avocado salad; chicken salad; chocolate fruit smoothie; fruit chocolate parfait; fruit nut chocolate parfait; fruit parfait; fruit salad; fruit smoothie; grain salad; pasta salad; peanut butter oatmeal smoothie; veggie salad	prepared entirely without stove

Preference	Recipes that Match	Reasoning
hot-dish	avocado cheese stew; cheese pasta; chicken red sauce pasta; chicken red sauce veggie pasta; chicken stew; chicken veggie stew; chicken white sauce pasta; chicken white sauce veggie pasta; fruity chocolate oatmeal; fruity oatmeal; peanut butter chocolate oatmeal; tomato cheese pasta; veggie red sauce pasta; veggie white sauce pasta	requires stove cooking
hydrating	avocado cheese stew; avocado grain salad; avocado salad; chicken salad; chicken stew; chicken veggie stew; chocolate fruit smoothie; fruit salad; fruit smoothie; grain salad; pasta salad; veggie salad	high water content or liquid serving
portable	avocado grain salad; avocado salad; chicken salad; chocolate fruit smoothie; fruit chocolate parfait; fruit nut chocolate parfait; fruit parfait; fruit smoothie; grain salad; pasta salad; peanut butter oatmeal smoothie; veggie salad; yogurt parfait	travels well, eaten cold/-room temp with minimal mess
one-pot meal	avocado cheese stew; chicken stew; chicken veggie stew; fruity chocolate oatmeal; fruity oatmeal; peanut butter chocolate oatmeal	full dish finishes in one pan or pot

Preference	Recipes that Match	Reasoning
meal-prep friendly	avocado cheese stew; avocado grain salad; avocado salad; cheese pasta; chicken red sauce pasta; chicken red sauce veggie pasta; chicken stew; chicken veggie stew; chicken white sauce pasta; chicken white sauce veggie pasta; chicken salad; fruity chocolate oatmeal; fruity oatmeal; grain salad; pasta salad; peanut butter chocolate oatmeal; tomato cheese pasta; veggie red sauce pasta; veggie salad	keeps quality ≥ 2 days; reheats or serves cold well
low-calorie	fruit salad; fruit smoothie; veggie salad	≤ 250 kcal per typical serving
high-calorie	avocado cheese stew; cheese pasta; chicken red sauce pasta; chicken red sauce veggie pasta; chicken white sauce pasta; chicken white sauce veggie pasta; fruit nut chocolate parfait; peanut butter chocolate oatmeal; peanut butter oatmeal smoothie; tomato cheese pasta; veggie red sauce pasta; veggie white sauce pasta	≥ 400 kcal per serving

Preference	Recipes that Match	Reasoning
comfort-food	avocado cheese stew; cheese pasta; chicken red sauce pasta; chicken red sauce veggie pasta; chicken stew; chicken veggie stew; chicken white sauce pasta; chicken white sauce veggie pasta; fruity chocolate oatmeal; fruity oatmeal; peanut butter chocolate oatmeal; tomato cheese pasta; veggie red sauce pasta; veggie white sauce pasta	warm hearty pas- tas/stews or sweet oat bowls
vegan	avocado grain salad; avocado salad; fruit salad; fruity oatmeal; grain salad; pasta salad; veggie salad	contain no meat, dairy, or eggs

B.3 Implementation Details and Additional Statistics

Table B.3: Hyperparameters for Goal-Prediction Modules.

Parameter	Value	Notes
Data Handling		
Train/test split	test_size=0.2, random_state=42	Split per-goal sequence before dataset creation
Dataloader batch size	8	Used for both train and test loaders
Collate padding token	0	Pads variable-length sequences in batching
Random-Forest Training		
Estimators	200 trees	Number of trees in the forest baseline
Random seed	10	Ensures reproducible initialization
Min samples per leaf	2	Leaf-size constraint
Min samples per split	10	Minimum samples to split internal nodes
Goal-State Inference		
Filter strength	0.1	Smoothing for filtered transformer predictions
Temperature reuse	1.5	Global temperature scaling during inference

Table B.4: Hyperparameters for MPC Tree Search and Action Filtering.

Parameter	Value	Notes
Global Search Limits		
Tree depth (DEPTH)	2	Lookahead horizon for MPC expansion
N-gram horizon (N_GRAM)	2	Sequence length for heuristic pruning
Expansion width (TOP_K)	5	Candidate actions kept per node
Goal filter strength	0.5	Weight for goal-probability pruning
Question Scheduling		
Minimum question cost	0.01	Lower bound on query utility
Maximum question cost	1	Upper bound on penalty for asking
Question interval	5	Steps between successive queries

Table B.5: LLM Interface Hyperparameters and Defaults.

Parameter	Value	Notes
Model Selection & API Behavior		
Small-model ID	phi4-mini:latest	Default local model for fast runs
Large-model ID	gemini-2.5-flash	Remote model for stronger reasoning
Default model	SMALL_MODEL	Used unless overridden
Default temperature	0.0	Deterministic completions
Default top-p	0.0	Ensures no sampling
Max tokens (yes/no)	10	For binary responses
Max tokens (lists)	48	For short enumerations
Max tokens (text)	1000	For longer explanations

Bibliography

- [1] P. Abbeel and A. Y. Ng. Apprenticeship learning via inverse reinforcement learning. In *Proceedings of the twenty-first international conference on Machine learning*, page 1, 2004.
- [2] A. Abouelenin, A. Ashfaq, A. Atkinson, H. Awadalla, N. Bach, J. Bao, A. Benhaim, M. Cai, V. Chaudhary, C. Chen, et al. Phi-4-mini technical report: Compact yet powerful multimodal language models via mixture-of-loras. *arXiv preprint arXiv:2503.01743*, 2025.
- [3] T. Adamson, D. Ghose, S. C. Yasuda, L. J. S. Shepard, M. A. Lewkowicz, J. Duan, and B. Scassellati. Why we should build robots that both teach and learn. In *Proceedings of the 2021 ACM/IEEE International Conference on Human-Robot Interaction*, pages 187–196, 2021.
- [4] C. Aeronautiques, A. Howe, C. Knoblock, I. D. McDermott, A. Ram, M. Veloso, D. Weld, D. W. Sri, A. Barrett, D. Christianson, et al. Pddl| the planning domain definition language. *Technical Report, Tech. Rep.*, 1998.
- [5] P. Agrawal. The task specification problem. In *Conference on Robot Learning*, pages 1745–1751. PMLR, 2022.
- [6] M. Ahn, A. Brohan, N. Brown, Y. Chebotar, O. Cortes, B. David, C. Finn, C. Fu, K. Gopalakrishnan, K. Hausman, et al. Do as i can, not as i say:

- Grounding language in robotic affordances. *arXiv preprint arXiv:2204.01691*, 2022.
- [7] M. Ahn, M. Gonzalez Arenas, M. Bennice, N. Brown, C. Chan, B. David, A. Francis, G. Gonzalez, R. Hessmer, T. Jackson, N. J. Joshi, D. Lam, T.-W. E. Lee, A. Luong, S. Maddineni, H. Patel, J. Peralta, J. Quiambao, D. Reyes, R. M. J. Ruano, D. Sadigh, P. Sanketi, L. Takayama, P. Vodenski, and F. Xia. Vader: Visual affordance detection and error recovery for multi-robot human collaboration. *arXiv preprint arXiv:2405.16021*, May 2024.
- [8] I. n. Alonso, A. Sabater, D. Ferstl, L. Montesano, and A. C. Murillo. Semi-supervised semantic segmentation with pixel-level contrastive learning from a class-wise memory bank. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 8219–8228, October 2021.
- [9] L. Amado, R. Mirsky, and F. Meneguzzi. Goal recognition as reinforcement learning. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 36, pages 9644–9651, 2022.
- [10] L. Amado, S. P. Shainkopf, R. F. Pereira, R. Mirsky, and F. Meneguzzi. A survey on model-free goal recognition. In *IJCAI 2024: 33rd International Joint Conference on Artificial Intelligence*, 2024.
- [11] C. Amato and A. Baisero. Active goal recognition. *arXiv preprint arXiv:1909.11173*, 2019.
- [12] Z. Ankner, C. Blakeney, K. Sreenivasan, M. Marion, M. L. Leavitt, and M. Paul. Perplexed by perplexity: Perplexity-based pruning with small reference models. In *ICLR 2024 Workshop on Mathematical and Empirical Understanding of Foundation Models*, 2024.

- [13] M. Arnold, M. Goldschmitt, and T. Rigotti. Dealing with information overload: a comprehensive review. *Frontiers in Psychology*, 14:1122200, 2023.
- [14] R. M. Aronson and H. Admoni. Gaze complements control input for goal prediction during assisted teleoperation. In *Robotics science and systems*, 2022.
- [15] R. M. Aronson, N. Almutlak, and H. Admoni. Inferring goals with gaze during teleoperated manipulation. In *2021 IEEE/RSJ international conference on intelligent robots and systems (IROS)*, pages 7307–7314. IEEE, 2021.
- [16] A. Athalye, N. Kumar, T. Silver, Y. Liang, J. Wang, T. Lozano-Pérez, and L. P. Kaelbling. From pixels to predicates: Learning symbolic world models via pretrained vision-language models. *arXiv preprint arXiv:2501.00296*, 2024.
- [17] K. Ayush, B. Uzkent, C. Meng, K. Tanmay, M. Burke, D. Lobell, and S. Ermon. Geography-aware self-supervised learning. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 10181–10190, 2021.
- [18] A. Bajcsy, D. P. Losey, M. K. O’Malley, and A. D. Dragan. Learning from physical human corrections, one feature at a time. In *Proceedings of the 2018 ACM/IEEE International Conference on Human-Robot Interaction*, pages 141–149, 2018.
- [19] A. Bajcsy, D. P. Losey, M. K. O’Malley, and A. D. Dragan. Learning robot objectives from physical human interaction. In S. Levine, V. Vanhoucke, and K. Goldberg, editors, *Proceedings of the 1st Annual Conference on Robot Learning*, volume 78 of *Proceedings of Machine Learning Research*, pages 217–226. PMLR, 13–15 Nov 2017.
- [20] B. Baker, I. Kanitscheider, T. Markov, Y. Wu, G. Powell, B. McGrew, and I. Mordatch. Emergent tool use from multi-agent autotutorials. *arXiv preprint arXiv:1909.07528*, 2019.

- [21] C. L. Baker, R. Saxe, and J. B. Tenenbaum. Action understanding as inverse planning. *Cognition*, 113(3):329–349, 2009.
- [22] C. L. Baker, R. Saxe, and J. B. Tenenbaum. Action understanding as inverse planning. *Cognition*, 113(3):329–349, 2009.
- [23] T. Bandyopadhyay, K. S. Won, E. Frazzoli, D. Hsu, W. S. Lee, and D. Rus. Intention-aware motion planning. In *Algorithmic Foundations of Robotics X: Proceedings of the Tenth Workshop on the Algorithmic Foundations of Robotics*, pages 475–491. Springer, 2013.
- [24] S. Bansal, J. Xu, A. M. Howard, and C. Isbell. A bayesian framework for nash equilibrium inference in human-robot parallel play. *CoRR*, abs/2006.05729, 2020.
- [25] K. Baraka, I. Idrees, T. Kessler Faulkner, E. Biyik, S. Booth, M. Chetouani, D. H. Grollman, A. Saran, E. Senft, S. Tulli, A.-L. Vollmer, A. Andriella, H. Beierling, T. Horter, J. Kober, I. Sheidlower, M. E. Taylor, S. van Waveren, and X. Xiao. Human-interactive robot learning: Definition, challenges, and recommendations. *J. Hum.-Robot Interact.*, Dec. 2025. Just Accepted.
- [26] Y. Bastanlar and S. Orhan. Self-supervised contrastive representation learning in computer vision. 2022.
- [27] S. Basu, S. Tatlidil, M. H. Kim, S. Sloman, and R. I. Bahar. Using causal information to enable more efficient robot operation. In *Proceedings of the 61st ACM/IEEE Design Automation Conference (DAC)*, pages 1–4, June 2024.
- [28] S. Basu, S. Tatlidil, M. H. Kim, T. Tran, S. Saxena, T. Williams, S. Sloman, and R. I. Bahar. Robot planning under uncertainty for object assembly and troubleshooting using human causal models. In *Proceedings of the 2025 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2025.

- [29] A. Belsare, Z. Karimi, C. Mattson, and D. S. Brown. Toward zero-shot user intent recognition in shared autonomy. *arXiv preprint arXiv:2501.08389*, 2025.
- [30] F. Bisson, F. Kabanza, A. R. Benaskeur, and H. Irandoust. Provoking opponents to facilitate the recognition of their intentions. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 25, pages 1762–1763, 2011.
- [31] E. Bıyık, M. Palan, N. C. Landolfi, D. P. Losey, and D. Sadigh. Asking easy questions: A user-friendly approach to active reward learning. *arXiv preprint arXiv:1910.04365*, 2019.
- [32] J. Blankenburg, M. Zagainova, S. M. Simmons, G. Talavera, M. Nicolescu, and D. Feil-Seifer. Human-robot collaboration and dialogue for fault recovery on hierarchical tasks. In *International Conference on Social Robotics (ICSR)*, CO, October 2020.
- [33] A. Bobu, A. Peng, P. Agrawal, J. Shah, and A. D. Dragan. Aligning robot and human representations. *arXiv preprint arXiv:2302.01928*, 2023.
- [34] A. Bobu, A. Peng, P. Agrawal, J. A. Shah, and A. D. Dragan. Aligning human and robot representations. In *Proceedings of the 2024 ACM/IEEE International Conference on Human-Robot Interaction*, pages 42–54, 2024.
- [35] D. Bolya, C. Zhou, F. Xiao, and Y. J. Lee. Yolact: Real-time instance segmentation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 9157–9166, 2019.
- [36] T. Bonjour, M. Haliem, A. Alsalem, S. Thomas, H. Li, V. Aggarwal, M. Kejriwal, and B. Bhargava. Decision making in monopoly using a hybrid deep reinforcement learning approach. *IEEE Transactions on Emerging Topics in Computational Intelligence*, 6(6), 2022.

- [37] J. Brawer. *Fusing Symbolic and Subsymbolic Approaches for Natural and Effective Human-Robot Collaboration*. PhD thesis, Yale University, 2023.
- [38] J. Brawer, D. Ghose, K. Candon, M. Qin, A. Roncone, M. Vázquez, and B. Scassellati. Interactive policy shaping for human-robot collaboration with transparent matrix overlays. In *In Proceedings of the 2023 ACM/IEEE International Conference on Human-Robot Interaction (HRI '23)*, pages 525–533. Association for Computing Machinery, New York, NY, USA, 2023.
- [39] J. Brawer, O. Mangin, A. Roncone, S. Widder, and B. Scassellati. Situated human–robot collaboration: predicting intent from grounded natural language. In *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 827–833. IEEE, 2018.
- [40] T. V. Bui, T. Mai, and T. H. Nguyen. Imitating opponent to win: Adversarial policy imitation learning in two-player competitive games. In *Proceedings of the 2023 International Conference on Autonomous Agents and Multiagent Systems*, pages 1285–1293, 2023.
- [41] K. Candon, Q. Zhang, A. Lew, H. Claire, L. Qian, A. Quarles, C. Sarkar, and M. Vázquez. Learning human preferences over a human-robot collaboration based on explicit and implicit human feedback. 2026.
- [42] K. Candon, H. Zhou, S. Gillet, and M. Vázquez. Verbally soliciting human feedback in continuous human-robot collaboration: Effects of the framing and timing of reminders. In *Proceedings of the 2023 ACM/IEEE International Conference on Human-Robot Interaction*, pages 290–300, 2023.
- [43] M. Caron, I. Misra, J. Mairal, P. Goyal, P. Bojanowski, and A. Joulin. Unsupervised learning of visual features by contrasting cluster assignments. *Advances in Neural Information Processing Systems*, 33:9912–9924, 2020.

- [44] C. M. Carpinella, A. B. Wyman, M. A. Perez, and S. J. Stroessner. The robotic social attributes scale (rosas) development and validation. In *Proceedings of the 2017 ACM/IEEE International Conference on human-robot interaction*, pages 254–262, 2017.
- [45] M. Carroll, R. Shah, M. K. Ho, T. Griffiths, S. Seshia, P. Abbeel, and A. Dragan. On the utility of learning about humans for human-ai coordination. *Advances in neural information processing systems*, 32, 2019.
- [46] M. Caterino, M. Rinaldi, V. Di Pasquale, A. Greco, S. Miranda, and R. Macchiaroli. A human error analysis in human–robot interaction contexts: Evidence from an empirical study. *Machines*, 2023.
- [47] L. Chan, D. Hadfield-Menell, S. Srinivasa, and A. Dragan. The assistive multi-armed bandit. *arXiv preprint arXiv:1901.08654*, 2019.
- [48] M. L. Chang, R. A. Gutierrez, P. Khante, E. S. Short, and A. L. Thomaz. Effects of integrated intent recognition and communication on human-robot collaboration. In *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2018.
- [49] A. Chari, R. Chen, J. Grover, and C. Liu. An optimal control framework for influencing human driving behavior in mixed-autonomy traffic. In *2024 American Control Conference (ACC)*, pages 292–298. IEEE, 2024.
- [50] D. Chen, T. Moutakanni, W. Chung, Y. Bang, Z. Ji, A. Bolourchi, and P. Fung. Planning with reasoning using vision language world model. *arXiv preprint arXiv:2509.02722*, 2025.
- [51] K. Chen, J. Wang, J. Pang, Y. Cao, Y. Xiong, X. Li, S. Sun, W. Feng, Z. Liu, J. Xu, et al. Mmdetection: Open mmlab detection toolbox and benchmark. *arXiv preprint arXiv:1906.07155*, 2019.

- [52] M. Chen, S. Nikolaidis, H. Soh, D. Hsu, and S. Srinivasa. Planning with trust for human–robot collaboration. In *ACM/IEEE International Conference on Human-Robot Interaction*, 2018.
- [53] S. Chen, D. Fried, and U. Topcu. Human-agent cooperation in games under incomplete information through natural language communication. *arXiv preprint arXiv:2405.14173*, 2024.
- [54] T. Chen, S. Kornblith, M. Norouzi, and G. Hinton. A simple framework for contrastive learning of visual representations. In *International conference on machine learning*, pages 1597–1607. PMLR, 2020.
- [55] X. Chen, H. Fan, R. Girshick, and K. He. Improved baselines with momentum contrastive learning. *arXiv preprint arXiv:2003.04297*, 2020.
- [56] X. Chen and K. He. Exploring simple siamese representation learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 15750–15758, 2021.
- [57] S. Chernova and A. L. Thomaz. *Robot learning from human teachers*. Springer Nature, 2022.
- [58] C. Chi, Z. Xu, S. Feng, E. Cousineau, Y. Du, B. Burchfiel, R. Tedrake, and S. Song. Diffusion policy: Visuomotor policy learning via action diffusion. *The International Journal of Robotics Research*, 44(10-11):1684–1704, 2025.
- [59] A. Choi, M. K. Jawed, and J. Joo. Preemptive motion planning for human-to-robot indirect placement handovers. In *2022 International Conference on Robotics and Automation (ICRA)*, pages 4743–4749. IEEE, 2022.
- [60] S. Chopra, R. Hadsell, and Y. LeCun. Learning a similarity metric discriminatively, with application to face verification. In *2005 IEEE Computer Society*

Conference on Computer Vision and Pattern Recognition (CVPR'05), volume 1, pages 539–546. IEEE, 2005.

- [61] H. H. Clark and S. E. Brennan. Grounding in communication. In L. B. Resnick, J. M. Levine, and S. D. Teasley, editors, *Perspectives on Socially Shared Cognition*, pages 127–149. American Psychological Association, 1991.
- [62] M. Colledanchise and P. Ögren. *Behavior trees in robotics and AI: An introduction*. CRC Press, 2018.
- [63] G. Comanici, E. Bieber, M. Schaekermann, I. Pasupat, N. Sachdeva, I. Dhillon, M. Blistein, O. Ram, D. Zhang, E. Rosen, et al. Gemini 2.5: Pushing the frontier with advanced reasoning, multimodality, long context, and next generation agentic capabilities. *arXiv preprint arXiv:2507.06261*, 2025.
- [64] M. Cramer, K. Kellens, and E. Demeester. Probabilistic decision model for adaptive task planning in human-robot collaborative assembly based on designer and operator intents. *IEEE Robotics and Automation Letters*, 6(4):7325–7332, 2021.
- [65] Y. Cui, S. Karamcheti, R. Palleti, N. Shivakumar, P. Liang, and D. Sadigh. No, to the right: Online language corrections for robotic manipulation via shared autonomy. In *Proceedings of the 2023 ACM/IEEE International Conference on Human-Robot Interaction*, pages 93–101, 2023.
- [66] H. Damgacioglu, M. Hornilla, O. Bafail, and N. Celik. Recovering value from single stream material recovery facilities—an outbound contamination analysis in florida. *Waste Management*, 102:804–814, 2020.
- [67] A. Das, S. S. Srinivasa, and S. Chernova. Explainable AI for robot failures: Generating explanations that improve user assistance in fault recovery. In *Pro-*

- ceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 632–639, 2021.
- [68] D. Das, S. Banerjee, and S. Chernova. Explainable AI for robot failures: Generating explanations that improve user assistance in fault recovery. *CoRR*, abs/2101.01625, 2021.
- [69] D. De Lazzari, M. Terreran, G. Giacomuzzo, S. Jain, P. Falco, R. Carli, and D. Romeres. Pace: Proactive assistance in human-robot collaboration through action-completion estimation. In *2025 IEEE International Conference on Robotics and Automation (ICRA)*, pages 6725–6731. IEEE, 2025.
- [70] J. DeCastro, A. Silva, D. Gopinath, E. Sumner, T. M. Balch, L. Dees, and G. Rosman. Dreaming to assist: Learning to align with human objectives for shared control in high-speed racing. In *8th Annual Conference on Robot Learning*.
- [71] D. C. Dennett. *The Intentional Stance*. The MIT Press, Cambridge, MA, 1987.
- [72] S. Desai, D. Ghose, and D. Chakraborty. A survey on data curation for visual contrastive learning: Why crafting effective positive and negative pairs matters. *arXiv preprint arXiv:2502.08134*, 2025.
- [73] R. Dhakal, D. M. Nguyen, T. Silver, X. Xiao, and G. J. Stein. Anticipatory task and motion planning. *arXiv preprint arXiv:2407.13694*, 2024.
- [74] K. Doctor, H. Ghosh, and R. Grupen. Learning intrinsically motivated transition models for autonomous systems. In *2022 IEEE International Conference on Development and Learning (ICDL)*, pages 9–14. IEEE, 2022.
- [75] A. D. Dragan, K. C. Lee, and S. S. Srinivasa. Legibility and predictability of

- robot motion. In *2013 8th ACM/IEEE International Conference on Human-Robot Interaction (HRI)*, pages 301–308. IEEE, 2013.
- [76] D. Dwibedi, Y. Aytar, J. Tompson, P. Sermanet, and A. Zisserman. With a little help from my friends: Nearest-neighbor contrastive learning of visual representations. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 9588–9597, 2021.
- [77] I. El Makrini, M. Omid, F. Fusaro, E. Lamon, A. Ajoudani, and B. Vanderborght. A hierarchical finite-state machine-based task allocation framework for human-robot collaborative assembly tasks. In *2022 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 10238–10244. IEEE, 2022.
- [78] A. Favier, S. Shekhar, and R. Alami. Robust planning for human-robot joint tasks with explicit reasoning on human mental state. *arXiv preprint arXiv:2210.08879*, 2022.
- [79] J. Felip, D. Gonzalez-Aguirre, and L. Nachman. Intuitive & efficient human-robot collaboration via real-time approximate bayesian inference. In *2022 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 3093–3099. IEEE, 2022.
- [80] T. Fitzgerald, P. Koppol, P. Callaghan, R. Q. J. H. Wong, R. Simmons, O. Kroemer, and H. Admoni. Inquire: Interactive querying for user-aware informative reasoning. In *6th Annual Conference on Robot Learning*, 2022.
- [81] flowai. Flow Judge: An Open Small Language Model for LLM System Evaluations | Flow AI — flow-ai.com. <https://www.flow-ai.com/blog/flow-judge>. [Accessed 17-09-2025].

- [82] M. Forbes, M. Chung, M. Cakmak, and R. Rao. Robot programming by demonstration with crowdsourced action fixes. In *Proceedings of the AAAI Conference on Human Computation and Crowdsourcing*, volume 2, pages 67–76, 2014.
- [83] R. G. Freedman, Y. R. Fung, R. Ganchin, and S. Zilberstein. Responsive planning and recognition for closed-loop interaction. *arXiv preprint arXiv:1909.06427*, 2019.
- [84] H. A. Frijns, M. Hirschmanner, B. Sienkiewicz, P. Hönig, B. Indurkha, and M. Vincze. Human-in-the-loop error detection in an object organization task with a social robot. *Frontiers in Robotics and AI*, 11, Apr. 2024. Section: Human-Robot Interaction.
- [85] C. E. Garcia, D. M. Prett, and M. Morari. Model predictive control: Theory and practice—a survey. *Automatica*, 1989.
- [86] D. Ghose. Adapting robot actions to human intentions in dynamic shared environments. In *Companion of the 2025 ACM/IEEE International Conference on Human-Robot Interaction*, 2025.
- [87] D. Ghose, O. Gitelson, R. Jin, G. Abawe, M. Vázquez, and B. Scassellati. I’ve changed my mind: Robots adapting to changing human goals during collaboration. *arXiv preprint arXiv:2511.15914*, 2025.
- [88] D. Ghose, O. Gitelson, R. Jin, G. Abawe, M. Vázquez, and B. Scassellati. I’ve changed my mind: Robots adapting to changing human goals during collaboration. *IEEE Robotics and Automation Letters*, 11(2):1490–1497, 2026.
- [89] D. Ghose, O. Gitelson, and B. Scassellati. Integrating multimodal affective signals for stress detection from audio-visual data. In *Proceedings of the 26th International Conference on Multimodal Interaction*, pages 22–32, 2024.

- [90] D. Ghose, O. Gitelson, M. Vazquez, and B. Scassellati. Open-ended goal inference through actions and language for human-robot collaboration, 2025.
- [91] D. Ghose, O. Gitelson, M. Vazquez, and B. Scassellati. Open-ended goal inference through actions and language for human-robot collaboration. *arXiv preprint arXiv:2512.04453*, 2025.
- [92] D. Ghose, M. Lewkowicz, D. Dong, A. Cheng, T. Doan, E. Adams, M. Vázquez, and B. Scassellati. Planning with critical decision points: Robots that influence humans to infer their strategy. In *2024 33rd IEEE International Conference on Robot and Human Interactive Communication (RO-MAN)*, 2024.
- [93] D. Ghose, M. A. Lewkowicz, K. Gezahegn, J. Lee, T. Adamson, M. Vázquez, and B. Scassellati. Tailoring visual object representations to human requirements: A case study with a recycling robot. In *Conference on Robot Learning*, pages 583–593. PMLR, 2023.
- [94] M. Gienger, D. Ruiken, T. Bates, M. Regaieg, M. MeiBner, J. Kober, P. Seiwald, and A.-C. Hildebrandt. Human-robot cooperative object manipulation with contact changes. In *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 1354–1360. IEEE, 2018.
- [95] O. Gitelson, S. P. Nayak, R. Raha, and A.-K. Schmuck. Maximal adaptation, minimal guidance: Permissive reactive robot task planning with humans in the loop. *arXiv preprint arXiv:2510.12662*, 2025.
- [96] A. Gopnik and H. M. Wellman. Why the child’s theory of mind really is a theory. *Mind & Language*, 7(1-2):145–171, 1992.
- [97] O. C. Görür, B. Rosman, F. Sivrikaya, and S. Albayrak. Fabric: A framework for the design and evaluation of collaborative robots with extended human adaptation. *ACM Transactions on Human-Robot Interaction*, 12(3):1–32, 2023.

- [98] A. Gottardi, M. Terreran, E. Pagello, and E. Menegatti. Had-tamp: Human adaptive task and motion planning for human-robot collaboration in industrial scenario. *Robotics and Autonomous Systems*, page 105318, 2025.
- [99] C. Goubard and Y. Demir. Cooking up trust: Eye gaze and posture for trust-aware action selection in human-robot collaboration. In *Proceedings of the First International Symposium on Trustworthy Autonomous Systems*, pages 1–5, 2023.
- [100] J. Grannen, S. Karamcheti, S. Mirchandani, P. Liang, and D. Sadigh. Vocal sandbox: Continual learning and adaptation for situated human-robot collaboration. *arXiv preprint arXiv:2411.02599*, 2024.
- [101] J. Grannen, S. Karamcheti, B. Wulfe, and D. Sadigh. Provox: Personalization and proactive planning for situated human-robot collaboration. *IEEE Robotics and Automation Letters*, 2025.
- [102] R. A. Grier, A. Bangor, P. Kortum, and S. C. Peres. The system usability scale: Beyond standard usability testing. In *Proceedings of the human factors and ergonomics society annual meeting*, volume 57, pages 187–191. SAGE Publications Sage CA: Los Angeles, CA, 2013.
- [103] J.-B. Grill, F. Strub, F. Altché, C. Tallec, P. Richemond, E. Buchatskaya, C. Doersch, B. Avila Pires, Z. Guo, M. Gheshlaghi Azar, et al. Bootstrap your own latent-a new approach to self-supervised learning. *Advances in Neural Information Processing Systems*, 33:21271–21284, 2020.
- [104] J. Gu, X. Jiang, Z. Shi, H. Tan, X. Zhai, C. Xu, W. Li, Y. Shen, S. Ma, H. Liu, et al. A survey on llm-as-a-judge. *arXiv preprint arXiv:2411.15594*, 2024.
- [105] E. Guerra, S. T. Büttner, and M. Prilla. Understanding task diversity in

- human-robot collaboration: A scoping review. *International Journal of Human-Computer Interaction*, pages 1–21, 2026.
- [106] Y. Guo, P. Yi, X. Wei, et al. Active interaction strategy generation for human-robot collaboration based on trust. *Visual Computing for Industry, Biomedicine, and Art*, 8(1):16, 2025.
- [107] M. Hagenow, E. Senft, R. Radwin, M. Gleicher, B. Mutlu, and M. Zinn. Informing real-time corrections in corrective shared autonomy through expert demonstrations. *IEEE Robotics and Automation Letters*, 6(4):6442–6449, 2021.
- [108] M. Hagenow and J. A. Shah. Realm: Real-time estimates of assistance for learned models in human-robot interaction. *IEEE Robotics and Automation Letters*, 2025.
- [109] K. Haninger, C. Hegeler, and L. Peternel. Model predictive control with gaussian processes for flexible multi-modal physical human robot interaction. In *2022 international conference on robotics and automation (ICRA)*, pages 6948–6955. IEEE, 2022.
- [110] C. Hannum, R. Li, and W. Wang. A trust-assist framework for human-robot co-carry tasks. *Robotics*, 12(2), 2023.
- [111] S. G. Hart. Nasa-task load index (nasa-tlx); 20 years later. In *Proceedings of the human factors and ergonomics society annual meeting*, volume 50, pages 904–908. Sage publications Sage CA: Los Angeles, CA, 2006.
- [112] S. G. Hart and L. E. Staveland. Development of nasa-tlx (task load index): Results of empirical and theoretical research. In P. A. Hancock and N. Meshkati, editors, *Human Mental Workload*, pages 139–183. North-Holland, 1988.

- [113] B. Hayes and B. Scassellati. Effective robot teammate behaviors for supporting sequential manipulation tasks. In *2015 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 6374–6380. IEEE, 2015.
- [114] B. Hayes and B. Scassellati. Autonomously constructing hierarchical task networks for planning and human-robot collaboration. In *2016 IEEE International Conference on Robotics and Automation (ICRA)*, pages 5469–5476. IEEE, 2016.
- [115] K. He, H. Fan, Y. Wu, S. Xie, and R. Girshick. Momentum contrast for unsupervised visual representation learning. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 9729–9738, 2020.
- [116] K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.
- [117] S. Hiramatsu and S. Murata. Deep predictive network for inference and dynamic optimization of task goals during human-robot collaboration. In *2023 International Joint Conference on Neural Networks (IJCNN)*, pages 1–6, 2023.
- [118] S. Hochreiter and J. Schmidhuber. Long short-term memory. *Neural computation*, 9(8):1735–1780, 1997.
- [119] J. Hoegerman, S. Sagheb, B. A. Christie, and D. P. Losey. Aligning learning with communication in shared autonomy. In *2024 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 11530–11536. IEEE, 2024.
- [120] G. Hoffman. Evaluating fluency in human-robot collaboration. *IEEE Transactions on Human-Machine Systems*, 49(3):209–218, 2019.

- [121] G. Hoffman, T. Bhattacharjee, and S. Nikolaidis. Inferring human intent and predicting human action in human–robot collaboration. *Annual Review of Control, Robotics, and Autonomous Systems*, 2024.
- [122] G. Hoffman and C. Breazeal. Cost-based anticipatory action selection for human–robot fluency. *IEEE transactions on robotics*, 23(5):952–961, 2007.
- [123] G. Hoffman and C. Breazeal. Effects of anticipatory action on human-robot teamwork. In *Proceedings of the ACM/IEEE International Conference on Human-Robot Interaction*, pages 1–8, 2007.
- [124] J. Hong, S. Levine, and A. Dragan. Learning to influence human behavior with offline reinforcement learning. *Advances in Neural Information Processing Systems*, 36, 2024.
- [125] J.-H. Hong, Y.-S. Song, and S.-B. Cho. Mixed-initiative human–robot interaction using hierarchical bayesian networks. *IEEE Transactions on Systems, Man, and Cybernetics-Part A: Systems and Humans*, 37(6):1158–1164, 2007.
- [126] S. Hopko, J. Wang, and R. Mehta. Human factors considerations and metrics in shared space human-robot collaboration: A systematic review. *Frontiers in Robotics and AI*, 9:799522, 2022.
- [127] E. Horvitz. Principles of mixed-initiative user interfaces. *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, pages 159–166, 1999.
- [128] E. Horvitz, A. Jacobs, and D. Hovel. Models of attention in computing and communication. *Communications of the ACM*, 46(3):52–59, 2003.
- [129] Z. Huang, Y.-J. Mun, X. Li, Y. Xie, N. Zhong, W. Liang, J. Geng, T. Chen, and K. Driggs-Campbell. Hierarchical intention tracking for robust human-

- robot collaboration in industrial assembly tasks. In *2023 IEEE International Conference on Robotics and Automation (ICRA)*, pages 9821–9828. IEEE, 2023.
- [130] N. Hulle, S. Aroca-Ouellette, A. J. Ries, J. Brawer, K. von der Wense, and A. Roncone. Eyes on the game: Deciphering implicit human signals to infer human proficiency, trust, and intent. *arXiv preprint arXiv:2407.03298*, 2024.
- [131] A. Ibarguren, I. Eimontaite, J. L. Outón, and S. Fletcher. Dual arm co-manipulation architecture with enhanced human–robot communication for large part manipulation. *Sensors*, 20(21), 2020.
- [132] I. Idrees, T. Yun, N. Sharma, Y. Deng, N. Gopalan, G. Konidaris, and S. Tellex. Improved inference of human intent by combining plan recognition and language feedback. In *2023 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 7976–7983. IEEE, 2023.
- [133] M. Ionova and J. K. Behrens. Cobos: Constraint-based online scheduler for human-robot collaboration. In *2024 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 7131–7137. IEEE, 2024.
- [134] S. Iregui, J. De Schutter, and E. Aertbeliën. Reconfigurable constraint-based reactive framework for assistive robotics with adaptable levels of autonomy. *IEEE Robotics and Automation Letters*, 6(4):7397–7405, 2021.
- [135] L. Jacqmin, L. M. Rojas-Barahona, and B. Favre. " do you follow me?": A survey of recent approaches in dialogue state tracking. *arXiv preprint arXiv:2207.14627*, 2022.
- [136] A. Jain, D. Chen, D. Bansal, S. Scheele, M. Kishore, H. Sapra, D. Kent, H. Ravichandar, and S. Chernova. Anticipatory human-robot collaboration via multi-objective trajectory optimization. In *2020 IEEE/RSJ International Con-*

- ference on Intelligent Robots and Systems (IROS)*, pages 11052–11057. IEEE, 2020.
- [137] S. Jain and B. Argall. Recursive bayesian human intent recognition in shared-control robotics. In *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 3905–3912, 2018.
 - [138] S. Jain and B. Argall. Probabilistic human intent recognition for shared autonomy in assistive robotics. *ACM Transactions on Human-Robot Interaction (THRI)*, 9(1):1–23, 2019.
 - [139] A. Jaiswal, A. R. Babu, M. Z. Zadeh, D. Banerjee, and F. Makedon. A survey on contrastive self-supervised learning. *Technologies*, 9(1):2, 2020.
 - [140] A. Jonnavittula and D. P. Losey. I know what you meant: Learning human objectives by (under) estimating their choice set. In *2021 IEEE International Conference on Robotics and Automation (ICRA)*, pages 2747–2753. IEEE, 2021.
 - [141] S. Karamcheti, M. Srivastava, P. Liang, and D. Sadigh. Lila: Language-informed latent actions. In *Conference on robot learning*, pages 1379–1390. PMLR, 2022.
 - [142] U. B. Karli, Z. Shangguan, and T. Fitzgerald. Insight: Inference-time sequence introspection for generating help triggers in vision-language-action models. *arXiv preprint arXiv:2510.01389*, 2025.
 - [143] H. Karnan, G. Warnell, X. Xiao, and P. Stone. Voila: Visual-observation-only imitation learning for autonomous navigation. In *2022 International Conference on Robotics and Automation (ICRA)*, pages 2497–2503. IEEE, 2022.
 - [144] T. Kaupp, A. Makarenko, and H. Durrant-Whyte. Human-robot communica-

- tion for collaborative decision making—a probabilistic approach. *Robotics and Autonomous Systems*, 58(5):444–456, 2010.
- [145] C. C. Kemp, A. Edsinger, H. M. Clever, and B. Matulevich. The design of stretch: A compact, lightweight mobile manipulator for indoor human environments. *arXiv preprint arXiv:2109.10892*, 2021.
- [146] A. Khamis, A. Hussein, and A. Elmogy. Multi-robot task allocation: A review of the state-of-the-art. *Cooperative robots and sensor networks 2015*, pages 31–51, 2015.
- [147] O. Khatib. Real-time obstacle avoidance for manipulators and mobile robots. *The international journal of robotics research*, 1986.
- [148] P. Khosla, P. Teterwak, C. Wang, A. Sarna, Y. Tian, P. Isola, A. Maschinot, C. Liu, and D. Krishnan. Supervised contrastive learning. *Advances in Neural Information Processing Systems*, 33:18661–18673, 2020.
- [149] W. Kim, L. Peternel, M. Lorenzini, J. Babič, and A. Ajoudani. A human–robot collaboration framework for improving ergonomics during dexterous operation of power tools. *Robotics and Computer-Integrated Manufacturing*, 68:102084, 2021.
- [150] A. Kirillov, E. Mintun, N. Ravi, H. Mao, C. Rolland, L. Gustafson, T. Xiao, S. Whitehead, A. C. Berg, W.-Y. Lo, et al. Segment anything. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 4015–4026, 2023.
- [151] R. F. Kizilcec. How much information? effects of transparency on trust in an algorithmic interface. In *Proceedings of the 2016 CHI Conference on Human Factors in Computing Systems (CHI '16)*, pages 2390–2395, 2016.

- [152] R. A. Knepper, S. Tellex, A. Li, and N. Roy. Recovering from failure by asking for help. *Autonomous Robots*, 39(3):347–362, 2015.
- [153] M. Kollmitz, K. Hsiao, J. Gaa, and W. Burgard. Time dependent planning on a layered social cost map for human-aware robot navigation. In *2015 European Conference on Mobile Robots (ECMR)*, pages 1–6. IEEE, 2015.
- [154] H. S. Koppula, A. Jain, and A. Saxena. Anticipatory planning for human-robot teams. In *Experimental Robotics: The 14th International Symposium on Experimental Robotics*, pages 453–470. Springer, 2016.
- [155] M. Kwon, E. Biyik, A. Talati, K. Bhasin, D. P. Losey, and D. Sadigh. When humans aren’t optimal: Robots that collaborate with risk-aware humans. In *Proceedings of the 2020 ACM/IEEE international conference on human-robot interaction*, pages 43–52, 2020.
- [156] M. Lagomarsino, E. Merlo, A. Pupa, T. Birr, F. Krebs, C. Secchi, T. Asfour, and A. Ajoudani. Intuitive programming, adaptive task planning, and dynamic role allocation in human–robot collaboration. *Annual Review of Control, Robotics, and Autonomous Systems*, 9.
- [157] J. Laplaza, A. Garrell, F. Moreno-Noguer, and A. Sanfeliu. Context and intention for 3d human motion prediction: experimentation and user study in handover tasks. In *2022 31st IEEE international conference on robot and human interactive communication (RO-MAN)*, pages 630–635. IEEE, 2022.
- [158] K. Lee, L. Smith, and P. Abbeel. Pebble: Feedback-efficient interactive reinforcement learning via relabeling experience and unsupervised pre-training. *arXiv preprint arXiv:2106.05091*, 2021.
- [159] G. Lemasurier, G. Bejerano, V. Albanese, J. Parrillo, H. A. Yanco, N. Amerson, R. Hetrick, and E. Phillips. Methods for expressing robot intent for human–

- robot collaboration in shared workspaces. *ACM Transactions on Human-Robot Interaction (THRI)*, 10(4):1–27, 2021.
- [160] A. Lew, S. Thompson, N. Tsoi, and M. Vázquez. Shutter, the robot photographer: Leveraging behavior trees for public, in-the-wild human-robot interactions. *arXiv preprint arXiv:2302.00191*, 2023.
- [161] M. Li, M. Kwon, and D. Sadigh. Influencing leading and following in human–robot teams. *Autonomous Robots*, 45:959–978, 2021.
- [162] Y. Li, P. Hu, Z. Liu, D. Peng, J. T. Zhou, and X. Peng. Contrastive clustering. In *2021 AAAI Conference on Artificial Intelligence (AAAI)*, 2021.
- [163] K. Liang, Z. Zhang, and J. F. Fisac. Introspective planning: Aligning robots’ uncertainty with inherent task ambiguity. In *The Thirty-eighth Annual Conference on Neural Information Processing Systems*.
- [164] T.-Y. Lin, M. Maire, S. Belongie, J. Hays, P. Perona, D. Ramanan, P. Dollár, and C. L. Zitnick. Microsoft coco: Common objects in context. In *European conference on computer vision*, pages 740–755. Springer, 2014.
- [165] Z. C. Lipton, C. Elkan, and B. Naryanaswamy. Optimal thresholding of classifiers to maximize f1 measure. In *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*, pages 225–239. Springer, 2014.
- [166] C. Liu, J. B. Hamrick, J. F. Fisac, A. D. Dragan, J. K. Hedrick, S. S. Sastry, and T. L. Griffiths. Goal inference improves objective and perceived performance in human-robot collaboration. *arXiv preprint arXiv:1802.01780*, 2018.
- [167] Y. Liu, F. Guo, and Y. Ma. A stackelberg trust-based human–robot collaboration framework for warehouse picking. *Systems*, 13(5), 2025.

- [168] C. Lohrmann, M. Stull, A. Roncone, and B. Hayes. Generating pattern-based conventions for predictable planning in human–robot collaboration. *ACM Transactions on Human-Robot Interaction*, 13(4):1–23, 2024.
- [169] D. P. Losey, A. Bajcsy, M. K. O’Malley, and A. D. Dragan. Physical interaction as communication: Learning robot objectives online from human corrections. *The International Journal of Robotics Research*, 41(1):20–44, 2022.
- [170] C. Ma, K. Lu, R. Desai, X. Puig, A. Markham, and N. Trigoni. Coopera: Continual open-ended human-robot assistance. *arXiv preprint arXiv:2510.23495*, 2025.
- [171] R. Ma, J. Qu, A. Bobu, and D. Hadfield-Menell. Goal inference from open-ended dialog. *arXiv preprint arXiv:2410.13957*, 2024.
- [172] X. Ma and D. A. Castanon. Receding horizon planning for dubins traveling salesman problems. In *Proceedings of the 45th IEEE Conference on Decision and Control*, pages 5453–5458. IEEE, 2006.
- [173] Y. Ma, Z. Song, Y. Zhuang, J. Hao, and I. King. A survey on vision-language-action models for embodied ai. *arXiv preprint arXiv:2405.14093*, 2024.
- [174] W. Macke, R. Mirsky, and P. Stone. Expected value of communication for planning in ad hoc teamwork. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35, pages 11290–11298, 2021.
- [175] A. J. Mahmud, W. Li, and X. Wang. Mutual adaptation in human–robot co-transportation with human preference uncertainty. *arXiv preprint arXiv:2503.08895*, 2025.
- [176] K. Majd and R. S. Zarrin. Robot behavior adaptation in physical human–

- robot interactions based on learned safety preferences. In *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2025.
- [177] O. Majumder, A. Ravichandran, S. Maji, M. Polito, R. Bhotika, and S. Soatto. Revisiting contrastive learning for few-shot classification. *CoRR*, abs/2101.11058, 2021.
- [178] O. Mangin, A. Roncone, and B. Scassellati. How to be helpful? supportive behaviors and personalization for human-robot collaboration. *Frontiers in Robotics and AI*, page 426, 2022.
- [179] S. Mannem, W. Macke, P. Stone, and R. Mirsky. Exploring the cost of interruptions in human-robot teaming. In *2023 IEEE-RAS 22nd International Conference on Humanoid Robots (Humanoids)*, pages 1–8, 2023. arXiv:2311.00785.
- [180] S. Mannem, W. Macke, P. Stone, and R. Mirsky. Exploring the cost of interruptions in human-robot teaming. In *2023 IEEE-RAS 22nd International Conference on Humanoid Robots (Humanoids)*, pages 1–8. IEEE, 2023.
- [181] D. K. Misra, J. Sung, K. Lee, and A. Saxena. Tell me dave: Context-sensitive grounding of natural language to manipulation instructions. *The International Journal of Robotics Research*, 35(1-3):281–300, 2016.
- [182] D. Mukherjee, K. Gupta, L. H. Chang, and H. Najjaran. A survey of robot learning strategies for human-robot collaboration in industrial settings. *Robotics and Computer-Integrated Manufacturing*, 73:102231, 2022.
- [183] S. Murata, W. Masuda, J. Chen, H. Arie, T. Ogata, and S. Sugano. Achieving human-robot collaboration with dynamic goal inference by gradient descent. In *Neural Information Processing*. Springer International Publishing, 2019.

- [184] T. Muxfeldt and J. J. Steil. Recovering from assembly errors by exploiting human demonstrations. *Procedia CIRP*, 72:1244–1249, 2018.
- [185] V. Myers, E. Ellis, S. Levine, B. Eysenbach, and A. Dragan. Learning to assist humans without inferring rewards. *Advances in Neural Information Processing Systems*, 37:71540–71567, 2024.
- [186] S. Nashed and S. Zilberstein. A survey of opponent modeling in adversarial domains. *Journal of Artificial Intelligence Research*, 73:277–327, 2022.
- [187] Z. Nazari, G. M. Lucas, and J. Gratch. Opponent modeling for virtual human negotiators. In *Intelligent Virtual Agents: 15th International Conference, IVA 2015, Delft, The Netherlands, August 26-28, 2015, Proceedings 15*, pages 39–49. Springer, 2015.
- [188] E. Ng, Z. Liu, and M. Kennedy. It takes two: Learning to plan for human-robot cooperative carrying. In *2023 IEEE International Conference on Robotics and Automation (ICRA)*, pages 7526–7532. IEEE, 2023.
- [189] S. M. Nguyen and P.-Y. Oudeyer. Active choice of teachers, learning strategies and goals for a socially guided intrinsic motivation learner. *Paladyn, Journal of Behavioral Robotics*, 3(3):136–146, 2012.
- [190] S. M. Nguyen and P.-Y. Oudeyer. Socially guided intrinsic motivation for robot learning of motor skills. *Autonomous Robots*, 36(3):273–294, 2014.
- [191] S. Nikolaidis, D. Hsu, and S. Srinivasa. Improved human–robot team performance through cross-training. *The International Journal of Robotics Research*, 34(13):1711–1730, 2015.
- [192] S. Nikolaidis, A. Kuznetsov, D. Hsu, and S. Srinivasa. Formalizing human-robot

- mutual adaptation: A bounded memory model. In *2016 11th ACM/IEEE International Conference on Human-Robot Interaction (HRI)*, pages 75–82, 2016.
- [193] S. Nikolaidis, M. Kwon, J. Forlizzi, and S. Srinivasa. Planning with verbal communication for human-robot collaboration. *ACM Transactions on Human-Robot Interaction (THRI)*, 2018.
- [194] S. Nikolaidis, P. Lasota, R. Ramakrishnan, and J. Shah. Improved human–robot team performance through cross-training, an approach inspired by human team training practices. *The International Journal of Robotics Research*, 34(14):1711–1730, 2015.
- [195] S. Nikolaidis, P. Lasota, G. Rossano, and J. Shah. Human–robot mutual adaptation in collaborative tasks: Models and experiments. *The International Journal of Robotics Research*, 36(5-7):618–634, 2017.
- [196] S. Nikolaidis, S. Nath, A. D. Procaccia, and S. Srinivasa. Game-theoretic modeling of human adaptation in human-robot collaboration. In *Proceedings of the 2017 ACM/IEEE International Conference on Human-Robot Interaction (HRI)*, pages 323–331. ACM, 2017.
- [197] S. Nikolaidis, R. Ramakrishnan, K. Gu, and J. A. Shah. Efficient model learning from joint-action demonstrations for human–robot collaborative tasks. In *Proceedings of the 10th ACM/IEEE International Conference on Human-Robot Interaction*, pages 189–196, 2015.
- [198] S. Nikolaidis and J. Shah. Human-robot cross-training: Computational formulation, modeling and evaluation of a human team training strategy. In *2013 8th ACM/IEEE international conference on human-robot interaction (HRI)*, pages 33–40. IEEE, 2013.

- [199] S. Nikolaidis, Y. X. Zhu, D. Hsu, and S. Srinivasa. Human-robot mutual adaptation in shared autonomy. In *Proceedings of the 2017 International Conference on Human-Robot Interaction*, 2017.
- [200] S. Nikolaidis, Y. X. Zhu, D. Hsu, and S. Srinivasa. Human-robot mutual adaptation in shared autonomy. In *Proceedings of the 2017 ACM/IEEE International Conference on Human-Robot Interaction, HRI '17*, pages 294–302, New York, NY, USA, 2017. Association for Computing Machinery.
- [201] A. Noormohammadi-Asl, S. L. Smith, and K. Dautenhahn. To lead or to follow? adaptive robot task planning in human-robot collaboration. *IEEE Transactions on Robotics*, 2025.
- [202] I. Oguntola, J. Campbell, S. Stepputtis, and K. Sycara. Theory of mind as intrinsic motivation for multi-agent reinforcement learning. *arXiv preprint arXiv:2307.01158*, 2023.
- [203] T. Panambur, D. Chakraborty, M. Meyer, R. Milliken, E. Learned-Miller, and M. Parente. Self-supervised learning to guide scientifically relevant categorization of martian terrain images. *arXiv preprint arXiv:2204.09854*, 2022.
- [204] R. Pandya, C. Liu, and A. Bajcsy. Robots that learn to safely influence via prediction-informed reach-avoid dynamic games. *arXiv preprint arXiv:2409.12153*, 2024.
- [205] R. Pandya, Z. Wang, Y. Nakahira, and C. Liu. Towards proactive safe human-robot collaborations via data-efficient conditional behavior prediction. In *2024 IEEE International Conference on Robotics and Automation (ICRA)*, pages 12956–12963. IEEE, 2024.
- [206] R. Pandya, M. Zhao, C. Liu, R. Simmons, and H. Admoni. Multi-agent strat-

- egy explanations for human-robot collaboration. In *2024 IEEE International Conference on Robotics and Automation (ICRA)*, pages 17351–17357, 2024.
- [207] S. Parekh and D. P. Losey. Learning latent representations to co-adapt to humans. *Autonomous Robots*, pages 1–26, 2023.
- [208] R. Paul, A. Barbu, S. Felshin, B. Katz, and N. Roy. Temporal grounding graphs for language understanding with accrued visual-linguistic context. *arXiv preprint arXiv:1811.06966*, 2018.
- [209] T. Pearce, T. Rashid, A. Kanervisto, D. Bignell, M. Sun, R. Georgescu, S. V. Macua, S. Z. Tan, I. Momennejad, K. Hofmann, et al. Imitating human behaviour with diffusion models. *arXiv preprint arXiv:2301.10677*, 2023.
- [210] J. Pearl. *Causality*. Cambridge university press, 2009.
- [211] S. Pellegrinelli, H. Admoni, S. Javdani, and S. Srinivasa. Human-robot shared workspace collaboration via hindsight optimization. In *2016 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 831–838. IEEE, 2016.
- [212] P. Perera, P. Oza, and V. M. Patel. One-class classification: A survey. *arXiv preprint arXiv:2101.03064*, 2021.
- [213] L. Peternel, N. G. Tsagarakis, and D. G. Caldwell. Robot adaptation to human physical fatigue in human–robot co-manipulation. *Autonomous Robots*, 42(5):1011–1021, 2018.
- [214] L. Peternel, N. G. Tsagarakis, and D. G. Caldwell. Robot adaptation to human physical fatigue in human–robot co-manipulation. *Autonomous Robots*, 42(5):1011–1021, 2018.

- [215] M. Pfeiffer, U. Schwesinger, H. Sommer, E. Galceran, and R. Siegwart. Predicting actions to act predictably: Cooperative partial motion planning with maximum entropy models. In *2016 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2016.
- [216] A. Pupa, C. T. Landi, M. Bertolani, and C. Secchi. A dynamic architecture for task assignment and scheduling for collaborative robotic cells. In *Human-Friendly Robotics 2020: 13th International Workshop*, pages 74–88. Springer, 2021.
- [217] P. Qian, H. Huang, and V. Unhelkar. Pps: Personalized policy summarization for explaining sequential behavior of autonomous agents. In *Proceedings of the AAAI/ACM Conference on AI, Ethics, and Society*, volume 7, pages 1167–1179, 2024.
- [218] C. Z. Qiao, M. Sakr, K. Muelling, and H. Admoni. Learning from demonstration for real-time user goal prediction and shared assistive control. In *2021 IEEE International Conference on Robotics and Automation (ICRA)*, pages 3270–3275. IEEE, 2021.
- [219] J. Qin, S. Ban, W. Zhu, Y. Wang, and D. Samaras. Learning human-aware robot policies for adaptive assistance. *arXiv preprint arXiv:2412.11913*, 2024.
- [220] A. H. Qureshi, Y. Nakamura, Y. Yoshikawa, and H. Ishiguro. Intrinsically motivated reinforcement learning for human–robot interaction in the real-world. *Neural Networks*, 107:23–33, 2018.
- [221] N. Rabinowitz, F. Perbet, F. Song, C. Zhang, S. A. Eslami, and M. Botvinick. Machine theory of mind. In *International conference on machine learning*, pages 4218–4227. PMLR, 2018.

- [222] A. Radford, J. W. Kim, C. Hallacy, A. Ramesh, G. Goh, S. Agarwal, G. Sastry, A. Askell, P. Mishkin, J. Clark, et al. Learning transferable visual models from natural language supervision. In *International conference on machine learning*, pages 8748–8763. PMLR, 2021.
- [223] K. Ramachandruni, C. Kent, and S. H. Chernova. Uhtp: A user-aware hierarchical task planning framework for communication-free, mutually adaptive human-robot collaboration. *ACM Transactions on Human-Robot Interaction*, Aug. 2024.
- [224] 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.
- [225] H. Renz, M. Krämer, and T. Bertram. Moving horizon planning for human-robot interaction. In *Proceedings of the 2024 ACM/IEEE International Conference on Human-Robot Interaction*, pages 939–943, 2024.
- [226] A. Romero, G. Baldassarre, R. J. Duro, and V. G. Santucci. A motivational architecture for open-ended learning challenges in robots. *arXiv preprint arXiv:2506.18454*, 2025.
- [227] A. Roncone, O. Mangin, and B. Scassellati. Transparent role assignment and task allocation in human robot collaboration. In *2017 IEEE International Conference on Robotics and Automation (ICRA)*, pages 1014–1021. IEEE, 2017.
- [228] D. Sadigh, A. D. Dragan, S. Sastry, and S. A. Seshia. *Active preference-based learning of reward functions*. 2017.
- [229] D. Sadigh, S. Sastry, S. A. Seshia, and A. D. Dragan. Planning for autonomous cars that leverage effects on human actions. In *Robotics: Science and systems*, volume 2. Ann Arbor, MI, USA, 2016.

- [230] D. Sadigh, S. S. Sastry, S. A. Seshia, and A. Dragan. Information gathering actions over human internal state. In *2016 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 66–73. IEEE, 2016.
- [231] B. Sadrfaridpour, H. Saeidi, J. Burke, K. Madathil, and Y. Wang. Modeling and control of trust in human-robot collaborative manufacturing. In R. Mittu et al., editors, *Robust Intelligence and Trust in Autonomous Systems*, pages 115–141. Springer, 2016.
- [232] S. Sagheb, Y.-J. Mun, N. Ahmadian, B. A. Christie, A. Bajcsy, K. Driggs-Campbell, and D. P. Losey. Towards robots that influence humans over long-term interaction. In *2023 IEEE International Conference on Robotics and Automation (ICRA)*, pages 7490–7496. IEEE, 2023.
- [233] S. Sagheb, S. Parekh, R. Pandya, Y.-J. Mun, K. Driggs-Campbell, A. Bajcsy, and D. P. Losey. A unified framework for robots that influence humans over long-term interaction. *arXiv preprint arXiv:2503.14633*, 2025.
- [234] N. Sao Mai. *The intrinsic motivation of reinforcement and imitation learning for sequential tasks*. PhD thesis, Ph. D. Dissertation. HAL Archive. <https://hal.science/tel-04853270>, 2024.
- [235] M. Schrepp. User experience questionnaire handbook. *All you need to know to apply the UEQ successfully in your project*, 10, 2015.
- [236] M. L. Schrum, E. Hedlund-Botti, N. Moorman, and M. C. Gombolay. Mind meld: Personalized meta-learning for robot-centric imitation learning. In *2022 17th ACM/IEEE International Conference on Human-Robot Interaction (HRI)*, pages 157–165. IEEE, 2022.
- [237] M. Schultz and T. Joachims. Learning a distance metric from relative comparisons. *Advances in neural information processing systems*, 16, 2003.

- [238] S. Sebo, B. Stoll, B. Scassellati, and M. F. Jung. Robots in groups and teams: a literature review. *Proceedings of the ACM on Human-Computer Interaction*, 4(CSCW2):1–36, 2020.
- [239] P. Sermanet, C. Lynch, Y. Chebotar, J. Hsu, E. Jang, S. Schaal, S. Levine, and G. Brain. Time-contrastive networks: Self-supervised learning from video. In *2018 IEEE international conference on robotics and automation (ICRA)*, pages 1134–1141. IEEE, 2018.
- [240] V. Sharma, M. Tapaswi, M. S. Sarfraz, and R. Stiefelhagen. Clustering based contrastive learning for improving face representations. In *2020 15th IEEE International Conference on Automatic Face and Gesture Recognition (FG 2020)*, pages 109–116. IEEE, 2020.
- [241] M. Shvo. *Theory of Mind Reasoning in Explanation, Plan Recognition, and Assistance: Theory and Practice*. PhD thesis, University of Toronto (Canada), 2023.
- [242] M. Shvo and S. A. McIlraith. Active goal recognition. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 9957–9966, 2020.
- [243] T. Silver and R. Chitnis. Pddl-gym: Gym environments from pddl problems. *arXiv preprint arXiv:2002.06432*, 2020.
- [244] S. Singh, E. Rantanen, and J. Heard. Human-robot teaming: A comprehensive survey on collaboration, communication, and cognition. *ACM Transactions on Human-Robot Interaction*, 2025.
- [245] S. Sloman and S. A. Sloman. *Causal models: How people think about the world and its alternatives*. Oxford University Press, 2009.

- [246] J. Smisek, M. Jancosek, and T. Pajdla. 3d with kinect. In *Consumer depth cameras for computer vision*, pages 3–25. Springer, 2013.
- [247] M. Song, X. Deng, Z. Zhou, J. Wei, W. Guan, and L. Nie. A survey on diffusion policy for robotic manipulation: Taxonomy, analysis, and future directions. *Authorea Preprints*, 2025.
- [248] J. Spencer, S. Choudhury, M. Barnes, M. Schmittle, M. Chiang, P. Ramadge, and S. Srinivasa. Learning from interventions: Human-robot interaction as both explicit and implicit feedback. 07 2020. doi: 10.15607/rss. 2020.
- [249] A. Sripathy, A. Bobu, D. S. Brown, and A. D. Dragan. Dynamically switching human prediction models for efficient planning. In *2021 IEEE International Conference on Robotics and Automation (ICRA)*, pages 3495–3501. IEEE, 2021.
- [250] A. St. Clair and M. Mataric. How robot verbal feedback can improve team performance in human-robot task collaborations. In *Proceedings of the tenth annual acm/ieee international conference on human-robot interaction*, pages 213–220, 2015.
- [251] M. Stiber, R. Taylor, and C.-M. Huang. Robot error awareness through human reactions: Implementation, evaluation, and recommendations, 2025.
- [252] P. S. Suresh, P. Doshi, and B. Banerjee. Adaptive human-robot collaboration using type-based irl. In *The 41st Conference on Uncertainty in Artificial Intelligence*, 2025.
- [253] J. Sweller. Cognitive load during problem solving: Effects on learning. *Cognitive Science*, 12(2):257–285, 1988.
- [254] A. Tabrez, S. Agrawal, and B. Hayes. Explanation-based reward coaching to improve human performance via reinforcement learning. In *Proceedings of the*

- 14th ACM/IEEE International Conference on Human-Robot Interaction (HRI)*, pages 249–257. IEEE, 2019.
- [255] A. Tabrez, M. B. Luebbbers, and B. Hayes. A survey of mental modeling techniques in human–robot teaming. *Current Robotics Reports*, 1(4):259–267, 2020.
- [256] A. Tabrez, M. B. Luebbbers, and B. Hayes. Descriptive and prescriptive visual guidance to improve shared situational awareness in human-robot teaming. In *Proceedings of the 21st International Conference on Autonomous Agents and Multiagent Systems (AAMAS)*, pages 1467–1475. IFAAMAS, 2022.
- [257] F. Tanguay-Rioux, M. Héroux, and R. Legros. Physical properties of recyclable materials and implications for resource recovery. *Waste management*, 136:195–203, 2021.
- [258] S. Tellex, T. Kollar, S. Dickerson, M. Walter, A. Banerjee, S. Teller, and N. Roy. Understanding natural language commands for robotic navigation and mobile manipulation. In *Proceedings of the AAAI conference on artificial intelligence*, volume 25, pages 1507–1514, 2011.
- [259] J. B. Tenenbaum, C. Kemp, T. L. Griffiths, and N. D. Goodman. How to grow a mind: Statistics, structure, and abstraction. *science*, 331(6022):1279–1285, 2011.
- [260] A. L. Thomaz and C. Breazeal. Robot learning via socially guided exploration. *Development and Learning*, pages 82–87, 2007.
- [261] S. Thompson, T. Horiuchi, and S. Kagami. A probabilistic model of human motion and navigation intent for mobile robot path planning. In *2009 4th international conference on autonomous robots and agents*, pages 663–668. IEEE, 2009.

- [262] S. Thompson, A. Lew, R. Phanse, A. Huang, E. Stanish, Y. Li, and M. Vázquez. Predicting human intent to interact with a public robot: The people approaching robots database (par-d). 2024.
- [263] R. Tian, M. Tomizuka, A. D. Dragan, and A. Bajcsy. Towards modeling and influencing the dynamics of human learning. In *Proceedings of the 2023 ACM/IEEE International Conference on Human-Robot Interaction*, HRI '23, page 350–358, New York, NY, USA, 2023. Association for Computing Machinery.
- [264] R. Tian, M. Tomizuka, A. D. Dragan, and A. Bajcsy. Towards modeling and influencing the dynamics of human learning. In *Proceedings of the 2023 ACM/IEEE international conference on human-robot interaction*, pages 350–358, 2023.
- [265] Y. Tian, K.-R. Kladny, Q. Wang, Z. Huang, and O. Fink. Multi-agent actor-critic with time dynamical opponent model. *Neurocomputing*, 517:165–172, 2023.
- [266] B. Tilekbay, S. Yang, M. A. Lewkowicz, A. Suryapranata, and J. Kim. Expressedit: Video editing with natural language and sketching. In *Proceedings of the 29th International Conference on Intelligent User Interfaces*, pages 515–536, 2024.
- [267] J. G. Trafton, E. M. Altmann, D. P. Brock, and F. E. Mintz. The dynamics of interruption: Memory for goals and goal activation. *Journal of Experimental Psychology: General*, 132(4):563–581, 2003.
- [268] D. Tsakona and Y. Demiris. Continuous real-time adaptation framework for enhancing trust and technology acceptance: An assistive feeding study. *ACM Transactions on Human-Robot Interaction*, 15(1):1–37, 2025.

- [269] N. Tsoi, A. Xiang, P. Yu, S. S. Sohn, G. Schwartz, S. Ramesh, M. Hussein, A. W. Gupta, M. Kapadia, and M. Vázquez. Sean 2.0: Formalizing and generating social situations for robot navigation. *IEEE Robotics and Automation Letters*, 7(4):11047–11054, 2022.
- [270] Y.-S. Tung, M. B. Luebbbers, A. Roncone, and B. Hayes. Workspace optimization techniques to improve prediction of human motion during human-robot collaboration. In *Proceedings of the 2024 ACM/IEEE International Conference on Human-Robot Interaction*, 2024.
- [271] V. V. Unhelkar, S. Li, and J. A. Shah. Decision-making for bidirectional communication in sequential human-robot collaborative tasks. In *Proceedings of the 2020 ACM/IEEE International Conference on Human-Robot Interaction*, pages 329–341, 2020.
- [272] V. V. Unhelkar, S. Li, and J. A. Shah. Semi-supervised learning of decision-making models for human-robot collaboration. In *conference on Robot Learning*, pages 192–203. PMLR, 2020.
- [273] M. Ürkmez and H. I. Bozma. Detecting 3d hand pointing direction from rgb-d data in wide-ranging hri scenarios. In *2022 17th ACM/IEEE International Conference on Human-Robot Interaction (HRI)*, pages 441–450. IEEE, 2022.
- [274] A. Valtazanos and S. Ramamoorthy. Bayesian interaction shaping: Learning to influence strategic interactions in mixed robotic domains. In *Proceedings of the 2013 international conference on Autonomous agents and multi-agent systems*, pages 63–70. International Foundation for Autonomous Agents and Multiagent Systems, 2013.
- [275] K. van de Merwe, S. Mallam, and S. Nazir. Agent transparency, situation

- awareness, mental workload, and operator performance: A systematic literature review. *Human Factors*, 2022.
- [276] L. van der Spaa, J. Kober, and M. Gienger. Simultaneously learning intentions and preferences during physical human-robot cooperation. *Autonomous Robots*, 48(4):11, 2024.
- [277] S. van Waveren, C. Pek, J. Tumova, and I. Leite. Correct me if i’m wrong: Using non-experts to repair reinforcement learning policies. In *Proceedings of the 17th ACM/IEEE International Conference on Human-Robot Interaction*, pages 1–9, 2022.
- [278] E. M. van Zoelen, E. I. Barakova, and M. Rauterberg. Adaptive leader-follower behavior in human-robot collaboration. In *2020 29th IEEE International Conference on Robot and Human Interactive Communication (RO-MAN)*, pages 1259–1265, 2020.
- [279] E. M. van Zoelen, K. van den Bosch, and M. Neerincx. Becoming team members: Identifying interaction patterns of mutual adaptation for human-robot co-learning. *Frontiers in Robotics and AI*, 8:692811, 2021.
- [280] A. Vaswani. Attention is all you need. *arXiv preprint arXiv:1706.03762*, 2017.
- [281] M. Vered, P. Howe, T. Miller, L. Sonenberg, and E. Velloso. Demand-driven transparency for monitoring intelligent agents. *IEEE Transactions on Human-Machine Systems*, 50(3):264–275, 2020.
- [282] F. B. Von Der Osten, M. Kirley, and T. Miller. The minds of many: Opponent modeling in a stochastic game. In *IJCAI*, 2017.
- [283] C. Wang, C. Perez-D’Arpino, D. Xu, L. Fei-Fei, C. K. Liu, and S. Savarese. Co-

- gail: Learning diverse strategies for human-robot collaboration. In *Proceedings of the 5th Conference on Robot Learning (CoRL)*, London, UK, 2021. PMLR.
- [284] S. Wang, A. Wang, S. Goncharova, B. Scassellati, and T. Fitzgerald. Effects of robot competency and motion legibility on human correction feedback. In *2025 20th ACM/IEEE International Conference on Human-Robot Interaction (HRI)*, pages 789–799. IEEE, 2025.
- [285] W. Z. Wang, M. Beliaev, E. Bıyık, D. A. Lazar, R. Pedarsani, and D. Sadigh. Emergent prosociality in multi-agent games through gifting. *arXiv preprint arXiv:2105.06593*, 2021.
- [286] W. Z. Wang, A. Shih, A. Xie, and D. Sadigh. Influencing towards stable multi-agent interactions. In A. Faust, D. Hsu, and G. Neumann, editors, *Proceedings of the 5th Conference on Robot Learning*, volume 164 of *Proceedings of Machine Learning Research*, pages 1132–1143. PMLR, 08–11 Nov 2022.
- [287] M. Wilber, I. S. Kwak, D. Kriegman, and S. Belongie. Learning concept embeddings with combined human-machine expertise. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 981–989, 2015.
- [288] N. Wilde, A. Blidaru, S. L. Smith, and D. Kulić. Improving user specifications for robot behavior through active preference learning: Framework and evaluation. *The International Journal of Robotics Research*, 39(6):651–667, 2020.
- [289] N. Wilde, D. Kulić, and S. L. Smith. Active preference learning using maximum regret. In *2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 10952–10959. IEEE, 2020.
- [290] H. Wu, A. Ghadami, A. E. Bayrak, J. M. Smereka, and B. I. Epureanu. Impact of heterogeneity and risk aversion on task allocation in multi-agent teams. *IEEE Robotics and Automation Letters*, 6(4):7065–7072, 2021.

- [291] H. Wu and X. Wang. Contrastive learning of image representations with cross-video cycle-consistency. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 10149–10159, October 2021.
- [292] S. A. Wu, R. E. Wang, J. A. Evans, J. B. Tenenbaum, D. C. Parkes, and M. K-Weiner. Too many cooks: Bayesian inference for coordinating multi-agent collaboration. *Topics in Cognitive Science*, 2021.
- [293] A. Xie, D. Losey, R. Tolsma, C. Finn, and D. Sadigh. Learning latent representations to influence multi-agent interaction. In *Conference on robot learning*, pages 575–588. PMLR, 2021.
- [294] A. Xie, D. P. Losey, R. Tolsma, C. Finn, and D. Sadigh. Learning latent representations to influence multi-agent interaction. In *Proceedings of the 4th Conference on Robot Learning (CoRL)*, Cambridge, MA, USA, 2020. PMLR.
- [295] A. Yadav, S. K. Hopko, P. R. Pagilla, and R. K. Mehta. Enhancing trust examinations with neural measures during human–robot collaboration under cognitive fatigue. *ACM Transactions on Human-Robot Interaction*, 15(2):1–27, 2025.
- [296] E. Yang and C. Mavrogiannis. Implicit communication in human-robot collaborative transport. In *2025 20th ACM/IEEE International Conference on Human-Robot Interaction (HRI)*, pages 23–33. IEEE, 2025.
- [297] J. Yang, A. Li, M. Farajtabar, P. Sunehag, E. Hughes, and H. Zha. Learning to incentivize other learning agents. *Advances in Neural Information Processing Systems*, 33:15208–15219, 2020.
- [298] L. Ying, T. Zhi-Xuan, V. Mansinghka, and J. B. Tenenbaum. Inferring the goals of communicating agents from actions and instructions. In *Proceedings of the AAAI symposium series*, volume 2, pages 26–33, 2023.

- [299] X. Yu, J. Jiang, W. Zhang, H. Jiang, and Z. Lu. Model-based opponent modeling. *Advances in Neural Information Processing Systems*, 35:28208–28221, 2022.
- [300] J. Zbontar, L. Jing, I. Misra, Y. LeCun, and S. Deny. Barlow twins: Self-supervised learning via redundancy reduction. In *International Conference on Machine Learning*, pages 12310–12320. PMLR, 2021.
- [301] D. Zhang, F. Nan, X. Wei, S. Li, H. Zhu, K. McKeown, R. Nallapati, A. Arnold, and B. Xiang. Supporting clustering with contrastive learning. *arXiv preprint arXiv:2103.12953*, 2021.
- [302] F. Zhang, V. Bazarevsky, A. Vakunov, A. Tkachenka, G. Sung, C.-L. Chang, and M. Grundmann. Mediapipe hands: On-device real-time hand tracking. *arXiv preprint arXiv:2006.10214*, 2020.
- [303] X. Zhang, C. Liu, Y. Yue, Q. Gong, F. Ma, and Y. Wang. Human-robot collaborative disassembly task planning for retired power battery based on stackelberg game and multi-agent deep reinforcement learning. *Journal of Manufacturing Systems*, 82:841–857, 2025.
- [304] Y. Zhang, H. Jiang, Y. Miura, C. D. Manning, and C. P. Langlotz. Contrastive learning of medical visual representations from paired images and text. *arXiv preprint arXiv:2010.00747*, 2020.
- [305] Y. Zhang, X. Zhang, J. Li, R. Qiu, H. Xu, and Q. Tian. Semi-supervised contrastive learning with similarity co-calibration. *IEEE Transactions on Multimedia*, pages 1–1, 2022.
- [306] M. Zhao, R. Simmons, and H. Admoni. Coordination with humans via strategy matching. In *2022 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 9116–9123. IEEE, 2022.

- [307] M. D. Zhao, R. Simmons, and H. Admoni. Learning human contribution preferences in collaborative human-robot tasks. In *Conference on Robot Learning*, pages 3597–3618. PMLR, 2023.
- [308] H. Zheng, W. Xia, and X. Xu. A human-robot collaborative assembly framework with quality checking based on real-time dual-hand action segmentation. *Robotics and Computer-Integrated Manufacturing*, 94:102976, 2025.
- [309] M. Zheng, F. Wang, S. You, C. Qian, C. Zhang, X. Wang, and C. Xu. Weakly supervised contrastive learning. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 10042–10051, 2021.
- [310] Y. Zheng, Y. Li, P. Wang, and H. Liu. A human-robot collaborative assembly framework with real-time quality checking. *Robotics and Computer-Integrated Manufacturing*, 86:102641, 2025.
- [311] T. Zhi-Xuan, G. Kang, V. Mansinghka, and J. B. Tenenbaum. Infinite ends from finite samples: Open-ended goal inference as top-down bayesian filtering of bottom-up proposals. *arXiv preprint arXiv:2407.16770*, 2024.
- [312] H. Zhong, J. Wu, C. Chen, J. Huang, M. Deng, L. Nie, Z. Lin, and X.-S. Hua. Graph contrastive clustering. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 9224–9233, 2021.
- [313] T. Zhou and J. P. Wachs. Early turn-taking prediction with spiking neural networks for human robot collaboration. In *2018 IEEE International Conference on Robotics and Automation (ICRA)*, pages 3250–3256. IEEE, 2018.
- [314] Y. Zhou, Y. Peng, W. Li, and D. T. Pham. Stackelberg model-based human-robot collaboration in removing screws for product remanufacturing. *Robotics and Computer-Integrated Manufacturing*, 77:102370, 2022.

- [315] Z. Zhuang, Y. Ben-Shabat, J. Zhang, S. Gould, and R. Mahony. Goferbot: A visual guided human-robot collaborative assembly system. In *2022 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 8910–8917. IEEE, 2022.
- [316] B. Zitkovich, T. Yu, S. Xu, P. Xu, T. Xiao, F. Xia, J. Wu, P. Wohlhart, S. Welker, A. Wahid, et al. Rt-2: Vision-language-action models transfer web knowledge to robotic control. In *Conference on Robot Learning*, pages 2165–2183. PMLR, 2023.
- [317] X. Zou, J. Yang, H. Zhang, F. Li, L. Li, J. Wang, L. Wang, J. Gao, and Y. J. Lee. Segment everything everywhere all at once. *Advances in neural information processing systems*, 36:19769–19782, 2023.
- [318] M. Zurek, A. Bobu, D. S. Brown, and A. D. Dragan. Situational confidence assistance for lifelong shared autonomy. In *2021 IEEE International Conference on Robotics and Automation (ICRA)*, pages 2783–2789. IEEE, 2021.