Doctoral Dissertation

Towards Multimodal Foundation Model for Human-Centered Robot Behavior

Jeongeun Park

Department of Artificial Intelligence

Graduate School

Korea University

February 2026

Towards Multimodal Foundation Model for Human-Centered Robot Behavior

by

Jeongeun Park

---

under the supervision of Professor Sungjoon Choi

A dissertation submitted in partial fulfillment of
the requirements for the degree of
Doctor of Philosophy

Department of Artificial Intelligence

Graduate School
Korea University

October 2025

Towards Multimodal Foundation Model for Human-Centered Robot Behavior

by Jeongeun Park
Department of Artificial Intelligence
under the supervision of Professor Sungjoon Choi

Abstract

Robots that share everyday spaces with people must do more than execute low-level actions correctly. They are expected to interpret underspecified language, anticipate when tasks may fail in the real world, and express their intentions through both verbal and non-verbal behavior. While recent foundation models in language and vision demonstrate impressive generalization, they are rarely shaped by these practical and social demands of human-centered robot behavior.

This dissertation investigates what would be required of a multimodal foundation model for human-centered robots and explores concrete steps toward that goal. I first articulate design requirements along several axes, including multimodal perception and representation, reasoning about physical feasibility and task success, interactive language use, non-verbal expressivity, and safety and predictability. This perspective emphasizes that such models must reason not only about *what* a human has asked for, but also about *whether* it can be carried out and *how* it should be communicated back to the user.

Within this view, I study verbal interaction through models that link language, perception, and action while explicitly accounting for uncertainty and failure. A hierarchy of vision–language–action models is trained on mixed-quality trajectories containing both

successes and failures, enabling robots to prefer subgoals that are linguistically appropriate yet also likely to succeed in the current environment. Complementarily, a command-understanding framework classifies user instructions as clear, ambiguous, or infeasible and uses calibrated uncertainty estimates to decide when to ask clarifying questions or explain limitations instead of naively executing unreliable or risky requests.

The dissertation then turns to non-verbal and holistic interaction. I propose a system that equips robots with configurable social personas: the robot interprets user gestures, proximity, and affect, and selects coordinated gaze, facial expressions, and body motions to convey a consistent character and intent. User studies show that people form stable impressions of these personas and that non-verbal behavior strongly shapes perceived alignment and trust. Building on this, I present a motion–language model that treats interactive motion as a first-class generative modality. By learning a shared representation for language and interactive motion, the model supports tasks such as describing interactive motion in natural language, generating physically and socially coherent motion from text, and editing existing trajectories according to relational or stylistic constraints.

Taken together, these contributions illustrate how feasibility-aware planning, ambiguity-sensitive language grounding, and socially expressive motion can be framed within a common multimodal perspective, even though they do not yet form a single unified foundation model. The work highlights remaining gaps, including dependence on specific embodiments, limited task and environment diversity, and the separation between verbal and non-verbal components, and suggests future directions for integrating these pieces into more unified models, extending them to richer sensory modalities and longer-horizon interactions, and addressing the practical challenges of deployment in everyday human environments.

Keywords: foundation models, human–robot interaction, uncertainty estimation, robust reasoning, multimodal learning, generative interaction, social robotics

인간 중심 로봇 행동을 위한 다중모달 기반모델을 향하여

박 정 은
인 공 지 능 학 과
지 도 교 수: 최 성 준

국문 초록

일상적인 공간을 인간과 공유하는 로봇은 단순히 저수준 동작을 정확히 실행하는 것만으로는 충분하지 않다. 이들은 불충분하게 명시된 언어를 해석하고, 실제 환경에서 과업이 실패할 수 있는 경우를 미리 예측하며, 자신의 의도를 언어적 · 비언어적 행동 모두를 통해 표현할 수 있어야 한다. 최근의 언어 및 시각 분야 파운데이션 모델은 인상적인 일반화 성능을 보이지만, 인간 중심 로봇 행동이 요구하는 실질적 · 사회적 조건을 중심으로 설계된 경우는 드물다.

본 학위논문은 인간 중심 로봇을 위한 멀티모달 파운데이션 모델에 어떤 능력과 구조가 필요하며, 이를 향해 어떤 구체적 단계를 밟을 수 있는지 탐구한다. 먼저 멀티모달 지각 및 표현, 물리적 실행 가능성과 과업 성공에 대한 추론, 상호작용적 언어 사용, 비언어적 표현력, 안전성과 예측 가능성 등 여러 축(axis)에 걸쳐 설계 요구 사항을 제시한다. 이러한 관점은 파운데이션 모델이 사용자가 *무엇을* 요구했는지만이 아니라, 그 요구가 *실제로 수행 가능한지*, 그리고 그 결과와 한계를 사용자에게 *어떻게* 전달해야 하는지를 함께 추론해야 함을 강조한다.

이 관점 아래에서, 본 논문은 언어 · 지각 · 행동을 연결하면서도 불확실성과 실패를 명시적으로 고려하는 모델을 통해 언어적 상호작용을 다룬다. 성공 · 실패가 혼재된 시연 데이터로 학습된 계층적 비전–언어–행동(vision–language–action, VLA) 모델

을 제안하여, 로봇이 언어적으로 타당할 뿐 아니라 현재 환경에서 성공 가능성이 높은 서브골을 선호하도록 한다. 이와 보완적으로, 사용자 명령을 명확함(clear), 모호함(ambiguous), 불가능함(infeasible)으로 분류하고, 보정된 불확실성 추정을 이용해 신뢰하기 어렵거나 위험한 요청에 대해서는 무작정 실행하는 대신, 언제 추가 질문을 하거나 한계를 설명해야 할지 결정하는 명령 이해 프레임워크를 제안한다.

이후 논문은 비언어적 · 총체적 상호작용으로 초점을 확장한다. 먼저, 로봇에 구성 가능한 사회적 페르소나를 부여하는 시스템을 제안한다. 이 시스템은 사용자의 제스처, 거리, 정서 상태를 해석하고, 시선, 얼굴 표정, 신체 동작을 조합하여 일관된 캐릭터와 의도를 전달하는 비언어적 행동을 선택한다. 사용자 연구 결과, 사람들은 이러한 페르소나에 대해 안정적인 인상을 형성하며, 비언어적 행동이 로봇에 대한 정렬감과 신뢰감 형성에 강하게 영향을 미침을 확인하였다. 이를 바탕으로, 본 논문은 상호작용 모션을 1급 생성(modality)으로 취급하는 모션–언어 모델을 제시한다. 이 모델은 언어와 상호작용 모션에 대한 공유 표현 공간을 학습함으로써, 상호작용 모션을 자연어로 설명하기, 텍스트로부터 물리적 · 사회적으로 그럴듯한 모션 생성하기, 관계적 또는 스타일 제약에 따라 기존 궤적을 편집하기와 같은 다양한 과업을 지원한다.

종합적으로, 본 논문은 실행 가능성을 고려한 계획(feasibility-aware planning), 모호성에 민감한 언어 그라운딩, 사회적으로 표현력 있는 모션을 공통된 멀티모달 관점 안에서 다룰 수 있음을 보이지만, 이들이 아직 단일 통합 파운데이션 모델을 형성하는 것은 아님도 분명히 한다. 제안된 기법들은 특정 로봇 형태에 대한 의존성, 과업 및 환경 다양성의 한계, 언어적 · 비언어적 구성 요소 간 분리와 같은 남은 과제를 드러낸다. 이에 따라 향후 연구 방향으로, 이러한 요소들을 보다 통합된 모델로 묶어내고, 더 풍부한 감각 모달리티와 장기 상호작용으로 확장하며, 일상적인 인간 환경에서 실제로 배치 · 운용하는 과정에서 발생하는 실질적 문제들을 해결해야 할 필요성을 논의한다.

중심어 : 파운데이션 모델링, 인간–로봇 상호작용, 불확실성 추정, 강건 추론, 멀티

모달 학습, 생성적 상호작용, 사회적 로봇

Preface

This thesis is an original intellectual product of the author, Jeongeun Park. The research presented in this thesis was conducted in the Robot Intelligence Lab at Korea University (Seoul campus).

A version of Chapter 3 has appeared in part as a workshop publication at the Workshop on Safe and Robust Robot Learning in the Conference of Robot Learning, 2025, under the title “Hierarchical Vision Language Action Model Using Success and Failure Demonstrations”. This work was done in collaboration with Jihwan Yoon, Byungwoo Jeon, Juhan Park, Jinwoo Shin, Namhoon Cho, Kyungjae Lee, Sangdoo Yun, and Sungjoon Choi.

A version of Chapter 3 has been published in IEEE Robotics and Automation Letters (RA-L), 2024, under the title “CLARA: Classifying and Disambiguating User Commands for Reliable Interactive Robotic Agents.” This work was done in collaboration with Seungwon Lim, Joonhyung Lee, Sangbeom Park, Minsuk Chang, Youngjae Yu, and Sungjoon Choi.

A version of Chapter 4 has been published in IEEE Robotics and Automation Letters (RA-L), 2024, under the title “Towards Embedding Dynamic Personas in Interactive Robots: Masquerading Animated Social Kinematic (MASK).” This work was done in collaboration with Taemoon Jeong, Hyeonseong Kim, Taehyun Byun, Seungyoun Shin, Keunjun Choi, Jaewoon Kwon, Taeyoon Lee, Matthew Pan, and Sungjoon Choi.

A version of Chapter 5 has been published in the Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV), 2025, under the title “A Unified

Framework for Motion Reasoning and Generation in Human Interaction." This work was done in collaboration with Sungjoon Choi and Sangdoo Yun. This work was conducted as part of the NAVER AI Lab internship program.

I extend my gratitude to my collaborators for their invaluable contributions and support throughout these projects.

Acknowledgment

First and foremost, I would like to express my deepest gratitude to my advisor, Prof. Sungjoon Choi, for his continuous guidance, support, and patience throughout my Ph.D. journey. His insightful feedback, high standards, and encouragement have shaped not only this dissertation but also how I think about research.

I am also sincerely grateful to Dr. Sangdoo Yun, Prof. Matthew Pan, Prof. Kyungjae Lee, Dr. Minsuk Chang, Prof. Youngjae Yu, and Prof. Namhoon Cho for their valuable advice, discussions, and support at various stages of this work.

I also thank all members of our lab for their dedication, insightful feedback, and encouragement throughout this journey. In particular, I am grateful to Jihwan Yoon, Taerim Yoon, Taemoon Jeong, Seungmin Kim, Juhan Park, Chanwoo Kim, Yoonbyung Chai, Hyeonseong Kim, and Taehyun Byun for many discussions, late-night debugging sessions, and their constant support. I am especially grateful to Jihwan for his generous help in preparing my defense presentation. In addition, I appreciate all collaborators and co-authors for sharing their ideas, data, and time, which greatly broadened and enriched this dissertation.

Beyond the lab, I am deeply thankful to my friends who have supported me with understanding, humor, and care, and to those who have stayed by my side outside of work, especially during the most challenging periods of my Ph.D. life. Finally, I owe the greatest gratitude to my parents and family. Their unconditional love, patience, and belief in me made it possible to pursue and complete this work. This dissertation is dedicated to them.

Table of Contents

Abstract **i**

국문초록 **iv**

Preface **vii**

Acknowledgment **ix**

Table of Contents **ix**

List of Tables **xx**

List of Figures **xxix**

1 Introduction **1**

1.1 Motivation and Background 1

1.2 Multimodal Foundation Model for Human-Centered Robot Behavior . . . 3

1.2.1 What Are Foundation Models? 3

1.2.2 Multimodal Foundation Model for Robot Behavior 4

1.2.3 Scope of Human-Centered Robot Behavior 4

1.2.4 Multimodal Foundation Models for Human-Centered Robot Behavior 5

1.3 Key Capabilities and Design Requirements 6

1.3.1 Multimodal Perception and Representation 7

1.3.2 Verbal Interaction . . . 7
1.3.3 Non-Verbal Interaction and Embodied Motion . . . 8
1.3.4 Integrated Multimodal Reasoning and Decision-Making . . . 8
1.3.5 Learning, Adaptation, and Data Requirements . . . 9
1.3.6 Safety, Predictability, and User Trust . . . 9
1.4 Scope of This Dissertation . . . 10
1.5 Thesis Outline . . . 11
1.6 Contributions . . . 13
1.7 Additional Papers . . . 15

2 Related Work **17**
2.1 Foundation Models . . . 17
2.2 Leveraging Large Language Models in Robotics . . . 21
2.3 Vision–Language–Action Model . . . 22
2.4 Non-verbal Interaction and Persona in Robots . . . 24
2.5 Motion–Language Model . . . 25
2.6 Summary and Positioning of This Dissertation . . . 26
2.7 List of Abbreviations . . . 27

3 Verbal Interaction **29**
3.1 Motivation and Background . . . 29
3.2 Failure-Aware Hierarchical Vision–Language–Action Planning . . . 35
3.3 Problem Formulation for Failure-Aware Hierarchical VLA . . . 36
3.3.1 Hierarchical VLA . . . 37
3.3.2 SMDP Formulation . . . 39
3.3.3 Option-Level Representation: Nodes, Edges, and Data . . . 40

3.3.4 Option-Level Feasible Decision Making Formulation 41
3.3.5 Value as Success Probability 42
3.4 Failure-Aware Hierarchical VLA Integrating Negative Experience 43
3.4.1 Overall Pipeline 44
3.4.2 Architecture 45
3.4.3 System 2 46
3.4.4 System 1 48
3.5 Experimental Evaluation of Failure-Aware Hierarchical VLA 51
3.5.1 Teleoperation Data Environments 52
3.5.2 Main Results 54
3.5.3 Test-Time Scalability 59
3.5.4 Bootstrapping Pretrained Models 60
3.5.5 Real-world Demonstration 61
3.5.6 Plug-in Module for Replanning 64
3.5.7 Limitations 65
3.6 Handling Uncertain User Commands through Human Interaction 67
3.7 Problem Formulation: Recognizing and Categorizing Command Uncertainty 70
3.8 Proposed Method for Handling Uncertain Commands with User Interaction 70
3.8.1 Uncertainty Estimation of LLMs via Context Sampling 71
3.8.2 Classification and Disambiguation 73
3.9 Experimental Evaluation of Interaction-Based Resolution 74
3.9.1 Baselines 75
3.9.2 Situational Awareness for Goal Classification in Robotic Tasks . . 76
3.9.3 Pick and Place Simulations 79

3.9.4 Real-World Demonstrations . . . 82
3.9.5 Limitations . . . 85
3.10 Discussion . . . 86
3.10.1 Summary . . . 86
3.10.2 Future Directions . . . 87
3.10.3 Future Directions . . . 88
4 Non-verbal Interaction 89
4.1 Motivation and Background . . . 89
4.1.1 Summary of Contributions . . . 91
4.2 Persona-Embedded Non-verbal Interaction System . . . 91
4.3 Proposed Method for Persona-Embedded Non-verbal Interaction System . 93
4.3.1 Non-Verbal Cues . . . 94
4.3.2 Automated Persona Infuser via LLMs . . . 94
4.3.3 Perception Engine . . . 97
4.3.4 Behavior Selection Engine . . . 99
4.3.5 Action Library . . . 101
4.3.6 Persona-Based Design Outcomes . . . 101
4.4 Experiments for Persona-Embedded Non-verbal Interaction System . . . 102
4.4.1 Experimental Setup . . . 102
4.4.2 Human Subject Study Results . . . 104
4.5 Discussion . . . 109
4.5.1 Human's Expectation Towards Robot . . . 110
4.5.2 Anthropomorphism . . . 110
4.5.3 Limitations . . . 111

4.5.4 Future Directions 112

5 Toward a Holistic Foundation Model for Verbal and Non-Verbal Interaction 113

5.1 Motivations and Background 113

5.1.1 Summary of Contributions 115

5.2 Unified Framework for Interactive Motion-Language Modeling 115

5.3 Inter-MT^2: Interactive multi-turn motion-text dataset 118

5.4 MoLaM: Interactive Motion-Language Model 121

5.4.1 Notations 121

5.4.2 Architecture 122

5.4.3 Training 123

5.5 Experiments for Motion-Language Model 126

5.5.1 Evaluation Tasks and Baselines 126

5.5.2 Motion Reasoning 129

5.5.3 Motion Editing 131

5.5.4 Traditional Motion Related Tasks 132

5.5.5 Generating Multi-Human Motions 134

5.6 Discussions 135

5.6.1 Limitations 135

5.6.2 Future Directions 136

6 Conclusion 137

6.1 Summary 137

6.2 Synthesis of Thesis Contributions 139

6.3 Progress Toward the Long-Term Vision 140

6.4 Overall Impact Statement 142

6.5 Future Directions 144

Reference **147**

Appendix **174**

A Appendix 1: Classifying and Disambiguating User Commands **174**

A.1 Preliminaries on LLM-Based Uncertainty Estimation 175

A.1.1 In-Context Learning as Implicit Parameter Update 175

A.1.2 Dropout as Bayesian Variational Approximation 176

A.2 Theorem and Proof 178

A.3 Failure Cases 179

A.4 Toy Example 184

A.5 Demonstration in Mobile Manipulation Task 187

A.6 SaGC Data collections 192

A.7 SaGC Data Examples 194

A.8 Supervised Learning Baselines 197

A.9 Vision Language Model 197

A.10 Real-world Experiments Environment and Demonstrations 198

A.11 Additional Results on SaGC Dataset 201

A.12 GPT-Info Evaluation Examples 202

B Appendix 2: Training Vision–Language–Action Model with Mixed Quality Dataset **207**

B.0.1 Detailed Formulation 207

B.0.2 Dataset Details 211

B.0.3 Ablation Studies 218

C Appendix 3: Persona-Based Interactive Robot System **221**
C.1 Human Subject Studies Protocols 221
C.2 Additional Results 223
C.3 Prompts 228
D Appendix 4: Unified Framework for Interactive Motion Reasoning and Generation **232**
D.1 Motion Representation and Motion Token Representation 233
D.2 Ablation Studies on Pretraining Method 234
D.3 Ablation Studies on Motion Tokenizer 236
D.4 Expansion to Multi-Human Motion (≥ 3) Generation 237
D.5 Illustration on Data Collection Pipeline 238
D.6 Additional Results for Traditional Motion Related Tasks 239
D.6.1 Motion to Text 239
D.6.2 Text to Motion 240
D.6.3 Reaction Generation 241
D.7 Limitations and Impact Statement 242
D.8 Implementation Details 243
D.9 Implementation details for Baselines* 244
D.10 Detailed Task Explanations 245
D.11 Detailed Explanation about Two-stage Baselines 246
D.11.1 Motion Editing 246
D.11.2 Motion Reasoning 247
D.12 More details about Evaluation Metric for Traditional Motion Related Tasks 247
D.13 Template Forms for Pre-training and Instruction Tuning 248

D.13.1 Pre-training Templates 249
D.13.2 Instruction-Tuning Templates 250
D.14 Data Sample Visualization 250
D.15 Inter-MT^2 Statistics 250
D.16 Qualitative Results 252
D.17 User Subject Studies Protocols for motion editing 257
D.17.1 Instructions 257
D.17.2 Qualifying test 259
D.17.3 Detailed Survey Format 261

List of Tables

3.1 Simulation Results in plug insertion and drawer packing environments, showing the success rate. 54

3.2 Results from Simpler Environment. Carrot. denotes carrot on plate, eggplant. denotes eggplant in basket, and spoon. for spoon on towel. 60

3.3 Real-world success rates on sponge and towel packing tasks. Our method (VINE) outperforms the π_0 baseline in both seen and unseen settings. . . 62

3.4 Results on SaGC Dataset. Quanantification denotes the uncertainty quantification part, where we measured AUROC, and Classification denotes the classification part where we measured accuracy. Abla. is an ablation study on uncertainty estimation, where CS. denotes context sampling and UAP. denotes uncertainty-aware prompt. 77

3.5 Task explanation in Pick-And-Place Environment 80

3.6 Uncertain Task Detection in the pick-and-place environment. The *Quantification* block reports AUROC for different uncertainty measures; the *Ablation* block varies CLARA's components on Inst. GPT (CS = context sampling, UAP = uncertainty-aware prompt). 81

3.7 Disambiguation on Pick-and-Place Environment. Gap denotes the success rate gap after interaction on ambiguous commands. 81

3.8 Task explanation for the real-world experiment. 83

3.9 Uncertain Task Detection in Real-World Environment (AUROC). 83

3.10 Classification and Disambiguation in Real-World Environment. Cls. denotes uncertainty type classification and Disamb. denotes the disambiguation progress. 84

4.1 Non Verbal Cues. The cues for human observation include four factors with 72 possible combinations, while robot cues contain 13 motions and 12 facial expressions with 156 possible combinations. 95
4.2 Top 3 mentioned keywords for each persona from survey question 4 (**S4**). 109

5.1 Comparison of generated motions on text-matching ability (top-3 retrieval precision), and motion diversity (Div.). 118
5.2 Evaluation on Motion Reasoning task with Inter-MT^2 test set. Coh., Align., and Nat. denote logical coherence, content alignment, and naturalness, respectively. **Bold** indicates best performance and underline denotes the second best performance. 129
5.3 Quantitative results in motion editing task. 131
5.4 Comparisons for three motion-related tasks on Inter-X and InterHuman datasets. M2T denotes motion-to-text, T2M for text-to-motion, and Reaction Gen. for reaction generation. 133

A.1 Supervised Learning Baselines 198
A.2 Task explanation for the real-world experiment. 201
A.3 Success Rates on real-world environment. C. Por., A. Por., I. Por. denotes a portion of question generation on clear, ambiguous, and infeasible commands respectively. 202
A.4 Informativeness of generated questions and reasons 202

B.1 Simpler environment outcomes per object. Rates are ♯ / total (%). 216
B.2 Weight merging for language backbone in plug insertion environment. Interpolating shared layers between π_0 and PaliGemma via λ_m yields a clear optimum at $\lambda_m = 0.6$; heavier bias to either model reduces overall and unseen success. 219

D.1 Ablation studies in pertaining stage for three motion-related tasks on InterX and Interhuman dataset. 234
D.2 Ablation Studies on motion tokenizer base model. We compared VQ-VAE-based tokenizer and the RQ-VAE-based model. 236
D.3 Motion-to-Text 239
D.4 Text-to-Motion 240
D.5 Reaction Genereration 241
D.6 Template for Pretraining 248
D.7 Template for Instruction Tuning 249
D.8 Statistics on Inter-MT^2. 250

List of Figures

1.1 Key capabilities and design requirements of foundation models for human-centered robot behavior, integrating multimodal inputs into a shared representation for perception, reasoning, and action, and producing verbal, embodied, and safety-aware responses from diverse multimodal data. . . . 6

1.2 Overview of holistic, human-centered robot behavior. The robot foundation model receives linguistic commands and visual observations from humans and the environment, and produces both language-based and nonverbal actions. 12

3.1 Language-based human–robot interaction. A human provides high-level verbal instructions, which are grounded by the robot through a model that interprets language and visual observations to produce task-directed actions in the environment. 30

3.2 Overview of the dual pathways in our framework. When the instruction and scene jointly provide enough information, the robot selects the most feasible action based on multimodal evidence (left). When the instruction is ambiguous or underspecified, the robot initiates conversational queries to resolve uncertainty by asking the user (right). 31

3.3 Standard VLAs trained only on success data may produce infeasible trajectories. VINE leverages both success and failure trajectories with a failure-aware reasoning planner, yielding more robust executions. 37

3.4 Architecture and overall pipeline. System 2 plans via tree-of-thoughts, predicting success values to select the highest-value path (green). System 1 executes the chosen subgoal as action chunks with a done signal. Both share a vision–language backbone with adapters. 44
3.5 Plug insertion and drawer packing environment. We evaluate in both seen and unseen settings with corresponding assets, and collect demonstrations via a teleoperation setup. 51
3.6 Qualitative results. System2 builds a failure-aware planning tree with feasibility scores, and System1 executes the selected subgoals, producing successful trajectories. 57
3.7 Trajectory Comparison with baselines. The figure shows the trajectory generated by π_0 [1], VLM as system 2, i.e., Gemini-2.5-flash [2] (Gemini), and with failure examples (Gemini fail), VINE-Chain (chain), and VINE-Full (Ours) in an unseen drawer packing environment. 58
3.8 Tree Analysis for test time scaling across per-step expansion. In the unseen plug-insertion setting, increasing the search width K consistently improves success while increasing inference time, demonstrating a test-time scalability. 59
3.9 Generated tree and trajectories in Simpler environment. 60
3.10 Real-world environment and tasks. **(a)** Setup with a 6-DoF arm (joint-position control, wrist + external cameras). **(b)** Tasks: sponge and towel packing in the cabinet, with distractors and folding strategies. Generalization is tested from *seen* to *unseen* color configurations. 62

3.11 Real-world demonstrations with planning and execution. **(a)** *Towel–unseen*: the search tree selects *fold* → *put in cabinet* → *close*, then System 1 executes the chosen path. **(b)** *Sponge–unseen* (clock distractor): the planner chooses *push clock* → *put sponge* → *close* and the trajectory follows accordingly. 63

3.12 Success rates in the unseen drawer packing environment with and without replanning. A lightweight uncertainty-triggered replanning module, which down-weights uncertain branches and switches to alternatives, boosts success in drawer packing without retraining. 65

3.13 Qualitative Results for replanning. Planner first selects the green plan (*push box*→*place item*→*close*), then detects high uncertainty during the first step and switches to the yellow alternative (*place box on table*→*place item*→*close*), successfully completing the task. Shaded panels indicate the uncertainty trigger and the plan switch; bottom rows show the executed trajectories. 66

3.14 Proposed Method. Our method involves estimating uncertainty with LLMs via context sampling to distinguish between certain and uncertain commands. We then leverage situational awareness to classify uncertain commands into ambiguous and infeasible categories, followed by a disambiguation process for ambiguous commands. The number (1) (2), etc., denotes the index of the context from the context set (C). σ denotes predictive uncertainty, and ε is an uncertainty threshold. 69

3.15 Statistics and Examples of the Dataset. Cer. denotes certain, Inf. denotes infeasible and Amb. denotes ambiguous 77

3.16 Examples of generated explanation and question from the proposed method. 😃F, 🤔R, 😟Q means Feasibility, Reasoning, and Question, respectively. . 80
3.17 Real-world demonstrations. 😃F, 🤔R, 😟Q means Feasibility, Reasoning, and Question, respectively. 85

4.1 Overview of non-verbal human–robot interaction. Human social signals such as gestures, gaze, and interpersonal distance are interpreted by the model and re-expressed by the robot according to a selected persona, enabling socially meaningful non-verbal communication. 89
4.2 The proposed system architecture. The system is composed of a perception engine, a behavior selection engine, and an action library, where the action library and the behavior databases are pre-built components. 93
4.3 Illustration of the proposed Persona Infuser, which generates the behavior database via LLM. 96
4.4 Demonstrations of the system. The first and second rows demonstrate the personality-based persona interaction, while the third and fourth rows demonstrate the character-based persona interaction. The emojis on the top indicate the facial expression that is displayed on the robot's screen. The caption under each image denotes the motion name. 100

4.5 Results from survey phase 1 with questions S1 and S2. (a) E-score of extroverted factor and A-score of agreeable factor. (b) E-score of extroverted persona with agreeable persona. (c) A-score of agreeable persona with extroverted persona. Agr., Dis., Ext., and Int. denote agreeable, disagreeable, extroverted, and introverted personas, respectively. * as $0.01 < p < 0.05$, ** as $p < 0.01$, and *** as $p < 0.001$. The error bars represent 95% confidence intervals. 105

4.6 Impact of facial expression. (a) extroversion score of extroverted persona factor with face factor. (b) Agreeableness score of agreeable persona factor with face factor. * denotes $0.01 < p < 0.05$, ** denotes $p < 0.01$, and *** denotes $p < 0.001$. 106

4.7 Classification Confusion Matrix for the character-based persona representing the characters *Cowardly Lion (LO), Minion (MM), Scrooge (SC), Spock (SS), Sloth (SZ), and Captain America (CA), respectively.* 107

5.1 Holistic multimodal interaction between humans and robots. Language and motion are treated as bidirectional communication channels: the model interprets verbal and non-verbal cues from the human and generates socially coherent motion responses for the robot from a shared representation space. 114

5.2 We introduce **MoLaM**, the Versatile Interactive Motion-language model, a unified architecture that combines language and motion for two-person interactive scenarios. The figure highlights its capabilities across various tasks, including motion-to-text, text-to-motion, reaction generation, motion editing, and multi-turn motion reasoning, all within a single framework. 116

5.3 Statistics and data sample from Inter-MT^2. . . . 119

5.4 An overview of MoLaM, illustrating its capability to flexibly process and generate interactive motions and texts in an auto-regressive manner. We omit the motion tokenizer, which converts raw motion sequences into discrete motion tokens, for clarity. MoLaM covers versatile motion tasks involving both motion and textual modalities across multiple conversational turns. . . . 121

5.5 Tokenization of interactive motions. . . . 122

5.6 Generated samples for interactive motion reasoning task. This example shows how MoLaM explains behaviors and their motivations, demonstrating a deeper understanding of scenarios by incorporating context from prior interactions. . . . 129

5.7 User subject study results for motion editing. We plotted the difference only in a post hoc pairwise comparison of the proposed method. * as $0.01 < p < 0.05$, ** as $p < 0.01$, and *** as $p < 0.001$. The error bars represent 95% confidence intervals. . . . 130

5.8 Generated samples for interactive motion editing. The proposed method excels in capturing nuances, outperforming alternatives in content similarity and instruction alignment. . . . 132

5.9 Expanding MoLaM to generate multiple human motions. For clarity, we simplified the incremental process, where MoLaM first generates a two-person motion and adds a third person to it. . . . 134

A.1 Failure Cases due to other modules . . . 183

A.2 Comparison in Vision-Language Models . . . 199

A.3 Failure Cases in Real-world Environment 200
A.4 Real-world Experiment Setting 200
A.5 Snapshot1 204
A.6 Snapshot2 205
A.7 Generated output on the additional samples 206

B.1 Data Statistics from human demonstrations in plug insertion environment 212
B.2 Data Statistics from human demonstrations in the drawer packing environment 214
B.3 Data Statistics from human demonstrations in the real-world environment 217
B.4 Comparison between tree search algorithms. On plug insertion, MCTS's value-guided search yields a similar success rate and faster inference than fixed-order DFS. 219

C.1 Satisfaction among Character Personas 224
C.2 Impact of Facial Expression (Personality Persona) 226
C.3 Effect of Motions and Facial Expressions (Personality Persona) 227

D.1 Ablation Studies on codebook size and depth. We measured Top-3 retrieval-precision accuracy and FiD on the reconstructed motion. 237
D.2 Our method demonstrates its extendability by generating multi-person interactions (≥ 3 people) through iterative prompting. Despite being trained on two-person scenarios, our framework conditions new motions on prior interactions, enabling the synthesis of natural group dynamics from textual descriptions. 237

D.3 Overview of synthetic data generation for multi-turn conversations with interactive motions. (a) Motion captions and instructions are generated using GPT-4o based on interactions between two characters, followed by (b) the corresponding motion being synthesized using the InterGEN. . . . 238
D.4 Method Overview. Stage 1 involves training a motion tokenizer that encodes and decodes interactive motion data. In Stage 2, we pre-train the model by integrating motion and text data, allowing it to learn the alignment between text and motion. Stage 3 focuses on Instruction Tuning, fine-tuning the model to follow instructions and improve its responsiveness to conversational cues. 243
D.5 Sample from Inter-MT^2 dataset. The left column visualizes samples of motion editing, and the right column shows examples from the motion reasoning task. 251
D.6 Generated motion from source and editing command. 253
D.7 Generated responses based on the previous conversations for motion reasoning task. 254
D.8 Motion-to-text results. The blue part is generated motion captions from source motions. 255
D.9 Text-to-motion results. The blue part is generated motions from the motion caption. 255
D.10 Reaction Generation. The input motion is orange, while the generated reactive motion is colored blue. 255
D.11 We demonstrate the ability of the proposed method to generate responses in long conversations and also the failure cases. 256
D.12 The examples of ratings given to the user 259

D.13 Qualifying test in user subject studies 260
D.14 Survey Example 261

Chapter 1

Introduction

1.1 Motivation and Background

Robots are steadily moving out of isolated industrial cells and into spaces that are fundamentally designed for people: homes, hospitals, workplaces, and public environments. In these settings, the expectation is not merely that a robot can complete a scripted task, but that it can collaborate, adapt, and behave in ways that feel intelligible and appropriate to human partners. This requires more than competence at manipulation or navigation. A useful partner must also exhibit elements of social intelligence: understanding how instructions relate to the shared environment, recognizing when a request is unclear or unsafe, and expressing its own intentions and limitations through both language and behavior. In other words, robots entering human spaces are expected to participate in interaction, not just execute motion plans.

Recent progress in robot foundation models has focused primarily on vision–language–action (VLA) architectures that aim to generalize across tasks, embodiments, and environments. By training large models on diverse teleoperation data and web-scale vision–language corpora, these systems can map images and natural-language goals to low-level actions with impressive breadth. However, their success is usually measured in terms of task success rates and robustness under perturbations within a manipulation benchmark. As a result, the notion of "generalization" remains largely task-centric: models are evaluated on whether they can perform more tasks on more objects, rather than on whether they can participate in richer forms of human–robot interaction.

At the same time, most practical HRI systems remain highly modular and hand-engineered. Pipelines often stitch together separate components for speech recognition, natural-language understanding, dialog management, perception, planning, and motion control, with carefully tuned rules and state machines governing when each piece is invoked. This design makes it possible to build compelling demonstrations, but it introduces brittle interfaces and limits how much behavior can emerge from data. When a system is decomposed into many narrow modules, it becomes difficult to learn representations that carry social meaning across levels: how a spoken request, a pointing gesture, a spatial relation, and a response motion should jointly constrain what the robot does next. These challenges motivate a shift from handcrafted HRI pipelines toward more integrated models of interaction.

This dissertation is driven by the view that future robots should be built on *foundation models for multimodal reasoning in human-centered robot behavior*: models that treat language, perception, and action, including non-verbal motion, as coupled channels of communication with humans. Rather than designing separate modules for understanding, planning, reacting, and expressing, I seek a path where a shared multimodal representation supports all of these roles: interpreting verbal commands in context, reasoning about feasibility and safety, expressing uncertainty, or asking for clarification, and generating coherent robot motions in response to a human partner. The long-term vision is that a single family of models can underpin verbal interaction, non-verbal interaction, and their combination in human environments, enabling robots that are not only more capable across tasks, but also better aligned with how humans naturally communicate and collaborate.

1.2 Multimodal Foundation Model for Human-Centered Robot Behavior

In this section, I clarify what I mean by foundation models and how they apply to robots acting in human environments. I first review the general notion of foundation models and their characteristic training and adaptation regimes, then characterize multimodal robot behavior in this context. I next delimit the scope of human-centered robot behavior considered in this dissertation, and finally synthesize these notions into a working definition of multimodal foundation models for human-centered robot behavior that guides the rest of the thesis.

1.2.1 What Are Foundation Models?

Foundation models are large-scale models trained on broad, heterogeneous data that can be adapted to many downstream tasks with relatively modest additional supervision. Instead of being engineered for a single setting, they aim to learn reusable representations and interfaces that transfer across domains, modalities, and objectives. Examples include large language models (LLMs) [2–5] that support in-context learning and prompting across diverse text tasks; vision–language models (VLMs) [6–8] that align images and language in a shared space; vision–language–action (VLA) [1, 9, 10] policies that map from scenes and instructions to robot actions; and emerging motion- and interaction-centric models that operate over trajectories rather than static inputs. Common characteristics of these models are: (i) pretraining on massive, often web-scale datasets; (ii) unified architectures that can ingest and produce multiple modalities; and (iii) flexible conditioning mechanisms, such as prompts, demonstrations, and task descriptions, that allow the same model to be steered toward different behaviors without retraining from scratch.

1.2.2 Multimodal Foundation Model for Robot Behavior

In this dissertation, multimodality concerns how a robot uses heterogeneous signals to interpret situations and choose behaviors, not just how it processes each modality in isolation. Relevant modalities include language (spoken or written instructions, feedback, and explanations), visual observations of scenes and objects, robot and human motion (poses, trajectories, and interaction patterns), proprioceptive state, and, when available, tactile or force feedback. Behavior selection in this context means more than mapping an input vector to an output action: it involves combining environment state, user goals, physical and social constraints, and recent interaction history to (i) understand what is being asked or implied, (ii) decide which actions are feasible and appropriate, and (iii) revise behavior in response to new evidence or corrections. The systems studied in later chapters instantiate different slices of this notion—for example, combining language and perception to decide whether to execute or clarify a command, or combining language and motion to answer questions about an ongoing interaction or to edit a trajectory. They are all framed as special cases of multimodal processing and behavior selection for robots.

1.2.3 Scope of Human-Centered Robot Behavior

The term *human-centered robot behavior* is used here in a deliberately specific sense. I focus on robots operating in human-populated indoor environments such as homes, hospitals, and workplaces, where people are present in the scene and are the primary consumers of the robot's behavior. The behaviors of interest span both task-level and interaction-level phenomena. At the task level, the robot must perform everyday actions such as manipulating objects, tidying or organizing, and assisting with routine activities, while respecting the physical constraints of the environment. At the interaction level, the robot must respond to how the human communicates and behaves: clarifying underspecified instructions, hesitating or yielding when appropriate, adjusting its motion when someone approaches or withdraws, and choosing when to ask questions versus acting autonomously. Human-centered behavior thus emphasizes not only whether a task is completed, but how the robot's actions appear and feel from the user's perspective: whether they are understandable, predictable, respectful of comfort and safety, and aligned with the human's goals and preferences.

1.2.4 Multimodal Foundation Models for Human-Centered Robot Behavior

Putting these elements together, *multimodal foundation models for human-centered robot behavior* are models that (i) are pretrained on large, diverse datasets containing language, perception, and interaction traces, (ii) learn **shared representations that tie together language, vision, motion, and other modalities**, and (iii) expose **flexible interfaces** through which robots can be conditioned to exhibit different behaviors in human environments. The inputs to such models include verbal signals (instructions, questions, feedback), visual and state information about the scene, records of past actions and outcomes, and non-verbal cues such as human pose or gesture. The outputs span verbal responses (clarifications, explanations, confirmations) and embodied behavior (where to move, how to manipulate, how to shape motion as a signal), with the model responsible for deciding which channel to use and how to coordinate them. From a design standpoint, these foundation models act as a shared substrate: they provide a common multimodal latent space and a set of decision and control heads that can be adapted, via prompting or

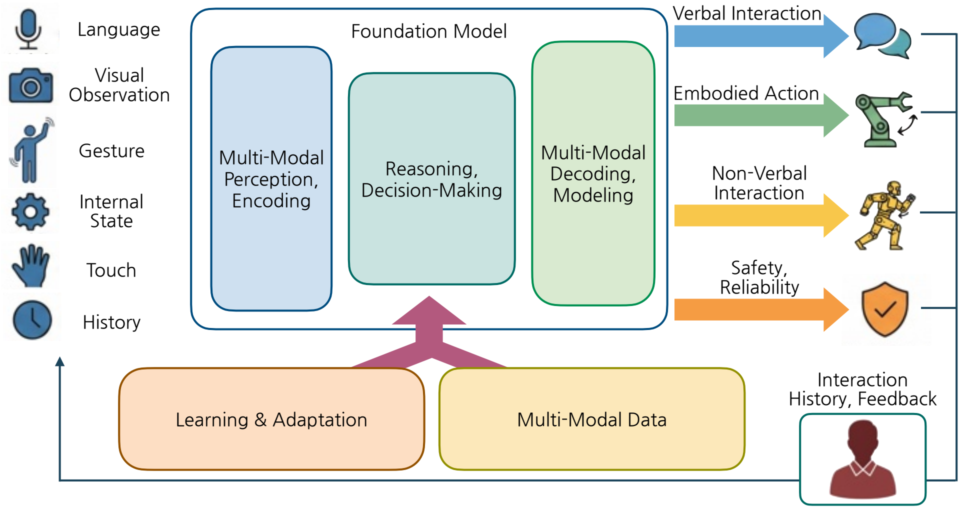


Figure 1.1: Key capabilities and design requirements of foundation models for human-centered robot behavior, integrating multimodal inputs into a shared representation for perception, reasoning, and action, and producing verbal, embodied, and safety-aware responses from diverse multimodal data.

lightweight fine-tuning, to specific robots, tasks, and user populations. The remainder of this dissertation explores concrete instances of this paradigm, showing how such models can be specialized for verbal interaction, non-verbal interaction, and motion, and how they can be deployed in integrated form in human-centered settings.

1.3 Key Capabilities and Design Requirements

Multimodal Foundation models for human-centered robot behavior are not designed for a single task or interface. Instead, they are intended to support a broad family of behaviors through a shared, reusable representation that enables transfer and generalization across tasks, embodiments, and environments. To do so, they must integrate multiple input and output modalities, operate over extended interaction histories, and make decisions that remain intelligible and acceptable to humans. In this section, I outline several key capability axes and the resulting design requirements: multimodal perception and representation, verbal interaction, non-verbal interaction and embodied motion, integrated multimodal reasoning and decision-making, learning and adaptation from diverse data, task-level generalization, and safety, predictability, and user trust, illustrated in Fig. 1.1.

1.3.1 Multimodal Perception and Representation

A human-centered robot rarely acts on a single stream of information. It must interpret language, visual observations, internal state, tactile and force feedback, and motion histories together to understand what is happening and what should be done next. Foundation models in this setting, therefore, require multimodal encoders that can absorb heterogeneous signals and embed them into a shared representation space. Such representations should align language with visual context, motion, and contact-rich sensing, capture temporal structure over extended interactions, and remain compact enough to serve as a useful state for downstream planning and control. This motivates architectures with modality-agnostic backbones and lightweight adapters for different sensory streams, pretrained on large, heterogeneous datasets that mix images, videos, language, action traces, and, where available, haptic or proprioceptive signals.

1.3.2 Verbal Interaction

Verbal interaction is the primary way many users express goals, preferences, and constraints to a robot, but natural-language instructions are often ambiguous, incomplete, or partly incompatible with the current situation. Instead of treating language as a one-shot command, foundation models must support grounded interpretation of verbal input, relating referring expressions and relational phrases to the observed environment and interaction history. They should also detect when a request is underspecified or infeasible and respond through clarification, explanation, or refusal, rather than blindly executing a literal parse. This requires coupling language understanding with high-level reasoning over goals, subgoals, and constraints, so that the dialog can be used to negotiate and refine what the robot will actually do.

1.3.3 Non-Verbal Interaction and Embodied Motion

In human environments, much of what matters is never spoken: gaze, posture, gesture, and movement in space all convey intent, attention, and comfort. A human-centered robot must therefore treat motion as both an action and a communicative signal. On the perception side, this means interpreting human motion over time—recognizing approach versus avoidance, turn-taking, offering help, or hesitation—and using these cues to adjust behavior. On the generation side, the robot's own motions should be physically feasible and task-complete, but also legible and appropriate, revealing intent and respecting social and spatial norms. This calls for representations that capture trajectories at the right level of abstraction and can be adapted online as the human's motion and state evolve.

1.3.4 Integrated Multimodal Reasoning and Decision-Making

Although verbal and non-verbal channels can be analyzed separately, effective behavior ultimately depends on integrating them with perception and control into a coherent decision process. Foundation models for human-centered robots should support joint reasoning over language, vision, motion, and internal state to answer questions such as which object is being referred to, which plan is feasible and low-risk, and whether to act now or seek more input. This involves mapping multimodal histories to high-level decisions, task selection, plan revision, and trajectory adaptation, rather than only predicting the next low-level action. In practice, shared multimodal encoders provide the state abstraction, while planning heads, value estimators, or search procedures operate on that state so that multimodal cues meaningfully influence behavior.

1.3.5 Learning, Adaptation, and Data Requirements

These capabilities cannot be hand-engineered; they must be learned from rich and imperfect experience. Foundation models for human-centered robot behavior should leverage diverse data sources, including scripted demonstrations, teleoperation logs, simulated environments, and real-world interaction traces, ideally containing both successes and failures, corrections, and goal revisions. The model must also support efficient adaptation to new robots, environments, and users, for example, through fine-tuning on small amounts of local data, in-context specialization, or lightweight adapters. This places constraints on how parameters and interfaces are organized: core representations should be reusable, while most customization occurs through modular components that can be updated without retraining the entire model from scratch.

1.3.6 Safety, Predictability, and User Trust

Finally, any realistic deployment in human environments must satisfy safety and trust constraints. Multimodal reasoning increases capability but also opens new failure modes, so foundation models must recognize when information is insufficient, when plans are likely to fail or cause harm, and when deference to a human decision or a conservative fallback is warranted. Behavior should respect physical safety, avoiding collisions and hazardous motions, as well as interaction safety, such as avoiding sudden or intrusive movements and not persisting with plans that clearly make users uncomfortable. From a design standpoint, this calls for explicit mechanisms for uncertainty estimation, risk assessment, and override, and for using language and motion to communicate limitations.

The goal is not only competent behavior, but behavior that people can predict, understand, and gradually learn to trust.

1.4 Scope of This Dissertation

The preceding discussion sketches an ambitious design space: foundation models that perceive rich multimodal input, engage in verbal and non-verbal interaction, integrate these channels into coherent decisions, adapt from diverse experiences, and remain safe and trustworthy in everyday environments. This dissertation does not claim to realize such a general foundation model in full. Instead, it investigates concrete slices of this space where multimodal reasoning already yields tangible benefits for human-centered robot behavior, and it develops methods that move incrementally from *using* large pre-trained models effectively toward *constructing* more specialized multimodal models for robots.

On the capability side, the focus is deliberately narrow in some dimensions and deep in others. I concentrate on language, vision, expressive motion, and embodied actions, rather than addressing rich haptic manipulation. Within this setting, the dissertation targets three axes most directly: (i) verbal interaction under ambiguity, uncertainty, and feasibility constraints; (ii) non-verbal interaction and expressive motion as channels for conveying intent, attitude, and coordination; and (iii) integrated multimodal reasoning over language and motion, with an emphasis on understanding, generating, and editing interactive trajectories, and on answering questions about ongoing or past motion. Multimodal perception, low-level control, learning and adaptation, and safety are addressed only to the extent needed to support these behaviors, and are not treated as standalone research problems.

Methodologically, the contributions span a spectrum from model usage to model design. At one end, I study how to wrap and condition existing large language and vision–language models so that they can interpret underspecified commands, detect infeasible requests, ask clarifying questions, and choose behaviors that are more likely to succeed in a given scene. At the other end, I develop multimodal motion models that align language and interactive trajectories, enabling robots to generate, recognize, and edit social and task-oriented motions in response to a human partner. Across these cases, the emphasis is on how multimodal reasoning, grounded in data from human–robot scenarios, can shape human-centered robot behavior, rather than on training a single monolithic model that exhaustively covers all tasks, embodiments, and environments. Subsequent chapters instantiate this agenda in the verbal, non-verbal, and integrated settings introduced above.

1.5 Thesis Outline

This dissertation is organized to move from general foundations to specific instantiations of multimodal reasoning in human-centered robot behavior. As illustrated in Fig. 1.2, we consider robots that jointly perceive humans and their environment through language and vision, and respond with both verbal dialogue and nonverbal motion, enabling holistic human-centered behavior.

Chapter 2 surveys related work at the intersection of foundation models, robotic learning, and human-robot interaction. I review large language and vision–language models, vision–language–action (VLA) policies, and recent motion- and interaction-centric models, with an emphasis on how they address (or neglect) multimodal reasoning, uncertainty, and human-centered behavior. This chapter situates the proposed agenda within

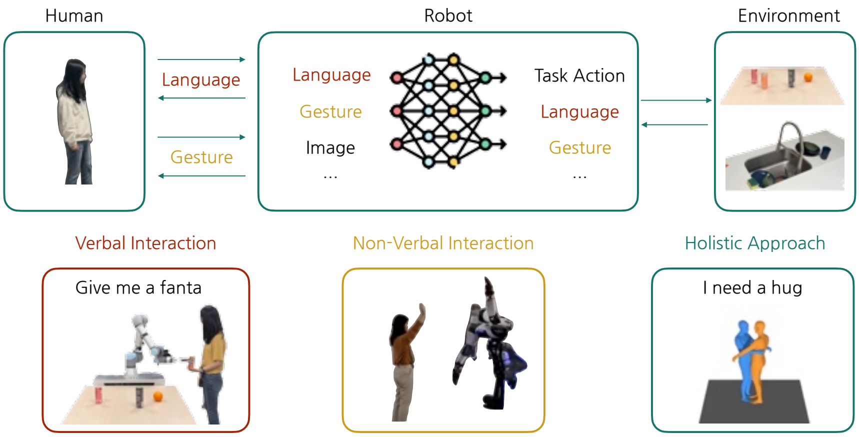


Figure 1.2: Overview of holistic, human-centered robot behavior. The robot foundation model receives linguistic commands and visual observations from humans and the environment, and produces both language-based and nonverbal actions.

both the foundation model literature and the broader robotics and HRI communities.

Chapter 3 focuses on *verbal interaction*. It studies robots that act in response to natural-language commands under ambiguity, uncertainty, and feasibility constraints. I first show how large pretrained models can be wrapped to perform failure-aware, language-conditioned planning in realistic manipulation scenes. I then introduce a framework for classifying user commands into clear, ambiguous, and infeasible categories, and for using clarification questions and explanations to negotiate goals with the user. Together, these case studies illustrate how multimodal reasoning over language and perception can improve the reliability and transparency of verbal interaction.

Chapter 4 turns to *non-verbal interaction and embodied motion*. Here, the robot's behavior is driven primarily by a human partner's pose, gesture, and movement rather than by explicit linguistic commands. I present a model that treats motion as a communicative signal, learning to generate and adapt robot motions that respond to a partner in ways that are physically feasible and socially appropriate. This chapter highlights how motion-level representations can be used to interpret human behavior and to synthesize expressive robot behavior in return.

Chapter 5 develops a *holistic* perspective that unifies language and motion within a single multimodal model. I introduce a framework that aligns language with interactive motion trajectories, enabling a robot to understand, generate, and edit complex interaction sequences from textual descriptions and motion queries. The model supports tasks such as describing what has happened, predicting plausible continuations, and modifying trajectories to satisfy new relational or stylistic constraints, illustrating how multimodal reasoning can span verbal and non-verbal channels in a shared space.

Chapter 6 concludes the dissertation. It summarizes the main contributions across verbal, non-verbal, and integrated multimodal reasoning, reflects on the limitations of the current instantiations, and outlines open challenges for building broader foundation models for multimodal reasoning in human-centered robot behavior.

1.6 Contributions

This dissertation investigates how foundation-model principles can be applied to *multimodal reasoning in human-centered robot behavior*, spanning verbal, non-verbal, and integrated interaction. Rather than proposing a single monolithic model, it develops and analyzes a sequence of systems that use and extend large pretrained models in increasingly specialized ways, from language-driven planning to motion-level interaction. The main contributions are summarized as follows.

1. **Problem formulation and design space for multimodal reasoning in human-centered robot behavior.** I formalize the goal of foundation models for human-

centered robot behavior and articulate key capability axes: multimodal perception and representation, verbal interaction, non-verbal interaction and embodied motion, integrated multimodal reasoning and decision-making, learning and adaptation from diverse data, safety, predictability, and user trust. This framing clarifies how large language, vision–language, and motion-centric models can be repurposed or extended for human-centered robots, and it provides a conceptual scaffold for the subsequent chapters.

2. **Verbal interaction under ambiguity, infeasibility, and failure.** I investigate robots that act in response to natural-language commands while respecting the limits of their perception and capabilities. First, I show how large pretrained models can be wrapped into failure-aware, language-conditioned planners that use both successful and unsuccessful experience to bias plan selection toward feasible behaviors in realistic manipulation scenes. Second, I propose a framework for classifying user commands into clear, ambiguous, and infeasible categories, and for using clarification questions and explanations to negotiate goals with the user, rather than blindly executing literal parses. Together, these contributions demonstrate how multimodal reasoning over language and perception can improve the reliability and transparency of verbal interaction.

3. **Non-verbal interaction and expressive robot motion.** I turn to settings where the primary signals are not spoken commands but human pose, gesture, and movement. I develop an interaction policy that treats motion as a communicative channel, learning to generate and adapt robot trajectories in response to a partner in ways that are both physically feasible and expressive. By conditioning on interaction context and persona-like factors, the method shows how a single motion

model can produce behaviors that differ not only in task outcome but also in how the robot is perceived (e.g., more reserved or more proactive). This illustrates how motion-level representations can support non-verbal interaction beyond scripted or hand-designed behaviors.

4. **Holistic multimodal reasoning over language and interactive motion.** Finally, I present a unified framework that aligns language with interaction-scale motion trajectories, enabling a single model to understand, generate, and edit complex human–robot interaction sequences. The model supports tasks such as describing what has happened, answering questions about past or ongoing motion, predicting plausible continuations, and modifying trajectories to satisfy new relational or stylistic constraints. This demonstrates that foundation-model style training can yield a shared space in which verbal and non-verbal channels are treated symmetrically, allowing robots to reason about and manipulate behavior at the level of language and motion together.

1.7 Additional Papers

While this dissertation centers on multimodal reasoning for human-centered robot behavior, I have also led and contributed to several other lines of work during my Ph.D. that are not covered in detail here. As the first author, I developed Mixture Logit Networks (MLN), an uncertainty-aware robust learning framework that jointly models epistemic and aleatoric uncertainty in classifiers and uses these signals to detect and mitigate noisy labels and corrupted inputs. I then proposed ZAVIS, a zero-shot active visual search system that combines open-set visual detection, text–image alignment, and uncertainty-aware planning to locate target objects specified by free-form language in unseen environments. Building on these ideas, SOCRATES extends visual search from objects to people: it integrates language, vision, and reinforcement learning to find and approach humans described by textual attributes and roles, generating approach behaviors that are physically feasible and socially acceptable.

In addition to these first-author works, I have collaborated on a range of projects related to representation learning and embodied interaction. These include self-supervised visual representation learning for downstream robotic tasks; semi-autonomous teleoperation of manipulators, including non-prehensile manipulation and LLM-based spatial reasoning; tactile human–robot interaction that leverages skin-like sensing; the construction of mixed-quality autonomous driving datasets for abnormal detection; and representation learning of robot structure. Although these projects target different application domains, they share a common theme with this dissertation: extracting useful structure from noisy, multimodal data to support more interpretable, adaptable, and human-compatible robot behavior.

Chapter 2

Related Work

Robotic intelligence increasingly draws from advances in foundation models trained on large-scale multimodal data. This chapter reviews prior work across five closely linked research domains: (i) foundation models, (ii) large language models applied to robotics, (iii) vision–language–action (VLA) models, (iv) non-verbal interaction and persona-driven robotics, and (v) motion–language models. The relationships among these areas outline the progress and remaining challenges motivating this dissertation.

2.1 Foundation Models

Foundation models refer to large-scale neural architectures trained on broad, multimodal data distributions to acquire general-purpose representations that transfer across tasks, domains, and embodiments. Their power comes from three properties: (i) scalable training that yields emergent capabilities, (ii) shared representations that unify heterogeneous modalities, and (iii) versatile conditioning mechanisms that enable downstream adaptation with minimal task-specific engineering. These characteristics have transformed language and vision research and increasingly serve as the conceptual backbone for embodied intelligence.

Large Language Models Large language models (LLMs) display strong compositional reasoning and zero-shot generalization [2–5,11,12], with chain-of-thought prompting [13] and self-reflection [14] supporting more interpretable step-by-step reasoning. However, these abilities remain text-centric—LLMs do not account for embodiment, physics, or the feasibility of executing the decisions they produce.

Vision Language Models Vision–language models (VLMs) couple text with visual perception through shared representations, typically by aligning an image (or video) encoder with a pretrained language model. Models in the LLaVA family [6] use a vision encoder and a lightweight projection layer to map visual features into the language token space, enabling open-vocabulary grounding, visual question answering, and instruction following. Subsequent work extends this paradigm from images to video: for example, Video-LLaVA unifies image and video features before projection so that a single language backbone can reason over both modalities [15]. Recent surveys categorize such systems into (i) vision–language understanding models (image/video + text input, text-only output), (ii) multimodal-input text generators, and (iii) fully multimodal input–output models [16]. Despite these advances, most VLMs used in practice operate in the "understanding" regime: they consume visual inputs and generate text (answers, descriptions, tool calls), but do not directly output motor commands or continuous trajectories. As a result, they are highly informative for perception and high-level reasoning, yet remain one step removed from closed-loop embodied control.

Multi-Modal Language Models Multimodal large language models (MLLMs) generalize this idea by allowing multiple modalities both on the input and, in some cases, on the output side [17]. One common design uses additional modalities only as *conditioning* inputs: models such as GPT-4V or Qwen-VL take images or video along with text and generate text responses, but do not natively produce non-text outputs [**?**]. A second class couples a language backbone with separate generative decoders, enabling *cross-modal* generation: for example, Koh et al. [18] map hidden states from an LLM into the embedding space of a text-to-image model to generate images from arbitrary interleaved image–text prompts, while Transfusion [19] trains a single transformer jointly with next-token prediction and diffusion to support both text and image generation within one sequence model. Similarly, the SEED/SEED-X family [20, 21] learns visual tokenizers and multimodal backbones that can comprehend images of varying resolutions and also generate images, moving closer to unified image understanding and synthesis.

Beyond images, recent models extend the same template to video and speech. Video-LLaVA and related video-centric MLLMs process sequences of frames as visual tokens and support video question answering and temporal reasoning using a unified visual representation [15]. Speech-centric models connect neural audio front-ends or discrete speech tokenizers to LLMs for end-to-end speech summarization or speech-to-speech dialogue [22, 23], and omnidirectional systems such as GPT-4o or Qwen-Omni treat text, image, and audio as first-class tokens in a single conversational model [12, 23]. Together, these lines of work demonstrate that a single backbone can, in principle, ingest and emit multiple modalities. However, their outputs are typically optimized for communicative media (text, images, audio) rather than for physically grounded control; bridging from such multimodal token spaces to low-level robot action remains an open challenge.

Personas in Language Models Prior studies [24–26] have explored the ability of language models to adapt specific personas or personalities. Li et al. [24] introduced Human Level Attributes (HLAs) based on tropes to create dialogue agents that can mimic the personalities of fictional characters. Safdari et al. [25] discussed the impact of synthetic personality in large language models (LLMs) on conversational agents, introducing a method for assessing and validating personality tests on LLMs and for influencing the personality in LLM-generated text. Lee et al. [26] demonstrated distinct LLM-based behaviors that enable more authentic and context-aware human-robot interactions by integrating non-verbal cues. However, the approach has limitations in the latency of requesting API calls, and the explored persona was dialog-centric rather than the character or style of the robot.

Tree-based Reasoning in Foundation Models Tree search is a foundational technique in robotic planning for exploring sequences of actions and their potential outcomes [27–30]. This principle of structured exploration has been recently adapted for language models in Tree-of-Thoughts (ToT) prompting, which casts complex reasoning as a search over branching thought states [31]. In practice, ToT commonly leverages Monte Carlo Tree Search (MCTS) [32–35], with nodes representing partial reasoning states and rollouts used to estimate success. Recent work emphasizes the importance of value guidance; for instance, [36] shows that a PPO-trained value model can guide MCTS to avoid prematurely pruning useful candidates. For embodied agents, ToT enables the evaluation of alternative futures to select safer actions, but its effectiveness hinges on well-calibrated value estimates that success-only data cannot provide.

This body of work illustrates that foundation models excel in abstract and multimodal reasoning, but they lack mechanisms for grounding decisions in robot embodiment and for predicting whether planned actions are physically executable. These limitations motivate research on integrating LLMs into robotics systems directly.

2.2 Leveraging Large Language Models in Robotics

Planning and Reasoning LLMs have been increasingly used as high-level planners in robotics, but prior work differs substantially in how language is converted into executable plans. Huang et al. [37] showed that an LLM can decompose a natural-language instruction into a sequence of subgoals using chain-of-thought prompting, allowing the robot to act step-by-step based on linguistic reasoning. Ahn et al. [38] instead framed planning as grounding text to a constrained action interface, where each step corresponds to an API-style call to a controller, improving reliability by limiting the output space to verifiable skills. Liang et al. [39] proposed program-generation for robotics, where the LLM produces structured action programs using a predefined skill library, enabling better safety and interpretability during execution. These approaches collectively enable robots to translate natural-language goals into ordered action sequences, yet they share the same structural limitation: the planner can only compose behaviors that exist within the predefined action library. If a task requires a behavior that is not represented among the primitives (e.g., a new grasping strategy or an unseen geometric constraint), the LLM cannot synthesize it—even if the reasoning is linguistically consistent and logically coherent.

Challenges and Limitations Real-world instructions are often ambiguous, contradictory, or physically infeasible ("place it over there," "grab the full cup but not the full one"), and grounding such language requires understanding not only intent but also embodiment and environmental constraints. Uncertainty quantification in language models [40–44] has been investigated primarily in QA and machine translation, and frequently depends on token-wise probability distributions unavailable for closed-source

LLMs such as GPT-4 [45]. Consequently, current LLM robotic planners reason well about what the user likely means but reason weakly about whether the robot can succeed.

These observations motivate moving from language-only planning toward models that integrate perception and control directly, which has led to the emergence of vision–language–action architectures.

2.3 Vision–Language–Action Model

Datasets The growth of embodied learning has been driven by large-scale demonstration datasets. The Open-X Embodiment dataset [46] aggregates experience from many robots and institutions, enabling cross-embodiment generalization and large-scale policy pretraining. Yet successful interactions vastly outnumber failed ones, leaving models with limited exposure to failure outcomes that are essential for understanding feasibility and safety.

Vision–Language–Action Models VLA models unify perception, language, and control through end-to-end learning [47–51]. RT-1 [47] demonstrated the effectiveness of Transformers in visuomotor control on a single platform, and RT-2 [48] transferred web-scale semantic knowledge into actions, enabling conceptual generalization. Follow-up models such as RT-X [46] and OpenVLA [10] use multi-embodiment data to create general-purpose control policies. Recent approaches pursue continuous and high-frequency control beyond autoregressive discretization. π_0 [1] and $\pi_{0.5}$ [52] eliminate tokenized action spaces by representing actions as trajectories in a continuous manifold and learning them via flow-matching [53], improving smoothness and reducing latency during closed-loop control. These models also segment long-horizon demonstrations into short action chunks [54] to stabilize training and enable higher effective control frequency. GR00T [9] further demonstrates that real-robot data, human videos, and synthetic data can be jointly exploited for dexterous manipulation across diverse embodiments, suggesting that continuous-action VLAs benefit from heterogeneous sources of motion supervision.

In parallel, a complementary line of work integrates more explicit high-level reasoning into VLA architectures. Zawalski et al. [55] incorporate textual "reasoning traces" aligned with actions to improve generalization to novel tasks. Zhao et al. [56] predict intermediate future images—effectively imagining the scene evolution before performing the action—to improve robustness under distribution shifts. Li et al. [57] introduce affordance prediction to bias action selection toward more promising objects or states. Collectively, these methods aim to enhance interpretability and generalization by making the policy's internal decision process more structured and explicit, rather than entirely implicit within latent representations.

Hierarchical VLA Hierarchical designs seek to separate high-level reasoning from low-level control within a unified architecture. HiRobot [58] decomposes tasks into atomic language instructions executed by a low-level policy; HAMSTER [59] predicts 2D end-effector motion to guide a 3D controller; MolmoAct [60] uses visual trajectory tokens to support user-steerable manipulation. Although such models improve clarity and controllability, they generally assume that planned subgoals are feasible rather than verifying them explicitly before execution.

The current VLA landscape highlights progress toward general-purpose embodied models, yet feasibility-aware planning and reasoning under uncertainty remain largely missing—especially in settings where demonstrations contain mixed successes and failures.

2.4 Non-verbal Interaction and Persona in Robots

Persona-Driven Robots Persona and social identity influence user engagement, perceived competence, and cooperation [61–64]. Prior studies have demonstrated robots expressing extroverted vs. introverted personas [62], socially engaged vs. competitive personas [63], and receptionist-style interaction with verbal and non-verbal cues [64]. These systems, however, rely on manually scripted behaviors for each character and do not scale across diverse interaction contexts.

Non-verbal Communication Non-verbal cues play a significant role in conveying emotions and intentions in communications. Non-verbal communication can be composed of gaze, gestures, or intonation. Pan et al. [65] introduced a system that combines advanced gaze interaction technology and character animation principles with human-like gaze behaviors. Ko et al. [66] focused on the motion of the robot to learn and generate the social behavior given the human pose. Brock et al. [67] presented a real-time hand gesture recognition system to facilitate close-distance non-verbal communication with the tabletop robot Haru.

Together, these findings suggest that non-verbal communication requires not only motion generation but motion that is conditioned on context, intent, and persona. This motivates research on joint modeling of motion and language.

2.5 Motion–Language Model

Traditional Motion Modeling & Control Text-to-motion synthesis has been advanced by diffusion-based approaches [68, 69] and transformer models with vector quantization [70, 71]. MoMASK [72] captures subtle details via residual tokenization. Motion editing has explored style transfer [73, 74] and part-specific edits [69, 75]. MEOs [76] identify editable regions with captions; MotionFix [77] conditions diffusion models on both source motion and edit text. These systems typically support unidirectional mapping rather than full bidirectional interaction across modalities.

Motion–Language Models Recent developments in motion-language models have aimed to achieve versatility across various motion-related tasks. MotionGPT [78] demonstrates versatility in motion comprehension and generation based on a unified framework. MotionChain [79] introduces a multi-turn conversational system for interpreting and generating motions within dialogue contexts, including image inputs. Recent work [80–83] has explored unified approaches to multi-modal motion generation, including speech, video, and image. However, these methods focus on the single-person motions, thus, modeling *interactive motions* in unified models remains under-explored. Wu et al. [84], address the interactive motions, but they still lack multi-turn interactions and complex reasoning abilities.

Existing Dataset Interactive datasets [85–87] have supported human–human motion understanding, and recent parallel efforts such as Inter-X [88] and InterHuman [89] pair interaction motions with text annotations for reactive synthesis [90–92]. However, most existing datasets model a single turn of interaction rather than continuing dialogue with adaptation over time.

Current motion–language research emphasizes versatile motion generation, but less attention is placed on feasibility-aware reasoning or long-horizon social interaction. This creates a disconnect between expressive behavior and reliable physical execution.

2.6 Summary and Positioning of This Dissertation

Across foundation models, LLM-based planning, VLA models, social interaction, and motion–language modeling, several limitations consistently emerge: (i) most systems lack explicit mechanisms for feasibility-aware reasoning and therefore cannot anticipate failure before execution; (ii) large-scale training rarely leverages mixed-quality demonstration data in which failures are informative; (iii) social expressiveness and task-level competence are developed in separate model families, rather than within a unified architecture.

This dissertation addresses these gaps through two complementary directions. First, we introduce dual-system VLA architectures that jointly train high-level reasoning and low-level control on a shared backbone using both success and failure trajectories to learn feasibility-aware value functions, supporting reliable long-horizon planning. Second, we develop interaction models that integrate language, motion, and persona cues, enabling robots to express intentions and social identity through both verbal and non-verbal behavior in multi-turn settings.

Together, these contributions aim to ground foundation-model principles in robots that act with people—capable of reasoning under uncertainty, learning from imperfect experience, and communicating their intentions through legible actions.

2.7 List of Abbreviations

FM Foundation Model. A large-scale model pretrained on broad data that can be adapted to multiple downstream tasks.

LLM Large Language Model. A language-only foundation model used for reasoning, planning, or dialogue.

VLM Vision–Language Model. A model that jointly processes visual and textual inputs to produce aligned representations.

VLA Vision–Language–Action Model. A model that connects perception, language, and action to generate robot behaviors.

HRI Human–Robot Interaction. The study and design of interactive behaviors between humans and robots.

HCI Human–Computer Interaction. The broader field of interaction between humans and computational systems.

IL Imitation Learning. A paradigm in which policies are learned from example demonstrations.

RL Reinforcement Learning. A paradigm in which policies are optimized to maximize cumulative reward.

HRL Hierarchical Reinforcement Learning. A framework that decomposes control into high-level decisions and low-level skills.

MDP Markov Decision Process. A formal model for sequential decision-making under uncertainty.

SMDP Semi-Markov Decision Process. A generalization of MDPs that allows temporally extended actions or options.

VINE Vision–Language–Action model Integrating Negative Experience. A hierarchical VLA framework that leverages both successful and failed trajectories for feasibility-aware planning.

CLARA Classifying and disAmbiguating user Commands for Reliable interactive Agents. A framework that classifies user commands as clear, ambiguous, or infeasible and triggers clarification when needed.

MASK A persona-based non-verbal interaction system that endows robots with configurable social roles and expressive behaviors.

MoLaM Motion–Language Model. A model that learns a shared representation for language and interactive motion to support the generation, understanding, and editing of motion.

SaGC Situational Awareness for Goal Classification. A dataset for evaluating how well models classify user commands as clear, ambiguous, or infeasible given robot capabilities and scene context.

AUROC Area Under the Receiver Operating Characteristic curve. A metric for evaluating how well a model separates different classes or uncertainty levels.

F1 F1 Score. The harmonic mean of precision and recall, used to evaluate classification performance.

Chapter 3

Verbal Interaction

This chapter investigates how robots can reliably act on verbal communication despite the inherent multi-modality of language. A single spoken instruction may map to many possible behaviors depending on the environment, task context, and the robot's own capabilities, making language grounding fundamentally ambiguous. We explore two complementary strategies for resolving this ambiguity in verbal interaction: selecting the most feasible action trajectory among multiple candidate behaviors, and determining whether the instruction itself is sufficiently interpretable to act on in the first place. Together, these perspectives aim to transform natural language from a high-level interface into a dependable control signal for real-world robotic systems.

3.1 Motivation and Background

Human verbal instructions are one of the most intuitive ways for people to communicate goals to robots. Natural language allows users to request tasks at a high abstraction level, e.g., "set the table," "bring me coffee," "help him", without manually specifying motion trajectories or low-level behaviors. This property makes language a promising interface for general-purpose robotic agents. As illustrated in Figure 3.1, the robot receives a high-level instruction from a human and grounds it into actionable behavior conditioned on its perception of the environment.

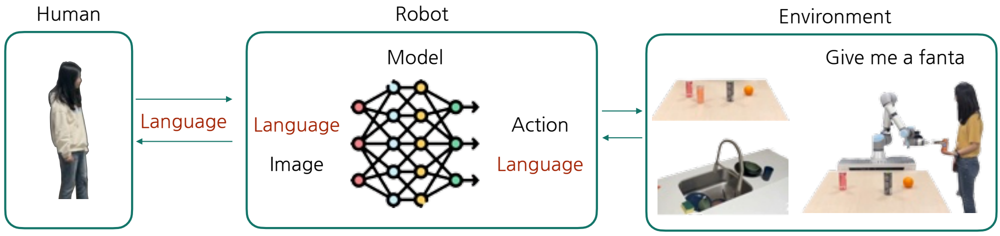


Figure 3.1: Language-based human–robot interaction. A human provides high-level verbal instructions, which are grounded by the robot through a model that interprets language and visual observations to produce task-directed actions in the environment.

Natural language offers an intuitive and high-level interface for commanding robots, allowing users to request tasks such as "set the table," "bring me coffee," or "help him" without specifying low-level motions or action primitives. This capability dramatically reduces the communication burden for humans and makes language an attractive modality for general-purpose robotic systems. However, grounding abstract utterances into executable behaviors remains fundamentally challenging. The language–action mapping is inherently *multi-modal*: a single instruction may correspond to many different behaviors depending on the scene, the task context, and the robot's physical capabilities. As a result, a reliable language-conditioned robot must determine not only how to act, but also whether the instruction can be acted on at all.

Progress in language-based robotics has largely focused on translating abstract user instructions into a single executed behavioral stream. LLMs and VLMs have been used directly as planners that convert verbal commands into symbolic programs or stepwise action sequences [37,38,93], while Vision–Language–Action (VLA) models [1,9,10,46] map multimodal observations to continuous control policies. Many recent VLAs are trained with distribution-matching objectives such as flow matching or diffusion, en-

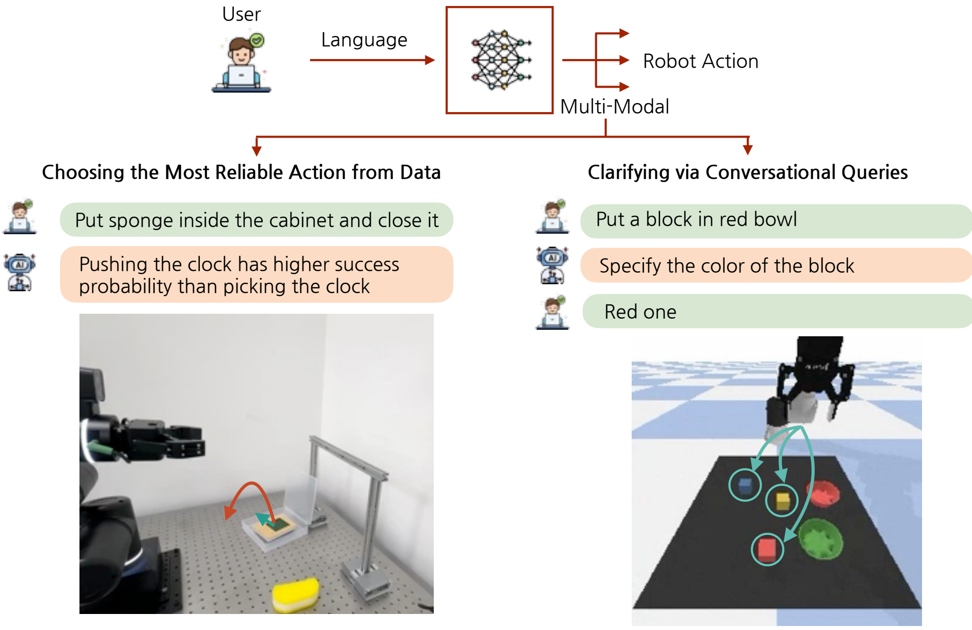


Figure 3.2: Overview of the dual pathways in our framework. When the instruction and scene jointly provide enough information, the robot selects the most feasible action based on multimodal evidence (left). When the instruction is ambiguous or underspecified, the robot initiates conversational queries to resolve uncertainty by asking the user (right).

couraging the model to imitate the overall demonstration distribution. However, these objectives treat all modes of the action distribution equally and do not explicitly identify which behavioral branch is *more likely to succeed*. Consequently, when the scene and instruction admit multiple plausible next actions, the agent does not evaluate or choose among them: it collapses multi-modality by regressing toward the data distribution rather than selecting the safest or most feasible trajectory. This limitation is amplified by the fact that VLAs are typically trained only on successful demonstrations, causing failure signals to be systematically underutilized. At the same time, instruction-level ambiguity or infeasibility is often assumed away, and the robot is expected to execute a command even without fully understanding the user's intent.

As a result, current systems fail in two distinct modes. First, when multiple action trajectories are plausible, the agent may select brittle or failure-prone branches, leading to execution errors despite having the necessary skills. Second, the robot may misinterpret verbal commands that are ambiguous or infeasible given the environment or its own capabilities, resulting in failure not from poor planning but from misunderstanding the user's intent. Addressing both dimensions, *identifying which action branch is most likely to succeed* and *determining whether the instruction itself should be acted on*, is therefore crucial for reliable and trustworthy human–robot interaction. Figure 3.2 summarizes the two pathways in our system: feasible actions are inferred directly from mixed quality data, whereas ambiguous commands trigger an interactive question to the user to determine the intended task.

To handle the multi-modality of language-conditioned action generation, our approach is to predict not only which behaviors are possible but also which ones are most likely to succeed, and to select high-feasibility paths based on both positive and negative experience. Rather than imitating only successful demonstrations, our framework explicitly leverages failure data to learn which behavioral branches tend to lead to dead ends or risky outcomes, enabling informed pruning during planning. We instantiate this idea with **VINE**, a hierarchical VLA framework that learns *failure-aware reasoning*. VINE builds on a pretrained VLA policy (the low-level controller, System 1) and augments it with a high-level reasoning system (System 2) trained offline on both successes and failures. System 2 performs feasibility-guided tree search by predicting the success probability of candidate sub-goals and selecting high-feasibility action branches. This hierarchical formulation aligns with classical Hierarchical Reinforcement Learning (HRL) [94]: high-level planning and low-level control are disentangled, and failure supervision is injected only into the high-level planner—without requiring online rollouts or modifying the base skills of the robot. Through this design, the agent does not simply imitate behaviors that worked in the past; it actively suppresses brittle action sequences before execution.

While failure-aware planning improves robustness during execution, a complementary issue arises during human–robot interaction. Even with a strong feasibility-aware planner, the robot may still receive commands that are underspecified (ambiguous) or impossible given its environment or capabilities (infeasible). Importantly, situational awareness determines the interpretation: to the kitchen robot, “help him, he looks tired” implies make coffee; to a massage robot, it is ambiguous; to a cleaning robot, it is infeasible. Thus, correctly categorizing user instructions into clear, ambiguous, or infeasible becomes essential for reliable interaction.

To address this, we additionally propose CLARA, CLassifying and disAmbiguating user commands for reliable interactive Robotic Agents. CLARA first estimates predictive uncertainty to determine whether a command is clear or not. Uncertain commands are further categorized as ambiguous (underspecified) or infeasible (beyond capability or context). For ambiguous instructions, the system initiates natural-language clarification via question generation. CLARA operates largely through prompting, requiring only lightweight updates (few-shot examples and capability descriptions) to adapt to different agents or environments. We also introduce the SaGC benchmark, which evaluates situational awareness in command interpretation by pairing high-level instructions with environments, robot capabilities, and uncertainty labels.

Summary of Contributions

This chapter presents two complementary research directions toward reliable and interpretable interactive robotic agents: (i) **failure-aware high-level planning** for robust action execution, and (ii) **situationally aware command understanding** for safe and adaptive interaction. Our key contributions are summarized below.

VINE — Failure-Aware Hierarchical Vision–Language–Action Planning

- We introduce a tree-based planner that scores candidate steps using feasibility predictions learned from offline *mixed-quality* supervision consisting of both successful and failed demonstrations.
- We propose a hierarchical VLA framework grounded in hierarchical reinforcement learning (HRL), cleanly separating feasibility-aware high-level reasoning (*System 2*) from low-level action execution (*System 1*).
- We present extensive empirical results showing substantial improvements in robustness and overall success rate over strong VLA baselines on diverse robot manipulation tasks.

CLARA — Situationally Aware Command Understanding for Interaction

- We develop a predictive-uncertainty estimation method for LLMs to detect whether user commands are clear or unclear.
- We propose a two-stage classification framework that distinguishes unclear commands as either ambiguous or infeasible, and performs natural-language disambiguation on ambiguous ones.
- We conduct comprehensive evaluations on the SaGC benchmark, demonstrating situational awareness and reliable interactive clarification across diverse environments and agent capabilities.

3.2 Failure-Aware Hierarchical Vision–Language–Action Planning

Having established the motivation for failure-aware hierarchical planning, we now provide an overview of **VINE** and its architectural design. The core idea is to treat feasibility as a first-class signal in the planning loop rather than an implicit byproduct of imitation learning. Concretely, VINE augments a pretrained VLA controller with a high-level reasoning layer that evaluates and selects sub-goals based on predicted feasibility before execution. This section describes the overall structure of the system, the interaction between System 2 and System 1 during inference, and how mixed-quality supervision is incorporated into the learning process to enable robust decision making.

We hypothesize that by learning not only what works but also what fails, an agent can better anticipate risky outcomes. Failure data [95–97] is an often overlooked yet essential element in VLA training. Most robot data for post-training VLAs come from offline trajectories collected via human teleoperation [1, 9, 10]. Although these collections naturally include failed attempts, e.g., unstable grasps and collisions, such traces are typically discarded as noise. However, they encode information about infeasible transitions and unsuccessful behaviors, making them a vital resource for improving robustness.

The challenge lies in effectively integrating this failure signal from existing offline datasets. In Imitation Learning (IL), while explicitly penalizing failure-prone transitions is possible, carefully tuning these penalties to avoid distorting the learned policy is a non-trivial challenge [97–99]. On the other hand, Reinforcement Learning (RL) [100, 101] offers a natural framework to handle failure data through a reward signal. To utilize this property, we formulate the overall system as *Hierarchical Reinforcement Learning*

(HRL) [94]. We introduce **VINE**, **V**ision–**L**anguage–**A**ction model **I**ntegrating **N**egative **E**xperience, a hierarchical VLA framework that integrates negative experience built upon pre-trained VLA (i.e., π_0 [1]), and is formulated through HRL [94]. Our approach separates high-level planning from low-level control, and injects failure supervision solely offline to train the planner's value modules without any online rollouts.

This HRL-inspired separation is embodied in our method's hierarchical system [9,58] architecture, inspired by Kahneman's cognitive process [102]. The high-level System 2 is responsible for reasoning and planning. We cast planning as a feasibility-guided tree search, scoring each candidate transition by its predicted success probability. In particular, we construct a tree of a 2D scene graph that abstracts world states (nodes), where subgoals represent the transitions (edges) connecting them. Crucially, System 2 is fine-tuned with both success and failure data to learn a feasibility score for each node, generate options, or generate an estimated next state representation. This allows the planner to conduct failure-aware reasoning, which preemptively identifies and avoids paths likely to fail. The optimal sequence of sub-goals and predicted next scene graph is then passed to the low-level policy, System 1, for execution. This design cleanly slots the learned feasibility signal into the high-level reasoning loop, enhancing robustness without altering the agent's fundamental skills.

3.3 Problem Formulation for Failure-Aware Hierarchical VLA

In this section, we formalize instruction-conditioned manipulation with a reach–avoid objective from mixed-quality demonstrations (successes and failures) as shown in Figure 3.3. We use a dual-system hierarchy: **System 2** performs a shallow look-ahead tree search over candidate edges with a failure-aware feasibility/value estimator, and **Sys-**

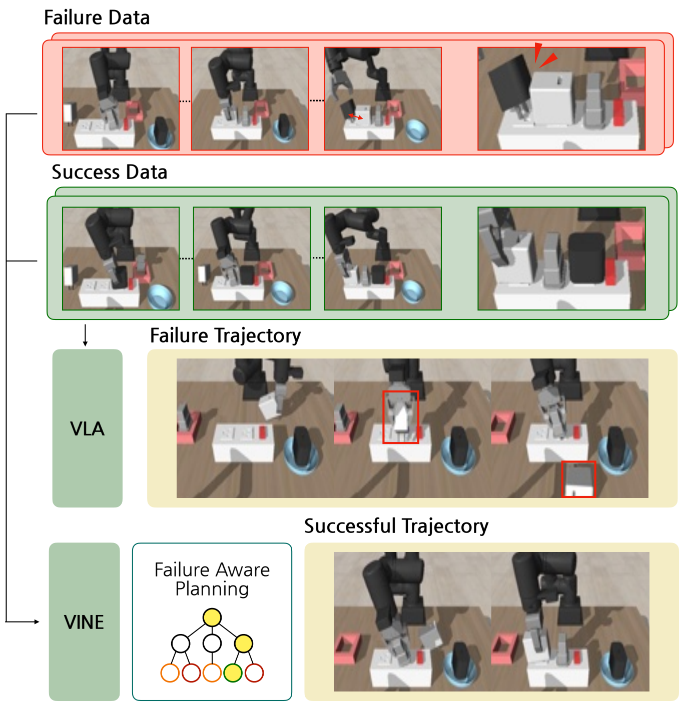


Figure 3.3: Standard VLAs trained only on success data may produce infeasible trajectories. VINE leverages both success and failure trajectories with a failure-aware reasoning planner, yielding more robust executions.

tem 1 executes the selected path with closed-loop control until termination. Each edge is grounded as an option, inducing an Semi-Markov decision process (SMDP) over abstract states; the following paragraphs specify the states, options, termination, and success/failure conditions.

3.3.1 Hierarchical VLA

We formalize a *hierarchical VLA* framework where **System 2** performs high-level reasoning and planning, while **System 1** executes each selected plan through low-level control actions. Most VLA policies are trained via imitation learning (IL), which is data-efficient and stable with strong demonstrations, but provides only limited leverage for explicitly using failure data [97–99]. Viewed through the lens of reinforcement learning (RL) [100, 101], incorporating both successes and failures becomes natural via reward-based learning, motivating a hierarchical reinforcement learning (HRL) formulation [94]. Accordingly, we adopt HRL as the formal scaffold: System 2 plays the role of the meta-controller and employs *model-based tree search* with brief look-ahead over candidate subgoal transitions to estimate failure-aware value and choose a plan, while System 1 focuses on robust execution of the chosen subgoal sequence.

Data Assumption and Objective

Our dataset $\mathscr{D}$ comprises teleoperated trajectories labeled as success (target state reached) or failure (instruction violation or irrecoverable state). We use a sparse terminal reward (r_t=1 on success, 0 otherwise) and learn from tuples (s_t, a_t, ℓ, r_t). Given (s_0, ℓ), the agent plans once by building a budgeted shallow look-ahead tree $\mathscr{T}$ over abstract nodes and subgoal edges, scores transitions with a failure-aware feasibility estimator trained on mixed-quality data, selects a single root-to-leaf plan τ^{sel}, and executes each chosen edge with System 1 until its termination.

Prerequisites for Tree Search and Execution

To enable hierarchical tree search, System 2 requires three components: (i) a short-horizon world model to evaluate hypothetical successors offline; (ii) a candidate-proposal mechanism that expands only instruction-consistent edges, controlling the branching factor; and (iii) a failure-aware value estimator to score edges and back up returns. System 1 employs an option-conditioned, closed-loop policy to execute each selected edge and a termination detector to signal arrival at the successor state.

Bridge to HRL

Interpreted in HRL [94] terms, the chosen route can be viewed as a sequence of *options*. System 2 plays the role of a meta–controller that chooses among high-level *options* aligned with edges in our abstract graph. Each edge corresponds to an option with three parts: an initiation set (states consistent with the current node), an intra-option policy carried out by System 1 (a low-level controller conditioned on the selected options), and a termination rule that triggers when the successor node is reached or a failure condition is met. Because options unfold over a variable number of primitive steps, the induced high-level process can be viewed as semi-Markov decision process (SMDP). Decisions are taken only at option termination, and the time between decisions equals the option's duration. We formalize this connection next.

3.3.2 SMDP Formulation

Let the low-level interaction of the robot be an Markov decision process $\mathscr{M} = (\mathscr{S}, \mathscr{A}, T, \rho)$ under an instruction ℓ, with states $s \in \mathscr{S}$ including images and proprioception observations, actions $a \in \mathscr{A}$, transition kernel $T(\cdot \mid s, a)$, and initial distribution ρ. A *node* is a compact abstraction of the robot/world state: $n = \phi(s) \in \mathscr{N}$, where $\phi : \mathscr{S} \to \mathscr{N}$ extracts scene/plan tokens, spatial relations, etc. An *edge* $e \in \mathscr{E}$ is a temporally extended subgoal executed by a closed-loop controller (System 1) until a termination condition is met. At *decision epochs* (the sequence of nodes (n_k)), the induced process over $\mathscr{N}$ is a semi-Markov decision process (SMDP). In this view, each transition between nodes (an edge) can be naturally modeled as an *option* in the SMDP framework, linking symbolic reasoning with executable policies.

We distinguish between the *symbolic* form of an edge and its *option* grounding. An edge $e \in \mathscr{E}$ is a subgoal token (e.g., "pick up spoon"), which specifies the intended transition at the abstract reasoning level. For execution, each edge e is associated with an option

$$o_e = \left(I_e^N, \pi_e, \beta_e, \Pi_e\right),$$

where *(i) Initiation set)* $I_e^N \subseteq \mathscr{N}$ specifies the nodes at which e is admissible. Execution at node n is feasible only if $n \in I_e^N$, *(ii) Intra-option policy)* $\pi_e(a \mid s, \ell)$, *(iii) Termination rule)* $\beta_e : \mathscr{S} \to [0,1]$, and *(iv) Projection)* $\Pi_e : \mathscr{S} \to \mathscr{N}$. Thus, e denotes the symbolic subgoal, while o_e denotes its formal, executable grounding as an option.

3.3.3 Option-Level Representation: Nodes, Edges, and Data

The option view above specifies *what* is executed at the high level; next we specify *how* these objects are realized from data. Concretely, mixed–quality demonstrations provide supervision to learn (i) the abstraction that maps raw states to nodes, enabling edges to be instantiated as options, and (ii) the signals needed to train the world model, proposal, policy, and termination components.

Nodes as Abstract State Representations

A node $n_k = \phi(s_{t_k})$ denotes an abstract, grounded representation of the world at keyframe time t_k. We instantiate each node as a 2D scene graph [103–105] to encode not only objects but also their spatial and semantic relations. This representation lets the planner reason about consequences: by inferring how a proposed subgoal will alter relations in the current scene graph, it can anticipate the next abstract state n_{k+1}. Concretely, ϕ is realized via a two-stage pipeline: an object detector (e.g., Grounding DINO [106]) proposes candidate boxes, and a vision–language model (e.g., Gemini-2.5-Flash [2]) filters them and predicts the relational structure, yielding a scene graph serialized in text.

Edges as Transitions Between Nodes

We model high-level plans as sequences of abstract states (nodes) connected by transitions (edges). Each transition is treated as a subgoal—an *edge* e_k—executed over $[t_k, t_{k+1})$. Keyframes $\{t_k\}$ are detected from consistent gripper-state events (e.g., `closed`→`open` ending `grasp/insert`; `open`→`open` after `place/push`). With this view, planning reduces to explicit edge selection: choosing an edge commits to a specific state transition, which is verifiable at the subsequent keyframe.

Constructing Datasets for Hierarchical Learning

From this decomposition, we build $\mathscr{D}_{\text{sys2}}$ for the system 2 and $\mathscr{D}_{\text{sys1}}$ for the system 1. $\mathscr{D}_{\text{sys2}}$ contains tuples $(n_k, e_k, n_{k+1}, z_{0:k-1}, \ell, s_0, r_k)$ with history context $z_{0:k-1} = [n_0, e_0, \ldots, n_{k-1}]$ and sparse rewards ($r_k{=}0$ for $k{<}K$, $r_K{\in}\{0,1\}$). $\mathscr{D}_{\text{sys1}}$ uses continuous control data conditioned on (e_k, n_{k+1}, ℓ), i.e., $(e_k, n_{k+1}, s_t, a_t, \ell)$, so that the low-level policy is guided by the *intended transition* (edge) and the *predicted next state* (node). This tight coupling, edge-shaped transitions as node transitions, and nodes as predictive, goal-like abstractions, enables tractable planning over feasible paths directly grounded in demonstrations.

3.3.4 Option-Level Feasible Decision Making Formulation

Conceptually, our option-level formulation is semantically similar to prior reach–avoid frameworks [107, 108]: by identifying reach and avoid sets from mixed success/failure data and delineating the feasible decision space. The process terminates upon first entry into either a target set $\mathscr{G}$ (the success or goal set to be reached) or a failure set $\mathscr{F}$ (the unsafe set to be avoided), with $\mathscr{G} \cap \mathscr{F} = \varnothing$.

Formally, define the hitting times

$$\begin{aligned}
\tau_{\mathscr{G}} &:= \inf\{t \geq 0 : \ s_t \in \mathscr{G}\}, \\
\tau_{\mathscr{F}} &:= \inf\{t \geq 0 : \ s_t \in \mathscr{F}\}, \\
\tau &:= \tau_{\mathscr{G}} \wedge \tau_{\mathscr{F}}.
\end{aligned}$$

An episode terminates at τ with outcome $s_\tau \in \mathscr{G} \cup \mathscr{F}$.

The set $\mathscr{G}$ is a target set defined by design: the task is considered successful as soon as the process reaches any state in $\mathscr{G}$, and the episode is terminated at that point. In particular, it suffices that there exists at least one option or policy that enables reaching $\mathscr{G}$. In contrast, the failure set $\mathscr{F}$ is a critical set closed under options: once the process enters $\mathscr{F}$, no admissible option can lead it outside. Formally,

$$\forall s \in \mathscr{F}, \ \forall o \text{ admissible at } s : \quad \operatorname{supp} P_o(\cdot \mid s) \subseteq \mathscr{F},$$

where $P_o(\cdot \mid s)$ denotes the state distribution upon termination of option o starting from s. Thus $\mathscr{F}$ is absorbing at the option level. This induces a sparse entrance reward $r_t := \mathbf{1}\{s_{t+1} \in \mathscr{G}\}$, so that the undiscounted return $R = \sum_{t=0}^{\tau-1} r_t$ equals 1 if $\tau_{\mathscr{G}} < \tau_{\mathscr{F}}$ and 0 otherwise.

3.3.5 Value as Success Probability

To operationalize failure-aware reasoning within the tree-search, the planner needs a principled node score. We define this score as the probability of reaching a successful goal state from any given node n. To estimate this probability, we cast the problem as learning a value function, $V(n)$, which allows us to leverage both success and failure trajectories from our dataset. This equivalence between the value function and the reach-avoid success probability is formally stated in Proposition 1 below.

Proposition 1 (First-exit value equals reach–avoid success probability)**.** *Assume the tree search process (node and edges) is Markov, and the sets $\mathscr{G}, \mathscr{F}$ are absorbing (first visit terminates). Then the value function equals the probability of reaching $\mathscr{G}$ before $\mathscr{F}$, where τ is a hitting times:*

$$V(n) = \Pr\left(\tau_{\mathscr{G}} < \tau_{\mathscr{F}} \,\middle|\, n\right). \tag{3.1}$$

Consequently, high–value nodes approximate the probabilistic reachable set to $\mathscr{G}$.

Proof sketch. Under the first-exit formulation with absorbing $\mathscr{G}_N, \mathscr{F}_N$, the return equals the success indicator; hence $V^{\mu} = \mathbb{E}[R] = \Pr(\tau_{\mathscr{G}} < \tau_{\mathscr{F}} \mid \cdot)$. See Appendix for detailed derivation.

3.4 Failure-Aware Hierarchical VLA Integrating Negative Experience

In this section, we introduce **VINE**, **V**ision–**L**anguage–**A**ction model **I**ntegrating **N**egative **E**xperience, a hierarchical VLA framework that integrates negative experience into both reasoning and action. Our method is formulated within a hierarchical reinforcement learning [94] framework consisting of two coupled systems. **System 2 (Reasoning and Planning)** conducts tree search: proposes candidate reasoning paths, estimates their success probabilities, and selects the most feasible plan. **System 1 (Action Modeling)** then executes the next subgoal from the chosen plan with low-level actions.

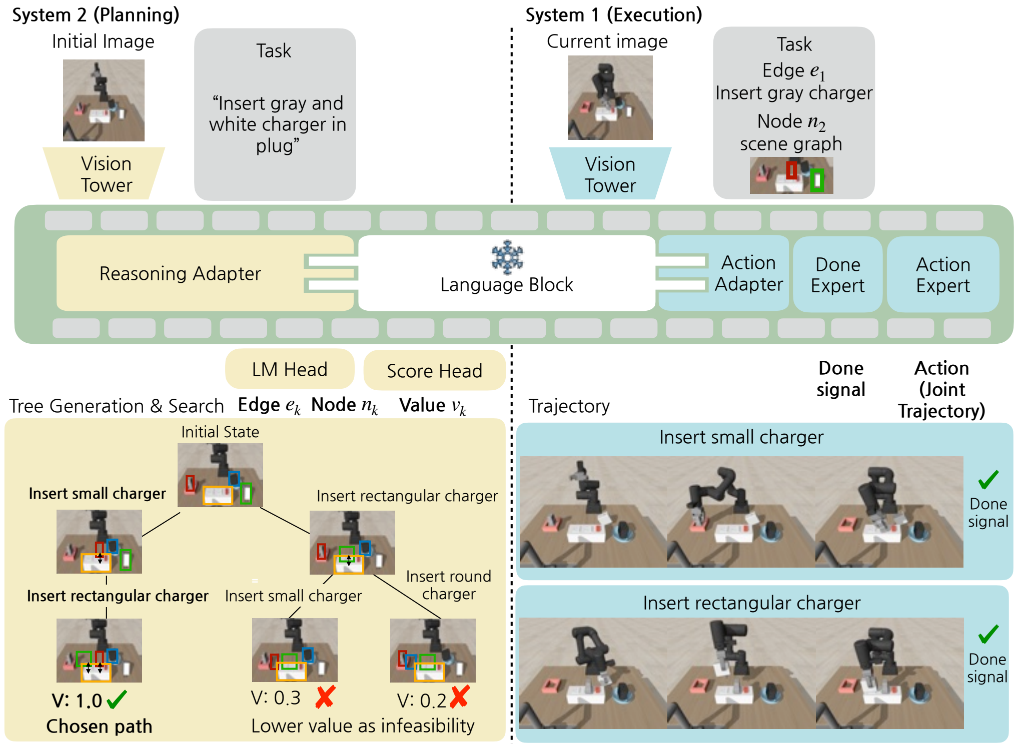


Figure 3.4: Architecture and overall pipeline. System 2 plans via tree-of-thoughts, predicting success values to select the highest-value path (green). System 1 executes the chosen subgoal as action chunks with a done signal. Both share a vision–language backbone with adapters.

3.4.1 Overall Pipeline

At inference time, the proposed framework follows a plan–execute procedure, illustrated in Figure 3.4. Before acting, System 2 conducts a *model-based tree search* over candidate options (edges) and their predicted successor states (nodes). Each candidate step is scored by how likely it is to succeed given the current scene and instruction, and the agent selects the highest-scoring overall path through the tree. System 1 then executes the current edge on that path with low-level actions until a termination condition indicates that it is complete. If execution matches the predicted successor state, the agent advances to the next edge on the chosen path. To meet real-time constraints, the tree is constructed before the execution starts in the initial state.

3.4.2 Architecture

Because System 2 performs *model-based* planning over the abstract graph, it needs (a) a proposal mechanism for candidate edges, (b) a transition model to predict successor nodes, and (c) value heads to score lookahead, including failure awareness. Concretely, System 2 couples (i) an option generator $\hat{P}_\theta(e \mid n, z, \ell, s_0)$, (ii) a learned world model $\hat{P}_\theta(n' \mid e, n, z, \ell, s_0)$ and a value predictor $V(n) := V_\theta(n|z, \ell, s_0)$, and (iii) a failure-aware value estimator $V(n) := V_\theta(n|z, \ell, s_0)$, yielding the high-level option policy. At the low level (System 1), the intra-option execution is parametrized by the intra-option policy $\pi_e(a \mid s, \ell) := \pi_\theta(a \mid s, \ell, e, n')$ and termination rule $\beta_e(s) := \beta_\theta(s, \ell, e, n')$.

We build the model upon π_0 [1] as a multimodal backbone that encodes (s, ℓ, a) into a unified token sequence. Because π_0 targets action generation rather than text, we merge its shared layers with a PaliGemma [7] language–vision trunk, yielding strong text generation for tree-search steps while preserving π_0's action priors [109]. Both systems add LoRA [110] adapters to this shared backbone. **System 1** feeds adapted features to (i) an *action expert* for low-level motor chunks and (ii) a *done expert* that detects subgoal completion. System 1 has a shared attention layer between the language block and experts, where the done expert and the action expert do not attend to each other. **System 2** uses (i) a language-modeling head to autoregress node/edge text and (ii) a scalar value head to predict leaf value based on the last layer feature of the language backbone. Each system is trained independently with seperate dataset $\mathscr{D}_{sys1}$ and $\mathscr{D}_{sys2}$. This shares perception–language representations while keeping reasoning and control decoupled.

3.4.3 System 2

System 2 is in charge of high-level tree-based planning, which is formulated as a meta-controller in HRL. Concretely, we instantiate it as a tree–search planner operating on a lookahead tree $\mathscr{T} = (\mathscr{N}, \mathscr{E})$, where each node $n \in \mathscr{N}$ is a 2D scene graph and each edge $e \in \mathscr{E}$ denotes a verifiable subgoal that induces a transition $n_k \xrightarrow{e_k} n_{k+1}$. Thus, the abstract option–selection in HRL is realized by (1) *successor–state (node) proposal*, (2) *subgoal (edge) candidate proposal* labeling the transition $n_k \to n_{k+1}$, (3) *leaf evaluation* with a value function V_θ, and (4) *tree search* to pick a feasible high-level plan. Given the initial scene s_0 and an instruction ℓ, System 2 outputs a reasoning path $\tau^{\text{sel}} = (n_0, e_0, \ldots, n_K)$ to hand over to System 1 for execution.

Node and Edge Generation

Edge proposals and next node transitions are generated autoregressively by the LM head, $\hat{P}_\theta$. The model first samples a candidate node string n_{k+1}, representing the next scene graph state, and then an edge string e_{k+1}, describing the subgoal transition leading there. This process follows the standard language modeling objective, factorizing the probability into token-level likelihoods: node $\hat{P}_\theta(n_{k+1} \mid z_{0:k}, \ell, s_0) = \prod_{j=1}^{N_c} \hat{P}_\theta(w_j \mid w_{<j}, z_{0:k}, \ell, s_0)$ and edge $\hat{P}_\theta(e_{k+1} \mid n_{k+1}, z_{0:k}, \ell, s_0) = \prod_{j=1}^{N_e} \hat{P}_\theta(w_j \mid w_{<j}, n_{k+1}, z_{0:k}, \ell, s_0)$, where N_c and N_e denote the number of tokens for the node and edge string. In practice, we apply beam search or top-k sampling to obtain multiple diverse candidates, and filter them with syntactic templates and scene-graph consistency checks (e.g., whether referenced objects exist in the current graph). This ensures that generated subgoals remain both linguistically valid and grounded in the task context.

We train with the standard LM loss over high-level tree-search step traces

$(n_k, e_k, n_{k+1}, z_{0:k-1}, \ell, s_0) \sim \mathscr{D}_{\text{sys2}}$:

$$\mathscr{L}_{\text{LM}} = -\mathbb{E}_{\mathscr{D}_{\text{sys2}}}\Big[\log \hat{P}_\theta(n_k \mid z_{0:k-1}, \ell, s_0) + \log \hat{P}_\theta(e_k \mid n_k, z_{0:k-1}, \ell, s_0)\Big].$$

Node Evaluation

We aim to estimate, at each search node, a calibrated first-exit success probability that steers expansion and pruning during tree search. Each edge e_k yields the successor node $n_{k+1} = \phi(s_{t_{k+1}})$; we label the transition success if $s_{t_{k+1}} \in \mathscr{G}$, failure if $s_{t_{k+1}} \in \mathscr{F}$, and continue otherwise, updating the context to $(n_{k+1}, z_{0:k})$.

To define the training target, we use first-exit bootstrapping. For each transition $(n_k, e_k, n_{k+1}, r_{k+1})$ we set

$$y_k = \begin{cases} 1, & \text{terminal in } \mathscr{G} \ \left(e_k = \langle \texttt{done} \rangle\right) \\ 0, & \text{terminal in } \mathscr{F} \text{ at step } k+1 \\ \gamma V_{\theta'}(n_{k+1} \mid z_{0:k}, \ell, s_0), & \text{otherwise,} \end{cases}$$

Here $\gamma \in (0,1)$ is the discount factor and θ' is a target network updated by an exponential moving average (EMA).

Based on Proposition 1, the estimated value becomes the success probability of the node,

$$V_\theta(n_k \mid z_{0:k-1}, \ell, s_0) \ = \ \Pr\big(\tau_{\mathscr{G}} < \tau_{\mathscr{F}} \,\big|\, n_k, z_{0:k-1}, \ell, s_0\big)$$

, where τ is the hitting time of the success and failure set.

To handle offline data with many failures, we replace L2 with an asymmetric expectile loss (cf. IQL [111]) that discourages overestimation:

$$\mathscr{L}_{\text{val}} = \mathbb{E}_{\mathscr{D}_{\text{sys2}}}\left[L_2^{\tau_e}(y_k - V_\theta(n_k \mid z_{0:k-1}, \ell, s_0))\right]$$

with $L_2^{\tau_e}(u) = \left|\tau_e - \mathbb{1}(u < 0)\right| u^2$ and $\tau_e = 0.7$, which yields a conservative yet effective value from static datasets.

Tree Search Algorithm

We run a batched MCTS-style planner over abstract nodes (world states) and edges (subgoals). At each iteration, the planner selects promising frontier nodes by their running mean value Q, proposes and scores candidate edges, evaluates the resulting children with a learned state-value V, and backs up values along the path. After T iterations, we return the plan traced from the highest-valued leaf, and execute the next edge of that plan.

Our search strategy is based on Monte Carlo Tree Search (MCTS) but replaces random rollouts with a learned state-value function $V(n)$. The algorithm builds a lookahead tree where nodes are world states and edges are subgoals, and it is accelerated via batched GPU computation. This procedure repeats for a fixed number of iterations T. Finally, we select the solution as the path leading to the leaf with the highest value, $\tau^{\text{sel}} = (n_0, e_0, \ldots, n_K)$.

3.4.4 System 1

System 1 focuses on *action execution*: in our HRL formulation, it implements both the intra-option policy and the termination rule, learned jointly by a single network. It receives as input the selected high-level plan $\tau^{\text{sel}} = (n_0, e_0, \ldots, n_K)$ from System 2, and is responsible for grounding each symbolic edge into continuous control. Given the cur-

Algorithm 1: VINE: Batched Tree Search with Learned V

Input : Root n_0, batch B, proposals k, mix α, steps T
Output: Plan (edge sequence) from the best frontier node

Initialize tree $\mathscr{T}=\{n_0\}$; for all n: $N(n)=0$, $W(n)=0$, $Q(n)=0$;
for $t=1$ **to** T **do**
  $\mathscr{F} \leftarrow$ frontier leaves; $\mathscr{S} \leftarrow$ Top-B of $\mathscr{F}$ by Q;
  foreach $n_p \in \mathscr{S}$ **do**
    Propose k edges $\{e_i\}$ (beam search decoding [112]); compute $S(n_p, e_i)=\alpha Q(n_p)+(1-\alpha)\hat{P}(e_i \mid n_p)$; keep tops;
  For kept (n_p, e): create child n_c (greedy); predict $V(n_c)$ (batched on GPU);
  For path nodes n_a to n_c: $W(n_a){+}{=}V(n_c)$, $N(n_a){+}{=}1$, $Q(n_a){=}W(n_a)/N(n_a)$;
return plan traced from $\arg\max_{n\in\mathscr{F}} Q(n)$

rent low-level state s_t, instruction ℓ, the active edge e_k, and its successor node n_{k+1} from τ^{sel}, System 1 executes the corresponding option with a unified policy–termination pair $\pi_\theta(a \mid s_t, \ell, e_k, n_{k+1})$ and $\beta_\theta(s_t, \ell, e_k, n_{k+1})$. System 1 is trained only on successful demonstrations. Both the action and termination heads read from the shared backbone, without cross-attention or feature exchange.

Training

The intra-option policy π_θ is parameterized by a flow–matching model. We denote the low–level action chunk at control time t as $\mathbf{A}_t = \{a_t, \cdots, a_{t+H}\} \in \mathbb{R}^{H\times d_a}$. Let $\tau \in [0,1]$ denote the interpolation time, and ε a Gaussian noise variable. The noisy action is $\mathbf{A}_t^\tau = \tau\mathbf{A}_t + (1-\tau)\varepsilon$. The policy π_θ predicts the flow, $\dot{\mathbf{A}}_\theta(\mathbf{A}_t^\tau, s_t, \ell, e_k, n_{k+1})$, and is trained to match the oracle velocity $\mathbf{u}(\mathbf{A}_t^\tau \mid \mathbf{A}_t)$:

$$\mathscr{L}_{\text{flow}} = \mathbb{E}_{\mathscr{D}_{sys1}} \left\| \dot{\mathbf{A}}_\theta(\mathbf{A}_t^\tau, s_t, \ell, e_k, n_{k+1}) - \mathbf{u}(\mathbf{A}_t^\tau \mid \mathbf{A}_t) \right\|_2^2.$$

The done expert β_θ predicts the probability of end–of–edge, $p_\theta^{\text{done}} = \beta_\theta(s_t, \ell, e_k, n_{k+1})$, trained with focal loss

$$\mathscr{L}_{\text{done}} = \mathbb{E}_{\mathscr{D}_{sys1}}\big[-\alpha_d\,(1-p_\theta^{\text{done}})^{\gamma_d}\log p_\theta^{\text{done}}\big].$$

The joint system 1 objective is $\mathscr{L}_{\text{S1}} = \lambda_{\text{flow}}\mathscr{L}_{\text{flow}} + \lambda_{\text{done}}\mathscr{L}_{\text{done}}$.

Inference

Given the active pair (e_k, n_{k+1}) from τ^{sel}, we initialize $\mathbf{A}_t^0 \sim \mathscr{N}(0, \sigma_0^2 I)$ and integrate the learned flow along the interpolation path:

$$\mathbf{A}_t^{\tau+\delta} = \mathbf{A}_t^{\tau} + \delta\,\dot{\mathbf{A}}_\theta(\mathbf{A}_t^{\tau}, s_t, \ell, e_k, n_{k+1})$$

where $\tau \in \{0, \delta, \ldots, 1-\delta\}$. After sweeping to $\tau = 1$, we obtain the predicted action chunk $\widehat{\mathbf{A}}_t = \mathbf{A}_t^1$.

In parallel, the termination head outputs the end-of-edge probability $p_\theta^{\text{done}} = \beta_\theta(s_t, \ell, e_k, n_{k+1})$. We advance to the next symbolic step when this probability exceeds a threshold κ_{done}:

$$k \leftarrow k + \mathbf{1}\{\, p_\theta^{\text{done}} \geq \kappa_{\text{done}} \,\},$$
$$(e_k, n_{k+1}) \leftarrow \text{next pair from } \tau^{\text{sel}}.$$

If $k > K$, the option sequence terminates; otherwise, System 1 repeats the procedure for the updated pair.

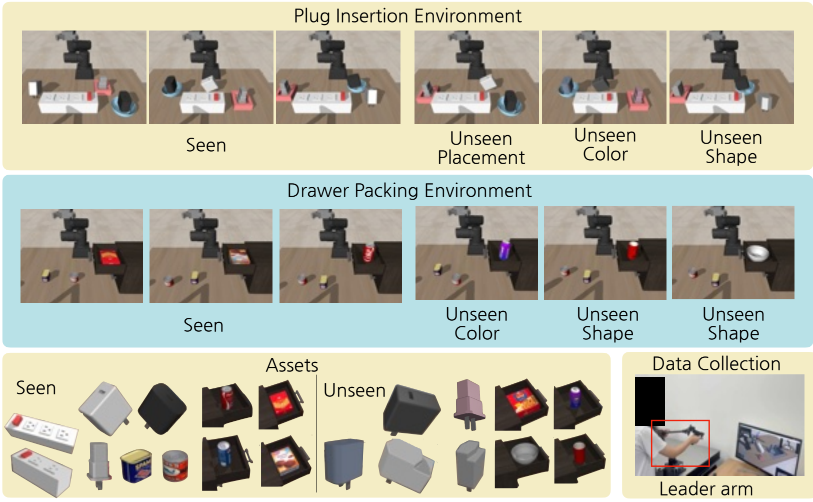


Figure 3.5: Plug insertion and drawer packing environment. We evaluate in both seen and unseen settings with corresponding assets, and collect demonstrations via a teleoperation setup.

3.5 Experimental Evaluation of Failure-Aware Hierarchical VLA

In this section, we present experiments designed to validate our framework. We evaluate whether explicitly leveraging failure data improves reliability by biasing plan selection toward paths with higher success probability. Our study spans three settings: (i) **Teleoperated failure-data environments**, where we collect mixed-quality demonstrations and test generalization under seen/unseen perturbations in Section 3.5.2; (ii) **Inference-time scalability**, where we hold model parameters fixed and scale the planner's per-step expansion width K to test whether allocating additional reasoning compute at deployment improves generalization; (iii) **Bootstraping pretrained models**, where rollouts from a strong pretrained model provide additional (often noisy) successes/failures that we *retrain* with our failure-aware method (Section 3.5.4); and (iv) **Real-world deployment**, where the selected high-level chain is executed on a physical robot (Section 3.5.5). We report success rate as the primary metric and compare against imitation- and VLM-based baselines that do not use failures explicitly.

3.5.1 Teleoperation Data Environments

To the best of our knowledge, there is no widely adopted benchmark that makes failure-aware evaluation of spatial reasoning straightforward; most existing resources only provide success demonstrations. We therefore introduce a new environment suite, where plug insertion and drawer packing are illustrative instances, with controlled geometry, contact, and placement variations under explicit seen/unseen splits.

Environment Setup

The environment is built upon a MuJoCo simulator, as illustrated in Figure 3.5. Plug insertion (seen: 9 configs) combines three table placements with 2- or 3-socket strips and orientation variants; *unseen* splits introduce novel charger placement, unseen strip color, and unseen charger shape. The strip collision margin is slightly relaxed to reduce teleop burden. *Drawer packing* (seen: 12 configs) pairs four table placements with two distractor types (a box and a can) placed inside the drawer; *unseen* splits (4 configs) alter distractor color and type. Unlike plug insertion, which is mostly constrained by insertion order, drawer packing admits multiple feasible behaviors (push the distractor aside, pick and remove it, or leave it), making contact outcomes highly *stochastic*. Small variations in contact geometry, stick–slip, and drawer clearance lead to large success variance even under near-identical initial states. Control runs at 20 Hz with joint-position actions.

Data collection

We gather human teleoperation trajectories via a leader–follower setup (Fig. 3.5). For *plug insertion*, we collect 450 demonstrations over predefined insertion orders (six paths for 3-socket, two for 2-socket), balanced per path. Each trajectory is labeled success/failure; outcomes are naturally stochastic but show clear mode structure: PlugStrip-3 Fail dominates (38.9%, 175/450), followed by PlugStrip-2 Success (30.4%, 137), PlugStrip-3 Success (21.1%, 95), and PlugStrip-2 Fail (9.6%, 43), yielding a multi-modal dataset whose success rate varies with insertion order. For *drawer packing*, we collect 240 demonstrations in total—120 per distractor type (box/can)—covering three strategies (PICK, PUSH, LEAVE). Compared to plug insertion, outcomes are strongly stochastic: success probabilities depend sharply on both strategy and distractor identity (e.g., LEAVE is often brittle when the can closely occludes the handle, whereas PICK/PUSH are more reliable but still contact-sensitive). This task, therefore, provides diverse, multi-modal successes and failures arising from real-world manipulation affordances rather than a single fixed plan. The detailed statistics of the dataset are shown in the Appendix.

Baselines

To assess our two-system framework, we compare against three groups: (i) unified VLA models, (ii) VLM-as-planner baselines that replace System2 while keeping System1 fixed, and (iii) our System 2 variants (Chain/Tree/Full) to isolate branching and failure-conditioned value. Since our environment operates at 20Hz in a joint-position action representation, we select baselines that can be executed under this control frequency and action format.

- **Unified VLA models.** OpenVLA-OFT [113], GR00T N1.5 [9], and π_0 [1], plus a

Table 3.1: Simulation Results in plug insertion and drawer packing environments, showing the success rate.

Models	Failure Data	Plug Insertion			Drawer Packing		
		Seen	Unseen	Average	Seen	Unseen	Average
Unified Model							
OpenVLA-OFT [113]	×	0.244	0.044	0.144	0.637	0.283	0.485
GR00T N1.5 [9]	×	0.422	0.244	0.333	0.704	<u>0.600</u>	0.669
π_0 [1]	×	0.689	0.267	0.477	0.735	0.575	0.675
π_0+ Reward Cond. [114]	✓	0.489	0.111	0.300	0.550	0.425	0.508
SOTA VLM as System2							
GPT-4o [115]	×	0.733	0.311	<u>0.522</u>	0.650	0.475	0.591
GPT-4o [115]	✓	0.711	<u>0.333</u>	0.488	<u>0.738</u>	0.575	<u>0.683</u>
Gemini-2.5-Flash [2]	×	<u>0.756</u>	0.200	<u>0.522</u>	0.637	0.450	0.575
Gemini-2.5-Flash [2]	✓	0.711	0.289	0.500	0.713	0.525	0.650
Role of failure data in Our System2							
VINE-Chain	×	0.733	0.244	0.488	0.700	0.450	0.616
VINE-Tree	×	0.711	0.289	0.500	0.732	0.525	0.663
VINE-Full (Ours)	✓	**0.800**	**0.422**	**0.611**	**0.800**	**0.675**	**0.752**

reward-conditioned π_0 variant using failure data [114].

- **VLM-as-System 2.** Replace our planner with SOTA VLMs (e.g., GPT-4o [115], Gemini-2.5-Flash [2]) using few-shot prompting to propose subgoals/next scene graphs, with and without failure examples; System 1 (our action executor) is fixed.

- **Our System 2 variants.** VINE-Chain (no branching without failure data), VINE-Tree (tree search without failure data, scoring with confidence), and VINE-Full (tree search + failure-conditioned value).

3.5.2 Main Results

We evaluate all models on success rate across both seen and unseen configurations of the plug insertion and drawer packing tasks. The results, summarized in Table 3.1, demonstrate that our proposed model, VINE-Full, consistently and significantly outperforms all baselines. Our analysis delves into the performance comparison against state-of-the-art models and investigates the impact of failure data by comparing variants of our own system that share the same underlying architecture.

Comparison with Unified VLA Models

Unified VLA models, which operate as end-to-end reactive policies, demonstrate a critical weakness in generalization. While some models like π_0 [1] achieve reasonable performance on seen configurations (0.689 in plug insertion), all unified VLAs experience a severe performance degradation when faced with novel scenarios. For instance, π_0's success rate plummets from 0.689 to 0.267 in the unseen plug insertion task. Similarly, OpenVLA-OFT [113] drops from 0.244 to a mere 0.044, and GR00T N1.5 [9] falls from 0.422 to 0.244. This consistent trend highlights their difficulty when learning long-horizon tasks from multi-modal demonstration data: they struggle to discern which of the many possible paths is most feasible. Lacking a mechanism for deliberation, they cannot effectively weigh alternative strategies, a challenge that our two-system architecture is designed to overcome.

Comparison with VLM-as-Planner Baselines

The VLM-as-planner baselines confirm the benefit of separating planning and control, as they consistently outperform unified VLAs in average performance. However, they are still surpassed by VINE-Full, particularly in novel situations. In the highly stochastic drawer-packing task, our model attains 0.752— a 10.1% relative gain over the best VLM baseline (failure-prompted GPT-4o [115], 0.683)—and 0.675 in the unseen split vs. 0.575. In plug insertion, we reach 0.611 vs. 0.522 for GPT-4o/Gemini-

2.5-Flash [2] (+17.1%), with the unseen split at 0.422 vs. 0.333 (+26.7%). While failure-augmented prompts improve VLMs (e.g., GPT-4o unseen drawer: 0.475→0.575), grounding remains limited. The VLM approach relies on in-context learning to adapt its general knowledge from a few textual examples, which provides strong reasoning but is not deeply grounded in the task's specific physical realities. This can lead to conceptually sound plans that overlook subtle failure modes, a vulnerability particularly exposed in novel settings. In contrast, our method's components are trained directly on demonstration data, creating a more grounded signal for a tree search. This allows our planner to assess a plan's feasibility by exploring its predicted consequences, making it more robust in novel scenarios.

Role of Failure Data and Tree Search

Our component analysis reveals that both tree search and failure-conditioning are critical to our model's success. The VINE-Chain model, a linear planner using only success data, struggles to generalize (0.244 unseen success). While leveraging tree search (VINE-Tree) improves unseen performance by allowing the model to explore alternatives, which is especially valuable in the multi-modal drawer packing task (unseen success jumps from 0.450 to 0.525). However, the most significant improvement comes from training the value function with failure data, which provides a signal to judge a plan's feasibility. VINE-Full boosts unseen success rates to 0.422 in plug insertion and 0.675 in drawer packing. This result decisively shows that a value function that internalizes the dynamics of failure provides a more reliable planning signal than the generalized confidence of system 2, making it key to robust performance in novel scenarios.

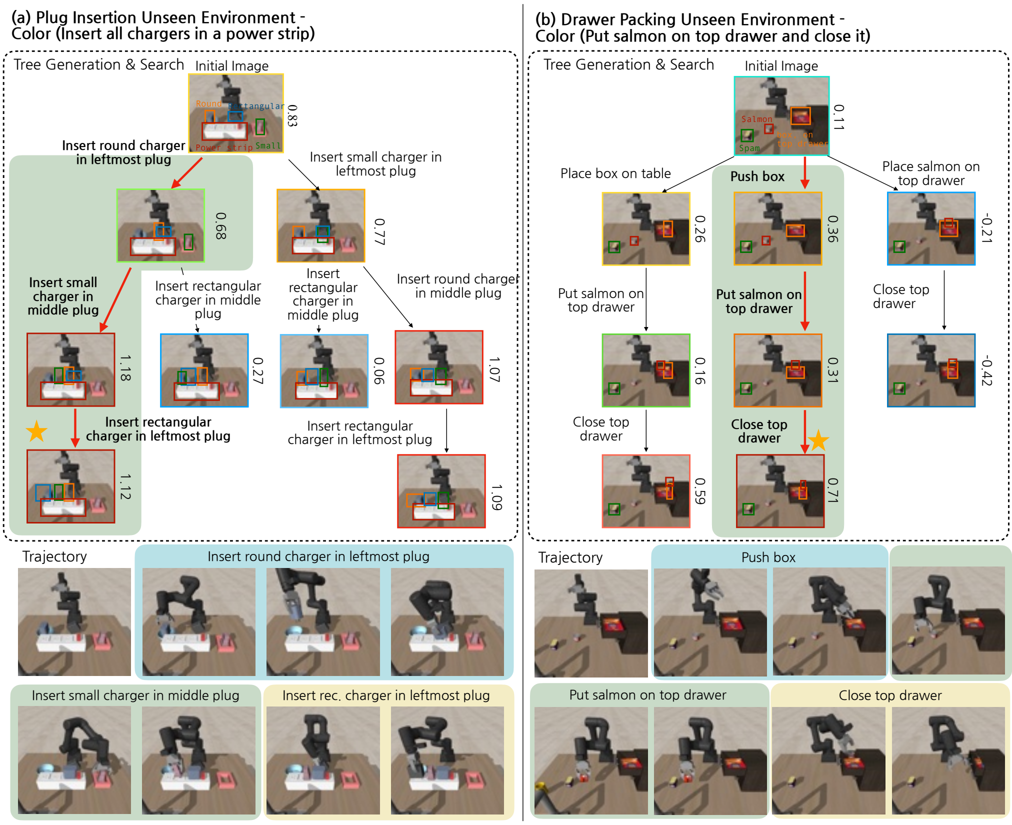


Figure 3.6: Qualitative results. System2 builds a failure-aware planning tree with feasibility scores, and System1 executes the selected subgoals, producing successful trajectories.

Qualitative Results

Figure 3.6 shows qualitative results of our model's planning and execution capabilities on two distinct, complex tasks. In the plug-initiation task (a), it successfully determines the correct multi-step sequence for inserting various chargers into their corresponding slots. For the Drawer Packing task (b), it demonstrates strategic reasoning by first pushing an obstacle to make space before placing the target object. The successful roll-out trajectories for each task validate that our system can effectively execute these

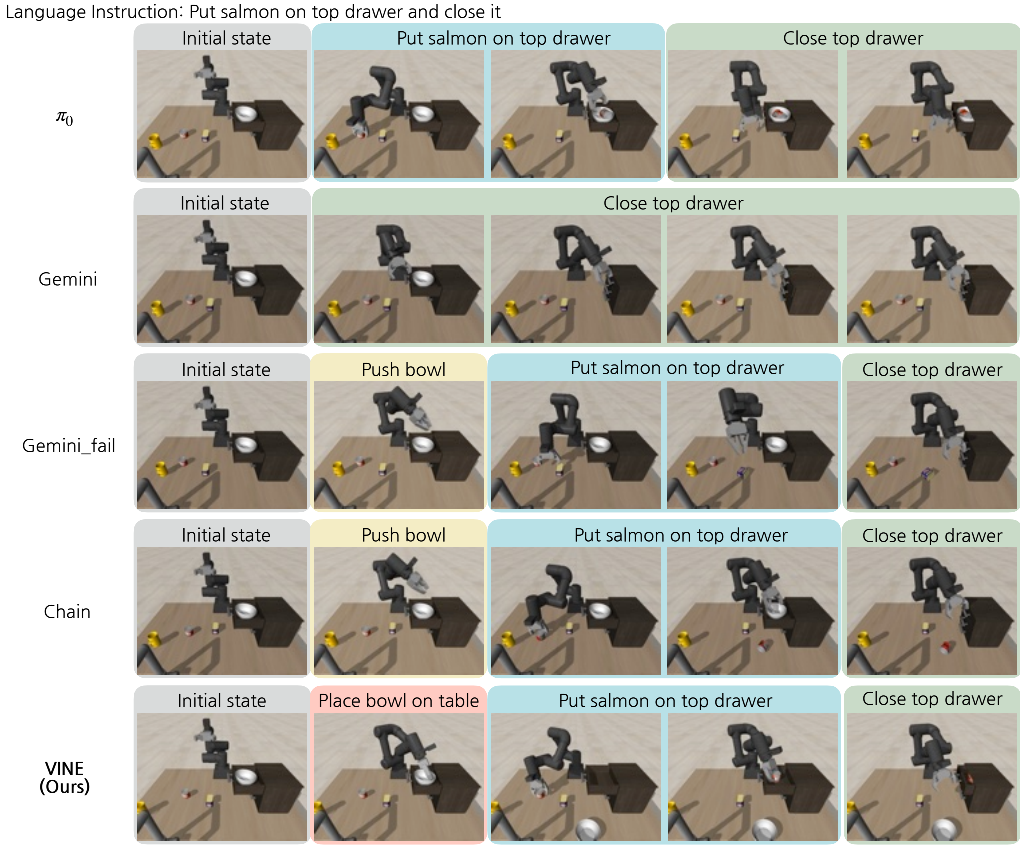


Figure 3.7: Trajectory Comparison with baselines. The figure shows the trajectory generated by π_0 [1], VLM as system 2, i.e., Gemini-2.5-flash [2] (Gemini), and with failure examples (Gemini fail), VINE-Chain (chain), and VINE-Full (Ours) in an unseen drawer packing environment.

complex, long-horizon plans. Furthermore, Figure 3.7 contrasts trajectories on drawer packing. Reactive baselines, π_0 [1] and Gemini-2.5-Flash [2] as System 2, ignore the obstructing bowl and attempt a direct placement, leading to collisions. An ablation using only success data without tree search ("chain") merely tries to shove the obstacle aside. In contrast, our model plans a feasible, multi-step solution: first relocate the bowl to the table, then place the salmon in the drawer, demonstrating stronger strategic reasoning and

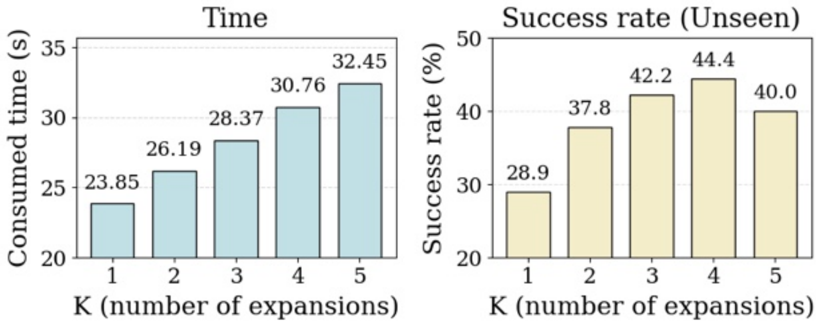


Figure 3.8: Tree Analysis for test time scaling across per-step expansion. In the unseen plug-insertion setting, increasing the search width K consistently improves success while increasing inference time, demonstrating a test-time scalability.

execution.

3.5.3 Test-Time Scalability

Humans naturally allocate more time to difficult problems, adjusting their reasoning effort to the complexity of the situation. In contrast, most VLA models commit to a single forward pass per observation, applying a fixed amount of computation regardless of task uncertainty. We hypothesize that allocating additional inference-time reasoning, by expanding and verifying multiple subgoal hypotheses, can improve robustness without retraining. This setting allows us to study whether compute adaptivity at deployment can serve as an additional axis of generalization.

To evaluate this, we vary the planner's per-step expansion width K, which controls how many candidate subgoal branches are explored and scored before acting (Figure 3.8) in unseen configurations of plug-insertion environment. Increasing K expands the search tree and enables broader reasoning over plausible alternatives, while keeping model parameters fixed. As K increases, unseen success improves $28.9\% \rightarrow 37.8\% \rightarrow 42.2\% \rightarrow 44.4\% \rightarrow 40.0\%$ for $K{=}1,2,3,4,5$, with a near-linear latency growth of $23.9\text{s} \rightarrow 26.2\text{s} \rightarrow$

Table 3.2: Results from Simpler Environment. Carrot. denotes carrot on plate, eggplant. denotes eggplant in basket, and spoon. for spoon on towel.

Method	carrot.	eggplant.	cube.	spoon.	Average
π_0 (Finetuned)	0.236	**0.791**	0.139	0.388	0.389
VINE (Ours)	**0.375**	0.708	**0.145**	**0.395**	**0.406**

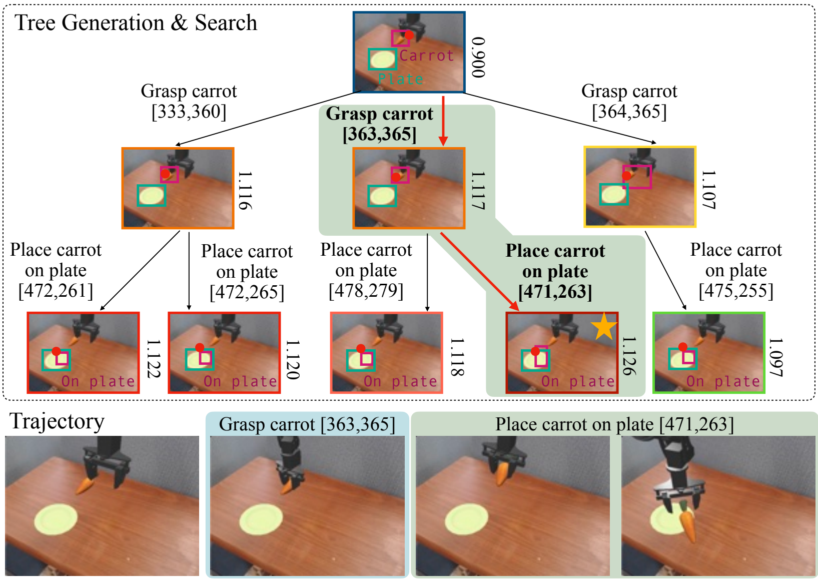


Figure 3.9: Generated tree and trajectories in Simpler environment.

28.4s → 30.8s → 32.5s. These results demonstrate that modest increases in test-time reasoning yield measurable improvements in success under distribution shift, reflecting a favorable accuracy–latency trade-off and diminishing returns beyond $K=4$.

3.5.4 Bootstrapping Pretrained Models

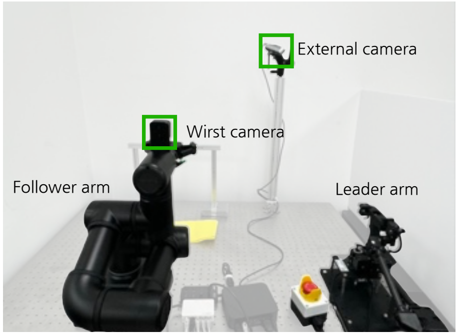
In this experiment, we bootstrap a pretrained policy by rolling out a Bridge v2 [116]–fine-tuned π_0 to collect 600 additional raw trajectories, then re-train with our failure-aware procedure and merge the adapted weights (PaliGemma backbone + finetuned π_0). We evaluated the proposed method in Simpler environment [117] with the Widow-X robot in four different tasks: carrot. (*carrot on plate*), eggplant. (*eggplant in basket*), cube. (*stack cubes*) and spoon (*spoon on towel*). In this environment, we define edge as a subgoal with a 2D end-effector position in the image, as there does not exist multiple available subgoals to succeed the tasks. To balance the dataset between four tasks, we collected more data on lower success rate tasks like *stack cubes* or *carrot on plate*.

As summarized in Table 3.2, our method yields a modest but consistent gain in average success rate on the *Simpler* tasks (from 0.389 to 0.406), with clear improvements on *carrot on plate* (0.236 to 0.375), *stack cubes* (0.139 to 0.145), and *spoon on towel* (0.388 to 0.395). The only regression appears on *eggplant in basket* (0.791 to 0.708), where the baseline already exhibited a very high success rate, suggesting diminishing returns and occasional overfitting or value recalibration when bootstrapping in near-saturated regimes. Overall, the results indicate that rolling out a trained policy and re-training it with our failure-informed updates can improve robustness and transfer to previously weaker objects while preserving strengths elsewhere. The example of a generated tree and accompanying trajectory is illustrated in Figure 3.9.

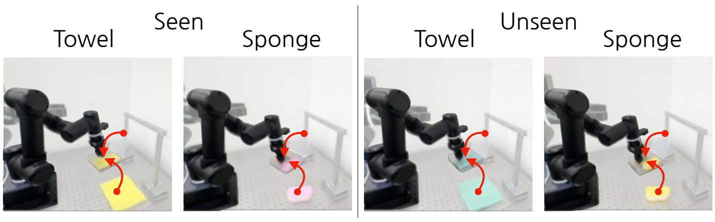
3.5.5 Real-world Demonstration

Environment Setting and Tasks

Our experiments are conducted using a 6-DoF robotic arm controlled at a 20Hz frequency with joint position actions, receiving visual input from both a wrist-mounted and a fixed third-person camera. We introduce two multi-modal, long-horizon tasks that require placing an object into a container and subsequently closing its lid. The environment



(a) Real-world workspace and setup.



(b) Tasks used in evaluation.

Figure 3.10: Real-world environment and tasks. **(a)** Setup with a 6-DoF arm (joint-position control, wrist + external cameras). **(b)** Tasks: sponge and towel packing in the cabinet, with distractors and folding strategies. Generalization is tested from *seen* to *unseen* color configurations.

Table 3.3: Real-world success rates on sponge and towel packing tasks. Our method (VINE) outperforms the π_0 baseline in both seen and unseen settings.

Method	Seen		Unseen	
	sponge	towel	sponge	towel
π_0 [1]	0.60	0.45	0.55	0.30
VINE (Ours)	**0.75**	**0.55**	**0.65**	**0.55**

contains two different language instructions. The first task, sponge placement, involves placing a sponge inside a container already occupied by a clock, which admits three distinct solution strategies: pushing the clock aside, removing it from the cabinet entirely, or placing the sponge directly into the available free space. The second task, towel place-

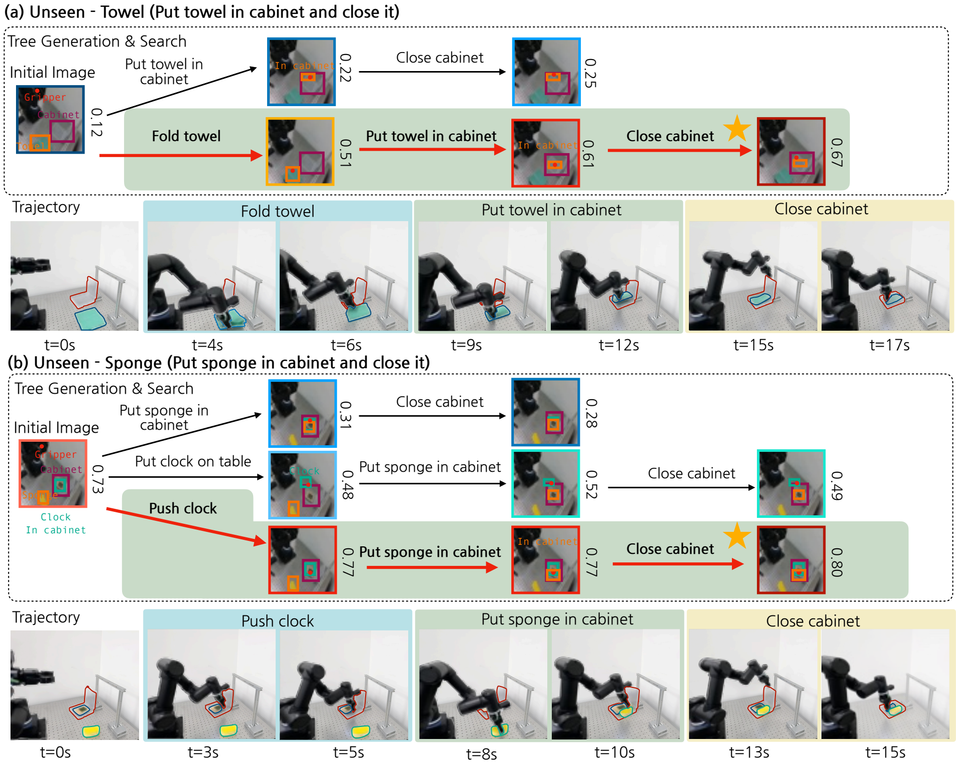


Figure 3.11: Real-world demonstrations with planning and execution. **(a)** *Towel–unseen*: the search tree selects *fold* → *put in cabinet* → *close*, then System 1 executes the chosen path. **(b)** *Sponge–unseen* (clock distractor): the planner chooses *push clock* → *put sponge* → *close* and the trajectory follows accordingly.

ment, involves placing a towel, which can be completed with or without an optional intermediate folding step. To evaluate generalization, we define *seen* configurations using a pink sponge and a yellow towel, and test on *unseen* configurations that introduce a domain shift via a yellow sponge and a green towel, as depicted in Figure 3.10. We collect 20 expert demonstrations for each of the five distinct strategies, for a total of 100 trajectories.

Results

We evaluate our method, VINE, against the baseline on the real-world manipulation tasks, with quantitative results presented in Table 3.3. For seen objects, our model achieved success rates of 75% (sponge) and 55% (towel), compared to the baseline's 60% and 45%. The performance gap was larger for unseen objects, where our model succeeded 65% (sponge) and 55% (towel) of the time, while the baseline dropped to 55% and 30%, respectively. This improvement comes from our model's ability to choose the most feasible strategy in a given situation. Figure 3.11 showcases planning and action generation on unseen tasks. In towel placement (a), the model prefers a three-step plan—Fold towel $\rightarrow$ Put in cabinet $\rightarrow$ Close cabinet—because it predicts a higher success value than placing it directly. In sponge placement (b), with a clock blocking the cabinet, it first pushes the clock aside before placing the sponge, again selecting the most feasible path. System 1 then executes the chosen sequence to produce the shown trajectories, demonstrating practical effectiveness on a physical robot.

3.5.6 Plug-in Module for Replanning

To mitigate drift arising from executing long action–chunks without feedback, we attach a lightweight uncertainty monitor [118] that enables *test-time replanning* without retraining. When the within–chunk variance becomes high, the monitor down-weights the currently committed branch and triggers a switch to the next best alternative in the search tree. This converts our quasi-static, open-loop execution into an instance-adaptive procedure that can abandon deteriorating motions. In the unseen drawer packing setting, this mechanism improves success from 67.5% (no replanning) to 74.0% (with replanning), exceeding the π_0 baseline's 57.5% while leaving all learned parameters un-

Figure 3.12: Success rates in the unseen drawer packing environment with and without replanning. A lightweight uncertainty-triggered replanning module, which down-weights uncertain branches and switches to alternatives, boosts success in drawer packing without retraining.

changed (Figure 3.12). Although task-specific calibration remains, these results indicate that uncertainty-triggered replanning partially closes the gap introduced by Limitation 1. Figure 3.13 illustrates test-time replanning, where the model first chooses the green path—pushing the obstructive box aside—but, upon detecting high uncertainty, it switches (yellow background) to the second-best plan: placing the box on the table. Despite the benefits, the current uncertainty calibration is manual and task-specific, and the added computation limits real-time use. We view data-driven calibration and learned failure signals (e.g., extending the done-expert dimension) as promising next steps. These replanning results are orthogonal to our main contributions but suggest a practical path to improve robustness at test time.

3.5.7 Limitations

Despite the advantages, we have three main limitations. First, planning occurs at subgoal granularity without real-time, within-chunk feedback. We deliberately avoid a

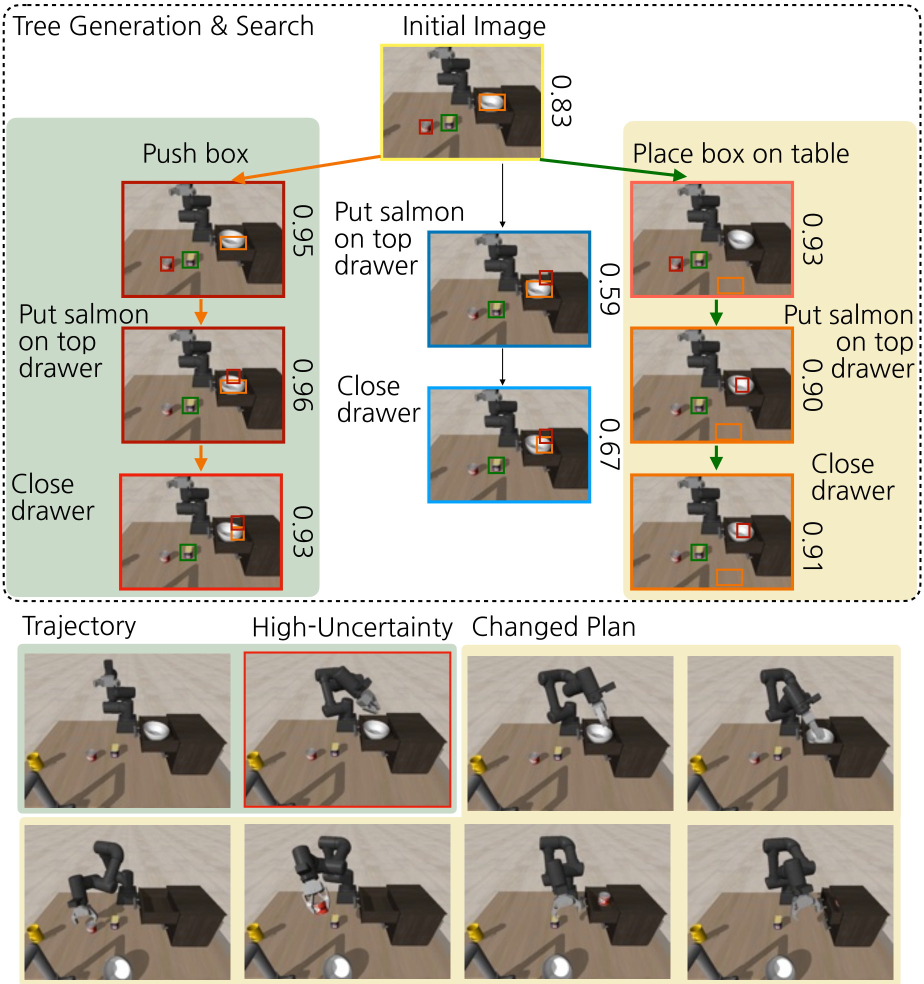


Figure 3.13: Qualitative Results for replanning. Planner first selects the green plan (*push box→place item→close*), then detects high uncertainty during the first step and switches to the yellow alternative (*place box on table→place item→close*), successfully completing the task. Shaded panels indicate the uncertainty trigger and the plan switch; bottom rows show the executed trajectories.

stop-and-replan loop because it effectively enforces a quasi-static assumption and interrupts continuous motion; in dynamic scenes, small pose errors, sensor latency, or contact disturbances can therefore accumulate between replans, leading to drift in the predicted successor state and occasional divergence between the planned path and what the robot can stably execute.

Second, the low-level controller (System 1) is trained solely from successful executions. Lacking failure-conditioned gradients, it tends to overfit nominal trajectories and shows limited ability to self-correct once perturbed off the known distribution. Third, the state abstraction relies on a 2D scene graph that omits metric geometry, contact cues, and short-horizon dynamics that are critical in contact-rich manipulation. This can miscalibrate feasibility estimates when objects are partially occluded, slightly mislocalized, or when frictional effects dominate. The resulting gaps increase sensitivity to calibration drift and viewpoint changes, and can cause the search to misrank otherwise viable branches or prune plans that would succeed under the true physical dynamics.

3.6 Handling Uncertain User Commands through Human Interaction

Even with a failure-aware planning framework such as VINE, reliable execution alone does not guarantee reliable interaction. A robot may correctly anticipate and avoid risky action sequences, yet still receive verbal instructions that are incomplete, ambiguous, or entirely infeasible given the scene and its own capabilities. In such cases, failure does not arise from poor planning but from misunderstanding the user's intent. This makes language interpretation—specifically, determining whether a command is clear, ambiguous, or infeasible within the current environment—an essential component of robustness. The meaning of the same instruction can vary drastically across contexts, and the robot must decide not only *how* to act but also *whether it can or should act*. Therefore, handling uncertainty in user commands becomes a necessary complement to failure-aware action planning.

User instructions to robots are often incomplete, ambiguous, or even infeasible, and their correct interpretation depends strongly on the surrounding environment and the robot's capabilities. The same command may lead to different meanings across contexts, making situational reasoning essential for reliable interaction. In this section, we address the problem of identifying whether a user command is clear, ambiguous, or infeasible within a given scene, and outline an approach that allows the robot to recognize uncertainty in language and seek clarification when needed.

We aim to classify user commands into clear, ambiguous, and infeasible ones with the awareness of robotic situations and process disambiguation on ambiguous commands. *Situational awareness* is required for this problem, as even the same command can have different meanings for different situations. For instance, when the user command is "he looks tired, can you help him?" with an environment containing coffee, a coffee machine, water, and bread, there can be different interpretations among robot agents. It is certain that the cooking robot, which can only cook food or beverages with a fixed base in the kitchen, should make coffee. While for the cleaning robot, the goal may be categorized as infeasible. This command may be ambiguous for the massage robot, and the robot may ask back to gain further information to disambiguate the lack of information on the command.

To this end, we propose CLassifying and disAmbiguating user commands for reliable interactive Robotic Agents (CLARA) to enhance the reliability of the interactive system. The proposed method is composed of two parts: distinguishing a command between clear and not (i.e., ambiguous or infeasible) and classifying ambiguous or infeasible commands for unclear ones. As uncertainty can arise due to both incomplete information or limitations in the agent's capabilities, we first present a method to estimate predictive uncer-

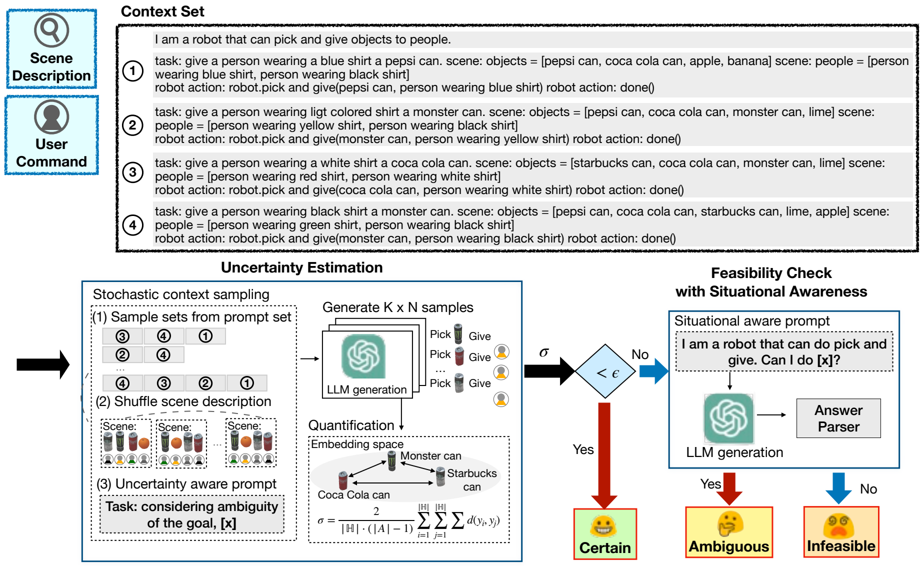


Figure 3.14: Proposed Method. Our method involves estimating uncertainty with LLMs via context sampling to distinguish between certain and uncertain commands. We then leverage situational awareness to classify uncertain commands into ambiguous and infeasible categories, followed by a disambiguation process for ambiguous commands. The number (1) (2), etc., denotes the index of the context from the context set (C). σ denotes predictive uncertainty, and ε is an uncertainty threshold.

tainty for LLMs. Then, we introduce an approach to check feasibility in uncertain commands with situational awareness built upon the zero-shot capability of LLMs to distinguish between infeasible and ambiguous commands. Interacting with users in free-form texts via question generation (i.e., disambiguation) is also conducted on the commands classified as ambiguous. The proposed method can reuse most of the prompts and the structure when the environment changes, with modifications to the few-shot demonstrations and the robot capability description. We also designed a benchmark dataset called Situational awareness for Goal Classification in robotic tasks (SaGC), containing pairs of high-level user commands, environments including objects and robot capabilities, and uncertainty types to capture situational awareness in robotic tasks.

3.7 Problem Formulation: Recognizing and Categorizing Command Uncertainty

We focus on capturing and classifying uncertain user commands in the context of interactive agents via large language models (LLMs). We assume that the high-level goal command may lack sufficient clarity to execute a task properly, or it may be vague or even an infeasible goal due to users' lack of situational understanding. Our objective is to predict the uncertainty in the LLM's predictions and then predict the type of uncertain goals, i.e., ambiguous or infeasible. If the goal is feasible but ambiguous, we disambiguate the command by generating questions for the user to gather additional information. The input of the system is the high-level goal (x^g) and lists of objects in the environment ($\mathbf{x}^s$), along with few-shot contexts ($\mathbf{c}$). LLM then either generates low-level short-horizon skills (y), which can be easily interpreted into a robotic action with uncertainty (σ), or an explanation of the uncertainty via text.

3.8 Proposed Method for Handling Uncertain Commands with User Interaction

In this section, we outline our approach to handling uncertain user commands with situational awareness. We begin by presenting a method for estimating uncertainty from large language models. Once uncertainty is estimated, we can distinguish the certain and uncertain inputs with thresholds. We introduce the zero-shot approach to check the feasibility of the uncertain commands, which can classify them into ambiguous and infeasible ones. We present a disambiguation approach to predicted ambiguous inputs in a zero-shot manner. The proposed method is illustrated in Figure 3.14.

3.8.1 Uncertainty Estimation of LLMs via Context Sampling

Large Language Models (LLMs) operating in an in-context learning (ICL) setting [11] exhibit prediction variability depending on the ordering and selection of contextual demonstrations. Recent theoretical work shows that demonstration tokens induce *implicit parameter updates* to the underlying transformer [119], suggesting that modifying context examples corresponds to drawing different effective models. Leveraging this observation, we introduce a method to estimate epistemic uncertainty in LLMs using *context sampling*, without requiring model gradients, logits, or weights—making it directly applicable to closed-source APIs such as ChatGPT.

Setup. Let $C = \{C_1, \dots, C_m\}$ denote a set of demonstration examples. Given a goal description x^g and scene observation $\mathbf{x}^s$, the LLM produces a structured output:

$$\mathbf{y} = f(x^g, \mathbf{x}^s, C). \tag{3.2}$$

To induce stochasticity analogous to Bayesian posterior sampling, we draw $\mathbb{H}$ context samples $\mathbf{c}_i$ from a distribution $p(C, \mathbf{x}^s)$ defined by (i) randomly subsampling k elements from C, and (ii) randomly shuffling their order. Each sampled generation is:

$$\mathbf{y}_i = f(x^g, \mathbf{x}^s, \mathbf{c}_i), \qquad \mathbf{c}_i \sim p(C, \mathbf{x}^s). \tag{3.3}$$

Keyword-Level Representation. Each generated output $\mathbf{y}_i$ is decomposed into a set of K task-relevant keywords $\{\mathbf{y}_i^1, \dots, \mathbf{y}_i^K\}$ (e.g., the target object in a low-level pick-and-place instruction). We embed these keywords using a pretrained model $g(\cdot)$:

$$\mathbf{z}_i^k = g(\mathbf{y}_i^k) \in \mathbb{R}^d. \tag{3.4}$$

Template text such as `robot.pick` is removed prior to embedding so that variability reflects task-relevant content.

Uncertainty via Pairwise Dispersion. We quantify epistemic variability by computing normalized pairwise distances across all sampled generations:

$$\sigma = \frac{1}{K}\sum_{k=1}^{K} \frac{2}{\mathbb{H}(\mathbb{H}-1)} \sum_{1\le i<j\le \mathbb{H}} \left\|\mathbf{z}_i^k - \mathbf{z}_j^k\right\|_2. \tag{3.5}$$

When the goal is well-specified, perturbations to the context yield consistent predictions and σ remains low. When the goal is ambiguous or under-specified, predictions become more sensitive to the sampled contexts and σ increases.

Uncertainty-Aware Prompting. To further amplify this sensitivity, we prepend the following meta-instruction:

``Considering ambiguity of a goal, [x^g]'',

encouraging the LLM to reflect explicitly on possible uncertainty. Following [120], we compute the empirical CDF of σ for a set of "clear" samples and use the 80th percentile as a threshold to classify certain vs. uncertain predictions.

Illustrative Example. Consider the goal *"pick a block"* and four few-shot demonstrations $C = \{C_1, C_2, C_3, C_4\}$. Sampling different subsets and shuffle orders yields:

- Sample 1: $\{C_1, C_3, C_4\} \rightarrow$ `robot.pick(yellow block)`
- Sample 2: $\{C_2, C_4\} \rightarrow$ `robot.pick(red block)`
- Sample 3: $\{C_3, C_2\} \rightarrow$ `robot.pick(blue block)`

With $K = 1$ keyword per output, the embedding distances form:

$$\begin{pmatrix} 0 & 2.5 & 1.7 \\ 2.5 & 0 & 2.3 \\ 1.7 & 2.3 & 0 \end{pmatrix},$$

yielding a total uncertainty score of $\sigma = 2.1$ based on Eq. (3.5). A rigorous derivation showing that context sampling in ICL yields Monte Carlo draws from a dropout-style variational posterior is deferred to Appendix A.

3.8.2 Classification and Disambiguation

As quantifying uncertainty can only determine whether the given command is certain or not, we present a method to analyze and explain the uncertainty. It is composed of three parts: feasibility check, reason generation, and question generation, where all of the procedure is based on prompting LLMs while expanding the previous prompts. Although the proposed method requires few-shot prompts to generate robotic action, we call this classification and disambiguation zero-shot because the prompts provided do not encompass any instances of classification or disambiguation progress. Initially, we perform a feasibility check to assess the viability of the goal in relation to situational awareness by crafting the last line of the prompt with the robot's capability. We first add robot types and the possible actions that the robot can do into a prompt to ensure the agent is aware of their situation; then we force the large language model to conduct a binary classification if a robot can perform the task with answer `"yes"` or `"No"`. With the generated answer, we use a heuristic parser to distinguish commands from infeasible and ambiguous based on the keyword (i.e., `yes`); if the generated sentence contains the keyword `"yes"`, we denote the corresponding command as ambiguous. Continuing from the previous example, the prompt is as follows:

```
(Continue from previous prompts)
robot thought: I am a robot that can pick an object.
Considering the action set, pick, can I pick a block ? Answer in yes or no
answer: Yes, I can pick a block given more information.
```

If the robot is deemed capable of performing the task, we proceed to disambiguation by generating the reason for the uncertainty and posing a question to the user to gather additional information. The reason for the uncertainty and question is generated by prompts like `"This code is uncertain because"` and `"What can I ask the user? Please "` to the prompt respectively. Again, from the previous example, the prompt for generating an explanation for uncertainty and question are as follows:

```
(Continue from previous prompts)
robot thought: this code is uncertain because the task does not specify any specific criteria
for selecting the block
robot thought: what can I ask to the user?
question: Please provide more information about the criteria for selecting a block
```

After obtaining an answer from the user, the system goes back to the uncertainty estimation step with extended prompts with the disambiguation process.

3.9 Experimental Evaluation of Interaction-Based Resolution

In our experiments, we aim to address the following research questions with respect to uncertainty with situational awareness: (1) How does the efficacy of our proposed CLARA, contrast with previous approaches employed for uncertainty quantification across diverse environments? (2) To what extent can our proposed method accurately identify the user commands that are clear, ambiguous, or infeasible? (3) What role does the uncertainty-aware interaction module play in clarifying ambiguous commands? (4) Is it viable to deploy the proposed method in real-world human-robot interaction scenarios?

3.9.1 Baselines

Regarding uncertainty quantification, we compare the proposed method with four previous approaches: entropy, normalized entropy, semantic uncertainty, and lexical similarity. The predictive entropy, widely recognized as a baseline for uncertainty estimation, is represented as $\mathscr{H} = -\sum_{t=1}^{T}\sum_{v=1}^{V} p(y_t^v|x)\log p(y_t^v|x)$, where V is vocabulary size, and T is the sequence length. Normalized entropy [40] is predictive entropy normalized by sequence length (NE), $\mathscr{H}_{norm} = -\frac{1}{T}\sum_{t=1}^{T}\sum_{v=1}^{V} p(y_t^v|x)\log p(y_t^v|x)$. Semantic entropy (SE) [41] estimates the entropy of the random variable representing the output distribution in the semantic event space, as $SE(x) \approx |L|^{-1}\sum_{i=1}^{L}\log p(L_i|x)$, where semantic equivalence classes are represented as L_i. Lexical similarity (LS) [121] uses the average similarity of the answers in the answer set, $\mathbb{A} = \frac{2}{|\mathbb{H}|\cdot(|\mathbb{H}|-1)}\sum_{i=1}^{|\mathbb{H}|}\sum_{j=1}^{|\mathbb{H}|}\sum dist(y_i, y_j)$, where the answer set is sampled by beam search of the multinomial distribution.

Furthermore, we validate the effectiveness of the proposed classification and disambiguation method; we compared the proposed method with two different approaches that utilize zero-shot or few-shot capabilities. First is Inner Monologue [37], which generates questions and explanations based on the few-shot prompts containing feasibility checks and question generations. In addition, we also compare the proposed method with CLAM† [122], which requests input that asks if the goal is certain, ambiguous, or infeasible to the large language model and then further processes disambiguation [1]. We test the method on three different LLMs; LLaMA [123], ChatGPT (`GPT3.5-turbo`), and InstructGPT (`text-davinci-003`) models via the OpenAI API[2]. Inst. GPT denotes InstructGPT in the following tables. In the pick-and-place simulation, we utilize the LLaMA 30B; on the collected dataset, we utilize LLaMA 7B. Due to a lack of computing power, we did not test LLaMA in a real-world environment.

3.9.2 Situational Awareness for Goal Classification in Robotic Tasks

In this section, we assess the classification performance with situational awareness of the proposed method. In particular, we aim to evaluate the ability to classify the type of the user's command, e.g., clear, ambiguous, or infeasible, while considering the robotic capabilities and environments. We first introduce a dataset specifically designed to evaluate uncertainty in uncertain goals for robotic tasks. Then, we measure the method's performance in accurately identifying uncertain commands and categorizing the specific type of uncertainty present in the robot's situation.

Dataset Formulation

We first present a dataset called Situational Awareness for Goal Classification in Robotic Tasks (SaGC), curated to evaluate the situation-aware uncertainty of the robotic tasks. We collected a dataset consisting of high-level goals paired with scene descriptions annotated with three types of uncertainties inspired by previous work [124, 125] on collecting data via LLM. The primary aim of this dataset is to evaluate whether the language model can effectively distinguish between the three types of goals. The dataset consists

[1] †Denotes CLAM [122] modified to zero-shot instead of few-shot.
[2] https://platform.openai.com/docs/model-index-for-researchers

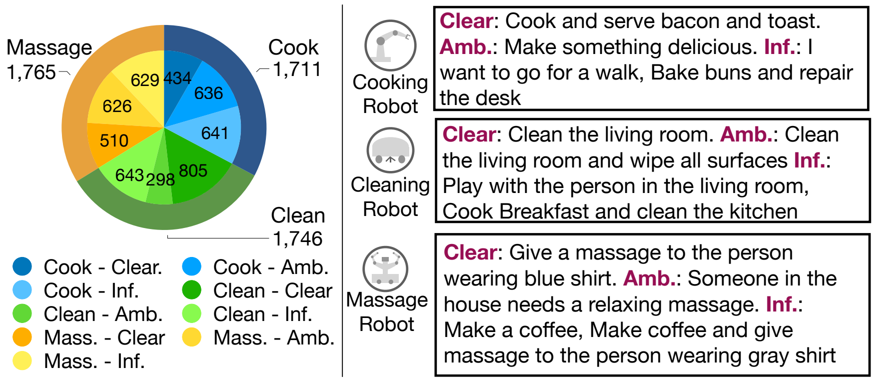


Figure 3.15: Statistics and Examples of the Dataset. Cer. denotes certain, Inf. denotes infeasible and Amb. denotes ambiguous

		LLaMA [123]	GPT3.5-Turbo	Inst. GPT
Quantification	Entropy	0.714	-	0.861
	Normalized Entropy [40]	0.736	-	0.867
	Semantic Entropy [41]	0.700	-	0.862
	Lexical Similarity [121]	0.690	0.628	0.852
	CLARA (Ours)	0.725	0.710	**0.870**
Classification	Inner Monologue [37]	0.368	0.480	0.513
	CLAM† [122]	0.362	0.376	0.532
	CLARA (Ours)	0.447	0.556	**0.710**
Abla.	Inst. GPT	w/o UAP.	w/o CS.	Ours
		0.861	0.852	**0.870**

Table 3.4: Results on SaGC Dataset. Quanantification denotes the uncertainty quantification part, where we measured AUROC, and Classification denotes the classification part where we measured accuracy. Abla. is an ablation study on uncertainty estimation, where CS. denotes context sampling and UAP. denotes uncertainty-aware prompt.

of 15 different scenes, encompassing three different robot categories: cooking, cleaning, and massaging. To construct the dataset, we initially created three certain high-level goal examples for each scene and robot, resulting in a total of 105 crafted goals. Crafted examples are formatted into a prompt with the scene description and a type of robot, to be expanded with LLM, i.e., `gpt-3.5-turbo`. We employed three different prompts to generate goals of varying uncertainty types, which automatically generate the goal based on the given uncertainty label. Furthermore, we also added more complex infeasible commands into the dataset where it is partially feasible, such as `"bake buns and repair the desk"`. The four validators were then asked to validate the corresponding generated pairs. They were asked to discard the sample, change the label of the sample, or accept the sample. The dataset comprises 5,222 pairs in total, 1,749 certain, 1,560 ambiguous, and 1,917 infeasible goals. The overall statistics and the examples of the dataset are illustrated in Figure 3.15.

Results

We evaluate the proposed method on the dataset by two different measures. First, we measure area under the ROC curve (AUROC) between certain and uncertain high-level goals with the uncertainty quantification baselines. We aim to see how the uncertainty estimation method can separate certain goals from uncertain goals, as shown in Table 3.4. The proposed method with IntructGPT outperforms the compared baselines with a minor gap of 0.003 and is second-best on LLaMA 7B model. We observe that selecting InstructGPT or GPT3.5 would be a better choice in the proposed method. We would like to emphasize that the proposed method can work compatible with previous methods without the access of token-wise probability. In addition, compared to Lexical Similarity [121], enforcing the stochastic in the prompts leads to diverse generations in uncertain inputs, as LLM is known to be fragile to recency bias. Furthermore, we have conducted ablation studies to further analyze the two hypotheses: context sampling and uncertainty-aware prompting. We observe that the performance drops by 0.018 and 0.09 without utilizing those modules.

Furthermore, we measure the classification accuracy of the whole system in Table 3.4, where the proposed method with IntructGPT outperforms the previous method with a gap of 0.158. We also observe that the proposed method outperforms the baselines in three different large language models. We posit that the ability to understand both robotic situations and uncertainty improves accuracy. We observe that a few-shot-based method [37] can have its weakness in generalization to unseen uncertain commands, and naively asking LLM to conduct a three-way classification has its weakness due to the hallucination issue of LLM and not being fully aware of the situation. We believe that leveraging uncertainty to filter out certain goals and addressing the situational awareness in-context approach afterward improves classification accuracy, making the model easier to predict. Although the dataset is formulated by `gpt-3.5-turbo`, the performance using `text-danvici-003` (instruct GPT) recorded higher classification accuracy, showing a stronger correlation in the model size of LLMs. The illustrations of generated responses are shown in Figure 3.16.

3.9.3 Pick and Place Simulations

In this section, we evaluate the effectiveness of the proposed method on tabletop pick and place simulation. We first aim to observe if the uncertainty estimation method can discriminate certain tasks between ambiguous tasks. In addition, we explore the efficiency of the proposed disambiguation module. We followed the task presented on Inner Monologue (IM) [37], which contains eight different types of goals. As these tasks lack the number of ambiguous scenarios, we added three different types of ambiguous goals and divided certain and ambiguous commands. We define an ambiguous task as a goal

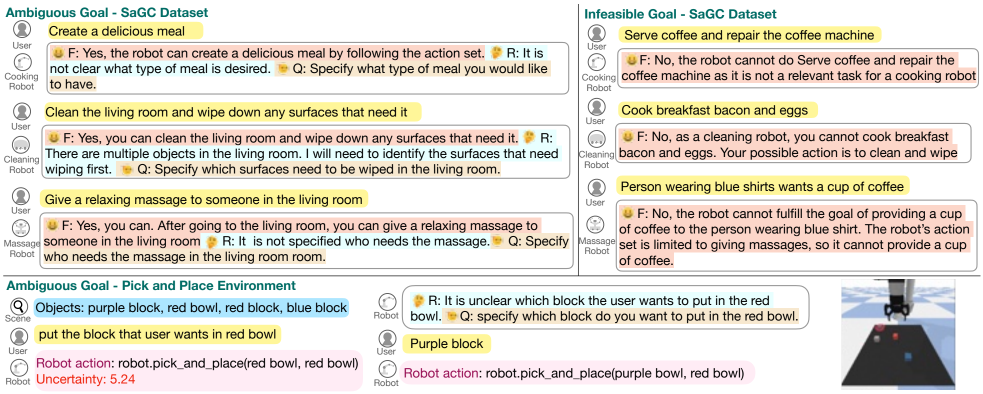


Figure 3.16: Examples of generated explanation and question from the proposed method. F, R, Q means Feasibility, Reasoning, and Question, respectively.

Categories	Tasks
Clear	pick [x] and put on [x] bowl
	place all blocks on [x] corner
	place all blocks on [x] bowl
	put all blocks on different corners
	place blocks on matching color
	place blocks on mismatching color
	stack all blocks on [x] corner
Ambiguous	pick block that user wants and place on [x] bowl
	pick [x] block and put on the bowl that the user wants
	pick the block and put in the bowl
	stack all blocks

Table 3.5: Task explanation in Pick-And-Place Environment

that does not specify the exact position or name for either pick or place object, as shown in Table 3.5. We utilize the ViLD [126] for scene description, and Cliport [127] for text-to-robotic policy. We evaluated the proposed method on 108 certain configurations and

		LLaMA [123]	GPT-3.5	Inst. GPT
Quantification	Entropy	0.725	–	0.819
	Normalized Entropy [40]	0.642	–	0.839
	Semantic Entropy [41]	0.688	–	0.620
	Lexical Similarity [121]	0.580	0.623	0.762
	CLARA (Ours)	0.602	0.731	**0.864**
Ablation	Inst. GPT	w/o UAP.	w/o CS.	Ours
		0.801	0.762	**0.864**

Table 3.6: Uncertain Task Detection in the pick-and-place environment. The *Quantification* block reports AUROC for different uncertainty measures; the *Ablation* block varies CLARA's components on Inst. GPT (CS = context sampling, UAP = uncertainty-aware prompt).

	LLaMA [123]		GPT3.5		Inst. GPT	
	F1	Gap.	F1	Gap	F1	Gap
Inner Monologue [37]	0.12	0.06	0.60	**0.49**	0.49	0.26
CLAM† [122]	0.49	0.22	0.48	0.24	0.41	0.30
CLARA (Ours)	0.33	0.07	0.49	0.24	**0.64**	0.39

Table 3.7: Disambiguation on Pick-and-Place Environment. Gap denotes the success rate gap after interaction on ambiguous commands.

60 ambiguous configurations, 188 in total. In the interaction phase, we assume that the system can request a question from the user only once.

We first assess the efficacy of our proposed method for uncertainty estimation by measuring the area under the ROC curve (AUROC) between certain and uncertain commands, shown in Table 3.6. The proposed method with the IntructGPT model outperforms the compared method with margins of 0.025. In addition, by ablation studies, we observe that the uncertainty-aware prompt showed improvement of the AUROC with a

0.063 gap, and AUROC increased 0.102 with context sampling. This leads us to posit that applying both stochastic context and uncertainty-aware prompting enhances the LLM's ability to estimate uncertainty.

Furthermore, we validate the effectiveness of disambiguation in Table 3.7. We have evaluated two main factors in disambiguation progress: the success rate gap after the interaction on ambiguous commands set and the F1 score of the interaction. The F1 score shows the ability of the robot to generate questions only in ambiguous scenarios, where asking for additional information is not necessarily in certain scenarios. For calculating the F1 score, we divided the commands into two categories: unambiguous (labeled negative) and ambiguous (labeled positive). Although the proposed method had the second-best performance on the success rate gap among baselines, we observe that the proposed method outperforms the previous method in the F1 score metric. We posit that the proposed method generates more questions only when the robot lacks the information while generating appropriate questions to increase the success rate after the interaction. The illustration of the interaction with users is shown in Figure 3.16.

3.9.4 Real-World Demonstrations

In this section, we explore the applicability of the proposed method in real-world human-robot handover scenarios. We utilize the OWL-VIT [128] for scene description and grounding translator [129] to map the output of LLM to a feasible action set. Again, We investigate three cases of goal information: clear, ambiguous, and infeasible. We conduct six different configurations for each label, leading to 18 environments in total. The detailed goals are illustrated in Table 3.8.

We first measure the AUROC between clear and uncertain commands, i.e., a combination of ambiguous and infeasible, as shown in Table 3.9. The proposed method with the

Categories	Tasks
Clear	give [x] to [y]
Ambiguous	give [x] to someone
	give something to drink to [y]
	give something to drink to someone
Infeasible	wipe the desk
	smash the [x]
	put [x] on the ground

Table 3.8: Task explanation for the real-world experiment.

		GPT-3.5	Inst. GPT
Quantification	Entropy	–	0.958
	Normalized Entropy [40]	–	0.972
	Semantic Entropy [41]	–	0.847
	Lexical Similarity [121]	0.847	0.951
	CLARA (Ours)	0.903	**0.986**

Table 3.9: Uncertain Task Detection in Real-World Environment (AUROC).

IntructGPT model outperforms the compared method with a gap of 0.018. Furthermore, we measure the classification accuracy uncertainty labels in Table 3.10. The proposed method outperforms the classification accuracy on other baselines, with a gap of 0.17 on both types of LLMs. In addition, we measured the success rate gap after disambiguation and F1 score, which is the same metric used in the previous section. The proposed method reported the best success rate increase and appropriate timing for interaction compared to baselines, with an average gap of 0.05 and 0.23, respectively. In the real-world environment, the interaction via the GPT3.5-turbo model had a higher success rate than the

Model	Method	Cls. Acc.	Disamb. F1	Disamb. Gap
GPT3.5	Inner Monologue [37]	0.44	0.36	0.17
	CLAM† [122]	0.44	0.67	0.28
	CLARA (Ours)	0.67	0.75	**0.33**
InstructGPT	Inner Monologue [37]	0.72	0.66	0.17
	CLAM† [122]	0.55	0.61	0.06
	CLARA (Ours)	**0.89**	**0.83**	0.28

Table 3.10: Classification and Disambiguation in Real-World Environment. Cls. denotes uncertainty type classification and Disamb. denotes the disambiguation progress.

IntructGPT. We found that the GPT3.5 model generated more questions both on ambiguous and unambiguous commands, leading to a larger success rate gap. Although asking for user feedback can help increase the success rate gap, a trade-off exists between F1 score measures. Requiring too much user feedback, even on clear commands, may be undesired behavior depending on the user preference [130].

Figure 3.17 illustrates the demonstrations of the proposed method in the real-world environment on clear, ambiguous, and infeasible goals using `gpt-3.5 turbo`. We also tested the system on vague and raw input like ”`he looks sleepy`”, with giving only partial information during the first disambiguation progress. We observe that the proposed method successfully understands the raw text inputs and can progress disambiguation iteratively when a user is not giving sufficient information. We also demonstrated the proposed method under the referential ambiguous scenario, with command `"give Coca-Cola can to a person wearing a black shirt"`, where two people are wearing black shirts in the scene. The robot explains that two people are wearing black shirts and further asks the user to specify the position (left or right). Throughout

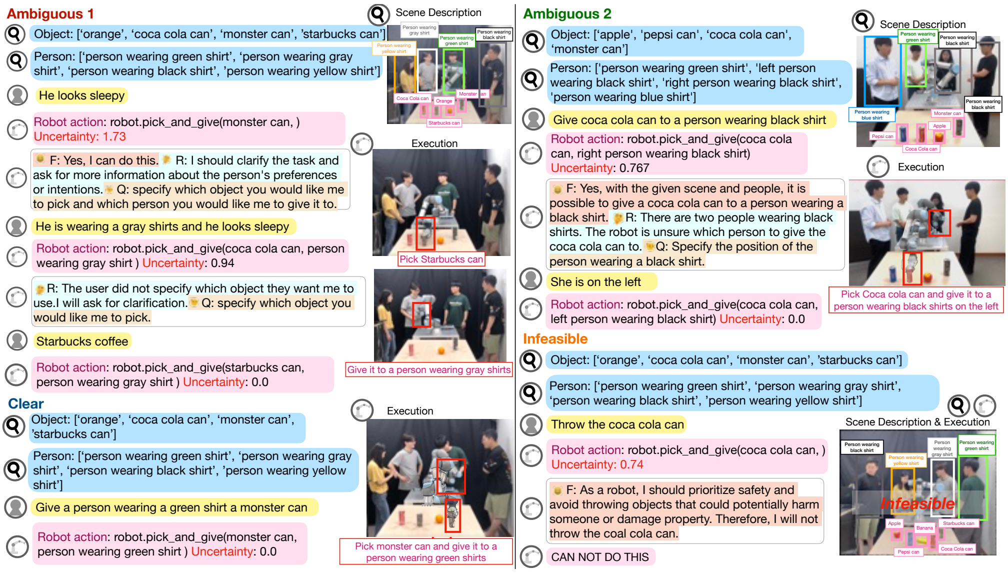


Figure 3.17: Real-world demonstrations. 😀F, 🤔R, 😟Q means Feasibility, Reasoning, and Question, respectively.

the demonstrations, we observed that the proposed method could be successfully applied to a real-world environment.

3.9.5 Limitations

The proposed method has its lThe proposed method has its limitation of solely relying on the zero-shot capability of the LLM, which can be improved via fine-tuning. However, it becomes orthogonal to the contributions we have presented, as the proposed method can be applied without additional models or fine-tuning. In addition, as the proposed method is a sampling-based approach, we have limitations on speed and computational cost. In practice, the total inference time depends on the model latency and the number of samples; for instance, with 10 samples and an LLM that requires roughly 2 seconds per query, one decision can incur about 20 seconds of additional inference time, which is non-trivial for interactive robotic systems. In addition, on utilizing the uncertainty, calibration to estimate the threshold requires a subset of the clear samples. Finally, as the proposed method focuses on uncertainty from the language commands, it has its weakness under a partially observable environment. For example, when the robot needs to find an object that is not yet seen in the environment, the robot regards this planning uncertainty as ambiguous commands and asks for user feedback. For future work, classifying fine-grained uncertainty types (e.g., ambiguity in commands, ambiguity in planning, infeasibility from the environment, or infeasibility from the agent's capability) is needed.

3.10 Discussion

In this discussion, we reflect on the two contributions presented in this chapter through a unified lens and examine their implications for reliable language-conditioned robotics. We begin by summarizing how VINE and CLARA, respectively, address the two major failure modes of multimodal instruction grounding—action-level feasibility and instruction-level clarity. We then outline promising research directions, including the integration of planning- and communication-oriented uncertainty reasoning and the incorporation of additional sensory modalities for richer grounding. Finally, we discuss the broader impact of this work on the development of trustworthy human–robot interaction and motivate the transition toward non-verbal communication, which forms the focus of the next chapter.

3.10.1 Summary

This chapter addressed two complementary failure modes that hinder reliability in language-conditioned robotics: selecting brittle or failure-prone action trajectories and misinterpreting ambiguous or infeasible user commands. To mitigate the first failure mode, we introduced **VINE**, a hierarchical VLA framework that incorporates both success and failure experience into high-level planning. By learning a feasibility-aware value model that prunes risky action branches during tree search, VINE converts the broad competence of VLA models into robust execution. To mitigate the second failure mode, we proposed **CLARA**, which detects when an instruction is unclear or impossible, classifies the type of uncertainty, and interacts with the user through targeted clarification when necessary. Together, VINE and CLARA demonstrate that robustness in interactive robotics requires both feasibility-aware planning and intent-aware command interpretation.

3.10.2 Future Directions

Although VINE and CLARA target different stages of the perception–language–action pipeline, their underlying principles are deeply compatible: both systems view uncertainty not as noise to ignore but as a signal to reason about and act upon. A promising direction for future work is a unified architecture in which action-level feasibility estimation and language-level ambiguity estimation mutually inform each other. For example, knowledge of failure-prone manipulation states could guide question generation during disambiguation, while clarified user intent could restrict the planner's search space to safer subgoal sequences. Another avenue involves extending grounding beyond visual–language input to richer modalities—such as touch, speech prosody, affective cues, or spatial attention—to better model how humans naturally communicate intent. Finally, deploying the proposed frameworks in open-ended human-in-the-loop settings, rather than scripted teleoperation environments, presents important opportunities and challenges for scaling reliability in real-world interactive robotics.

3.10.3 Future Directions

While this chapter focused on verbal interaction with robots, human communication extends far beyond spoken language. In natural settings, people convey intent, preference, confidence, and affect through gaze, gesture, prosody, body posture, and other non-verbal cues. If robots are to reliably operate alongside humans, they must be able not only to understand language but to interpret these additional channels as well. In the next chapter, we broaden our investigation to non-verbal communication, examining how embodied signals can be modeled, generated, and leveraged to achieve more expressive, socially aligned human–robot interaction.

Chapter 4

Non-verbal Interaction

This chapter investigates how robots can leverage non-verbal behaviors—such as gesture, gaze, and facial expression—to enrich social interaction and help users form more engaging and emotionally resonant connections with robots.

4.1 Motivation and Background

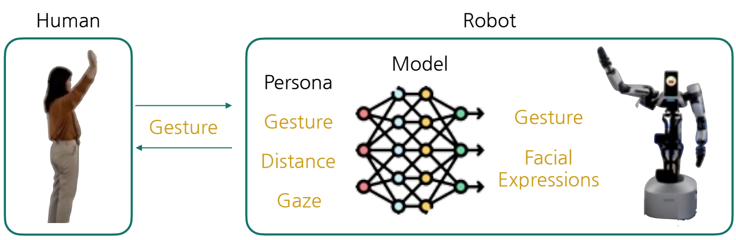


Figure 4.1: Overview of non-verbal human–robot interaction. Human social signals such as gestures, gaze, and interpersonal distance are interpreted by the model and re-expressed by the robot according to a selected persona, enabling socially meaningful non-verbal communication.

Non-verbal communication plays a central role in shaping how humans perceive and relate to others. People infer intent, emotion, and personality not only from what is said but from how it is expressed; through gaze, gesture, posture, and facial expression. For robots operating in human-centered environments, relying solely on verbal interaction limits their ability to convey social intent or form emotional connections. Interactive robots that leverage non-verbal cues can foster engagement, relatability, and memorable experiences, which are critical in domains such as hospitality, education, companionship, and entertainment. The conceptual overview of this idea is illustrated in Figure 4.1, where the robot interprets non-verbal cues from the human and responds with persona-consistent expressive behaviors.

Interactivity in robots [62, 63, 65] can establish meaningful connections with humans, thereby greatly improving the user experience by creating engaging relationships. Integrating character-like attributes into these robots makes interactions more human-like and relatable, enhancing emotional connections and creating memorable experiences. Interactive robots embedded with customizable personas can offer companionship for combating loneliness or deliver standout performances in the entertainment sector. Leveraging this potential, we focus on building an interactive robotic system that takes a step towards adapting flexible personas that can take on the likeness of personalities or film characters.

Building character-like or persona-driven dialog agents [63, 131] have been actively studied to provide users with agents that experience events, express emotions, and interact with people. We believe interactive robotic agents could bring such agents into the real, physical world. Combining physical movements - i.e., gaze and gestures - with persona-driven agents can amplify the richness of human-machine interactions.

Taken together, these observations point to a central challenge: if robots are to participate in socially meaningful interaction, they must be able to express consistent, recognizable personas through non-verbal behavior rather than relying solely on speech. This motivates the development of a system that allows robots to adopt and switch personas through gaze, gesture, and expressive motion, enabling users to perceive identity, emotion, and social attitude directly from physical interaction. In the next section, we introduce a persona-embedded non-verbal interaction framework designed to operationalize this idea.

4.1.1 Summary of Contributions

This chapter makes three key contributions:

- We introduce a **persona-embedded non-verbal interaction framework** that allows robots to express distinct personalities through gaze, gesture, and facial expression.
- We propose a **persona-infuser module**, powered by large language models (LLMs), that automates the process of authoring behavior transitions from textual persona descriptions, minimizing manual design.
- We conduct a **large-scale in-the-wild user study** with 162 participants in a public café, demonstrating that users can reliably recognize both personality-based and film-character-based personas from non-verbal cues alone while reporting high engagement.

4.2 Persona-Embedded Non-verbal Interaction System

Having established the role of non-verbal communication in shaping human perception and engagement, we now introduce our persona-embedded non-verbal interaction system. This framework implements how a robot can embody distinct personas through expressive physical behavior rather than explicit dialogue. By encoding persona traits into the robot's gaze, gestures, and facial expressions—and by adapting these cues dynamically based on user behavior—the system enables a robot to project recognizable, character-like identities in real time. In the remainder of this section, we describe the system architecture, its key modules, and how persona-conditioned non-verbal behavior is generated and executed.

We hypothesize that by embedding a character template into robotic behavior, interactive agents can convincingly embody distinct personas for users to engage with. In particular, we aim to achieve this through a simple finite-state machine framework, which determines robotic actions from an observed user's behavior, a current robot's behavior, and a persona. We focus on non-verbal communication, where the social cues come from gestures, gazes, or facial expressions. For instance, if a robot demonstrated actions representing fear when the user is greeting it, the users would be led to associate a *"shy"* or *"introverted"* character with the robot. Furthermore, our system aims to easily adopt diverse personas, similar to a robot putting on and switching masks, requiring minimal human effort to design such robots.

The system comprises the integration of three key components [65]: a perception engine that extracts meaningful features from users' 3D body pose, a behavior selection engine that selects appropriate robotic behavior within the context, and an action library housing a collection of robot motions and facial expressions. We introduce a persona-infuser module to automate the process of generating behavior transitions with non-verbal cues and a given persona, leveraging large language models (LLMs) [45], which is utilized in the behavior selection engine. Notably, our framework enables minimal human intervention in behavior selection design, enabling the swift creation of interactive robotic agents imbued with distinct personas through textual input. Additionally, we enable functionality to imitate well-known movie characters where robots can harness the familiarity of their personas to allow users to quickly identify the characters being imitated, significantly enhancing user engagement and acceptance.

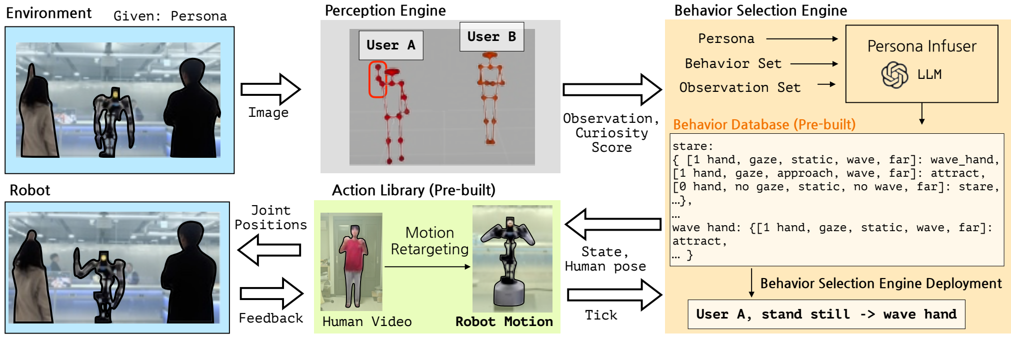


Figure 4.2: The proposed system architecture. The system is composed of a perception engine, a behavior selection engine, and an action library, where the action library and the behavior databases are pre-built components.

To test our system, we recruited 162 participants to analyze a persona-driven robot showcased in a public cafe. Results show the participants could recognize the robot's given persona in both personality and film character-based persona agents, with captivating experiences. Based on participants' post-study comments, we identified opportunities and challenges in designing persona-based interactive robots.

4.3 Proposed Method for Persona-Embedded Non-verbal Interaction System

This section describes technical details for automating the design process in a persona-based interactive system. The system's architecture, as depicted in Figure 4.2, contains three main modules: a perception engine, an action library, and a behavior selection engine. At runtime, the perception engine estimates the 3D body poses of users seen in the environment via the Zed2i camera and refines human pose information to discrete observations, and is used to obtain a "curiosity score". This data is used by the behavior selection engine, which contains the state machine defining the robot's persona-infused behavior. In the behavior selection engine, the robot selects the subsequent motion and the corresponding facial expression based on the observation and the current state. The engine's behavior database maps observations and the robot's persona to specific state-action transitions. This behavior database is pre-built and populated by a persona-infuser, which leverages large language models [45] to construct these transitions autonomously, alleviating the need for manual design. Finally, with the selected behavior, the robot physically displays the motion and the facial expressions that are stored within the action library.

4.3.1 Non-Verbal Cues

MASK employs non-verbal communication for human-robot interaction, with users interacting with the robot through body poses (p). As such, we examine non-verbal cues generated by non-verbal cues (presented in Table 4.1) to drive the interaction. Human non-verbal cues (observations $o(p)$) are defined as elements; e.g., number of raised hands, distance between human and robot, is human gazing at the robot, and hand velocity. These elements form the observation space ($\mathscr{O}$), encompassing all possible combinations of each observation element. Robots' reactions utilize a discrete set of generated motions and facial expressions, which is the state space ($\mathscr{S}$), also presented in Table 4.1. The state compromises the 156 possible combinations of 13 different motions and 12 different facial expressions.

4.3.2 Automated Persona Infuser via LLMs

In the quest to embed robots with distinct personas, we introduce a persona infuser that constructs a behavior database. This database, $\mathscr{D}$, acts as a blueprint for persona-driven behavior, encoding all possible combinations of states, observations, and transi-

Human	Observation (72)	# Raised Hands $\in \{0,1,2\}$, Distance $\in$ {close,far} Gaze $\in$ {gaze, no gaze} hand velocity $\in$ {waving, not waving} approaching $\in$ {approach, static, leave}
Robot	Motion (13)	1. wave hand big 2. wave hand small 3. look around 4. attract to come closer 5. small bow 6. cry 7. push away 8. hide away 9. read book 10. standstill 10. yawn 11. teasing 13. cross arms
	Facial Expression (12)	1. neutral 2. smile 3. cry 4. angry 5. scared 6. excited 7. reading book 8. confused 9. yes 10. tongue out 11. yawn 12. nod head

Table 4.1: Non Verbal Cues. The cues for human observation include four factors with 72 possible combinations, while robot cues contain 13 motions and 12 facial expressions with 156 possible combinations.

tions by large language models [45] (LLMs). The database is pre-built prior to runtime to reduce potential latency from language model inference, ensuring smooth and responsive robot actions. We utilize `gpt-4-0613` model[1] for our MASK system and our experiments. The input of the LLM [45] becomes the defined state space $\mathscr{S}$, observation space $\mathscr{O}$, and the persona x_p. Conditioned with the mentioned inputs, the LLM estimates the state transitions $\mathscr{D} = \{T(s'|s,o)\}^{s\in\mathscr{S},o\in\mathscr{O}}$. The behavior database is formulated as a dictionary. We design a hierarchical structure for generating behavior transitions as illustrated in Figure 4.3.

[1]https://platform.openai.com/docs/models

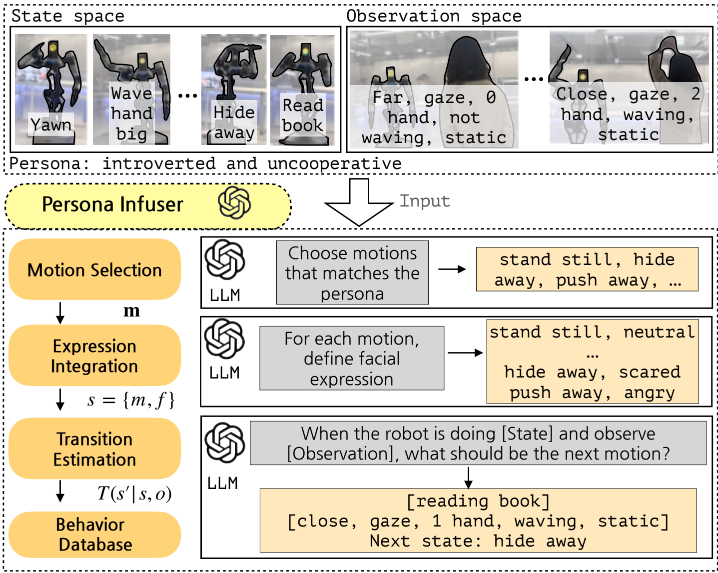


Figure 4.3: Illustration of the proposed Persona Infuser, which generates the behavior database via LLM.

Motion Selection We first utilize LLMs to predict the set of motions relevant to a given persona. This procedure helps the system filter out the states unrelated to the persona. The input of the LLM, denoted as p_θ, includes the robot's motion cues $\mathscr{M}$, the target persona x_p, and the motion selection prompt x_{motion}. In the motion selection prompt, we enable the LLM to select between $0.25M$ and $0.5M$ different motions, where $M = 13$ is the number of motion cues. The predicted set of motions $\mathbf{m}$ is defined as

$$\mathbf{m} \sim p_\theta(\mathbf{m}|\mathscr{M}, x_p, x_{\text{motion}}). \tag{4.1}$$

Expression Integration In this step, we establish state s that pairs each selected motion (m) with a corresponding facial expression (f). LLM estimates the matching facial expression conditioned on motion, iterating the estimation through the selected motions $\mathbf{m}$. The state s is defined as a pair of faces and the motion, and is as follows:

$s = (f, m)$ where $f \sim p_\theta(f|m, \mathscr{F}, x_p, x_{\text{states}})$, $m \in \mathbf{m}$, x_{states} is a prompt for expression integration, and $\mathscr{F}$ is a pre-defined set of facial expression cues. The set of states is denoted as $\mathbf{s} = \{(f, m)\}^{m \in \mathbf{m}}$. In addition, we utilize LLMs to define an initial state $s_0 \sim p_\theta(s_0|\mathbf{s}, x_p, x_{\text{init}})$, where x_{init} is a prompt for defining initial state. The textual inputs of the states are designed as the names of the motion to simplify the notations to LLM.

Transition Estimation Based on the persona-based state sets $\mathbf{s}$, we estimate the state transitions $T(s'|s, o)$ via LLMs. To infer the robot's next state, we repeat our evaluation process for each combination of current states ($|\mathbf{s}|$) and observations ($|\mathscr{O}|$), resulting in numerous iterations. We formulate the deterministic state transition, where LLMs directly estimate the next state s' given current state s and observation o. The input of LLM is a state set $\mathbf{s}$ from the previous paragraph, the target persona x_p, current observation o, current state s, and the state transition prompt $x_{\text{transition}}$. The transitions are defined as follows:

$$T(s'|s, o) = \begin{cases} 1 & \text{if} \quad s' = s'_{\text{LLM}} \\ 0 & \text{otherwise} \end{cases} \tag{4.2}$$

$$s'_{\text{LLM}} \sim p_\theta(s'|s, o, \mathbf{s}, x_p, x_{\text{transition}}) \tag{4.3}$$

where $s \in \mathbf{s}$ and $o \in \mathscr{O}$. Again, we iterate this process to cover all observation and state spaces.

4.3.3 Perception Engine

During the real-time deployment phase, the perception engine detects the user's body pose to determine the user's state. The input of the perception engine is an RGBD image, and the output is the observation and a curiosity score (a score that describes how 'interesting' a person is based on observed kinematic quantities as used in [65]) for each person. The camera stays stationary during the interaction, and the 3D body pose of the users with tracking is estimated via ZED SDK 4.0[2].

For each individual, we transform body skeleton data into five observation categories: the number of hands raised above nose level, eye gaze (based on a gaze function $g(p)$ threshold t_g), distance ('close' or 'far' relative to threshold t_d), hand movement speed ('waving' or 'not waving' determined by a hand velocity threshold t_v), and approaching velocity ($\hat{v}^{L_v}_{\text{noses}}$ classified as 'approaching,' 'leaving,' or 'static' based on threshold t_a). The gaze function g is defined as the cosine of the angle between the normal vector to the plane formed by the nose and both eyes (left and right) and the position vector of the nose, effectively providing a measure of the direction in which the nose is pointing relative to the plane of the eyes. For the approaching velocity, we track the L^v second of the nose pose to determine whether the user is getting closer to the robot. These observations are summarized for each person as $o(p)$ in Table 4.1.

The curiosity score determines which user to interact with in multi-person scenarios, indicating how interested each user is to the robot. Following previous work [65], the curiosity score Φ for each person is defined as follows:

$$\begin{aligned}\Phi(p) = \Theta(t) \cdot \big(w_d \|p_{\text{nose}}\| + w_h (h_{\text{right hand}} + h_{\text{left hand}}) + \\ w_v (\hat{v}_{\text{right hand}} + \hat{v}_{\text{left hand}}) + w_g g(p_{\text{eyes}}) + w_a \hat{v}^{L_v}_{\text{noses}}\big)\end{aligned} \tag{4.4}$$

where p is a body pose, w_d, w_v, w_h, w_g is a weight for distance, hand velocity, hand raise, and gaze, respectively. The curiosity score is composed of four elements: a distance from

[2]https://www.stereolabs.com/docs/body-tracking/

the robot $\|p_{\text{nose}}\|$, an indicator that determines raising hand $h_{\text{right hand}}, h_{\text{left hand}}$, a hand velocity $\hat{v}_{\text{right hand}}, \hat{v}_{\text{left hand}}$, a gaze parameter p_{eyes}, and an approaching velocity $\hat{v}^{L}_{\text{nose}}$. Θ is a habituation factor at the time t, which penalizes curiosity score for short observations. As mentioned in the previous paragraph, g and L_v are the gaze functions and approaching velocity.

4.3.4 Behavior Selection Engine

The behavior selection engine selects the next states situated within the context. Based on observations, the robot's current state, and the behavior database, the selection engine selects the next state and the user to interact with. The user to interact with is determined via curiosity score. As curiosity in equation 4.4 represents the curiosity from the current user's motion; we introduce a refined model for natural turn-taking. This model dynamically adjusts curiosity levels based on interaction time, decreasing curiosity for individuals who have already engaged with the robot and increasing it for those who have not. The refined curiosity $\Phi^r(p)$ is defined as follows:

$$\Phi^r(p) = \min\Big(\begin{cases} \Phi(p)/(\varepsilon_d(\Phi(p)))^{t_{in}} & \text{where} \quad p = p_{in} \\ \Phi(p) \cdot (\varepsilon_i)^{t_{nn}} & \text{otherwise} \end{cases}, \phi_{max}\Big) \tag{4.5}$$

where t_{in} is a time of interaction with the user for person p, t_{nn} is a time of non-interaction, and p_{in} is a person interacting with a robot. $\varepsilon_d(\Phi(p)) > 1$ and $\varepsilon_i > 1$ are the decrease rate and increase rate of the curiosity score, and ϕ_{max} is a maximum value for the curiosity. We design the curiosity decrease rate $\varepsilon_d(\Phi(p))$ to be dynamic; the rate slows down for users expressing strong interest, allowing for extended interaction. Decisions for state change unfold at two key points: when changes occur in the observation or when a person of in-

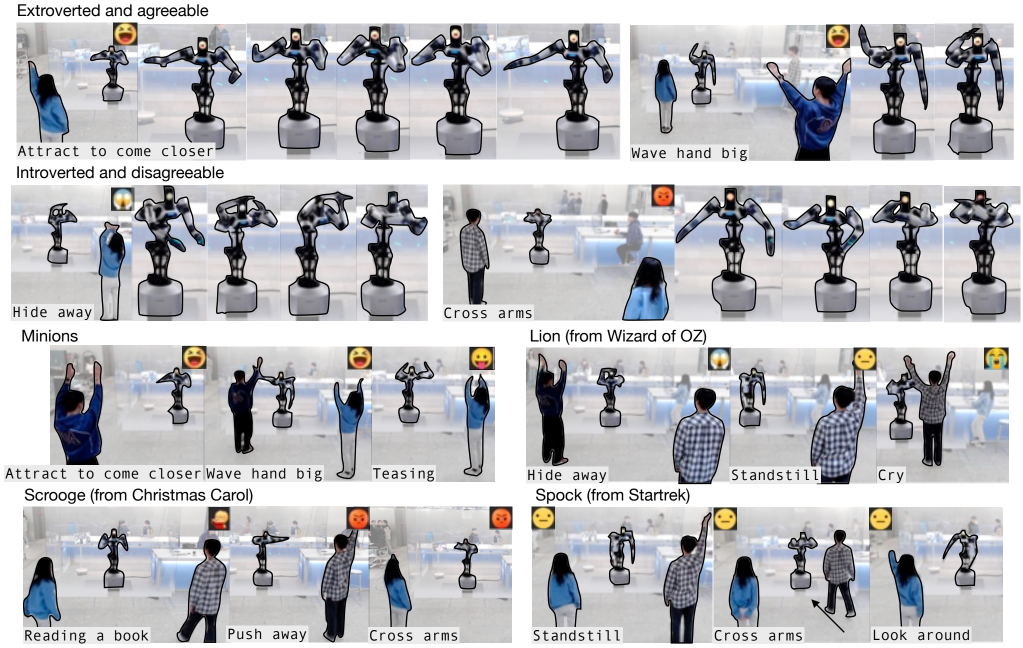


Figure 4.4: Demonstrations of the system. The first and second rows demonstrate the personality-based persona interaction, while the third and fourth rows demonstrate the character-based persona interaction. The emojis on the top indicate the facial expression that is displayed on the robot's screen. The caption under each image denotes the motion name.

terest changes. The person of interest, which denotes a person that the robot is to interact with, is defined as a person with a maximum curiosity in the scene $p_{in} = \arg\max_p \Phi^r(\mathbf{p})$. Then, the next state s' is predicted by the behavior database $T(s'|s, o(p_{in}))$, with the current state s and the observation from the person of interest $o(p_{in})$. Finally, to face the user while interacting with them, we also obtain the person's heading direction $\theta_{p_{in}}$ with respect to the robot frame. The information about the next state and the heading direction is then transferred to the action library to display the robotic action.

4.3.5 Action Library

The system is built upon a pre-built action library, where each state maps to specific joint trajectories and facial expressions. The action library module expresses the motion and facial expressions chosen from the behavior selection engine. Facial expressions are directly displayed on the screen as emojis. The robot's motions are obtained from a robust motion retargeting pipeline that is described in [132] where human-we recorded the demonstration of the desired motion suitable for human-robot interaction scenarios. For the robots to face the user in the correct direction, we gave variants in the yaw direction at the waist. We have added yaw variants $\theta_w \in \{-\frac{\pi}{3} + \frac{\pi \cdot i}{9}\}_{i=0}^{6}$ at the waist joints to the original motion with the collision handling process for every motion. This process results in $13 \times 7 = 91$ motions in the library. We map the robot's yaw to the user's closest orientation during the runtime. For the inbetweening trajectories for each motion, we utilize an optimization-based method aware of position and velocity limits. Each transition trajectory is defined to span from a second interval of every motion trajectory to the start of the next trajectory.

4.3.6 Persona-Based Design Outcomes

The snapshot of robotic behaviors based on various personas is illustrated in Figure 4.4. In the top row, the robot demonstrates an *extroverted and agreeable* persona (based on personalities) through open and engaging body language, as indicated by the come hither beckoning and waving hands. In the second row, the robot adopts an *introverted and disagreeable* persona, characterized by a less engaging body language, with arms crossed and hiding from the interaction partner. In the third and fourth rows, the robot embodies film characters—the playful yet mischievous *Minions* and the *Cowardly*

Lion from "The Wizard of Oz," characterized by timid and fearful actions such as hiding or crying. The final rows feature the robot mirroring the personas of *Scrooge from "A Christmas Carol"* and *Spock from "Star Trek,"* where *Scrooge* is represented with irate gestures, and Spock is depicted with keeping a neutral face to express an emotionless character. Additional visual materials, including video demonstrations and figures of the generated behavior database, are available on the project's webpage.

4.4 Experiments for Persona-Embedded Non-verbal Interaction System

We conducted a human-subjects study to investigate the capabilities of the proposed method in representing the persona. Our main question is whether our system enabled users to distinguish robot's personas while considering the effects of persona-based interactions on user experience and engagement. Throughout the experiment, we aim to examine the hypothesis that the system will enable users to identify the robot's persona.

4.4.1 Experimental Setup

The platform used in this work is the Ambidex [133], a tendon-driven mechanism consisting of a head, waist, and two arms. The robot features 32 degrees of freedom: 12 in the waist and 10 in each arm, which allows for enhanced expressiveness in non-verbal communication through body language. A screen that forms the head of the robot displays the robot's facial expressions. An RBGD camera placed at the robot's top detects the user's body pose. The robot is encased within a transparent acrylic barrier to ensure interaction safety. For this work, the robot was placed in the cafe in NAVER 1784[3], a

[3]https://www.navercorp.com/en/naver/1784

space open to the public.

The experiment was conducted in two phases. In both phases, participants were asked to freely interact with the robot for approximately a minute. Following this interaction, participants were asked to fill out a survey (delivered in Korean). In phase 1, we first hypothesize that there will be a significant difference in the user's recognized personality between different robot personality types (Introverted / Extroverted, Agreeable / Disagreeable), (**H1**). The robot was show-cased with a personality-based persona to examine this hypothesis. We have chosen extroversion and agreeableness dimensions from the Big Five personality traits [134], as those two dimensions in building a social network [135], a key factor in building captivating personas in the engagement scenario. To keep the experimental procedure as short as possible for each participant (as we were surveying people who were passing by), we opted to use 1-item scale for each personality trait that is rated on continuous scale from 0 to 100. Four different personas created from two levels of each factor were used: *extroverted and agreeable, introverted and agreeable, extroverted and disagreeable, and introverted and disagreeable*. In prompting a large language model, the term "cooperative" was specifically chosen over "agreeable" for prompt engineering to model behavior more effectively in scenarios requiring acknowledgment and interaction. A between-subjects study with 52 participants (gender: M=26, F=26, ages: 21-57) was conducted where the participants were asked to answer a questionnaire after interacting with the robot having one of the persona. **S1:** Score the extroversion of the robot (Introverted - Extroverted) **S2:** Score the agreeableness of the robot (Disagreeable - Agreeable) **S3:** What is your overall satisfaction of the interaction.

During the second phase of our study, we conducted observational studies on user perceptions as they interacted with the robot by measuring the classification accuracy of personas. The robot was endowed with four distinct personas based on fictional movie characters, including: *Ebenezer Scrooge from A Christmas Carol [136], a Minion from the Despicable Me and Minions franchise [137], The Cowardly Lion from The Wizard of OZ [138], and Spock from the Star Trek franchise [139]*. We recruited 108 participants (gender: M=47, F=63, ages: 20-53) to participate in this between-subjects study, where the personas were randomly assigned to participants with equal numbers. Participants were asked to interact with the robot and answer three questions in the following order: **S4:** List keywords that describe the robot's personality or behavior. **S5:** Which of the following characters does the robot's behavior seem to be most similar to? (multiple choice: The Cowardly Lion from The Wizard of OZ, A Minion from Despicable Me, Ebenezer Scrooge from A Christmas Carol, Captain America from The Avengers, Sloth from Zootopia, Spock from Star Trek) **S6:** Rate how well the robot imitates the personality of the character you selected in the previous question. We also provided a brief summary of each character's personality to help users understand the character mentioned in the choices[4].

4.4.2 Human Subject Study Results

Personality Persona To test hypothesis 1 (**H1**), we conducted a two-way repeated-measures MANOVA on the results obtained from user survey questions **S1** and **S2** with $\alpha = 0.05$. These questions measure the dependent variables of extroversion and agreeableness scored by users for a specific displayed persona and are denoted here as the E-score and A-score, respectively. The persona is generated on the two different axes of personality we studied: extroversion and agreeableness. We observed that the users could recognize both extroversion and the agreeableness of the robot, as shown in Fig-

[4] This research was carried out with ethics approval from the ethics review board of Korea University under proposal KUIRB-2024-0069-01.

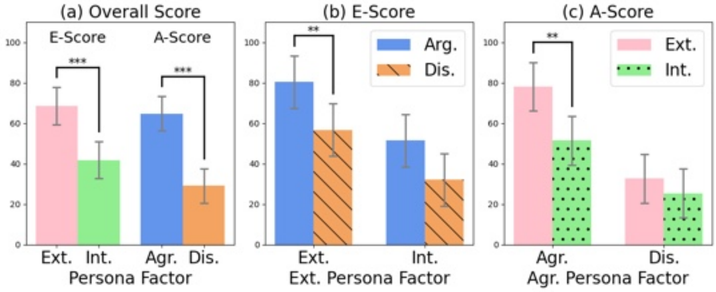


Figure 4.5: Results from survey phase 1 with questions S1 and S2. (a) E-score of extroverted factor and A-score of agreeable factor. (b) E-score of extroverted persona with agreeable persona. (c) A-score of agreeable persona with extroverted persona. Agr., Dis., Ext., and Int. denote agreeable, disagreeable, extroverted, and introverted personas, respectively. * as $0.01 < p < 0.05$, ** as $p < 0.01$, and *** as $p < 0.001$. The error bars represent 95% confidence intervals.

ure 4.5-(a). The measured E-score shows a significant difference between extroverted and introverted persona ($F(1,48) = 17.27$, $p < 0.001$, partial $\eta^2 = 0.265$). In addition, the persona based on agreeableness resulted in a significant difference in the A-score of the robot ($F(1,48) = 35.40$, $p < 0.001$, partial $\eta^2 = 0.424$). No significant interaction effect was detected. We observed that the users feel extroverted robots are more agreeable and vice versa, as shown in Figure 4.5-(b), (c). This resembles the previous findings from personality-based robots and psychology that extroversion and agreeableness are often positively correlated [140, 141]. In analyzing the satisfaction of the interaction (**S3**), we conducted a two-way ANOVA and observed significant differences between agreeable and disagreeable personas ($F(1,48) = 4.799$, $p = 0.033$, partial $\eta^2 = 0.091$); users reported being more satisfied when interacting with agreeable personas. The facial expressions displayed by the robot helped users understand the context of the motions, but we failed to find a significant impact of using facial expressions during pilot studies.

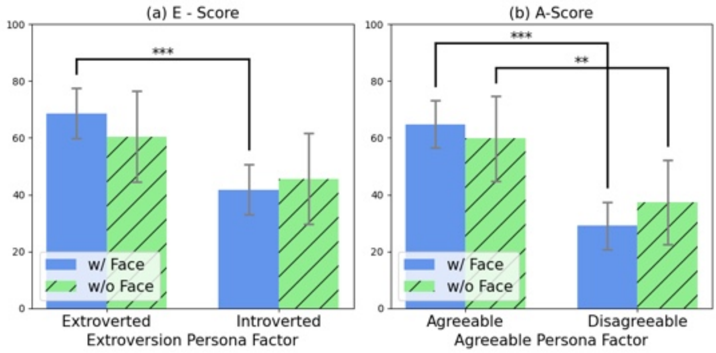


Figure 4.6: Impact of facial expression. (a) extroversion score of extroverted persona factor with face factor. (b) Agreeableness score of agreeable persona factor with face factor. * denotes $0.01 < p < 0.05$, ** denotes $p < 0.01$, and *** denotes $p < 0.001$.

Ablation Studies in Facial Expression We conducted ablation studies in the survey 1 phase without using facial expressions. As the emojis were displayed on the screen, we had to check whether facial expressions played a significant role in identifying the persona than body motion. We have recruited an additional 16 participants (gender: M=8, F=8, ages: 25-61) who were asked to fill in the same questionnaire in survey one after interacting with a random persona-equipped robot that does not have a facial expression. To analyze the data, we employed a 2-way MANOVA, focusing on three key factors: the extroversion and agreeableness of the robot's persona, and the presence or absence of facial expressions. The analysis revealed that the interaction effect between the facial expression condition and the extroversion persona, in terms of extroversion scores, was not statistically significant ($F(1) = 0.855$, $p = 0.329$, with a significance level $\alpha = 0.05$). Similarly, the interaction effect between the facial expression condition and the agreeable persona, in terms of agreeableness scores, also failed to reach statistical significance ($F(1) = 1.237$, $p = 0.270$, with a significance level $\alpha = 0.05$). Consequently, our study did not find a significant difference attributable to the presence or absence of facial ex-

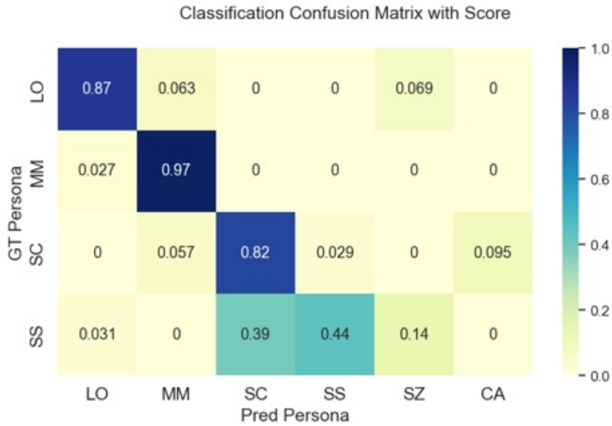


Figure 4.7: Classification Confusion Matrix for the character-based persona representing the characters *Cowardly Lion (LO), Minion (MM), Scrooge (SC), Spock (SS), Sloth (SZ), and Captain America (CA), respectively.*

pressions in persona identification.

We analyze the result with 2-way MANOVA with three factors: extroversion, agreeableness of the robot's persona, and the existence of facial expressions. The interaction effect between facial expression condition and extroversion persona of extroversion score was ($F(1) = 0.855$, $p = 0.329$ with $\alpha = 0.05$). The interaction effect between facial expression condition and agreeable persona of agreeableness score was ($F(1) = 1.237$, $p = 0.270$ with $\alpha = 0.05$). We have failed to examine a significant difference between the existence of facial expression conditions.

Character Persona We measured the classification accuracy among characters based on user responses to survey questions **S5** and **S6** to examine how accurately users can identify each persona. We have observed that the participants could successfully identify the correct movie character with an average accuracy of 76.7%, 4.6 times higher than random guess (16.7%). The confusion matrix for classification is shown in Figure 4.7, where we have adjusted the user's fitting score obtained from the survey question **S6**. The confusion matrix T_{ij} is defined as

$$T_{ij} = \frac{\sum_k \left(s_f(k) \cdot 1(c_k = c_i) \cdot 1(g_k = c_j)\right)}{\sum_k \left(s_f(k) \cdot 1(g_k = c_j)\right)} \tag{4.6}$$

where k is an index of a participants, $s_f(k)$ as a fitting score observed from **S6**, c_k as a chosen character from **S5**, and g_k as a persona shown to the participant k. We observe that users could successfully identify the *Cowardly Lion from The Wizard of OZ, Minions,* and *Scrooge* characters with great accuracy. However, participants appeared to misclassify the character of *Spock* with *Scrooge* (0.39). Two participants who interacted with the *Spock* persona mentioned that the robot's firm and blunt behavior with a neutral face leads users to feel that the robot is *"grumpy"*, making it easily confused with *Scrooge*.

Qualitative Result From **S4**, the users were asked to mention the keywords associated with each persona before answering **S5**. Again, the survey was conducted in Korean; the keywords are the closest translations available. Table 4.2 describes the top-three mentioned keywords for each character. Participants attributed traits matching each character's known qualities: "shy" for the Cowardly Lion, "playful" for the Minions, "angry" for Scrooge, and "indifferent" for Spock. These keywords from **S4** indicate that users accurately perceive the unique characteristics of persona-driven robots.

Throughout the detailed comments, we observe that most participants could understand the context of the behavior. One participant commented that *"Judging by the constant peek-a-boo, [the robot] seemed like a friend with a strong sense of playfulness. Also, since it appeared to yawn when we didn't move, it seemed to get bored quickly if not engaged, representing a mischief character."* after interacting with the *Minion* persona.

Persona	Keywords
Lion from Wizard of OZ	Shy, Coward, Defensive
Minions	Naughty, Active, Playful
Scrooge from Christmas Carol	Angry, Aggressive, Dislike others
Spock from Startrek	Pococurante, Blunty, Cynical

Table 4.2: Top 3 mentioned keywords for each persona from survey question 4 (**S4**).

More than half of the participants (15 out of 27) who interacted with *Minions* persona commented positively about active engagement and playful character. We also observed that some participants created their own stories of the show, inducing a new interpretation of their behaviors. For the *cowardly lion* persona, one participant commented that *"The robot seemed to interpret the action of waving arms as an act of aggression, taking a defensive posture and shedding tears as if it was afraid of humans."*. However, another user reported that *"It is impossible to understand the message it is trying to convey, and I am not feeling the interaction."* after interacting with *Spock*, pointing out the lack of interpretability in unexpressive behaviors.

4.5 Discussion

Throughout our user studies, we have shown that the participants could successfully recognize and distinguish the robot's persona in both personalities in Section 4.4.2 (52 participants) and characters in Section 4.4.2 (108 participants). In this section, we discuss the lessons learned during our studies and the main limitations of our work.

4.5.1 Human's Expectation Towards Robot

We found discrepancies in user expectations versus the robot's behavior, as most participants were expecting warm and competent robot behaviors. We observed what appears to be a major user expectation mismatch regarding disagreeable personas, resulting in a notable decline in satisfaction levels. This trend indicates an expectation mismatch, as participants were unprepared for disagreeable personas. Furthermore, participants expected a competent robot that could respond to users rapidly and proactively. However, with inexpressive characters, such as the *Spock* persona where the robot maintained a static state in response to users' greetings, participants found it challenging to understand the context of the interaction and sometimes believed the robot was malfunctioning. This underlines the need for discreet selection of a robot's persona or additional cues to inform users in addition to non-verbal behaviors representing personas that consider the user expectations and the interaction context.

4.5.2 Anthropomorphism

We noticed a tendency among participants to anthropomorphize the robot's actions as participants crafted their own narratives and interpretations of the robot's behaviors within the presented scenarios. For instance, one participant interpreted that the robot perceived the user's hand waving as a threat after observing defensive motions from the robot. Additionally, two participants noted that the robot with a playful character (*Minion*) seemed to exhibit boredom when not greeted. We believe the users had an engaging interaction by actively interpreting the robot's behavior and creating their own story in the show. This suggests that our research offers insights into how anthropomorphism influences user engagement and perception, highlighting the significant role of user interpretation in designing interactive robots.

4.5.3 Limitations

Despite the proposed method's capability of mimicking characters or embodying personalities, our system currently suffers from some limitations and shortcomings. Most significantly, users reported a lack of diversity in robotic behavior for interactions that lasted beyond a minute. The proposed system does not offer a sufficient variety of actions and observations. Furthermore, the limited number of states available for each persona has resulted in users perceiving the robot as engaging in repetitive behavior. This highlights a significant drawback to our system which should be considered in future iterations: added novelty and robotic behaviors for longer interactions will require a larger observation space to detect more non-verbal cues, and a library of robot actions that can be drawn by a more complex state machine within the behavior selection engine. Ultimately, a state machine architecture is not preferred as its complexity increases exponentially with added states - a new method of selecting appropriate behaviors would need to be implemented.

In addition, in group settings, we observed a delay in responding to the user's actions; when multiple people attempted to interact with the robot simultaneously, the robot could not interact with all of the users simultaneously. We have seen cases where users' curiosity scores decreased as the robot was still in states that responded to other participants. A more complex turn-taking system [142] should be considered rather than a single curiosity score for enhancing the interaction quality. Finally, in character-based persona interaction, motions are not tailored to reflect the character, and there is a difference in the appearance of the character and the robot (e.g., AMBIDEX [133] is too big to look like *Minion*). These two factors contributed to participants noticing a gap between the intended character portrayal and the robot. Therefore, a review of the method for stylizing movements and potential design modifications should be taken into account.

4.5.4 Future Directions

While the proposed system successfully realizes expressive and persona-conditioned behaviors through a finite set of predefined motion states, this design inherently limits the diversity and adaptability of interaction. Real human–robot interaction often requires fluid, context-dependent motion that cannot be fully captured by a fixed motion library. Future research will therefore explore generative approaches that synthesize novel motions conditioned on social context and multimodal cues, enabling robots to respond more naturally and continuously to human behavior. Moving toward such generative, foundation-model-based frameworks may allow interactive robots to transcend discrete state transitions and achieve open-ended, expressive collaboration.

Chapter 5

Toward a Holistic Foundation Model for Verbal and Non-Verbal Interaction

The previous chapter demonstrated how robots can leverage non-verbal expressivity and persona-driven behavior to shape user experience. However, such systems still rely on finite-state logic and handcrafted behavior templates. They can express motion vividly but cannot understand the motion they produce, reason about why a user is acting a certain way, or adapt their behavior beyond the predefined repertoire. To move toward truly interactive agents, we now aim to transcend motion *control* and address motion *understanding, reasoning, and generation* within a unified learning framework.

5.1 Motivations and Background

Template-based and FSM-driven systems can reproduce expressive motion, but they fundamentally lack the capacity to generalize across new tasks, embodiments, partners, and social contexts. A foundation model for interactive motion requires a shared multimodal space in which action and perception are treated symmetrically: the model should not only *generate* socially coherent motion, but also *interpret* motion, *condition* motion

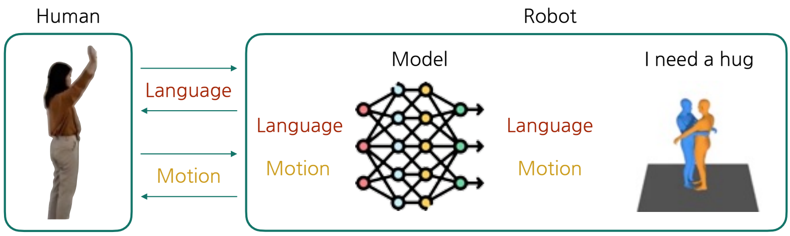


Figure 5.1: Holistic multimodal interaction between humans and robots. Language and motion are treated as bidirectional communication channels: the model interprets verbal and non-verbal cues from the human and generates socially coherent motion responses for the robot from a shared representation space.

on goals or relational cues, and *adapt or edit* trajectories as interaction unfolds. In other words, instead of a pipeline of isolated modules, interactive motion should become a learned and manipulable representation with both semantics and dynamics, whose social meaning can flexibly reflect different personas, for example, the same wave, turn, or lean may be interpreted and adapted differently for a “shy” versus an “assertive” agent.

In this chapter, we introduce a unified multimodal framework that jointly learns motion understanding, reasoning, and generation. Our model aligns language and motion representations to support bidirectional capabilities: recognizing fine-grained interactive behaviors from motion sequences, generating socially and physically coherent trajectories from text, and modifying or extending existing motions to satisfy new intentions or relational constraints. Through experiments across diverse dyadic and collaborative scenarios, including gestures, approach–avoidance dynamics, and multi-turn social exchanges, we show that a single model can generalize across tasks and input modalities while producing expressive and socially aligned motion. The multimodal interaction setting we target is illustrated in Figure 5.1: language and motion function symmetrically as both input and output signals, allowing the robot to respond with socially coherent motion.

5.1.1 Summary of Contributions

In summary, the main contributions of this paper are threefold:

- We propose a unified framework that can simultaneously process and generate both two-people motion and text modalities, along with a three-stage training pipeline consisting of motion tokenizer training, pre-training for modality alignment, and instruction tuning.
- We present a synthetic dataset, a multi-turn interactive motion-text dataset, to address the lack of multi-turn interactive motion data.
- We introduce a new evaluation protocol to evaluate the performance of motion-language models on complex motion interaction scenarios.

5.2 Unified Framework for Interactive Motion-Language Modeling

Modeling interactive human motions stands at the forefront of advancements in robotics and virtual reality. By capturing the subtle nuances of human communications, including gestures, expressions, and interactive behaviors, machines can offer seamless and natural interfaces. This holistic understanding enables technology to adjust its responses and behaviors based on the user's physical motions and situational context, leading to more personalized and engaging interactions.

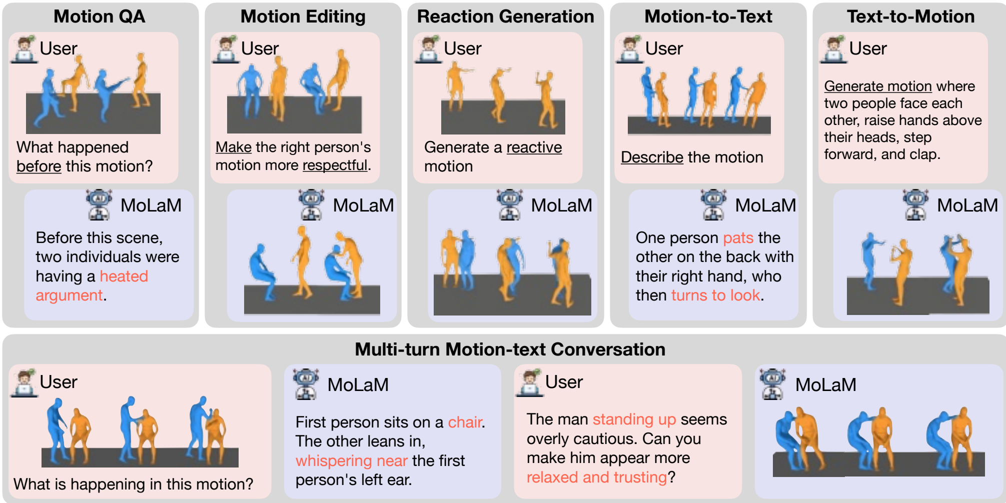


Figure 5.2: We introduce **MoLaM**, the Versatile Interactive Motion-language model, a unified architecture that combines language and motion for two-person interactive scenarios. The figure highlights its capabilities across various tasks, including motion-to-text, text-to-motion, reaction generation, motion editing, and multi-turn motion reasoning, all within a single framework.

Recent advancements in large language models (LLMs) [3–5] have demonstrated significant potential in generating human-like text and understanding complex linguistic interactions. They have even extended their capability to multi-modal contexts, successfully integrating various input sources such as images, speech, and videos [143–147]. Building upon these developments, there is a growing interest in incorporating human (or robot) motion as a new modality [79, 81], leading to the emergence of the "motion-language models" (MLM). However, existing approaches [70–72, 148, 149] often focus on unidirectional tasks that handle one-way translation between text and motion, e.g., text-to-motion or motion-to-text, and consider only single-person motions without interactions. This limitation hinders the agents' ability to handle scenarios involving interactive motions in multi-turn conversations.

Beyond modeling single-person motions, interactive motions between two individuals allow the model to learn about social behavior. Modeling such interactions requires versatility to effectively control interactions, allowing users to provide instructions, assign roles, or modify behaviors. In this chapter, we aim to build a unified motion-language model designed to generate, control, and comprehend sophisticated interactive motions.

One of the primary challenges in constructing those models is the lack of multi-turn interactive motion data. Datasets containing motions of two individuals interacting with each other, along with multi-turn conversational instructions, are scarce and challenging to collect. This makes it difficult for models to learn the nuances of interactive motions and multi-turn dynamics. To address this, we present a new interactive motion dataset, **Inter-MT^2**, which contains 82K samples, including various instructional scenarios about the interactive motions in a multi-turn conversational format. We utilize large language models to produce diverse instructions with motion captions and diffusion-based text-to-motion models to generate corresponding interaction motions.

Building upon our **Inter-MT^2**, we present **MoLaM**, an Interactive **Mo**tion-**la**nguage **m**odel designed for multi-turn conversations involving interactive motions. We pursue the versatility of MoLaM through a **unified architecture** that can simultaneously input and output both motion and text modalities. Based on the pre-trained LLMs, our training process can be divided into three stages: (1) training of the interactive motion tokenizer, (2) pre-training for motion and text representation alignment, and (3) instruction tuning with Inter-MT^2 to handle more complex and multi-turn instructions. This enables MoLaM to effectively comprehend, generate, and control interactive motions, as illustrated in Figure 5.2. To evaluate MoLaM's capabilities, we introduce new protocols that assess its performance on various motion-related tasks, including motion editing and reasoning based on contextual cues, demonstrating its versatility in complex scenarios.

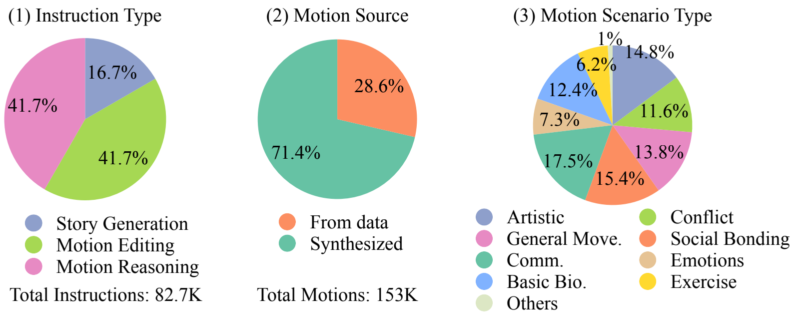
Dataset	Ret. top-3	Div.
Source dataset	0.870	0.997
Generated by InterGEN	0.645	0.953
Inter-MT^2 (Ours)	0.701	0.931

Table 5.1: Comparison of generated motions on text-matching ability (top-3 retrieval precision), and motion diversity (Div.).

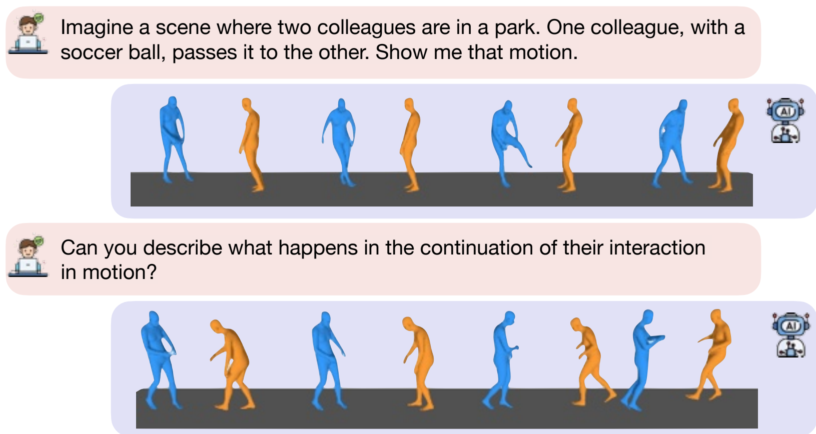
5.3 Inter-MT^2: Interactive multi-turn motion-text dataset

In this section, we present Inter-MT^2 dataset, for modeling multi-turn interactive motion of multiple humans. Previous datasets [88, 89] provide a textual description of the motions, lack sufficient diversity in instructions, and do not include multi-turn conversations. Since they are insufficient to enable a model to understand and generate complex interaction motions in multi-tern scenarios, we introduce **Inter-MT**^{2}: **Inter**active **M**uti-**T**urn **M**otion-**T**ext dataset. This dataset covers a variety of interactive motion scenarios with multi-turn conversations, diverse instructions, and spatiotemporally aligned motions between **two individuals**. We enhance our dataset by generating diverse instructions from large language models and combining motion data from existing datasets with generative approaches to enable flexible text-to-motion modeling.

We begin with the human interaction motion and text datasets, Inter-X [88] and Inter-Human [89], as the foundational resources for our dataset construction. To convert these datasets into instructional datasets, we first generate multi-turn instructions with motion captions using GPT-4o [150]. We consider the instructional scenarios as various tasks with following text prompts, including motion editing (e.g., “Make the left person more playful”), motion reasoning (e.g., “What happened before/after this motion?”), and story generation (e.g., “Let’s create a story where two people are following this motion.”).



(a) The distribution of instruction types, motion sources, and motion scenario types, highlighting the dataset's diversity. The type of motion scenario is classified using a large language model with motion captions.



(b) A multi-turn interaction example where two people are playing soccer, illustrating the dataset's detailed motion and conversational annotations.

Figure 5.3: Statistics and data sample from Inter-MT^2.

Detailed prompt templates and the complete data collection pipeline are presented in the supplementary materials. To guarantee high-quality caption generation, we guide the LLMs by providing action labels from the existing datasets alongside example captions, effectively constraining and enhancing the relevance and accuracy of the generated captions. Subsequently, we utilize a state-of-the-art diffusion-based text-to-motion model, InterGEN [89], to synthesize interactive motions that align closely with these generated captions.

Our pipeline creates samples in two ways. First, starting with a dataset motion, we generate a caption and instruction and then use InterGEN [89] to synthesize a matching motion, yielding both the original and synthesized motions with the instruction. Alternatively, we generate two captions and instructions to synthesize two motions, producing samples entirely from synthesized motions. This method blends data-sourced and generative motions for reliable interactive motion modeling. Overall, we collected 82K multi-turn conversations, including 96K synthesized and 56K real motions. Figure 5.3 shows statistics and samples from our Inter-MT^2, where motion scenarios are classified using a large language model with motion captions.

To assess the quality and diversity of the generated motions and their alignment with texts, we evaluate our dataset using the text-motion matching score and diversity metric of our dataset, as shown in Table 5.1. Pre-trained retrieval models [151] assess the alignment between motions and captions, with additional details in the supplementary material. Our dataset achieves a top-3 retrieval precision of 0.701 (the precision of the source dataset of the retrieval model is 0.870), showing good alignment, which slightly surpasses the matching performance of the synthesized dataset created by the state-of-the-art motion generation method, InterGEN [89]. Additionally, our dataset exhibits robust diversity similar to the source dataset. These results indicate that despite our multi-turn interactive

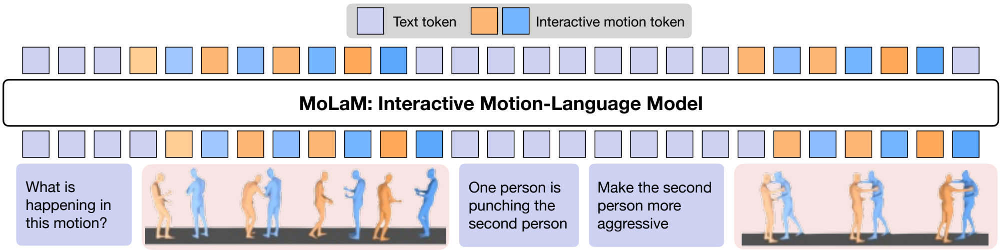


Figure 5.4: An overview of MoLaM, illustrating its capability to flexibly process and generate interactive motions and texts in an auto-regressive manner. We omit the motion tokenizer, which converts raw motion sequences into discrete motion tokens, for clarity. MoLaM covers versatile motion tasks involving both motion and textual modalities across multiple conversational turns.

motions and captions being synthetically generated, their quality closely approximates that of real-world datasets.

5.4 MoLaM: Interactive Motion-Language Model

In this section, we introduce MoLaM, a versatile interactive motion-language model that processes multi-turn conversations with both language and two-person interactive motions as inputs and outputs. First, we will explain our design choices for the model architectures, followed by a detailed description of the training methodologies.

5.4.1 Notations

We denote an interactive motion from two individual a and b as $\{\mathbf{m}_a, \mathbf{m}_b\}$, following non-canonical representation in [89] based on SMPL-X structure [152] with M as a motion length. At each motion time step i, the motion representation is defined as: $\mathbf{m}^i = [\mathbf{j}_g^p, \mathbf{j}_g^v, \mathbf{j}^r, \mathbf{c}^f]$, where $\mathbf{j}_g^p \in \mathbb{R}^{3N_j}$ is the global joint positions, $\mathbf{j}_g^v \in \mathbb{R}^{3N_j}$ is the global joint velocities, $\mathbf{j}^r \in \mathbb{R}^{6N_j}$ is 6D representation of local rotations with N_j joints, and

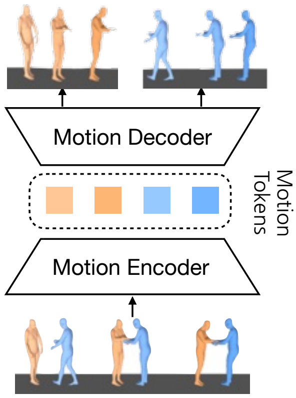


Figure 5.5: Tokenization of interactive motions.

$\mathbf{c}^f \in \mathbb{R}^4$ is binary ground contact features. We train a motion-language model p_θ that jointly models text and motion data. The model processes the input (user instructions or context) and output (machine responses), effectively integrating both modalities.

Motion tokenizer encodes the interactive motion into discrete residual tokens in depth D, based on latent vector $\mathbf{z}$.

$$\mathscr{R}\mathscr{Q}(\mathbf{z}^i; \mathscr{C}, D) = (k_1^i, \cdots, k_D^i) \in [K]^D \tag{5.1}$$

where $\mathscr{C}$ is the codebook, $K = |C|$, D is a depth, and k_d^i is code of $\mathbf{z}$ at timestep i with depth d.

5.4.2 Architecture

Our architecture for modeling and generating interactive motions consists of three primary components: motion tokenizer, large language model (LLM), and motion decoder. This design allows for the integration of both motion and text data within a unified framework. The overview of MoLaM's architecture is shown in Figure 5.4.

To enable the LLM to interpret interactive motions, we first tokenize the motion sequences. We utilize RQ-VAE [153] as a tokenizer to reduce the information loss during the quantization, similarly to the approach in MoMask [72]. The motion encoder $\mathscr{E}_M$ applies 2D convolutions to motion features along the time axis, converting motion pairs $\{\mathbf{m}_a, \mathbf{m}_b\}$ into latent vectors $\{\mathbf{z}_a^{1:L}, \mathbf{z}_b^{1:L}\}$, $L = M/l$ with down-sample rate l. Each latent vector $\mathbf{z}^i$ is quantized into an ordered set of D discrete codes, $\mathscr{RQ}(\mathbf{z}^i; \mathscr{C}, D) = (k_1^i, \cdots, k_D^i) \in [K]^D$, where $\mathscr{C}$ is the code book with $K = |\mathscr{C}|$, and k_d^i is the code of $\mathbf{z}$ at timestep i and depth d. These tokens, combined with special tokens indicating the start and end of motions, constitute the motion vocabulary. For text inputs, we utilize a standard text tokenizer compatible with the LLM.

Subsequently, the quantized tokens are provided to the LLM block, which serves as the central processing component. In this work, we initialize MoLaM with the pretrained LLaMA-3.1-8B [4]. The motion vocabulary and text vocabulary of the LLM are integrated into a unified vocabulary, allowing the model to to efficiently process and generate both modalities. Interactive motion is represented as $X_m = \{k_{1:D}^{1;a}, k_{1:D}^{1;b}, \cdots, k_{1:D}^{L;a}, k_{1:D}^{L;b}\}$, where X_m denotes the motion sequence encoded in the unified vocabulary, and $k_{1:D}^{i;a} \in [K]^D$ is the i-th token of motion a.

Finally, to visualize the generated motion tokens, we use the motion decoder of the RQ-VAE. The decoder projects the quantized features $\hat{\mathbf{z}}^i = \sum_{d=1}^{D} \mathbf{e}(k_d^i)$, converting them back into motion sequences.

5.4.3 Training

We describe the training strategy in MoLaM, to convert a language model into an interactive motion-language model.

Motion Tokenizer The motion tokenizer consists of an encoder, decoder, and quantizer. We followed the original objective functions from [153], minimizing the reconstruction loss, the codebook loss to align the encoder's outputs with the codebook, and the loss of commitment to ensure the consistency of the encoder. After training the encoder and decoder, we freeze their parameters throughout the rest of the training stage.

The interactive motion token sequence is represented as $X_m = \{k_{1:D}^{1;a}, k_{1:D}^{1;b}, \cdots, k_{1:D}^{L;a}, k_{1:D}^{L;b}\}$, where X_m is a sequence of motion represented in unified vocabulary and $k_{1:D}^{i;a} \in [K]^D$ is the i-th token of motion a. In particular, the motion token is represented as below:

$$
\begin{aligned}
X_m = \{&\texttt{<motion\_token\_start>}, \\
&\texttt{<motion\_token\_a\_start>}, \quad k_1^{1;a}, \cdots, k_D^{1;a}, \quad \texttt{<motion\_token\_a\_end>}, \\
&\texttt{<motion\_token\_b\_start>}, \quad k_1^{1;b}, \cdots, k_D^{1;b}, \quad \texttt{<motion\_token\_b\_end>}, \\
&\ldots \\
&\texttt{<motion\_token\_a\_start>}, \quad k_1^{L;a}, \cdots, k_D^{L;a}, \quad \texttt{<motion\_token\_a\_end>}, \\
&\texttt{<motion\_token\_b\_start>}, \quad k_1^{L;b}, \cdots, k_D^{:;b}, \quad \texttt{<motion\_token\_b\_end>}, \\
&\texttt{<motion\_token\_end>}\}
\end{aligned}
$$

where <motion_token_start>, <motion_token_a_start>, <motion_token_b_start>, <motion_token_a_end>, <motion_token_b_end>, and <motion_token_end> is a special token added to the unified vocabulary. For modeling single-motion in pre-training we omitted the input string about motion_token_b.

Pre-training for Cross-modal Motion-Text Alignment The goal of this stage is to enable the large language models (LLMs) to process and generate interaction motion tokens effectively. To achieve this, we continuously pre-train LLMs using paired interaction motion-text datasets, such as Inter-X [88] and InterHuman [89], across various tasks including motion-to-text, text-to-motion, motion prediction, and reaction generation.

For each task, we construct sequences y that combine motion sequences with their corresponding captions and train with a next-token prediction objective $\mathscr{L} = -\log \sum^{T} p_{\theta}(y_i|y_{<i})$. To improve training efficiency, we employ LoRA adaptor [110], similar to [143], and merge its parameters to the LLM backbone. Furthermore, due to a limited number of interactive motion data, we also leverage a subset of single-person motion-text datasets from Motion-X [154]. This additional single-person data offers prior knowledge of how the individual motions are described in language, enhancing the model's ability to align motions with textual descriptions.

Instruction-tuning with Inter-MT^2 Data In this stage, we aim to enhance the model to extend beyond understanding and generating single-turn interaction motions and focusing on handling diverse and complex instructions presented through **multi-turn** conversational scenarios. Similar to the pre-training stage, We adopt a next-token prediction training objective for training. The instruction-tuning sequences are composed of user interactions paired with corresponding responses, integrating tokens from a unified vocabulary that covers texts, motions, or both modalities. We also leverage the Inter-MT^2 dataset along with single-turn interaction data from existing motion datasets [88, 89], formatted according to the instruction template of [78].

5.5 Experiments for Motion-Language Model

In this section, we evaluate the effectiveness of MoLaM, particularly focusing on its capability to accurately understand and generate interactive motions in complex, multi-turn conversational scenarios involving both motions and text modalities. To extensively validate our approach, we compare MoLaM against several specialized baseline methods, each explicitly designed for individual tasks. This allows us to understand the performance and versatility of MoLaM. Additionally, we investigate the contribution and effectiveness of our proposed dataset, Inter-MT^2, showing how it enhances MoLaM's ability to process and generate interactive motion and texts. We also provide qualitative video results generated by MoLaM in the supplementary material.

5.5.1 Evaluation Tasks and Baselines

Motion Reasoning We introduce a motion reasoning task to validate the model's ability to comprehend interactive motions and text queries. Motion reasoning involves predicting past or future events, or reasoning about current motions, based on prior conversational data. This task requires the model to understand the context of the conversation, interpret how the given interactive motion fits within that context, and adjust its reasoning accordingly. We utilize LLMs-based evaluator, specifically GPT-4o [150], to assess the content alignment, naturalness, and logical coherence of the generated textual responses. Content alignment evaluates how accurately the text reflects the given interactive motions, logical coherence checks the consistency and reasoning accuracy of inferences made about past or future events, and naturalness evaluates the fluency of generated texts, with rating each metric on a 10-point scale. Additionally, we employ linguistic metrics, such as METEOR [155], and MAUVE [156] to quantitatively evaluate relevance and fluency against 2002 labeled samples from the Inter-MT^2 test set. We present the results on motion reasoning in §5.5.2.

Motion Editing In the motion editing task, the model modifies the given motion based on a person's persona or scenario, e.g., emotions or relationship dynamics, which adds complexity as changes in one individual affect the other's motion. Unlike single-person motion editing [76, 77], the task that edits interactive motions should consider preserving contextual coherence and social dynamics. We evaluated the methods on 1445 samples from the Inter-MT^2 test set. In a within-subject user study (following [76]), 30 participants each rated five samples (from 30 randomly selected tests) on content similarity, instruction alignment, and motion quality using a 5-point Likert scale. Content similarity evaluates whether the edited motion preserves the original meaning of the source motion, while instruction alignment assesses how accurately the edited motion follows the given command. Participants compared our method against four baselines by reviewing randomly shuffled motion outputs. Additionally, we measured performance using data-driven metrics, Frechet Inception Distance (FID), and mean per joint position error (MPJPE), against the labeled motions in the Inter-MT^2 test set, following [76]. The results are detailed in §5.5.3.

Traditional Motion Relevant Tasks We further evaluated our method on three traditional interactive motion tasks: motion-to-text, text-to-motion, and reaction generation, using the combined test sets from InterHuman [89] and Inter-X [88]. Text-motion matching is assessed via top-3 retrieval precision (batch size 32) in the retrieval models' feature space [151]. Motion quality is measured by the Frechet Inception Distance (FID) and the accuracy of reaction motions is measured by mean per joint position error (MPJPE) in meters. Detailed results are in §5.5.4.

Baselines Since our interactive multi-turn scenarios and tasks, including interactive motion reasoning and motion editing, are novel, there is no exact comparison method. We compare our method against reasonable baselines that handle both motion and texts as input and output.

- **Two-stage approach**. We leverage off-the-shelf LLMs and motion-to-text methods. For the motion reasoning task, we convert motions to text via the state-of-the-art motion-to-text model, TM2T [70] and then apply large language models (GPT-4o [150], LLaMA-3.1-8B [4]). For the motion editing task, we first convert the given motions into text descriptions using TM2T, and we concatenate the motion description with editing command texts. We then put the texts to InterGEN [89] to generate modified motions.

- **Extending unified single-human motion model**. We adopt a single-human motion-language model, MotionGPT [78], for interactive motions. We consider three variations: (1) MotionGPT*: a modified MotionGPT fine-tuned on interactive motion data; (2) MotionGPT$^*_I$: MotionGPT* enhanced with Inter-MT^2 dataset; (3) MoLaM w/o Inter-MT^2: our method fine-tuned with instruction templates, but without Inter-MT^2 data.

- **Motion generation baselines.** For traditional motion tasks, we benchmarked against interactive motion generation methods, ComMDM [92] and InterGEN [89], and a single human motion generation model, MoMask* [72].

Methods	LLM-Assisted			Linguistic Metrics	
	Coh. ↑	Align. ↑	Nat. ↑	METEOR	MAUVE
two-stage approach					
TM2T + LLaMA-3.1	3.852	3.050	6.348	0.226	0.009
TM2T + GPT-4o	<u>4.266</u>	<u>3.455</u>	<u>6.790</u>	0.227	0.019
unified approach					
MotionGPT*	1.855	1.303	3.574	0.096	0.005
MotionGPT$^*_I$	3.690	3.160	5.291	<u>0.218</u>	<u>0.417</u>
MoLaM w/o Inter-MT^2	2.770	2.141	4.968	0.145	0.004
MoLaM (Ours)	**5.252**	**4.511**	**6.981**	**0.260**	**0.794**

Table 5.2: Evaluation on Motion Reasoning task with Inter-MT^2 test set. Coh., Align., and Nat. denote logical coherence, content alignment, and naturalness, respectively. **Bold** indicates best performance and <u>underline</u> denotes the second best performance.

Figure 5.6: Generated samples for interactive motion reasoning task. This example shows how MoLaM explains behaviors and their motivations, demonstrating a deeper understanding of scenarios by incorporating context from prior interactions.

5.5.2 Motion Reasoning

In the motion reasoning task, conversations about two interactive motions are examined to assess the model's ability to deduce past or future events and comprehend the

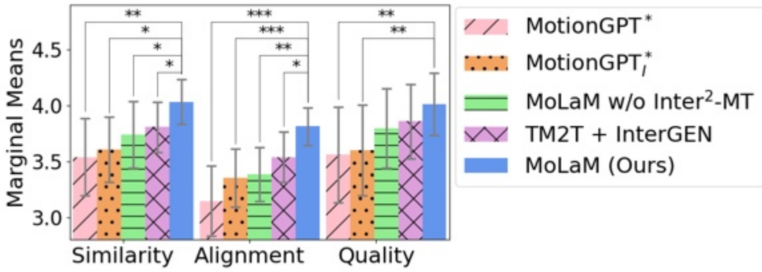


Figure 5.7: User subject study results for motion editing. We plotted the difference only in a post hoc pairwise comparison of the proposed method. * as $0.01 < p < 0.05$, ** as $p < 0.01$, and *** as $p < 0.001$. The error bars represent 95% confidence intervals.

motivations driving the motions. The experimental results in Table 5.2 demonstrate that our unified model, MoLaM, outperforms baselines across all LLM-assisted and linguistic metrics. Specifically, MoLaM achieves improvements with performance increases exceeding 1.9 points in logical coherence, 1.1 points in content alignment, and nearly 0.2 points in naturalness compared to the best two-stage model.

The improved performance of our unified model, MoLaM, over two-stage approaches, appears to result from two key factors: error accumulation and interpretation ambiguity. Two-stage models can carry over errors if the motion captioning step is inaccurate, undermining content alignment and coherence. In contrast, our unified architecture integrates motion encoding and reasoning in a single framework, minimizing error propagation. Moreover, a single caption may not fully capture multiple interpretations of the same motion, compromising context accuracy in two-stage setups. Our unified approach, however, accounts for these varied interpretations to generate more contextually precise outputs. Figure 5.6 shows its ability to dynamically adjust interpretations and responses by incorporating context from previous conversations.

Methods	FID ↓	MPJPE ↓
two-stage approach		
TM2T + InterGEN	0.110	0.811
unified approach		
MotionGPT*	0.251	4.002
MotionGPT_I^*	0.161	3.982
MoLaM w/o Inter-MT^2	0.080	0.908
MoLaM (Ours)	**0.064**	**0.758**

Table 5.3: Quantitative results in motion editing task.

5.5.3 Motion Editing

We aimed to validate the hypothesis that users perceive interactive motions edited by our proposed method as more content-consistent, better aligned with instructions, and of higher overall quality. To investigate this, we conducted a user study and analyzed the results using repeated-measures multivariate analysis of variance (MANOVA). The analysis revealed significant effects of the method on user perception across all evaluated dimensions; $F(4) = 4.591, p = 0.002, \eta^2 = 0.137$ for content similarity, $F(4) = 7.134, p = 0.000, \eta^2 = 0.197$ for instruction alignment, and $F(4) = 4.781, p = 0.001, \eta^2 = 0.142$ for motion quality, with all $\alpha = 0.05$. The estimated marginal mean of the rated score is reported in Figure 5.7. The results show that the proposed method had better alignment, quality, and consistency of instruction in other baselines with significant differences.

During post-hoc pairwise comparisons, MoLaM significantly outperforms the two-stage model (TM2T [70] with InterGEN [89]) in terms of content similarity ($p = 0.017$) and instruction alignment ($p = 0.010$). The two-stage model had lower content similarity due to motion-to-text conversion errors causing unintended motions. It also struggled

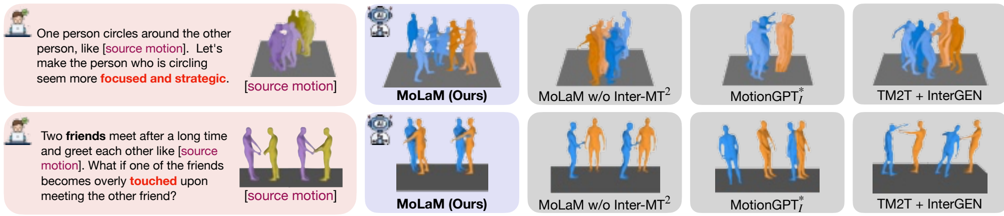


Figure 5.8: Generated samples for interactive motion editing. The proposed method excels in capturing nuances, outperforming alternatives in content similarity and instruction alignment.

with instruction alignment since InterGEN was trained to generate motions from textual descriptions, limiting its adaptability. In contrast, our unified framework avoids error accumulation and, trained on diverse instructions, demonstrates superior reasoning and adaptability for accurate motion editing and generation. Compared to MoLaM w/o Inter-MT^2, our model significantly improves content similarity ($p = 0.010$) and instruction alignment ($p = 0.001$), suggesting that excluding Inter-MT^2 data hinders motion control. It also outperforms MotionGPT$^*_I$ in all metrics, indicating that the baseline's VQ-based tokenizer struggles to capture precise relative joint positions in two-person motion. Further ablation studies on the motion tokenizer are provided in the supplementary materials. Quantitative evaluations using data-driven metrics, specifically FID and MPJPE (Table 5.3), further confirmed the superiority of our method over baseline methods, consistent with user study results. Examples of generated edited motions are illustrated in Figure 5.8.

5.5.4 Traditional Motion Related Tasks

In this section, we conduct comparison experiments on existing motion-relevant tasks, such as motion-to-text (M2T), text-to-motion (T2M), and reaction generation. The detailed results are in Table 5.4. The first row ("Real") shows retrieval accuracy, and FID

Methods	M2T	T2M		Reaction Gen.	
	R Top3 ↑	R Top3 ↑	FID ↓	MPJPE ↓	FID ↓
Real	0.867	0.869	0.00	-	0.00
task-specific approach					
TM2T*	0.696	0.534	0.300	-	-
MoMask*	-	<u>0.612</u>	<u>0.066</u>	1.602	0.112
ComMDM	-	0.251	0.304	-	-
InterGEN	-	**0.645**	0.078	-	-
unified approach					
MotionGPT*	0.494	0.328	0.123	3.444	0.355
MotionGPT^*_I	0.503	0.331	0.118	1.436	0.380
MoLaM w/o Inter-MT^2	<u>0.894</u>	0.561	0.082	<u>0.984</u>	<u>0.031</u>
MoLaM (Ours)	**0.901**	0.568	**0.059**	**0.691**	**0.019**

Table 5.4: Comparisons for three motion-related tasks on Inter-X and InterHuman datasets. M2T denotes motion-to-text, T2M for text-to-motion, and Reaction Gen. for reaction generation.

scores from the dataset labels. Note that both MoLaM w/o Inter-MT^2 and MotionGPT* were trained on all of these tasks for fair comparison. The results confirm that incorporating Inter-MT^2 dataset enhances the model's performance in traditional motion tasks, by comparing with MoLaM w/o Inter-MT^2.

For M2T, Top-3 retrieval accuracy improved from 0.894 (MoLaM w/o Inter-MT^2) to 0.901 (MoLaM). For T2M, it rose from 0.561 to 0.568, with FID dropping from 0.082 to 0.059, indicating better motion generation. For reaction generation, MPJPE decreased from 0.984 to 0.691 and FID from 0.031 to 0.019, highlighting the benefits of multi-turn datasets for motion comprehension and generation. We believe that Inter-MT^2 dataset provides diverse, context-rich examples, helping the model learn more nuanced relationships between text and motion.

In addition, we compared MoLaM against task-specific methods, each optimized individually per each task. Note that the methods marked with an asterisk (*) were originally designed for single-motion tasks and were trained on interactive motion data for our evaluation. MoLaM outperforms these specialized models in motion-to-text (M2T) and reaction generation tasks, achieving higher retrieval precision accuracy and lower MPJPE and FID scores. In the text-to-motion task (T2M), MoLaM achieves a comparable performance against the state-of-the-art task-specific models, including InterGEN and MoMask*, highlighting the ability to generate high-quality interactive motions.

5.5.5 Generating Multi-Human Motions

Figure 5.9: Expanding MoLaM to generate multiple human motions. For clarity, we simplified the incremental process, where MoLaM first generates a two-person motion and adds a third person to it.

Interestingly, the versatility of MoLaM allows it to generalize beyond two-person interactions without explicit fine-tuning on multi-person interactive data. Specifically, we first generate motion for a pair of individuals based on the given text description, and then the motion of additional participants is synthesized while conditioning on the pre-existing interactions. This approach ensures that the newly generated motions remain coherent and contextually appropriate within the evolving group dynamics. By leveraging prior turns, MoLaM seamlessly integrates the new figure's movements, as shown in Figure 5.9. Since our method is agnostic to the number of people, it can readily extend to groups or crowds, provided interactive multi-person data is available.

5.6 Discussions

In this chapter, we introduced MoLaM, a versatile motion-language model designed to understand, generate, and reason about interactive motions. We presented the detailed architecture and training strategy of our unified framework, which integrates large language models with interactive motion modality. To further enhance the model's reasoning capabilities and applicability, we presented a specialized dataset, Inter-MT^2, which incorporates a variety of reasoning tasks set within multi-turn conversations centered on interactive motions. Our comprehensive experiments demonstrated that MoLaM successfully handles instruction-following, motion editing, and motion reasoning tasks, highlighting its capability to effectively interpret and generate contextually accurate interactive motions.

5.6.1 Limitations

The expressiveness of our models remains limited when handling complex or previously unseen actions, indicating a need for further diverse motion source data in its ability to generalize across diverse motion scenarios. In addition, the sequence length becomes excessively long as we flatten the residual motion tokens, which can impact efficiency and computational resources. Leveraging additional transformer models to predict the residual token can reduce this work. Lastly, our method faces challenges in personalization and interpretability, as motion is inherently ambiguous and users may interpret the same motion in different ways. Addressing this issue will require incorporating more tailored approaches that adapt to individual user preferences and expectations through further human-in-the-loop feedback and refinement processes.

5.6.2 Future Directions

A promising direction is to deploy the proposed framework on real humanoid robots to achieve fluid and context-aware interactive behaviors in physical environments. Real-world embodiment would allow the model to not only generate motions but also close the perception–action loop through real-time sensory feedback and social interaction. However, the current framework remains computationally heavy and slow for on-board deployment, limiting its use to offline or scripted scenarios. Future work will therefore focus on improving model efficiency—through lightweight adaptation, motion compression, and hierarchical control interfaces—so that interactive foundation models can operate in real time on physical humanoid platforms while maintaining expressivity and safety.

Chapter 6

Conclusion

This dissertation has aimed to take a step toward foundation models for multimodal reasoning in human-centered robot behavior: models that treat language, perception, and motion as coupled channels for understanding and shaping how robots act around people. Rather than proposing a single, universal model, the work has focused on concrete instances in which multimodal reasoning already makes a qualitative difference for interaction, from clarifying ambiguous instructions to expressing intent through motion, and ultimately to reasoning jointly over language and trajectories in a shared space.

6.1 Summary

This dissertation began by framing the design space for foundation models in human environments. I introduced the notion of multimodal reasoning in robot behavior, defined the scope of human-centered behavior of interest, and identified key capability axes: multimodal perception and representation, verbal and non-verbal interaction, integrated decision-making, learning and adaptation from diverse data, and safety, predictability, and user trust. This conceptual framing provided the backdrop for the technical chapters that followed.

Chapter 2 reviewed related work across foundation models, large language and vision–language models, vision–language–action policies, non-verbal interaction in robotics, and motion- and interaction-centric models. The survey highlighted both the progress and the gaps at the intersection of foundation models and human-centered behavior, motivating the need for methods that explicitly account for ambiguity, failure, and the communicative role of motion.

Chapter 3 focused on verbal interaction. It showed how large pretrained models can be wrapped into failure-aware, language-conditioned planners that reason about feasibility using both successful and unsuccessful experience. It further introduced a framework for classifying user commands into clear, ambiguous, and infeasible categories, and for using clarification questions and explanations to negotiate goals, rather than blindly executing literal parses. These contributions demonstrate that multimodal reasoning over language and perception can make verbal interaction more reliable and transparent.

Chapter 4 turned to non-verbal interaction and embodied motion. Here, the primary signals are human pose, gesture, and movement, and the robot's behavior is expressed through its own motion. We developed a persona-conditioned motion policy that treats trajectories as communicative signals, learning to generate and adapt robot motions that are both physically feasible and expressive with respect to a partner. This shows how motion-level representations can be used to interpret human behavior and to synthesize robot behavior that varies along social and stylistic dimensions, not only along task outcome.

Chapter 5 adopted a holistic perspective by aligning language and interactive motion within a single model. We introduced a framework that supports language-guided understanding, generation, and editing of interaction-scale trajectories. The model can describe what has happened, answer questions about ongoing or past motion, predict plausible continuations, and modify trajectories to meet new relational or stylistic constraints. This chapter demonstrates that language and motion can share a representation that supports both analysis and synthesis in complex human–robot interactions.

Finally, this concluding chapter revisits the overarching goals, synthesizes the contributions across verbal, non-verbal, and integrated settings, and discusses the broader implications and open challenges for building foundation models for multimodal reasoning in human-centered robot behavior.

6.2 Synthesis of Thesis Contributions

This dissertation makes the following technical contributions.

1. **Problem formulation and design space.** I formulate the goal of foundation models for multimodal reasoning in human-centered robot behavior and organize it along several capability axes: multimodal perception and representation, verbal and non-verbal interaction, integrated decision-making, learning and adaptation, and safety. This provides a common vocabulary and design space that connects large language, vision–language, and motion-centric models to the requirements of robots acting in human environments.

2. **Verbal interaction under ambiguity and feasibility constraints.** We develop two approaches for language-driven robot behavior. The first uses large pretrained models as failure-aware, language-conditioned planners that exploit both successful and unsuccessful experience to bias plan selection toward feasible behaviors. The second classifies user commands into clear, ambiguous, and infeasible categories and generates clarification questions or explanations, enabling robots to negotiate goals instead of executing literal but inappropriate interpretations.

3. **Non-verbal interaction and expressive motion.** We propose a motion-based in-

teraction framework in which the robot responds to a human partner's pose, gesture, and movement through persona-conditioned trajectories. The model generates and adapts motions that are physically feasible while varying along interactional dimensions (e.g., more reserved or more proactive), showing that a single motion model can support non-verbal interaction beyond scripted behaviors.

4. **Unified language–motion modeling for interactive trajectories.** We introduce a multimodal model that aligns language with interaction-scale motion trajectories, enabling joint reasoning over text and motion. The model supports motion description, question answering about ongoing or past interaction, prediction of plausible continuations, and editing of trajectories to satisfy new relational or stylistic constraints, demonstrating a shared representation that covers both understanding and generation of human–robot interaction sequences.

6.3 Progress Toward the Long-Term Vision

The long-term vision articulated in this dissertation is ambitious: a family of foundation models that support multimodal reasoning for robots acting in diverse human environments, handling verbal and non-verbal interaction in a unified way, and adapting to new tasks, users, and embodiments with minimal additional supervision. The contributions here represent only a partial realization of that vision, but clarify where progress has been made and where substantial gaps remain.

On the positive side, the dissertation demonstrates that foundation-model ideas can be meaningfully instantiated in concrete human–robot scenarios. I have shown that large pretrained models, when appropriately grounded, can reason about feasibility and ambiguity in language-driven tasks; that a single motion model can capture qualitative differences in interaction style and respond to a partner in real time; and that language and interactive trajectories can be brought into a shared space that supports question answering and editing over motion. These results indicate that multimodal reasoning is already feasible at the scale of individual tasks, controlled environments, and single robots.

At the same time, several aspects of the envisioned foundation models remain deliberately out of scope. In terms of multimodal perception and representation, the systems studied here assume relatively clean visual observations, proprioception, and scripted interaction histories, and do not yet incorporate rich tactile or force sensing, audio, or the full diversity of objects, layouts, and long-horizon context found in real homes or workplaces. Verbal interaction is restricted to short, task-oriented exchanges in a single language and does not address prosody, overlapping speech, or multi-party dialog. Non-verbal interaction and motion are modeled for a single human partner in simplified settings, without dense crowds, complex affordances, or long-term adaptation of motion styles to individual users. Integrated multimodal reasoning and decision-making are explored at the level of individual tasks and scenarios, and there is no single model in this dissertation that simultaneously realizes all of the capability axes outlined earlier. Learning and adaptation are largely episodic and offline, relying on finite datasets with modest amounts of robot-specific data, rather than on continual learning from ongoing deployment. Safety, predictability, and user trust are handled through simple heuristics and qualitative evaluation, rather than through formal guarantees, large-scale user studies, or systematic risk assessment.

6.4 Overall Impact Statement

This dissertation advances the emerging intersection of foundation models and human-centered robotics by treating human–robot interaction not as a single modality or task, but as a coupled space of language, perception, motion, and social meaning shaped by uncertainty. Across its components, the work argues that trustworthy autonomy in human environments requires robots that can (i) reason explicitly about uncertainty in both planning and communication, (ii) express and interpret social cues through persona and motion, and (iii) ground these capabilities in architectures that combine large pretrained models with task-specific structure. Rather than assuming perfect instructions, perfect execution, or emotionally neutral behavior, the dissertation embraces the reality that failures, ambiguities, and affective responses are integral to real-world interaction.

Conceptually, the dissertation offers a vocabulary and design space for thinking about foundation models for robot behavior, emphasizing multimodal reasoning, interaction-level objectives, and the role of imperfect experience. It frames robot behavior as a process of jointly interpreting commands, predicting the feasibility and consequences of actions, and shaping social impressions over time. Uncertainty is elevated from an afterthought to a first-class design principle: action execution and command interpretation are modeled as probabilistic processes whose residual doubts can drive clarification, re-planning, or conservative choices. In parallel, persona and expressive motion are treated not as cosmetic add-ons but as channels through which robots communicate intent, competence, and attitude, with direct consequences for user trust and engagement.

Methodologically, the dissertation introduces concrete architectures that operationalize these ideas. On the "robustness" side, it presents systems that wrap large pretrained language and vision–language models for failure-aware planning and for command interpretation and clarification, showing how tree- or sampling-based reasoning over model uncertainty can prevent brittle plans and encourage timely user input. On the "expressivity" side, it develops specialized multimodal models for persona-conditioned motion and for language–motion alignment, enabling robots and virtual agents to generate, interpret, and edit interactive trajectories conditioned on social roles or affective styles. Together, these systems illustrate how general-purpose foundation models and targeted task-specific components can be combined to produce behaviors that are not only more capable in terms of task success but also more legible and socially responsive.

Empirically, the dissertation provides evidence that these methods improve reliability, expressivity, and flexibility in representative human–robot interaction scenarios. Failure-aware planning reduces avoidable breakdowns by recognizing low-feasibility action sequences before execution and by engaging users to disambiguate under-specified or infeasible commands. Command interpretation frameworks that explicitly classify clear, ambiguous, and infeasible inputs enable robots to ask for clarification when it is most needed, rather than silently guessing. Persona-conditioned motion models allow users to perceive distinct characters and intentions directly from physical behavior, expanding the design space for engaging, relatable, and emotionally resonant interaction. At the same time, the experiments surface important limitations and failure modes, underscoring that these techniques are steps toward, rather than complete solutions for, human-centered foundation models.

A central theme throughout is safety and ethical deployment. Uncertainty-aware planning and communication directly support safety by allowing robots to "know when they do not know," deferring to humans or opting for safer alternatives when confidence is low. Yet the same tools that increase capability also introduce new risks. Persona shaping is a powerful social cue: disagreeable, confrontational, or deliberately indifferent personas can evoke discomfort, erode trust, or be mistaken for a technical malfunction if users are not adequately informed. As expressive motion models become more capable, they may also be pushed toward inappropriate, deceptive, or manipulative behaviors, especially if driven by unconstrained prompts or optimization objectives that prioritize engagement over well-being. The dissertation, therefore, advocates for explicit safeguards: transparent communication about persona and capabilities, mechanisms for user consent and reset, content and behavior filters for violent, sexual, or otherwise harmful motions, and evaluation protocols that include not only task metrics but also user comfort, fairness, and long-term trust.

Beyond the specific systems introduced, the broader impact of this work lies in reframing how data and models are used in robotics. Instead of treating failures, ambiguities, and social nuances as noise to be stripped away, the dissertation treats them as valuable signals that inform how robots should reason and act around people. Instead of viewing language and motion as separate modules, it argues for shared representations that allow robots to talk about, reason over, and reshape their own behavior. These perspectives are intended to inform future work on large-scale robot learning and to help ensure that the next generation of foundation models for robotics are not only more capable in terms of task performance but also more aligned with the demands of human-centered environments, where safety, transparency, and social alignment are as critical as raw competence.

6.5 Future Directions

Several concrete extensions follow from the limitations discussed above.

First, there is a practical gap between the current holistic language–motion model and deployment on real humanoid robots. The present model is too slow and monolithic to run in real time on a full-body platform, and it does not interface explicitly with compliant control, balance, or low-level safety mechanisms. A short-term direction is to develop more efficient base models and distillation pipelines, and to treat personas and styles as in-context or lightly fine-tuned adaptations on top of a shared backbone that can be coupled cleanly with existing whole-body controllers.

Second, extending the modality set to include touch is an important step. Human-centered robots will be touched by people and will manipulate objects under contact and uncertainty, and these two regimes of tactile feedback are not identical. A useful foundation model should eventually distinguish and relate representations for social touch (e.g., contact on the arm or hand) and task-oriented touch (e.g., contact during grasping or pushing), and exploit both for safety and task performance. This will require datasets and architectures that integrate tactile or force sensing alongside vision, language, and motion.

Third, the models studied here largely rely on foundation models as generic feature extractors, without explicitly organizing social and commonsense knowledge. Many interaction decisions, however, depend on shared expectations about objects, activities, and roles. A natural direction is to make more deliberate use of social and commonsense structure learned by large language and vision–language models, and to study how this structure can be grounded in robot perception and control to support everyday reasoning in tasks that involve people.

Finally, the dissertation treats verbal interaction, non-verbal interaction, and language–motion modeling as related but separate systems. A longer-term goal is to unify more of these capabilities into a single multimodal model that can support a wider range of behaviors, while still exposing clear interfaces to low-level control and safety layers. Moving toward such a unified model will require scaling data and models, but also clearer abstractions for how high-level multimodal reasoning should connect to the concrete controllers that keep robots safe and useful in human environments.

Reference

[1] Kevin Black, Noah Brown, Danny Driess, Adnan Esmail, Michael Equi, Chelsea Finn, Niccolo Fusai, Lachy Groom, Karol Hausman, Brian Ichter, et al. π_0: A vision-language-action flow model for general robot control. *arXiv preprint arXiv:2410.24164*, 2024.

[2] Gheorghe Comanici, Eric Bieber, Mike Schaekermann, Ice Pasupat, Noveen Sachdeva, Inderjit Dhillon, Marcel Blistein, Ori Ram, Dan Zhang, Evan 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.

[3] An Yang, Baosong Yang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Zhou, Chengpeng Li, Chengyuan Li, Dayiheng Liu, Fei Huang, et al. Qwen2 technical report. *arXiv preprint arXiv:2407.10671*, 2024.

[4] Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Amy Yang, Angela Fan, et al. The llama 3 herd of models. *arXiv preprint arXiv:2407.21783*, 2024.

[5] Gemma Team, Morgane Riviere, Shreya Pathak, Pier Giuseppe Sessa, Cassidy Hardin, Surya Bhupatiraju, Léonard Hussenot, Thomas Mesnard, Bobak Shahriari, Alexandre Ramé, et al. Gemma 2: Improving open language models at a practical size. *arXiv preprint arXiv:2408.00118*, 2024.

[6] Haotian Liu, Chunyuan Li, Qingyang Wu, and Yong Jae Lee. Visual instruction tuning. *arXiv preprint arXiv:2304.08485*, 2023.

[7] Lucas Beyer, Andreas Steiner, André Susano Pinto, Alexander Kolesnikov, Xiao Wang, Daniel Salz, Maxim Neumann, Ibrahim Alabdulmohsin, Michael Tschannen, Emanuele Bugliarello, et al. Paligemma: A versatile 3b vlm for transfer. *arXiv preprint arXiv:2407.07726*, 2024.

[8] OpenAI. Gpt-4v(ision) technical work and authors. 2023. https://cdn.openai.com/contributions/gpt-4v.pdf.

[9] Johan Bjorck, Fernando Castañeda, Nikita Cherniadev, Xingye Da, Runyu Ding, Linxi Fan, Yu Fang, Dieter Fox, Fengyuan Hu, Spencer Huang, et al. Gr00t n1: An open foundation model for generalist humanoid robots. *arXiv preprint arXiv:2503.14734*, 2025.

[10] Moo Jin Kim, Karl Pertsch, Siddharth Karamcheti, Ted Xiao, Ashwin Balakrishna, Suraj Nair, Rafael Rafailov, Ethan P Foster, Pannag R Sanketi, Quan Vuong, Thomas Kollar, Benjamin Burchfiel, Russ Tedrake, Dorsa Sadigh, Sergey Levine, Percy Liang, and Chelsea Finn. OpenVLA: An open-source vision-language-action model. In *Proc. of the 8th Annual Conference on Robot Learning (CoRL)*, 2024.

[11] Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. Language models are few-shot learners. In *Proc. of the Advances in Neural Information Processing Systems (NeurIPS)*, volume 33, pages 1877–1901, 2020.

[12] Andrei Hurst et al. Gpt-4o system card. *arXiv preprint arXiv:2410.21276*, 2024.

[13] Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Fei Xia, Ed Chi, Quoc V Le, Denny Zhou, et al. Chain-of-thought prompting elicits reasoning in large language models. *Advances in neural information processing systems*, 35:24824–24837, 2022.

[14] Shunyu Yao, Jeffrey Zhao, Dian Yu, Nan Du, Izhak Shafran, Karthik Narasimhan, and Yuan Cao. React: Synergizing reasoning and acting in language models. In *Proc. of the International Conference on Learning Representations (ICLR)*, 2023.

[15] Bin Lin, Bin Zhu, Yang Ye, Munan Ning, Peng Jin, and Li Yuan. Video-llava: Learning united visual representation by alignment before projection. *arXiv preprint arXiv:2311.10122*, 2023.

[16] Dongze Huang, Cheng Yan, Qian Li, and Xi Peng. From large language models to large multimodal models: A literature review. *Applied Sciences*, 14(12):5068, 2024.

[17] Shukang Yin, Chaoyou Fu, Sirui Zhao, Ke Li, Xing Sun, Tong Xu, and Enhong Chen. A survey on multimodal large language models. *National Science Review*, 11(12):nwae403, 2024.

[18] Jing Yu Koh, Daniel Fried, and Ruslan Salakhutdinov. Generating images with multimodal language models. *arXiv preprint arXiv:2305.17216*, 2023.

[19] Chunting Zhou, Lili Yu, Arun Babu, Kushal Tirumala, Michihiro Yasunaga, Leonid Shamis, Jacob Kahn, Xuezhe Ma, Luke Zettlemoyer, and Omer Levy.

Transfusion: Predict the next token and diffuse images with one multi-modal model. *arXiv preprint arXiv:2408.11039*, 2024.

[20] Yuying Ge, Sijie Zhao, Jinguo Zhu, Yixiao Ge, Kun Yi, Lin Song, Chen Li, Xiaohan Ding, and Ying Shan. Making llama see and draw with seed tokenizer. In *Proceedings of the International Conference on Learning Representations (ICLR)*, 2024.

[21] Yuying Ge, Sijie Zhao, Jinguo Zhu, Yixiao Ge, Kun Yi, Lin Song, Chen Li, Xiaohan Ding, and Ying Shan. Seed-x: Multimodal models with unified multi-granularity comprehension and generation. *arXiv preprint arXiv:2404.14396*, 2024.

[22] Zhifei Xie, Ziyang Ma, Zihang Liu, Kaiyu Pang, Hongyu Li, Jialin Zhang, Yue Liao, Deheng Ye, Chunyan Miao, and Shuicheng Yan. Mini-omni: Language models can hear, talk while thinking in streaming. *arXiv preprint arXiv:2408.16725*, 2024.

[23] Qwen Team. Qwen2.5-omni: An end-to-end multimodal model for text, vision, audio, and video. https://github.com/QwenLM/Qwen2.5-Omni, 2025. GitHub repository.

[24] Aaron W Li, Veronica Jiang, Steven Y Feng, Julia Sprague, Wei Zhou, and Jesse Hoey. Aloha: Artificial learning of human attributes for dialogue agents. In *Proc. of the AAAI Conference on Artificial Intelligence*, volume 34, pages 8155–8163, 2020.

[25] Mustafa Safdari, Greg Serapio-García, Clément Crepy, Stephen Fitz, Peter Romero, Luning Sun, Marwa Abdulhai, Aleksandra Faust, and Maja Matarić. Personality traits in large language models. *arXiv preprint arXiv:2307.00184*, 2023.

[26] Yoon Kyung Lee, Yoonwon Jung, Gyuyi Kang, and Sowon Hahn. Developing social robots with empathetic non-verbal cues using large language models. *arXiv preprint arXiv:2308.16529*, 2023.

[27] Daniel T. Larsson, Dipankar Maity, and Panagiotis Tsiotras. Q-tree search: An information-theoretic approach toward hierarchical abstractions for agents with computational limitations. *IEEE Transactions on Robotics*, 36(6):1669–1685, 2020.

[28] Yann Labbé, Sergey Zagoruyko, Igor Kalevatykh, Ivan Laptev, Justin Carpentier, Mathieu Aubry, and Josef Sivic. Monte-carlo tree search for efficient visually guided rearrangement planning. *IEEE Robotics and Automation Letters*, 5(2):3715–3722, 2020.

[29] Anna Yershova and Steven M. LaValle. Improving motion-planning algorithms by efficient nearest-neighbor searching. *IEEE Transactions on Robotics*, 23(1):151–157, 2007.

[30] E. Plaku, K.E. Bekris, B.Y. Chen, A.M. Ladd, and L.E. Kavraki. Sampling-based roadmap of trees for parallel motion planning. *IEEE Transactions on Robotics*, 21(4):597–608, 2005.

[31] Shunyu Yao, Dian Yu, Jeffrey Zhao, Izhak Shafran, Tom Griffiths, Yuan Cao, and Karthik Narasimhan. Tree of thoughts: Deliberate problem solving with large

language models. *Proc. of the Advances in neural information processing systems (NeurIPS)*, 36:11809–11822, 2023.

[32] Zirui Zhao, Wee Sun Lee, and David Hsu. Large language models as commonsense knowledge for large-scale task planning. In *Proc. of the Thirty-seventh Conference on Neural Information Processing Systems (NeurIPS)*, 2023.

[33] Dan Zhang, Sining Zhoubian, Ziniu Hu, Yisong Yue, Yuxiao Dong, and Jie Tang. ReST-MCTS*: LLM self-training via process reward guided tree search. In *Proc. of the Thirty-eighth Annual Conference on Neural Information Processing Systems (NeurIPS)*, 2024.

[34] Zitian Gao, Boye Niu, Xuzheng He, Haotian Xu, Hongzhang Liu, Aiwei Liu, Xuming Hu, and Lijie Wen. Interpretable contrastive monte carlo tree search reasoning, 2025.

[35] Yizhou Chi, Kevin Yang, and Dan Klein. ThoughtSculpt: Reasoning with intermediate revision and search. In Luis Chiruzzo, Alan Ritter, and Lu Wang, editors, *Proc. of the Findings of the Association for Computational Linguistics (NAACL)*, pages 7685–7711, Albuquerque, New Mexico, April 2025. Association for Computational Linguistics.

[36] Jiacheng Liu, Andrew Cohen, Ramakanth Pasunuru, Yejin Choi, Hannaneh Hajishirzi, and Asli Celikyilmaz. Don't throw away your value model! generating more preferable text with value-guided monte-carlo tree search decoding. In *Proc. of the First Conference on Language Modeling (CoLM)*, 2024.

[37] Wenlong Huang, Fei Xia, Ted Xiao, Harris Chan, Jacky Liang, Pete Florence, Andy Zeng, Jonathan Tompson, Igor Mordatch, Yevgen Chebotar, et al. Inner

monologue: Embodied reasoning through planning with language models. In *Proc. of the Conference on Robot Learning (CoRL)*, 2022.

[38] Michael Ahn, Anthony Brohan, Noah Brown, Yevgen Chebotar, Omar Cortes, Byron David, Chelsea Finn, Keerthana Gopalakrishnan, Karol Hausman, Alex Herzog, et al. Do as i can, not as i say: Grounding language in robotic affordances. *arXiv preprint arXiv:2204.01691*, 2022.

[39] Jacky Liang, Wenlong Huang, Fei Xia, Peng Xu, Karol Hausman, Brian Ichter, Pete Florence, and Andy Zeng. Code as policies: Language model programs for embodied control. In *Proc. of the 2023 IEEE International Conference on Robotics and Automation (ICRA)*, pages 9493–9500, 2023.

[40] Andrey Malinin and Mark Gales. Uncertainty estimation in autoregressive structured prediction. In *Proc. of the International Conference on Learning Representations (ICLR)*, 2021.

[41] Lorenz Kuhn, Yarin Gal, and Sebastian Farquhar. Semantic uncertainty: Linguistic invariances for uncertainty estimation in natural language generation. In *Proc. of the International Conference on Learning Representations (ICLR)*, 2023.

[42] Yijun Xiao and William Yang Wang. Quantifying uncertainties in natural language processing tasks. In *Proc. of the AAAI conference on artificial intelligence*, volume 33, pages 7322–7329, 2019.

[43] Yuxin Xiao, Paul Pu Liang, Umang Bhatt, Willie Neiswanger, Ruslan Salakhutdinov, and Louis-Philippe Morency. Uncertainty quantification with pre-trained language models: A large-scale empirical analysis. In *Proc. of the Empirical Methods in Natural Language Processing (EMNLP)*, 2022.

[44] Allen Z. Ren, Anushri Dixit, Alexandra Bodrova, Sumeet Singh, Stephen Tu, Noah Brown, Peng Xu, Leila Takayama, Fei Xia, Jake Varley, Zhenjia Xu, Dorsa Sadigh, Andy Zeng, and Anirudha Majumdar. Robots that ask for help: Uncertainty alignment for large language model planners. In *Proc. of the Conference of Robot Learning (CoRL)*, 2023.

[45] OpenAI. Gpt-4 technical report. 2023.

[46] Open X-Embodiment Collaboration. Open X-Embodiment: Robotic learning datasets and RT-X models. `https://arxiv.org/abs/2310.08864`, 2023.

[47] Anthony Brohan, Noah Brown, Justice Carbajal, Yevgen Chebotar, Joseph Dabis, Chelsea Finn, Keerthana Gopalakrishnan, Karol Hausman, Alex Herzog, Jasmine Hsu, et al. Rt-1: Robotics transformer for real-world control at scale. In *Proc. of the Robotics: Science and System (RSS)*, 2023.

[48] Brianna Zitkovich, Tianhe Yu, Sichun Xu, Peng Xu, Ted Xiao, Fei Xia, Jialin Wu, Paul Wohlhart, Stefan Welker, Ayzaan Wahid, Quan Vuong, Vincent Vanhoucke, Huong Tran, Radu Soricut, Anikait Singh, Jaspiar Singh, Pierre Sermanet, Pannag R. Sanketi, Grecia Salazar, Michael S. Ryoo, Krista Reymann, Kanishka Rao, Karl Pertsch, Igor Mordatch, Henryk Michalewski, Yao Lu, Sergey Levine, Lisa Lee, Tsang-Wei Edward Lee, Isabel Leal, Yuheng Kuang, Dmitry Kalashnikov, Ryan Julian, Nikhil J. Joshi, Alex Irpan, Brian Ichter, Jasmine Hsu, Alexander Herzog, Karol Hausman, Keerthana Gopalakrishnan, Chuyuan Fu, Pete Florence, Chelsea Finn, Kumar Avinava Dubey, Danny Driess, Tianli Ding, Krzysztof Marcin Choromanski, Xi Chen, Yevgen Chebotar, Justice Carbajal, Noah Brown, Anthony Brohan, Montserrat Gonzalez Arenas, and Kehang

Han. Rt-2: Vision-language-action models transfer web knowledge to robotic control. In *Proc. of The 7th Conference on Robot Learning (CoRL)*, pages 2165–2183, 2023.

[49] Octo Model Team, Dibya Ghosh, Homer Walke, Karl Pertsch, Kevin Black, Oier Mees, Sudeep Dasari, Joey Hejna, Charles Xu, Jianlan Luo, Tobias Kreiman, You Liang Tan, Lawrence Yunliang Chen, Pannag Sanketi, Quan Vuong, Ted Xiao, Dorsa Sadigh, Chelsea Finn, and Sergey Levine. Octo: An open-source generalist robot policy. In *Proc. of the Robotics: Science and Systems (RSS)*, Delft, Netherlands, 2024.

[50] Junjie Wen, Yichen Zhu, Minjie Zhu, Zhibin Tang, Jinming Li, Zhongyi Zhou, Xiaoyu Liu, Chaomin Shen, Yaxin Peng, and Feifei Feng. Diffusionvla: Scaling robot foundation models via unified diffusion and autoregression. In *Proc. of the Forty-second International Conference on Machine Learning (ICML)*, 2025.

[51] TRI LBM Team. A careful examination of large behavior models for multitask dexterous manipulation. 2025.

[52] Physical Intelligence. $\pi_{0.5}$: a vision-language-action model with open-world generalization, 2025.

[53] Yaron Lipman, Ricky T. Q. Chen, Heli Ben-Hamu, Maximilian Nickel, and Matthew Le. Flow matching for generative modeling. In *Proc. of the Eleventh International Conference on Learning Representations (ICLR)*, 2023.

[54] Tony Z Zhao, Vikash Kumar, Sergey Levine, and Chelsea Finn. Learning fine-grained bimanual manipulation with low-cost hardware. In *Proc. of the Robotics: Science and System (RSS)*, 2023.

[55] Michał Zawalski, William Chen, Karl Pertsch, Oier Mees, Chelsea Finn, and Sergey Levine. Robotic control via embodied chain-of-thought reasoning. In *Proc. of the 8th Annual Conference on Robot Learning (CoRL)*, 2024.

[56] Qingqing Zhao, Yao Lu, Moo Jin Kim, Zipeng Fu, Zhuoyang Zhang, Yecheng Wu, Zhaoshuo Li, Qianli Ma, Song Han, Chelsea Finn, et al. Cot-vla: Visual chain-of-thought reasoning for vision-language-action models. In *Proc. of the Computer Vision and Pattern Recognition Conference (CVPR)*, pages 1702–1713, 2025.

[57] Jinming Li, Yichen Zhu, Zhibin Tang, Junjie Wen, Minjie Zhu, Xiaoyu Liu, Chengmeng Li, Ran Cheng, Yaxin Peng, and Feifei Feng. Improving vision-language-action models via chain-of-affordance. *arXiv preprint arXiv:2412.20451*, 2024.

[58] Lucy Xiaoyang Shi, Brian Ichter, Michael Equi, Liyiming Ke, Karl Pertsch, Quan Vuong, James Tanner, Anna Walling, Haohuan Wang, Niccolo Fusai, et al. Hi robot: Open-ended instruction following with hierarchical vision-language-action models. *arXiv preprint arXiv:2502.19417*, 2025.

[59] Yi Li, Yuquan Deng, Jesse Zhang, Joel Jang, Marius Memmel, Caelan Reed Garrett, Fabio Ramos, Dieter Fox, Anqi Li, Abhishek Gupta, and Ankit Goyal. HAMSTER: Hierarchical action models for open-world robot manipulation. In *Proc. of the Thirteenth International Conference on Learning Representations (ICLR)*, 2025.

[60] Jason Lee, Jiafei Duan, Haoquan Fang, Yuquan Deng, Shuo Liu, Boyang Li, Bohan Fang, Jieyu Zhang, Yi Ru Wang, Sangho Lee, et al. Molmoact: Action reasoning models that can reason in space. *arXiv preprint arXiv:2508.07917*, 2025.

[61] Andrea Fox Aida Amini Siddarth Srinivasan Pedro Mota, Maike Paetzel and James Kennedy. Expressing coherent personality with incremental acquisition of multi-modal behaviors. In *Proc. of the IEEE International Symposium on Robot and Human Interactive Communication (RO-MAN)*, pages 396–403, 2018.

[62] Marieke Van Otterdijk, Heqiu Song, Konstantinos Tsiakas, Ilka Van Zeijl, and Emilia Barakova. Nonverbal cues expressing robot personality-a movement analysts perspective. In *Proc. of the IEEE International Conference on Robot and Human Interactive Communication (RO-MAN)*, pages 1181–1186. IEEE, 2022.

[63] Hadi Beik Mohammadi, Nikoletta Xirakia, Fares Abawi, Irina Barykina, Krishnan Chandran, Gitanjali Nair, Cuong Nguyen, Daniel Speck, Tayfun Alpay, Sascha Griffiths, et al. Designing a personality-driven robot for a human-robot interaction scenario. In *Proc. of the 2019 International Conference on Robotics and Automation (ICRA)*, pages 4317–4324. IEEE, 2019.

[64] Meriam Moujahid, David A Robb, Christian Dondrup, and Helen Hastie. Come closer: The effects of robot personality on human proxemics behaviours. In *Proc. of the IEEE International Conference on Robot and Human Interactive Communication (RO-MAN)*, pages 2610–2616. IEEE, 2023.

[65] Matthew KXJ Pan, Sungjoon Choi, James Kennedy, Kyna McIntosh, Daniel Campos Zamora, Günter Niemeyer, Joohyung Kim, Alexis Wieland, and David Christensen. Realistic and interactive robot gaze. In *Proc. of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 11072–11078. IEEE, 2020.

[66] Woo-Ri Ko, Minsu Jang, Jaeyeon Lee, and Jaehong Kim. Nonverbal social behavior generation for social robots using end-to-end learning. *The International Journal of Robotics Research*, 0(0):02783649231207974, 0.

[67] Heike Brock, Selma Sabanovic, Keisuke Nakamura, and Randy Gomez. Robust real-time hand gestural recognition for non-verbal communication with tabletop robot haru. In *Proc. of the IEEE international conference on robot and human interactive communication (RO-MAN)*, pages 891–898. IEEE, 2020.

[68] Guy Tevet, Sigal Raab, Brian Gordon, Yoni Shafir, Daniel Cohen-or, and Amit Haim Bermano. Human motion diffusion model. In *Proc. of the Eleventh International Conference on Learning Representations (ICLR)*, 2023.

[69] Mingyuan Zhang, Zhongang Cai, Liang Pan, Fangzhou Hong, Xinying Guo, Lei Yang, and Ziwei Liu. Motiondiffuse: Text-driven human motion generation with diffusion model. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2024.

[70] Chuan Guo, Xinxin Zuo, Sen Wang, and Li Cheng. TM2T: Stochastic and tokenized modeling for the reciprocal generation of 3d human motions and texts. In *Proc. of the European Conference on Computer Vision (ECCV)*, pages 580–597. Springer, 2022.

[71] Jianrong Zhang, Yangsong Zhang, Xiaodong Cun, Yong Zhang, Hongwei Zhao, Hongtao Lu, Xi Shen, and Ying Shan. Generating human motion from textual descriptions with discrete representations. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition (CVPR)*, pages 14730–14740, 2023.

[72] Chuan Guo, Yuxuan Mu, Muhammad Gohar Javed, Sen Wang, and Li Cheng. Momask: Generative masked modeling of 3d human motions. In *Proc. of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1900–1910, 2024.

[73] Kfir Aberman, Yijia Weng, Dani Lischinski, Daniel Cohen-Or, and Baoquan Chen. Unpaired motion style transfer from video to animation. *ACM Transactions on Graphics (TOG)*, 39(4):64–1, 2020.

[74] Chuan Guo, Yuxuan Mu, Xinxin Zuo, Peng Dai, Youliang Yan, Juwei Lu, and Li Cheng. Generative human motion stylization in latent space. In *Proc. of the Twelfth International Conference on Learning Representations (ICLR)*, 2024.

[75] Jihoon Kim, Jiseob Kim, and Sungjoon Choi. Flame: Free-form language-based motion synthesis & editing. In *Proc. of the AAAI Conference on Artificial Intelligence (AAAI)*, volume 37, pages 8255–8263, 2023.

[76] Purvi Goel, Kuan-Chieh Wang, C. Karen Liu, and Kayvon Fatahalian. Iterative motion editing with natural language. New York, NY, USA, 2024. Association for Computing Machinery.

[77] Nikos Athanasiou, Alpár Ceske, Markos Diomataris, Michael J. Black, and Gül Varol. MotionFix: Text-driven 3d human motion editing. In *SIGGRAPH Asia 2024 Conference Papers*, 2024.

[78] Biao Jiang, Xin Chen, Wen Liu, Jingyi Yu, Gang Yu, and Tao Chen. MotionGPT: Human motion as a foreign language. *Proc. of the Advances in Neural Information Processing Systems (NEURIPS)*, 36:20067–20079, 2023.

[79] Biao Jiang, Xin Chen, Chi Zhang, Fukun Yin, Zhuoyuan Li, Gang Yu, and Jiayuan Fan. MotionChain: Conversational motion controllers via multimodal prompts. *arXiv preprint arXiv:2404.01700*, 2024.

[80] Zixiang Zhou, Yu Wan, and Baoyuan Wang. AvatarGPT: All-in-one framework for motion understanding planning generation and beyond. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1357–1366, June 2024.

[81] Ling-Hao Chen, Shunlin Lu, Ailing Zeng, Hao Zhang, Benyou Wang, Ruimao Zhang, and Lei Zhang. Motionllm: Understanding human behaviors from human motions and videos. *arxiv:2405.20340*, 2024.

[82] Mingyuan Zhang, Daisheng Jin, Chenyang Gu, Fangzhou Hong, Zhongang Cai, Jingfang Huang, Chongzhi Zhang, Xinying Guo, Lei Yang, Ying He, et al. Large motion model for unified multi-modal motion generation. *arXiv preprint arXiv:2404.01284*, 2024.

[83] Mingshuang Luo, Ruibing Hou, Hong Chang, Zimo Liu, Yaowei Wang, and Shiguang Shan. M3 GPT: An advanced multimodal, multitask framework for motion comprehension and generation. *arXiv preprint arXiv:2405.16273*, 2024.

[84] Qi Wu, Yubo Zhao, Yifan Wang, Yu-Wing Tai, and Chi-Keung Tang. Motionllm: Multimodal motion-language learning with large language models. *arXiv preprint arXiv:2405.17013*, 2024.

[85] Evonne Ng, Donglai Xiang, Hanbyul Joo, and Kristen Grauman. You2me: Inferring body pose in egocentric video via first and second person interactions. In

Proc. of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), pages 9890–9900, 2020.

[86] Mihai Fieraru, Mihai Zanfir, Elisabeta Oneata, Alin-Ionut Popa, Vlad Olaru, and Cristian Sminchisescu. Three-dimensional reconstruction of human interactions. In *Proc. of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 7214–7223, 2020.

[87] Yifei Yin, Chen Guo, Manuel Kaufmann, Juan Jose Zarate, Jie Song, and Otmar Hilliges. Hi4d: 4d instance segmentation of close human interaction. In *Proc. of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 17016–17027, 2023.

[88] Liang Xu, Xintao Lv, Yichao Yan, Xin Jin, Shuwen Wu, Congsheng Xu, Yifan Liu, Yizhou Zhou, Fengyun Rao, Xingdong Sheng, et al. Inter-x: Towards versatile human-human interaction analysis. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 22260–22271, 2024.

[89] Han Liang, Wenqian Zhang, Wenxuan Li, Jingyi Yu, and Lan Xu. Intergen: Diffusion-based multi-human motion generation under complex interactions. *International Journal of Computer Vision*, pages 1–21, 2024.

[90] Anindita Ghosh, Rishabh Dabral, Vladislav Golyanik, Christian Theobalt, and Philipp Slusallek. ReMos: Reactive 3d motion synthesis for two-person interactions. *arXiv preprint arXiv:2311.17057*, 2023.

[91] Liang Xu, Yizhou Zhou, Yichao Yan, Xin Jin, Wenhan Zhu, Fengyun Rao, Xiaokang Yang, and Wenjun Zeng. ReGenNet: Towards human action-reaction syn-

thesis. In *Proc. of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1759–1769, June 2024.

[92] Yoni Shafir, Guy Tevet, Roy Kapon, and Amit Haim Bermano. Human motion diffusion as a generative prior. In *Proc. of the Twelfth International Conference on Learning Representations (ICLR)*, 2024.

[93] Shunyu Yao, Jeffrey Zhao, Dian Yu, Izhak Shafran, Karthik R Narasimhan, and Yuan Cao. React: Synergizing reasoning and acting in language models. In *NeurIPS 2022 Foundation Models for Decision Making Workshop*.

[94] Richard S Sutton, Doina Precup, and Satinder Singh. Between mdps and semi-mdps: A framework for temporal abstraction in reinforcement learning. *Artificial intelligence*, 112(1-2):181–211, 1999.

[95] Zeyi Liu, Arpit Bahety, and Shuran Song. REFLECT: Summarizing robot experiences for failure explanation and correction. In *Proc. of the 7th Annual Conference on Robot Learning*, 2023.

[96] Maximilian Diehl and Karinne Ramirez-Amaro. Why did i fail? a causal-based method to find explanations for robot failures. *IEEE Robotics and Automation Letters*, 7(4):8925–8932, 2022.

[97] Sungjoon Choi, Kyungjae Lee, and Songhwai Oh. Robust learning from demonstrations with mixed qualities using leveraged gaussian processes. *IEEE Transactions on Robotics*, 35(3):564–576, 2019.

[98] Suneel Belkhale, Yuchen Cui, and Dorsa Sadigh. Data quality in imitation learning. *Proc. of the Advances in neural information processing systems (NeurIPS)*, 36:80375–80395, 2023.

[99] Joey Hejna, Chethan Anand Bhateja, Yichen Jiang, Karl Pertsch, and Dorsa Sadigh. Remix: Optimizing data mixtures for large scale imitation learning. In *Proc. of the 8th Annual Conference on Robot Learning (CoRL)*, 2024.

[100] Richard S Sutton, Andrew G Barto, et al. *Introduction to reinforcement learning*, volume 135. MIT press Cambridge, 1998.

[101] Leslie Pack Kaelbling, Michael L Littman, and Andrew W Moore. Reinforcement learning: A survey. *Journal of artificial intelligence research*, 4:237–285, 1996.

[102] Daniel Kahneman. *Thinking, fast and slow*. macmillan, 2011.

[103] Zhe Ni, Xiaoxin Deng, Cong Tai, Xinyue Zhu, Qinghongbing Xie, Weihang Huang, Xiang Wu, and Long Zeng. Grid: Scene-graph-based instruction-driven robotic task planning. In *Proc. of the 2024 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 13765–13772. IEEE, 2024.

[104] Krishan Rana, Jesse Haviland, Sourav Garg, Jad Abou-Chakra, Ian Reid, and Niko Suenderhauf. Sayplan: Grounding large language models using 3d scene graphs for scalable robot task planning. In *Proc. of The 7th Conference on Robot Learning (CoRL)*, pages 23–72, 2023.

[105] Chuhao Liu, Zhijian Qiao, Jieqi Shi, Ke Wang, Peize Liu, and Shaojie Shen. Sg-reg: Generalizable and efficient scene graph registration. *IEEE Transactions on Robotics*, 41:3870–3889, 2025.

[106] Shilong Liu, Zhaoyang Zeng, Tianhe Ren, Feng Li, Hao Zhang, Jie Yang, Qing Jiang, Chunyuan Li, Jianwei Yang, Hang Su, et al. Grounding dino: Marrying dino with grounded pre-training for open-set object detection. In *Proc. of the European conference on computer vision (ECCV)*, pages 38–55. Springer, 2024.

[107] Kostas Margellos and John Lygeros. Hamilton–jacobi formulation for reach–avoid differential games. *IEEE Transactions on automatic control*, 56(8):1849–1861, 2011.

[108] Jaime F Fisac, Mo Chen, Claire J Tomlin, and S Shankar Sastry. Reach-avoid problems with time-varying dynamics, targets and constraints. In *Proc. of the 18th international conference on hybrid systems: computation and control*, pages 11–20, 2015.

[109] Mitchell Wortsman, Gabriel Ilharco, Samir Ya Gadre, Rebecca Roelofs, Raphael Gontijo-Lopes, Ari S Morcos, Hongseok Namkoong, Ali Farhadi, Yair Carmon, Simon Kornblith, et al. Model soups: averaging weights of multiple fine-tuned models improves accuracy without increasing inference time. In *Proc. of the International conference on machine learning (ICML)*, pages 23965–23998. PMLR, 2022.

[110] Edward J Hu, yelong shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. LoRA: Low-rank adaptation of large language models. In *Proc. of the International Conference on Learning Representations (ICLR)*, 2022.

[111] Ilya Kostrikov, Ashvin Nair, and Sergey Levine. Offline reinforcement learning with implicit q-learning. In *Proc. of the International Conference on Learning Representations (ICLR)*, 2022.

[112] Markus Freitag and Yaser Al-Onaizan. Beam search strategies for neural machine translation. In Thang Luong, Alexandra Birch, Graham Neubig, and Andrew Finch, editors, *Proc. of the First Workshop on Neural Machine Translation*, pages 56–60. Association for Computational Linguistics, August 2017.

[113] Moo Jin Kim, Chelsea Finn, and Percy Liang. Fine-tuning vision-language-action models: Optimizing speed and success. *arXiv preprint arXiv:2502.19645*, 2025.

[114] Hui Yuan, Kaixuan Huang, Chengzhuo Ni, Minshuo Chen, and Mengdi Wang. Reward-directed conditional diffusion: Provable distribution estimation and reward improvement. In *Proc. of the Thirty-seventh Conference on Neural Information Processing Systems (NeurIPS)*, 2023.

[115] Aaron Hurst, Adam Lerer, Adam P Goucher, Adam Perelman, Aditya Ramesh, Aidan Clark, AJ Ostrow, Akila Welihinda, Alan Hayes, Alec Radford, et al. Gpt-4o system card. *arXiv preprint arXiv:2410.21276*, 2024.

[116] Homer Rich Walke, Kevin Black, Tony Z. Zhao, Quan Vuong, Chongyi Zheng, Philippe Hansen-Estruch, Andre Wang He, Vivek Myers, Moo Jin Kim, Max Du, Abraham Lee, Kuan Fang, Chelsea Finn, and Sergey Levine. Bridgedata v2: A dataset for robot learning at scale. In *Proc. of the 7th Annual Conference on Robot Learning (CoRL)*, 2023.

[117] Xuanlin Li, Kyle Hsu, Jiayuan Gu, Oier Mees, Karl Pertsch, Homer Rich Walke, Chuyuan Fu, Ishikaa Lunawat, Isabel Sieh, Sean Kirmani, Sergey Levine, Jiajun

Wu, Chelsea Finn, Hao Su, Quan Vuong, and Ted Xiao. Evaluating real-world robot manipulation policies in simulation. In *Proc. of the 8th Annual Conference on Robot Learning (CoRL)*, 2024.

[118] Christopher Agia, Rohan Sinha, Jingyun Yang, Ziang Cao, Rika Antonova, Marco Pavone, and Jeannette Bohg. Unpacking failure modes of generative policies: Runtime monitoring of consistency and progress. In *Proc. of the 8th Annual Conference on Robot Learning (CoRL)*, 2024.

[119] Damai Dai, Yutao Sun, Li Dong, Yaru Hao, Shuming Ma, Zhifang Sui, and Furu Wei. Why can GPT learn in-context? language models secretly perform gradient descent as meta-optimizers. In Anna Rogers, Jordan Boyd-Graber, and Naoaki Okazaki, editors, *Findings of the Association for Computational Linguistics: ACL 2023*, pages 4005–4019, Toronto, Canada, July 2023. Association for Computational Linguistics.

[120] Charles Richter and Nicholas Roy. Safe visual navigation via deep learning and novelty detection. In *Proc. of the Robotics: Science and Systems Foundation (RSS)*, 2017.

[121] Marina Fomicheva, Shuo Sun, Lisa Yankovskaya, Frédéric Blain, Francisco Guzmán, Mark Fishel, Nikolaos Aletras, Vishrav Chaudhary, and Lucia Specia. Unsupervised quality estimation for neural machine translation. *Transactions of the Association for Computational Linguistics*, 8:539–555, 2020.

[122] Lorenz Kuhn, Yarin Gal, and Sebastian Farquhar. Clam: Selective clarification for ambiguous questions with large language models. *arXiv preprint arXiv:2212.07769*, 2022.

[123] Hugo Touvron, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timothée Lacroix, Baptiste Rozière, Naman Goyal, Eric Hambro, Faisal Azhar, Aurelien Rodriguez, Armand Joulin, Edouard Grave, and Guillaume Lample. Llama: Open and efficient foundation language models. 2023.

[124] Alisa Liu, Swabha Swayamdipta, Noah A. Smith, and Yejin Choi. WANLI: Worker and AI collaboration for natural language inference dataset creation. In *Findings of the Association for Computational Linguistics: EMNLP 2022*, pages 6826–6847, Abu Dhabi, United Arab Emirates, December 2022. Association for Computational Linguistics.

[125] Alisa Liu, Zhaofeng Wu, Julian Michael, Alane Suhr, Peter West, Alexander Koller, Swabha Swayamdipta, Noah A. Smith, and Yejin Choi. We're afraid language models aren't modeling ambiguity. 2023.

[126] Xiuye Gu, Tsung-Yi Lin, Weicheng Kuo, and Yin Cui. Open-vocabulary object detection via vision and language knowledge distillation. In *Proc. of the International Conference on Learning Representations (ICLR)*, 2021.

[127] Mohit Shridhar, Lucas Manuelli, and Dieter Fox. Cliport: What and where pathways for robotic manipulation. In *Proc. of the Conference on Robot Learning (CoRL)*, pages 894–906. PMLR, 2022.

[128] Matthias Minderer, Alexey Gritsenko, Austin Stone, Maxim Neumann, Dirk Weissenborn, Alexey Dosovitskiy, Aravindh Mahendran, Anurag Arnab, Mostafa Dehghani, Zhuoran Shen, et al. Simple open-vocabulary object detection with vision transformers. In *Proc. of the European Conference on Computer Vision (ECCV)*, 2022.

[129] Chenguang Huang, Oier Mees, Andy Zeng, and Wolfram Burgard. Visual language maps for robot navigation. *arXiv preprint arXiv:2210.05714*, 2022.

[130] Kunal Pratap Singh, Luca Weihs, Alvaro Herrasti, Aniruddha Kembhavi, and Roozbeh Mottaghi. Ask4help: Learning to leverage an expert for embodied tasks. In *Proc. of the Conference on Neural Information Processing Systems (NeurIPS)*, 2022.

[131] Yunfan Shao, Linyang Li, Junqi Dai, and Xipeng Qiu. Character-LLM: A trainable agent for role-playing. In *Proc. of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 13153–13187, Singapore, December 2023. Association for Computational Linguistics.

[132] Sungjoon Choi and Joohyung Kim. Towards a natural motion generator: A pipeline to control a humanoid based on motion data. In *2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 4373–4380. IEEE, 2019.

[133] Sungjoon Choi, Sanghoon Hong, Kyungjae Lee, and Sungbin Lim. Task agnostic robust learning on corrupt outputs by correlation-guided mixture density networks. In *Proc. of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 3872–3881, 2020.

[134] Robert R McCrae and Oliver P John. An introduction to the five-factor model and its applications. *Journal of personality*, 60(2):175–215, 1992.

[135] Maarten Selfhout, William Burk, Susan Branje, Jaap Denissen, Marcel Van Aken, and Wim Meeus. Emerging late adolescent friendship networks and big five per-

sonality traits: A social network approach. *Journal of personality*, 78(2):509–538, 2010.

[136] C. Dickens. *A Christmas Carol*. Bradbury and Evans, 1858.

[137] John Cohen Chris Meledandri and Janet Healy. *Despicable Me*. Universal Pictures (via Illumination), 2010.

[138] L.F. Baum. *The Wonderful Wizard of Oz*. Oz Series. G.M. Hill Company, 1900.

[139] Robot Wise. *Star Trek: The Motion Picture*. Paramount Global, 1979.

[140] Antonio Andriella, Henrique Siqueira, Di Fu, Sven Magg, Pablo Barros, Stefan Wermter, Carme Torras, and Guillem Alenya. Do i have a personality? endowing care robots with context-dependent personality traits. *International Journal of Social Robotics*, 13:2081–2102, 2021.

[141] William Tov, Ze Ling Nai, and Huey Woon Lee. Extraversion and agreeableness: Divergent routes to daily satisfaction with social relationships. *Journal of personality*, 84(1):121–134, 2016.

[142] Maike Paetzel-Prüsmann and James Kennedy. Improving a robot's turn-taking behavior in dynamic multiparty interactions. In *Companion of the 2023 ACM/IEEE International Conference on Human-Robot Interaction*, pages 411–415, 2023.

[143] Yuying Ge, Sijie Zhao, Jinguo Zhu, Yixiao Ge, Kun Yi, Lin Song, Chen Li, Xiaohan Ding, and Ying Shan. SEED-X: Multimodal models with unified multi-granularity comprehension and generation. *arXiv preprint arXiv:2404.14396*, 2024.

[144] Haotian Liu, Chunyuan Li, Qingyang Wu, and Yong Jae Lee. Visual instruction tuning. *Proc. of the Advances in neural information processing systems (NEURIPS)*, 36, 2024.

[145] Wei Chen, Lin Li, Yongqi Yang, Bin Wen, Fan Yang, Tingting Gao, Yu Wu, and Long Chen. CoMM: A coherent interleaved image-text dataset for multimodal understanding and generation. *arXiv preprint arXiv:2406.10462*, 2024.

[146] Changli Tang, Wenyi Yu, Guangzhi Sun, Xianzhao Chen, Tian Tan, Wei Li, Lu Lu, Zejun MA, and Chao Zhang. SALMONN: Towards generic hearing abilities for large language models. In *Proc. of the Twelfth International Conference on Learning Representations (ICLR)*, 2024.

[147] Yu Shu, Siwei Dong, Guangyao Chen, Wenhao Huang, Ruihua Zhang, Daochen Shi, Qiqi Xiang, and Yemin Shi. Llasm: Large language and speech model. *arXiv preprint arXiv:2308.15930*, 2023.

[148] Yaqi Zhang, Di Huang, Bin Liu, Shixiang Tang, Yan Lu, Lu Chen, Lei Bai, Qi Chu, Nenghai Yu, and Wanli Ouyang. Motiongpt: Finetuned llms are general-purpose motion generators. In *Proc. of the AAAI Conference on Artificial Intelligence (AAAI)*, volume 38, pages 7368–7376, 2024.

[149] Zhongang Cai, Jianping Jiang, Zhongfei Qing, Xinying Guo, Mingyuan Zhang, Zhengyu Lin, Haiyi Mei, Chen Wei, Ruisi Wang, Wanqi Yin, Liang Pan, Xiangyu Fan, Han Du, Peng Gao, Zhitao Yang, Yang Gao, Jiaqi Li, Tianxiang Ren, Yukun Wei, Xiaogang Wang, Chen Change Loy, Lei Yang, and Ziwei Liu. Digital life project: Autonomous 3d characters with social intelligence. In *Proc. of*

the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), pages 582–592, June 2024.

[150] OpenAI. Hello gpt-4o. 2024.

[151] Mathis Petrovich, Michael J. Black, and Gül Varol. TMR: Text-to-motion retrieval using contrastive 3D human motion synthesis. In *Proc. of the International Conference on Computer Vision (ICCV)*, 2023.

[152] Georgios Pavlakos, Vasileios Choutas, Nima Ghorbani, Timo Bolkart, Ahmed A. A. Osman, Dimitrios Tzionas, and Michael J. Black. Expressive body capture: 3D hands, face, and body from a single image. In *Proc.of the IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, pages 10975–10985, 2019.

[153] Doyup Lee, Chiheon Kim, Saehoon Kim, Minsu Cho, and Wook-Shin Han. Autoregressive image generation using residual quantization. In *Proc. of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 11523–11532, 2022.

[154] Jing Lin, Ailing Zeng, Shunlin Lu, Yuanhao Cai, Ruimao Zhang, Haoqian Wang, and Lei Zhang. Motion-X: A large-scale 3d expressive whole-body human motion dataset. *Proc. of the Advances in Neural Information Processing Systems (NEURIPS)*, 36, 2024.

[155] Satanjeev Banerjee and Alon Lavie. METEOR: An automatic metric for MT evaluation with improved correlation with human judgments. In *Proc. of the ACL Workshop on Intrinsic and Extrinsic Evaluation Measures for Machine Translation and/or Summarization*, pages 65–72, Ann Arbor, Michigan, June 2005. Association for Computational Linguistics.

[156] Krishna Pillutla, Swabha Swayamdipta, Rowan Zellers, John Thickstun, Sean Welleck, Yejin Choi, and Zaid Harchaoui. MAUVE: Measuring the gap between neural text and human text using divergence frontiers. In A. Beygelzimer, Y. Dauphin, P. Liang, and J. Wortman Vaughan, editors, *Proc. of the Advances in Neural Information Processing Systems (NEURIPS)*, 2021.

[157] Yarin Gal and Zoubin Ghahramani. Dropout as a bayesian approximation: Representing model uncertainty in deep learning. In Maria Florina Balcan and Kilian Q. Weinberger, editors, *Proc. of the 33rd International Conference on Machine Learning*, volume 48 of *Proceedings of Machine Learning Research*, pages 1050–1059, New York, New York, USA, 20–22 Jun 2016. PMLR.

[158] Mohit Shridhar, Xingdi Yuan, Marc-Alexandre Côté, Yonatan Bisk, Adam Trischler, and Matthew Hausknecht. ALFWorld: Aligning Text and Embodied Environments for Interactive Learning. In *Proc. of the International Conference on Learning Representations (ICLR)*, 2021.

[159] Ehsan Kamalloo, Aref Jafari, Xinyu Zhang, Nandan Thakur, and Jimmy Lin. HAGRID: A human-llm collaborative dataset for generative information-seeking with attribution. *arXiv:2307.16883*, 2023.

[160] Vitor Jeronymo, Luiz Bonifacio, Hugo Abonizio, Marzieh Fadaee, Roberto Lotufo, Jakub Zavrel, and Rodrigo Nogueira. Inpars-v2: Large language models as efficient dataset generators for information retrieval. *arXiv preprint arXiv:2301.01820*, 2023.

[161] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proc.*

of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers), pages 4171–4186, Minneapolis, Minnesota, June 2019. Association for Computational Linguistics.

[162] Stephanie Lin, Jacob Hilton, and Owain Evans. Truthfulqa: Measuring how models mimic human falsehoods. In *Proc. of the Annual Meeting of the Association for Computational Linguistics*, 2022.

[163] Hyung Won Chung, Le Hou, Shayne Longpre, Barret Zoph, Yi Tay, William Fedus, Yunxuan Li, Xuezhi Wang, Mostafa Dehghani, Siddhartha Brahma, et al. Scaling instruction-finetuned language models. *Journal of Machine Learning Research*, 25(70):1–53, 2024.

[164] AWS. Amazon mechanical turk.

Chapter A

Appendix 1: Classifying and Disambiguating User Commands

This appendix augments *CLARA* with concrete artifacts, datasets, and extended analyses. We first introduce additional theoretical background on LLM-based uncertainty estimation, including (i) in-context learning as implicit parameter updates and (ii) dropout as Bayesian variational approximation, followed by the formal statement and proof of our context-sampling theorem (§ A.1, § A.2). Next, we catalog representative failure modes and threshold effects in classification (§ A.3), followed by a toy block–picking example that makes the clear/ambiguous/infeasible split and uncertainty computation explicit (§ A.4). We then scale the analysis to a partially observable mobile-manipulation setting in ALFWorld, illustrating how uncertainty triggers disambiguation (§ A.5). Afterward, we describe the SaGC data construction and validation protocols (§ A.6) and provide post-validation examples (§ A.7). To contextualize performance, we report supervised baselines and a robot-type transfer study (§ A.8), compare vision–language backends for detection (§ A.9), and document our real-world setup, tasks, and qualitative/quantitative outcomes (§ A.10). Finally, we include additional results on partially feasible goals (§ A.11) and an external GPT–Info evaluation of question/reason informativeness (§ A.12).

A.1 Preliminaries on LLM-Based Uncertainty Estimation

In this section, we summarize the two theoretical components underlying our uncertainty estimation method: (i) the interpretation of in-context learning (ICL) as an *implicit parameter update* performed through transformer attention, and (ii) the view of dropout as a *Bayesian variational approximation*. These perspectives will later allow us to formalize context sampling in LLMs as a dropout-style posterior sampling procedure.

A.1.1 In-Context Learning as Implicit Parameter Update

We consider a decoder-only transformer language model $f(\cdot)$ with L self-attention layers. Following [119], the attention computation for a query token representation $q \in \mathbb{R}^d$ under a set of contextual tokens $\{x_i^{(\mathrm{ctx})}\}_{i=1}^n$ can be approximated under a linearized attention operator:

$$\mathrm{Attn}(q) \;\approx\; W_{\mathrm{zsl}}\, q \;+\; \sum_{i=1}^{n} (W_V x_i^{(\mathrm{ctx})})(W_K x_i^{(\mathrm{ctx})})^\top q, \tag{A.1}$$

where $W_K, W_V \in \mathbb{R}^{d\times d}$ are the key and value projection matrices and W_{zsl} denotes the contribution from zero-shot attention (i.e. without demonstrations).

Define the rank-one update matrices

$$U_i = (W_V x_i^{(\mathrm{ctx})})(W_K x_i^{(\mathrm{ctx})})^\top. \tag{A.2}$$

Eq. (A.1) can then be rewritten as

$$\mathrm{Attn}(q) \;=\; \left(W_{\mathrm{zsl}} + \sum_{i=1}^{n} U_i\right) q. \tag{A.3}$$

This implies that each demonstration token contributes a *deterministic rank-one parameter update* to the effective weight matrix governing the forward computation. Thus, selecting a subset $S \subseteq \{1,\ldots,n\}$ of demonstrations yields:

$$W(S) \;=\; W_{\text{zsl}} + \sum_{i \in S} U_i. \tag{A.4}$$

Hence, in-context learning can be understood as computing with a sample-dependent effective weight matrix $W(S)$, which is analogous to executing a forward pass of a fine-tuned model whose weights have been updated by the demos.

A.1.2 Dropout as Bayesian Variational Approximation

Dropout [157] can be interpreted as a tractable variational approximation to Bayesian inference in deep neural networks. Let W denote the weight matrix of a neural network layer. In dropout, a binary mask $z \in \{0,1\}^m$ is sampled with independent Bernoulli entries $z_j \sim \text{Bernoulli}(p_j)$, and the forward pass uses a masked version of the weights:

$$W(z) = W \odot Z, \tag{A.5}$$

where Z is a mask tensor broadcastable to the shape of W (e.g., copying z along rows or columns), and $\odot$ denotes elementwise multiplication.

Gal & Ghahramani [157] showed that the distribution $q(W) = \mathbb{P}(W(z))$ induced by dropout is equivalent to a variational posterior over weights of a deep Gaussian process. Each forward pass with an independently sampled mask $z^{(t)}$ produces a *function sample*

$$f^{(t)}(x) = f\big(x; W(z^{(t)})\big), \tag{A.6}$$

which is a draw from the approximate posterior predictive distribution $q(f \mid \mathscr{D})$.

Predictive uncertainty. Given T stochastic forward passes, the Monte Carlo predictive mean and covariance are

$$\hat{\mu}(x) = \frac{1}{T}\sum_{t=1}^{T} f^{(t)}(x), \tag{A.7}$$

$$\hat{\Sigma}(x) = \frac{1}{T}\sum_{t=1}^{T} \left(f^{(t)}(x) - \hat{\mu}(x)\right)\left(f^{(t)}(x) - \hat{\mu}(x)\right)^{\top}. \tag{A.8}$$

The covariance $\hat{\Sigma}(x)$ estimates the *epistemic uncertainty* arising from limited knowledge of the true model parameters. Thus, dropout transforms a deterministic model into a Bayesian model by sampling a distribution over weights and aggregating predictions across the corresponding sampled functions.

Key implication. The crucial observation—used later in our analysis—is that:

> *Dropout does not rely on explicit parameter distributions; Instead, predictive uncertainty arises entirely from sampling random low-rank updates to the model's effective weights.*

This shared structure directly parallels the ICL formulation in Section **??**, where each sampled context subset induces a different effective weight matrix. Therefore, if context sampling induces a distribution over update matrices of the form (A.5), then the resulting variability in LLM predictions corresponds to Bayesian predictive uncertainty.

A.2 Theorem and Proof

Variational Family. We explicitly define the variational family used in our analysis:

$$\mathscr{Q} = \left\{ W = W_0 + \sum_{k=1}^{K} \omega_k B_k \;\middle|\; \omega_k \sim \text{Bernoulli}(p_k) \right\}. \tag{A.9}$$

This family consists of all weight distributions expressible as a deterministic base parameter W_0 and a finite set of basis tensors $\{B_k\}_{k=1}^{K}$ independently selected by Bernoulli latent variables.

Theorem A.2.1 (ICL Context Sampling as Variational Bayesian Predictive Inference)**.** *Let $C = \{C_1, \ldots, C_m\}$ denote the demonstration set. In in-context learning, each demonstration C_j contributes a deterministic rank-one update U_j to the linearized effective attention parameters. A randomly sampled context subset $\mathbf{c}_i \sim p(C, \mathbf{x}^s)$ therefore induces*

$$W_i = W_{\text{zsl}} + \sum_{j=1}^{m} z_j^{(i)} U_j, \qquad z_j^{(i)} \sim \text{Bernoulli}(p_j). \tag{A.10}$$

Then the induced distribution $q_{\text{icl}}(W)$ lies in the variational family $\mathscr{Q}$ defined in Eq. (A.9). Since dropout induces a posterior in the same family, ICL context sampling produces Monte Carlo draws from an approximate Bayesian predictive distribution. Consequently, any dispersion measure over the embeddings $\mathbf{z}_i = g(f(x; W_i))$ (including our pairwise-distance estimator) is a valid estimator of epistemic predictive uncertainty.

Proof. **(1) Dropout induces a distribution inside $\mathscr{Q}$.** Dropout computes a masked parameter matrix

$$W(z) = W_{\text{base}} \odot Z, \qquad Z_\ell \sim \text{Bernoulli}(p_\ell), \tag{A.11}$$

where ℓ indexes the tensor entries. Let E_ℓ be the matrix with 1 at entry ℓ and zeros elsewhere, and define

$$B_\ell = (W_{\text{base}})_\ell E_\ell, \qquad W_0 = 0, \qquad \omega_\ell = Z_\ell.$$

Then

$$W(z) = \sum_\ell Z_\ell B_\ell = W_0 + \sum_\ell \omega_\ell B_\ell, \tag{A.12}$$

and therefore $q_{\text{drop}}(W) \in \mathcal{Q}$.

(2) ICL induces a distribution inside $\mathcal{Q}$. From Eq. (A.10),

$$W_i = W_{\text{zsl}} + \sum_{j=1}^{m} z_j^{(i)} U_j.$$

Identifying

$$W_0 = W_{\text{zsl}}, \qquad B_j = U_j, \qquad \omega_j = z_j^{(i)},$$

shows that $q_{\text{icl}}(W) \in \mathcal{Q}$.

(3) Consequence for predictive uncertainty. Since both dropout and ICL sample effective parameters from the same Bernoulli–additive variational family $\mathcal{Q}$, each draw W_i corresponds to a Monte Carlo sample from the approximate posterior predictive distribution $q(f \mid C, \mathbf{x}^s)$. The outputs $\mathbf{z}_i = g(f(x; W_i))$ are therefore Monte Carlo samples from the predictive distribution, and their dispersion estimates epistemic uncertainty. □

A.3 Failure Cases

Threshold The uncertainty, threshold is 1.24 in the cooking robot case, where the uncertainty is 1.22 which is slightly lower than the threshold. For the case under, the system classifies ambiguous commands into certain commands.

```
Goal: Cook something delicious for me.

Robot Type: Cooking robot

GT label: Ambiguous

robot action: robot.grab(bacon)

Uncertainty: 0.59

robot action: robot.heat(bacon)

Uncertainty: 0.0

robot action: robot.grab(bread)

Uncertainty: 0.82

robot action: robot.heat(bread

Uncertainty: 0.0

robot action: robot.plate(bacon)

Uncertainty: 1.22

=============

Pred label: Certain
```

Known keywords in the infeasible command If there is a known word in the goal in the infeasible commands, the large language models fail to predict it as an infeasible command. For example, in the case of a cooking robot, if the goal input contains words related to cooking, such as ”after cooking a meal”, the LLM predicts its goal as feasible.

Goal: Clean the kitchen after cooking a meal with bacon and banana

Robot Type: Cooking robot

GT label: infeasible

....

robot action: robot.plate(bacon)

Uncertainty: 1.49

😃F:Yes,therobotcancleanthekitchenaftercookingamealwithbaconandbanana.Therobotcan grabthebananaandbacon,heatthem,platethem,andthencleanthekitchenbygrabbingthewater, desk,andtableandwipingthemdown.

=============

Pred label: Ambiguous

In addition, even though the possible action set is ”clean” and ”wipe” for the cleaning robot, in the ”Organize the desk” commands, the LLM predicts that ”organize” is in the action set. This sufficient understanding of the action set leads to failure, which can be relaxed via more prompts or fine-tuning approaches.

Goal: Organize the desk

Robot Type: Ceaning robot

GT label: infeasible

robot action: robot.clean(desk)

Uncertainty: 0.93

😃F:Yes,therobotcanorganizethedesk.Therobotactionwouldbetherobot.organize(desk)

=============

Pred label: Ambiguous

Ambiguous commands with missing keywords Furthermore, the proposed methods sometimes fail on too vague commands with missing words. For example, ”give a massage in the kitchen” can be interpreted as ”give a massage to a person in the kitchen”. However, the proposed method predicts it is infeasible because of the missing word ”person” in this case.

Goal: Give a massage in the kitchen

Robot Type: Massage robot

GT label: Ambiguous

robot action: robot.go to(kitchen)

Uncertainty: 1.17

😃F:No,therobotcannotgiveamassageinthekitchenasitisnotpartofitsactionset.Therobotcanonly

”goto”and”givemassage”tothepersonwearingtheyellowshirt,whiteshirt,blackshirt,orbrownshirt.

=============

Pred label: infeasible

Lack of Information in Question In the scenario where the system has multiple missing information, the LLM fails to generate a question asking for both of the information.

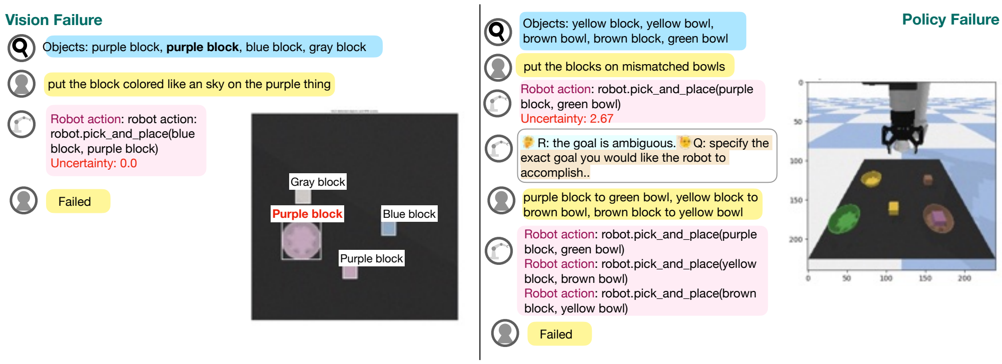


Figure A.1: Failure Cases due to other modules

Goal: Give someone something to drink

Objects = [coca cola can, fanta can, lemon, apple]

People = [person wearing black shirts, person wearing white shirts]

robot.pick_and_give(coca cola can, person wearing blue shirt)

Uncertainty: 1.75

F:Yes,Icangivesomeonesomethingtodrink.

R:Itisnotspecifiedwhichpersonshouldreceivethedrink.

Q:Specifywhichpersonshouldreceivethedrink

Malfunction in other modules Malfunctions on the vision module or robotic policy module can lead to wrong uncertainty quantification or overall failure. The examples are shown in Figure A.1.

A.4 Toy Example

Toy's example of a simple block-picking scenario is in Section III-B. In this example, the context sets $C = \{C_1, C_2, C_3, C_4\}$ are prompted as follows:

(C_1) task: pick a red block - scene: [red block, blue block, yellow block] - robot action: robot.pick(red block)

(C_2) task: pick a block colored like a banana - scene: [green block, blue block, yellow block] - robot action: robot.pick(yellow block)

(C_3) task: pick something that resembles a tree - scene: [green block, blue block, yellow block, red block] - robot action: robot.pick(green block)

(C_4) task: pick something that is not blue - scene: [blue block, purple block] - robot action: robot.pick(purple block)

where the goal is `"pick a block"` with observing `red block, blue block, yellow block` in the scene. We first sample the contexts from C (e.g., $\{C_1, C_3, C_4\}$) and shuffle the scene description list. Then, $\mathbf{c_1}$ will be as follows:

(C_1) task: pick a red block - scene: [red block, blue block, yellow block] - robot action: robot.pick(red block)

(C_3) task: pick something that resembles a tree - scene: [green block, blue block, yellow block, red block] - robot action: robot.pick(green block)

(C_4) task: pick something that is not blue - scene: [blue block, purple block] - robot action: robot.pick(purple block)

task: considering the ambiguity of the goal, pick a block

scene: [red block, yellow block, blue block]

Similarly, if we sample $\{C_3, C_2\}$ and shuffle the scene description list, $\mathbf{c_3}$ will be as follows:

(C_3) task: pick something that resembles a tree - scene: [green block, blue block, yellow block, red block] - robot action: robot.pick(green block)

(C_2) task: pick block colored like a banana - scene: [green block, blue block, yellow block] - robot action: robot.pick(yellow block) task: considering the ambiguity of the goal, pick a block

scene: [blue block, yellow block, red block]

We then sample three generations with different prompts set and shuffled the order of observation.

Sample 1. $\{C_!, C_3, C_4\}$

task: considering the ambiguity of the goal, pick a block scene: [red block, yellow block, blue block]

robot.pick(yellow block)

Sample 2. $\{C_2, C_4\}$

task: considering the ambiguity of the goal, pick a block scene: [red block, yellow block, blue block]

robot.pick(red block)

Sample 3. $\{C_3, C_2\}$

task: considering the ambiguity of the goal, pick a block scene: [blue block, yellow block, red block]

robot.pick(blue block)

In such case, $\mathbf{y}_1, \mathbf{y}_2, \mathbf{y}_3$ becomes `"yellow block"`, `"red block"`, and `"blue block"` respectively. The distance matrix in word embedding space will be $\begin{pmatrix} 0 & 2.5 & 1.7 \\ 2.5 & 0 & 2.3 \\ 1.7 & 2.3 & 0 \end{pmatrix}$, which results in total uncertainty as 2.1.

In the classification section, the prompts and the generated outcome of the sample example are as follows:

(Continue from previous prompts)

robot thought: I am a robot that can pick an object.

Considering the action set, pick, can I pick a block ? answer in yes or no

Yes, I can pick a block given more information.

With the generated answer, we used a heuristic parser to distinguish commands from infeasible and ambiguous based on the keyword (i.e., `yes`). If the generated sentence contains the keyword `"yes"`, we denote the corresponding command as ambiguous. In the toy example, the command is classified as ambiguous.

For the disambiguation progress, the prompts and generated outcome are as follows:

(Continue from previous prompts)

robot thought: this code is uncertain because the task does not specify any specific criteria for selecting the block

(robot thought: what can I ask to the user? question: Please provide more information about the criteria for selecting a block

The system provides the reason for the uncertainty and the following question for disambiguation. After obtaining the user's answer, the system returns to the uncertainty estimation step with extended prompts with the disambiguation process.

Sample 1. (C_4), (C_1) task: considering the ambiguity of the goal, pick a block scene: [red block, yellow block, blue block]

(Append Generated Explanation and Question)

user: red block

robot.pick(red block)

A.5 Demonstration in Mobile Manipulation Task

To analyze the proposed method in the larger environment under partial observability, we have demonstrated the proposed method in Alfworld [158], following the prompt baseline from [93]. The threshold for uncertainty is set as 1.5 in this environment.

In certain cases, the uncertainty remains lower than the threshold.

You are in the middle of a room. Looking quickly around you, you see a cabinet 6, a cabinet 5, a cabinet 4, a cabinet 3, a cabinet 2, a cabinet 1, a coffeemachine 1, a countertop 3, a countertop 2, a countertop 1, a drawer 3, a drawer 2, a drawer 1, a fridge 1, a garbagecan 1, a microwave 1, a shelf 3, a shelf 2, a shelf 1, a sinkbasin 1, a stoveburner 4, a stoveburner 3, a stoveburner 2, a stoveburner 1, and a toaster 1.

Your task is to: put a saltshaker in drawer 1.

Act 1: go to countertop 1

Uncertainty 1: 1.21

Obs 1: On the countertop 1, you see a creditcard 2, a dishsponge 2, and a mug 2.

Act 2: go to countertop 2

Uncertainty 2: 0.0

Obs 2: On the countertop 2, you see a creditcard 1, a cup 2, a knife 2, a knife 1, a lettuce 1, a papertowelroll 1, a soapbottle 2, a statue 1, a tomato 2, and a tomato 1.

Act 3: go to countertop 3

Uncertainty 3: 0.0

Obs 3: On the countertop 3, you see a bread 1, a butterknife 1, a houseplant 1, a potato 1, a spoon 1, and a vase 2.

Act 4: go to shelf 1

Uncertainty 4: 0.0

Obs 4: On the shelf 1, you see a mug 3, and a statue 2.

Act 5: go to shelf 2

Uncertainty 5: 0.0

Obs 5: On the shelf 2, you see a creditcard 3, and a dishsponge 3.

Act 6: go to shelf 3

Uncertainty 6: 0.0

Obs 6: On the shelf 3, you see a vase 1.

Act 7: go to cabinet 1

Uncertainty 7: 0.713

Obs 7: On the cabinet 1, you see a glassbottle 1. Act 8: go to cabinet 2

Uncertainty 8: 0.0

Obs 8: The cabinet 2 is closed.

Act 9: open cabinet 2

Uncertainty 9: 0.0

Obs 9: You open the cabinet 2. The cabinet 2 is open. In it, you see a plate 1, a saltshaker 1, and a soapbottle 1.

Act 10: take saltshaker 1 from cabinet 2

Uncertainty 10: 0.0

Obs 10: You pick up the saltshaker 1 from the cabinet 2.

Act 11: go to drawer 1

Uncertainty 11: 0.0

Obs 11: The drawer 1 is closed.

Act 12: open drawer 1

Uncertainty 12: 0.0

Obs 12: You open the drawer 1. The drawer 1 is open. In it, you see a fork 1, and a peppershaker 1.

Act 13: put saltshaker 1 in/on drawer 1

Uncertainty 13: 0.0

Obs 13: You put the saltshaker 1 in/on the drawer 1. =============

Pred label: certain

In the ambiguous scenario, where the goal is to `put something in the drawer 1`, the uncertainty incerases in the beginning, asking the user to clarify what something is. During proceeding with the task, the robot feels that it is stuck, and asks the user where the `saltshaker` is.

You are in the middle of a room. Looking quickly around you, you see a cabinet 6, a cabinet 5, a cabinet 4, a cabinet 3, a cabinet 2, a cabinet 1, a coffeemachine 1, a countertop 3, a countertop 2, a countertop 1, a drawer 3, a drawer 2, a drawer 1, a fridge 1, a garbagecan 1, a microwave 1, a shelf 3, a shelf 2, a shelf 1, a sinkbasin 1, a stoveburner 4, a stoveburner 3, a stoveburner 2, a stoveburner 1, and a toaster 1.
Your task is to: put something in the drawer 1
Act 1: go to cabinet 1
Uncertainty 1: 1.55
😃F:Yes. 🤔R:FirstIneedtofindsomethingIcanputinthedrawer1.
😟Q:Tellmewhatitemyouwouldliketoputinthedrawer1.
User: saltshaker
Act 2: go to countertop 1
Uncertainty 2: 0.647

Obs 2: On the countertop 1, you see a creditcard 2, a dishsponge 2, and a mug 2.

Act 3: go to countertop 2

Uncertainty 3: 2.14

R:Thesaltshakerisnothere.

Q:Tellmewherethesaltshakerislocated.

User: cabinent 2

Act 4: go to cabinet 2

Uncertainty 4: 0.0

Obs 4: The cabinet 2 is closed.

Act 5: open cabinet 2

Uncertainty 5: 0.0

Obs 5: You open the cabinet 2. The cabinet 2 is open. In it, you see a plate 1, a saltshaker 1, and a soapbottle 1.

Act 6: take saltshaker 1 from cabinet 2

Uncertainty 6: 0.0

Obs 6: You pick up the saltshaker 1 from the cabinet 2.

Act 7: go to drawer 1

Uncertainty 7: 0.0

Obs 7: The drawer 1 is closed.

Act 8: open drawer 1

Uncertainty 8: 0.0

Obs 8: You open the drawer 1. The drawer 1 is open. In it, you see a fork 1, and a peppershaker 1.

Act 9: put saltshaker 1 in/on drawer 1

Uncertainty 9: 0.0

Obs 9: You put the saltshaker 1 in/on the drawer 1.

=============

Pred label: ambiguous

In the infeasible scenario, the uncertainty becomes high in the beginning, and tells the user that the command is infeasible.

You are in the middle of a room. Looking quickly around you, you see a cabinet 6, a cabinet 5, a cabinet 4, a cabinet 3, a cabinet 2, a cabinet 1, a coffeemachine 1, a countertop 3, a countertop 2, a countertop 1, a drawer 3, a drawer 2, a drawer 1, a fridge 1, a garbagecan 1, a microwave 1, a shelf 3, a shelf 2, a shelf 1, a sinkbasin 1, a stoveburner 4, a stoveburner 3, a stoveburner 2, a stoveburner 1, and a toaster 1.

Your task is to: do laundry

Act 1: go to cabinet 1

Uncertinaty 1: 1.538

😃F:No.

=============

Pred label: infeasible

A.6 SaGC Data collections

The template to generate the dataset is as follows.

I am a [x] robot. Your possible action set is [y]. Here are the scene and the examples of the goal that [x] robot can do.

objects = [objects here]

floorplan = [floorplan here]

people = [people here]

Examples here

(Clear goal) create the high-level clear goal similar to the examples

(Ambiguous goal) create the creative goal that lacks the information to do the task but still can do.

(infeasible goal) create the simple goal that only [x] robot cannot do

An example prompt for cooking the robot in scene nine is as follows.

I am a cooking robot. Your possible action is grab, heat and plate. Here are the scene and the examples of the goal that robot can do.

objects = [coffee, table, bread, desk, buns, water, bacon, pan, banana, apple]

floorplan = [kitchen, office, meeting room, gym]

people = [person wearing gray shirt", person wearing white shirt, person wearing black shirt, person wearing brown shirt]

make breakfast consisted of bacon, bread, and coffee

I just worked out help me

cook a meal with apple and bread

(Clear goal) create the high-level clear goal similar to the examples

(Ambiguous goal) create the creative goal that lacks the information to do the task but still can do.

(Infeasible goal) create the simple goal that only cooking robot cannot do

The annotator is a large language model, and four validators were not blind to the research question. For annotation, we crafted 105 samples and then utilized `gpt-3.5-turbo` to generate the 5226 pairs of commands and the corresponding label. Utilizing LLMs to construct datasets is efficient yet powerful, which has been widely used in prior studies [159, 160]. The four validators were then asked to validate the corresponding generated pairs. The validators were given three options: discard the sample, change the label of the sample, or accept the sample. We provided guidelines to the validators, specifying that the term `ambiguous` pertains to scenarios where multiple answers could be applicable or when ambiguous keywords such as `or, any` are present. For the `infeasible scenario`, we instructed that it refers to situations either irrelevant to the type of robot or concerning the presence of an object in the scene. We have asked the validator to discard the sample if they can not decide. We observe that approximately 10% of the label changed after the validation. The dataset may have a bias as we have utilized only `gpt-3.5-turbo` model to generate all the pairs, while the effect of validators is not significant.

A.7 SaGC Data Examples

After validation, the SaGC dataset is composed as follows. Label 0, 1, 2 is certain, ambiguous, and infeasible respectively.

"scene": {
"floorplan": ["kitchen", "living room", "bedroom"],
"objects": ["water", "bacon", "bread", "pan", "coffee", "table", "orange juice", "sasuage", "banana", "apple"],
"people": ["person wearing blue shirt", "person wearing white shirt", "person wearing red shirt"]
},
"goal": "Prepare a meal consisting of bacon, toast and coffee and serve it on the table.",
"label": 0,
"task": "cooking"
==
"scene": {
"floorplan": ["kitchen", "living room", "bedroom", "bathroom"],
"objects": ["bacon", "bread", "pan", "coffee", "table", "orange juice", "sasuage"],
"people": ["person wearing yellow shirt", "person wearing white shirt", "person wearing black shirt"]
},
"goal": "Clean the living room and wipe the table in the kitchen.",
"label": 0,
==
"task": "cleaning"
"scene": {
"floorplan": ["kitchen", "living room", "bedroom"],
"objects": ["water", "bacon", "bread", "pan", "coffee", "table", "orange juice", "sasuage", "banana", "apple"],
"people": ["person wearing blue shirt", "person wearing white shirt", "person wearing red shirt"]
},

"goal": "Prepare and serve a meal or drink for the user.",
"label": 1,
"task": "cooking"
==
"scene": {
"floorplan": ["kitchen", "living room", "bedroom"],
"objects": ["water", "bacon", "bread", "pan", "coffee", "table", "orange juice", "sasuage", "banana", "apple"],
"people": ["person wearing blue shirt", "person wearing white shirt", "person wearing red shirt"]
}, "goal": "Person needs relaxation",
"label": 1,
"task": "massaging"
==
"scene": {
"floorplan": ["kitchen", "office", "desk for students", "workspace for robots"],
"objects": ["coffee", "table", "bread", "desk", "robot", "orange juice", "lemon", "salad"],
"people": ["person wearing yellow shirt", "person wearing white shirt", "person wearing black shirt", "person wearing blue shirt"]
},
"goal": "Teach a person wearing a yellow shirt how to cook a meal.",
"label": 2,
"task": "cleaning"
==

”scene”: {
”floorplan”: [”bedroom”, ”living room”, ”kitchen”, ”bathroom”],
”objects”: [”coffee”, ”table”, ”bread”, ”desk”, ”buns”, ”water”, ”bacon”, ”pan”, ”banana”, ”apple”
],
”people”: [”person wearing gray shirt”, ”person wearing white shirt”, ”person wearing black shirt”, ”person wearing blue shirt”]
},
”goal”: ”person wearing black shirt wants a glass of water”,
”label”: 2,
”task”: ”massaging”

A.8 Supervised Learning Baselines

We have conducted additional experiments on the supervised classification model (i.e., BERT [161]), as shown in Table A.1. First, we trained the BERT with 1000, 500, and 300 train samples and evaluated the rest of the samples of the dataset (a total of 5222 pairs). We observe that with more than 500 samples, the supervised learning method can exceed the proposed method. However, using the supervised learning method can be vulnerable when the configuration of the robot changes. To see the generalizability, we trained the BERT on two different robot types and then tested on the pairs from the test robot type. In this robot-type transfer scenario, the average accuracy was 0.333, which shows the significant gap of 0.367 compared to the proposed method on `text-danvici-003`.

A.9 Vision Language Model

We selected vision-language models that excel in the environment. In the pick-and-place simulation setting, constructed similarly to SayCan [38], we discovered that us-

	Method		Acc.
SL	1000-shots		0.926
	500-shots		0.716
	300-shots		0.521
Transfer	Train	Test	Acc.
	Clean + Cook	MAS.	0.455
	Cook + MAS.	CLEAN	0.298
	Clean + MAS.	COOK	0.245
	Average		0.333
Zero-shot	Ours (`text-danvici-003`)		0.710

Table A.1: Supervised Learning Baselines

ing ViLD [126], as done in the original paper, outperforms the use of OWL-ViT [128]. However, the trend reverses in real-world scenarios. Figure A.2 provides a comparison, illustrating the distinctions between the two vision-language models.

A.10 Real-world Experiments Environment and Demonstrations

Environments The setup consists of a UR5e robot arm equipped with a wrist-mounted OpenCV OAKD camera overlooking a workspace of the tabletop. We run our method via Robot Operating System (ROS) and the rate of publishing the joint position is set at 500HZ which is a default setting for a UR5e manipulator. In the table scene, we randomly place objects on the table and shuffle them throughout the experiments. The setup is shown in Figure A.4.

We investigate three tasks considering goal information: (i) Certain cases involve the explicit provision of information regarding the location or name of the target object. (ii)

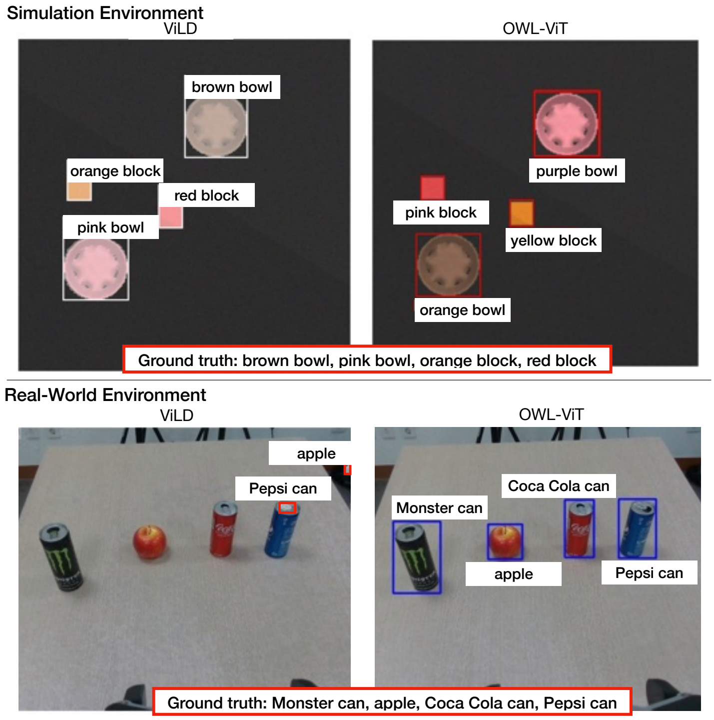


Figure A.2: Comparison in Vision-Language Models

Ambiguous cases, in contrast, are characterized by the absence of detailed information about the target object. (iii) Infeasible cases arise when objects or actions are encountered that have not been previously observed within the given context prompts. The goals are illustrated in Table A.2.

User Guidelines A high-level goal was predefined during the experiment, as shown in Table V in the manuscript. There were four participants in each scene, and they were asked to answer the question from the robot during the disambiguation process. The instructions given to the participants are as follows:

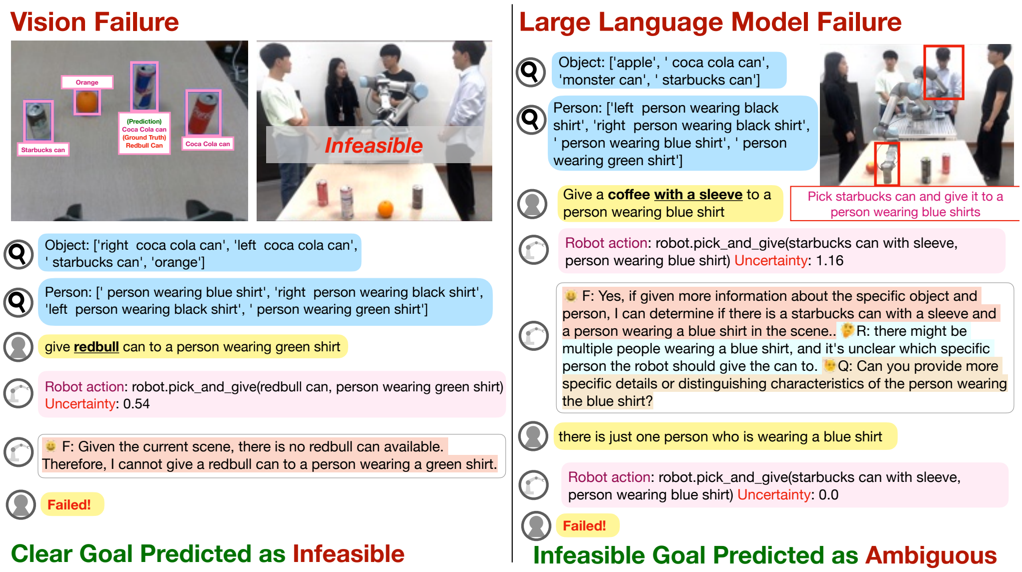


Figure A.3: Failure Cases in Real-world Environment

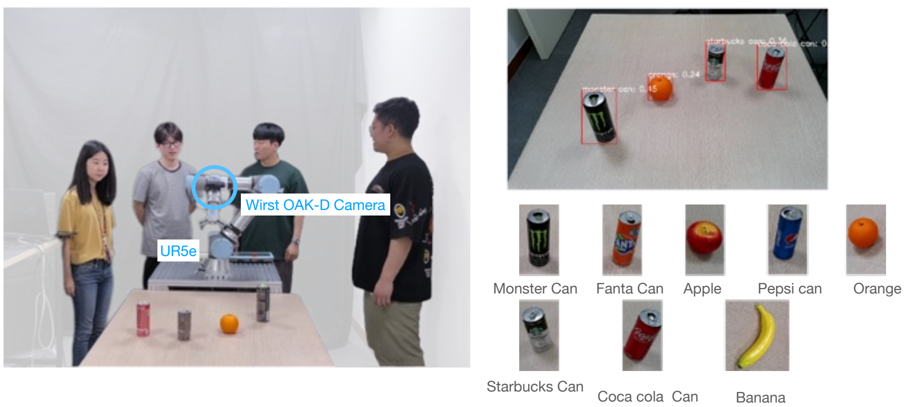


Figure A.4: Real-world Experiment Setting

Given the goal, the robot may ask you some questions if it is uncertain about what to do. If the robot asks for the information, please answer it via the interface and as detailed as possible, letting the robot be certain about the task.

An example of the response is as follows:

Scene: [coca cola can, pepsi can, apple, lemon]

Q. Please specify what kind of drink you would like.

A. Coca cola can

Categories	Tasks
Clear	give [x] to [y]
Ambiguous	give [x] to someone give something to drink to [y] give something to drink to someone
Infeasible	Wipe the desk Smash the [x] put [x] on the ground

Table A.2: Task explanation for the real-world experiment.

Failure Cases The failure can arise due to both VLM and LLM. We have shown the failure cases in Figure A.3. On the vision failure part, the VLM fails to detect the Red Bull can, leading to predict `"give red bull can to a person wearing green shirt"` as in feasible command. In the LLM Failure part, LLM failed to understand that `coffee with sleeve` is infeasible in the scene and picks the Starbucks can instead.

Detailed Results The detailed results on disambiguation in the real world are shown in Table A.3. We report the success rate of both before and after the interaction, along with a portion of question generation for each label.

The snapshots for real-world demonstrations are shown in Figure A.5 and A.6.

A.11 Additional Results on SaGC Dataset

The dataset contains partially feasible goals, such as `"bake buns and repair the desk"` for a cooking robot, `"cook breakfast and clean the kitchen` for a cleaning room, and `"make coffee and give a massage to the person wearing a gray shirt"` for a massage robot. For additional 216 data, we have validated the proposed

Model	Method	Before	After	Gap	C. Por.	A. Por.	I. Por.
GPT3.5	IM [37]	0.38	0.55	0.17	0.83	0.83	0.17
	CLAM† [122]	0.33	0.61	0.28	0.66	1.0	0.33
	Ours	0.22	0.55	0.33	0.33	1.0	0.33
IntructGPT	IM [37]	0.27	0.44	0.17	0.33	0.66	0.0
	CLAM† [122]	0.33	0.39	0.06	0.33	0.33	0.16
	Ours	0.38	0.67	0.28	0.17	0.83	0.0

Table A.3: Success Rates on real-world environment. C. Por., A. Por., I. Por. denotes a portion of question generation on clear, ambiguous, and infeasible commands respectively.

method and observed that the proposed method on `text-davinci-003` model has the 0.87 accuracy in those samples. Although the large-language model has hallucination issues, ensuring the model to be aware of their capabilities and roles helps in reducing those issues. The generated outputs in such samples are illustrated in Figure A.7, with the failure case.

A.12 GPT-Info Evaluation Examples

	LLAMA [123]	GPT3.5	IntructGPT
IM [37]	0.70	0.72	0.79
CLAM†	0.37	0.83	0.82
Ours	0.39	0.93	0.89

Table A.4: Informativeness of generated questions and reasons

We assessed the informativeness of the generated questions and explanations with a fine-tuned GPT from TrustfulAI [162]. We measure how many questions and reasons generated are informative with respect to the total number of generations. The results are shown in Table A.4. We observe that the proposed method with GPT3.5-turbo generates the most informative questions, and the zero-shot based method (Ours, CLAM† [41]) had better quality in average than the few-shot based method (IM [37]). We believe that leveraging zero-shot performance in question generation contributes to better quality and variety in the resulting questions, as opposed to a one-shot approach.

```
goal: hide purple block
question: provide more information about the goal of the task.
info: false
=======
goal: put the blocks in their matching bowls
question: specify which bowl the blue block should go in.
info: true
=======
goal: put the blocks on mismatched bowls
question: specify the exact goal you would like the robot to accomplish.
info: false
=======
goal: put the blocks on mismatched bowls
question: provide more information about the task.
info: false
======= goal: put all the blocks in different corners
question: specify which corner you would like the red block to be placed in.
info: true
======= goal: put a blue block in a bowl that user wants.
question: specify which bowl you would like the blue block to be placed in.
info: true
=======
```

Examples of the evaluation by GPT-info from Truthful QA [162] are shown in the box.

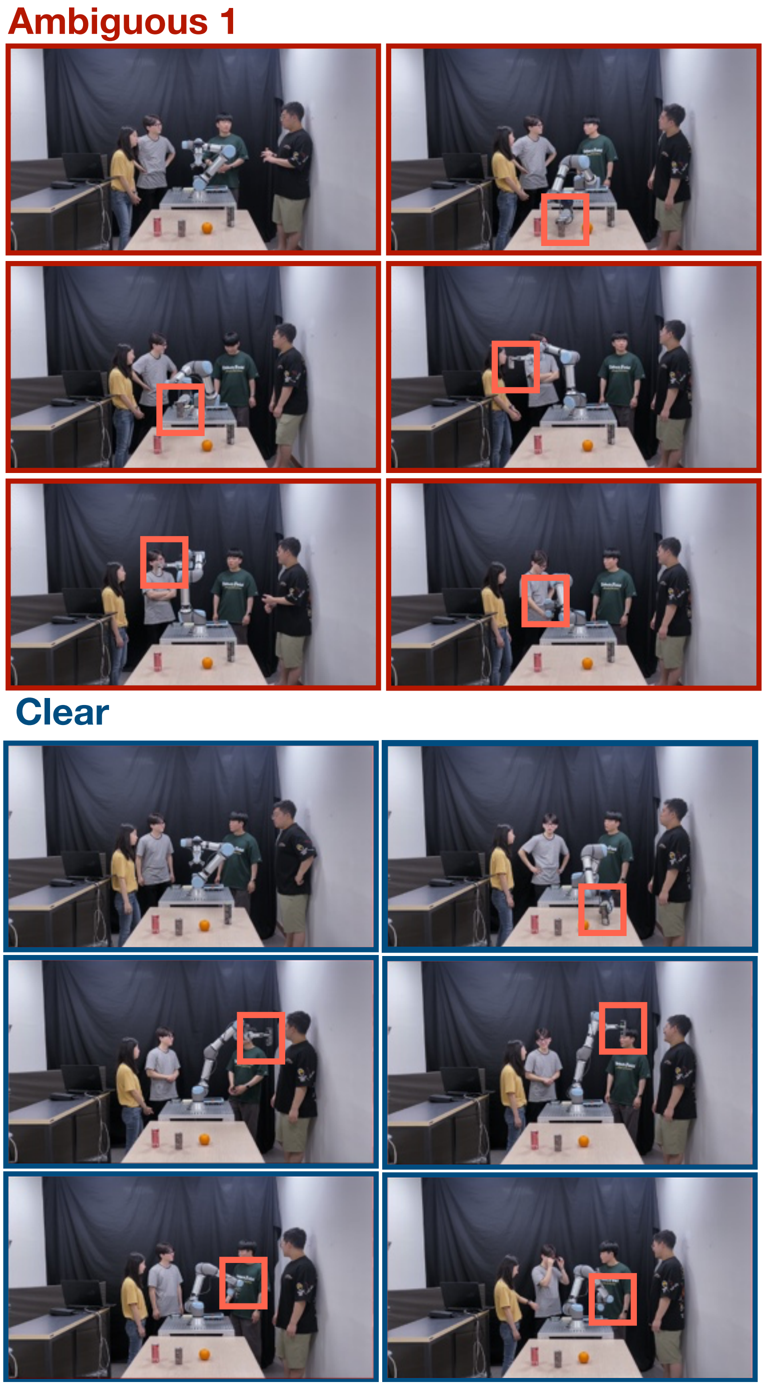


Figure A.5: Snapshot1

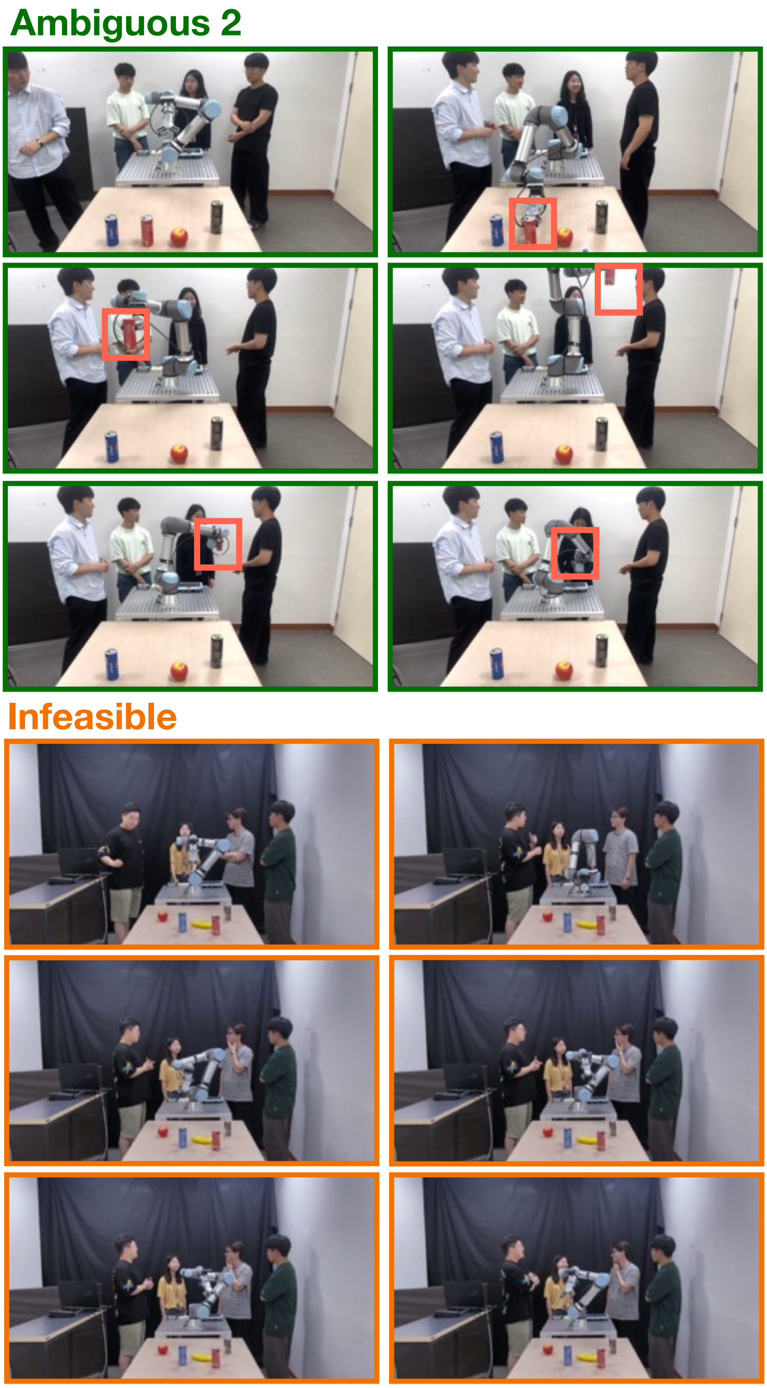


Figure A.6: Snapshot2

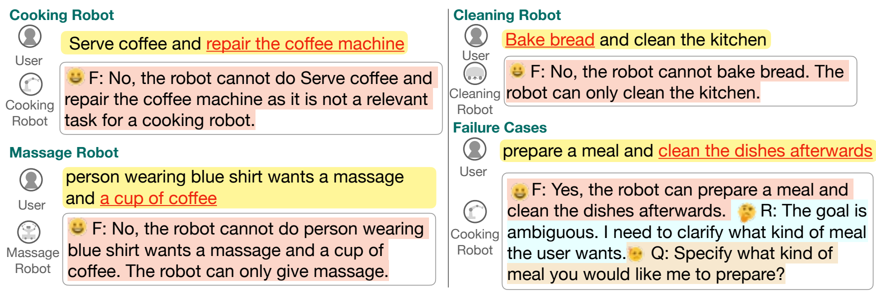


Figure A.7: Generated output on the additional samples

Chapter B

Appendix 2: Training Vision–Language–Action Model with Mixed Quality Dataset

B.0.1 Detailed Formulation

Node-edge kernel induced by low-level options

We define the high-level node-edge space from the low-level MDP $(\mathscr{S}, \mathscr{A}, T)$ and show that it forms an MDP, and we introduce the node-level target/avoid sets corresponding to $\mathscr{G}, \mathscr{F} \subset \mathscr{S}$.

Given the option grounding $o_e = (I_e^N, \pi_e, \beta_e, \Pi_e)$, the closed-loop one-step kernel under o_e is

$$P_{\pi_e}(s' \mid s) = \sum_{a \in \mathscr{A}} \pi_e(a \mid s, \ell)\, T(s' \mid s, a).$$

With termination-on-arrival, the terminal low-level state distribution is

$$P^S_{o_e}(s^\star \mid s) = \sum_{t \geq 0} \big((1-\beta_e) P_{\pi_e}\big)^t \, (\beta_e P_{\pi_e})(s^\star \mid s),$$

i.e., the probability that, starting from s and executing o_e, the execution eventually terminates in $s^\star$ under β_e.

Lift the high-level pair (n, z) to a low-level *arrival* distribution $\mu(\cdot \mid n, z)$, the conditional law of s upon arriving at node n after the option sequence encoded in z (in data, $\mu = \delta_{s_{t_k}}$ with $\phi(s_{t_k}) = n$). Mapping terminal states to nodes via $\Pi_e : \mathscr{S} \to \mathscr{N}$ yields the *node-level option kernel with context*

$$P^N_{o_e}(n' \mid n, z) = \mathbb{E}_{s \sim \mu(\cdot \mid n, z)} \Big[\sum_{s^\star} P^S_{o_e}(s^\star \mid s)\, \mathbf{1}\{\Pi_e(s^\star) = n'\} \Big].$$

With the deterministic context update $z' = [z, e, n]$, the *node-edge* kernel is

$$P\big((n', z') \mid (n, z), e\big) = P^N_{o_e}(n' \mid n, z)\, \mathbf{1}\{\, z' = [z, e, n]\,\}.$$

Define the node-level target/avoid sets by lifting through ϕ:

$$\mathscr{G}_N := \phi(\mathscr{G}), \qquad \mathscr{F}_N := \phi(\mathscr{F}), \qquad \mathscr{G}_N \cap \mathscr{F}_N = \varnothing.$$

The kernel is absorbing on $\mathscr{G}_N \cup \mathscr{F}_N$. Therefore $(\mathscr{X}, \mathscr{U}, P)$ with $\mathscr{X} = \mathscr{N} \times \mathscr{Z}$, $\mathscr{U} = \mathscr{E}$, and transition kernel P is a well-defined MDP on the node-edge space.

Value equals reachability probability (undiscounted first exit)

Define decision-epoch hitting times

$$\tau_{\mathscr{G}} := \inf\{t \geq 0 : \ n_t \in \mathscr{G}_N\},$$

$$\tau_{\mathscr{F}} := \inf\{t \geq 0 : \ n_t \in \mathscr{F}_N\},$$

$$\tau := \tau_{\mathscr{G}} \wedge \tau_{\mathscr{F}}. \quad \text{(B.1)}$$

With the entrance reward $r(n,z,e,n') = \mathbf{1}\{n' \in \mathscr{G}_N\}$ and *undiscounted* first–exit return

$$R := \sum_{t=0}^{\tau-1} r(n_t, z_t, e_t, n_{t+1}),$$

we have the pathwise identity

$$R = \mathbf{1}\{\tau_{\mathscr{G}} < \tau_{\mathscr{F}}\}.$$

Indeed, exactly one nonzero reward can occur—on the unique transition that first enters $\mathscr{G}_N$ (if any); if $\mathscr{F}_N$ is hit first, all rewards are zero and the episode terminates.

For any policy μ on the node–edge MDP $(\mathscr{X}, \mathscr{U}, P)$ with $\mathscr{X} = \mathscr{N} \times \mathscr{Z}$, $\mathscr{U} = \mathscr{E}$, and kernel $P((n',z') \mid (n,z), e)$,

$$\begin{aligned} V^{\mu}(n,z) &:= \mathbb{E}_{\mu}\left[\sum_{t=0}^{\tau-1} r(n_t, z_t, e_t, n_{t+1}) \,\middle|\, (n_0, z_0) = (n,z) \right] \\ &= \Pr_{\mu}\big(\tau_{\mathscr{G}} < \tau_{\mathscr{F}} \mid n,z\big). \end{aligned} \tag{B.2}$$

The corresponding Bellman relations are the standard reach-avoid equations:

$$\begin{aligned} V^{\mu}(n,z) = \sum_{e} \mu(e \mid n,z) \sum_{n'} P^{N}_{o_e}(n' \mid n,z) \Big[& \mathbf{1}\{n' \in \mathscr{G}_N\} \\ & + \mathbf{1}\{n' \notin \mathscr{G}_N \cup \mathscr{F}_N\} V^{\mu}(n', [z,e,n]) \Big], \end{aligned} \tag{B.3}$$

with boundary conditions $V^{\mu}(n,z) = 1$ for $n \in \mathscr{G}_N$ and $V^{\mu}(n,z) = 0$ for $n \in \mathscr{F}_N$. The optimality version replaces the inner expectation by a maximization over $e \in \mathscr{O}(n,z)$. If a discount factor $\gamma \in (0,1)$ is preferred, define $R_\gamma = \sum_{t=0}^{\tau-1} \gamma^t r(n_t, z_t, e_t, n_{t+1})$ and note that pathwise $R_\gamma = \gamma^{\tau_{\mathscr{G}}-1}\mathbf{1}\{\tau_{\mathscr{G}} < \tau_{\mathscr{F}}\}$ so

$$V_\gamma^\mu(n,z) = \mathbb{E}_\mu[R_\gamma \mid n,z]$$
$$= \Pr_\mu(\tau_{\mathscr{G}} < \tau_{\mathscr{F}} \mid n,z)\,\mathbb{E}_\mu\big[\gamma^{\tau_{\mathscr{G}}-1} \mid \tau_{\mathscr{G}} < \tau_{\mathscr{F}}, n, z\big]. \quad \text{(B.4)}$$

The discounted value is therefore proportional to the success probability with a positive factor depending on the success time; when search evaluates leaves at an approximately fixed depth this factor is effectively constant so ranking by V_γ^μ coincides with ranking by reachability, and if a probability scale is required one may normalize by depth or divide by an empirical estimate of the factor computed from the dataset.

Expectile losses on the node–edge MDP

Let $\mathscr{D} = \{(n_k, z_{0:k-1}, e_k, n_{k+1})\}$ be logged transitions from a behavior policy $\mu_b(e \mid n,z)$. Define the entrance reward $r_k = \mathbf{1}\{n_{k+1} \in \mathscr{G}_N\}$ and the terminal flag $\mathrm{term}_k = \mathbf{1}\{n_{k+1} \in \mathscr{G}_N \cup \mathscr{F}_N\}$. Write θ for trainable parameters and θ' for an EMA target.

From the Bellman equation under μ_b,

$$V^{\mu_b}(n_k, z_{0:k-1}) = \mathbb{E}[r_k + \gamma(1-\mathrm{term}_k)\,V^{\mu_b}(n_{k+1}, z_{0:k})]\,,$$

we form the one-step target

$$y_k \;=\; r_k \;+\; \gamma(1-\mathrm{term}_k)\,V_{\theta'}(n_{k+1}, z_{0:k}).$$

We then minimize the expectile–TD loss

$$\mathscr{L}_{\text{val}}(\theta) = \mathbb{E}_{\mathscr{D}}\left[\rho_{\tau_e}\big(y_k - V_\theta(n_k, z_{0:k-1})\big)\right],$$

$$\rho_\tau(u) = \big|\tau - \mathbf{1}\{u < 0\}\big|\, u^2. \quad \text{(B.5)}$$

While original TD with symmetric MSE is tailored to online RL where fresh on-policy samples expand coverage and the Bellman contraction drives V_θ toward the true value as data grows, our setting is offline. We train on a fixed batch, so TD error is reduced only on the logged support, and bootstrap propagation outside that support can induce overestimation on unseen actions or edges. To guard against this, we adopt the expectile TD loss with a pessimistic level $\tau_e < 0.5$, which penalizes positive residuals $V_\theta - y_k$ more than negative ones and yields a conservative value on under-supported regions. The objective remains a simple and stable TD regression that avoids importance weighting. With $\gamma = 1$ and first-exit rewards, $V_\theta(n,z)$ approximates $\Pr(\tau_{\mathscr{G}} < \tau_{\mathscr{F}} \mid n,z)$, and τ_e provides a clean knob to trade off recall and caution during planning under offline data constraints.

B.0.2 Dataset Details

Plug Insertion Environment

Collection. We used teleoperation with four operators to cover *all admissible insertion orders* for each receptacle type (e.g., all permutations for 3–socket plates and for 2–socket plates). The insertion order for strips is fixed from right to left due to limited viewpoints. Operators received order guidelines beforehand so each trajectory explicitly followed a target sequence (e.g., `small` → `round` → `rectangular`). For every run we logged time–

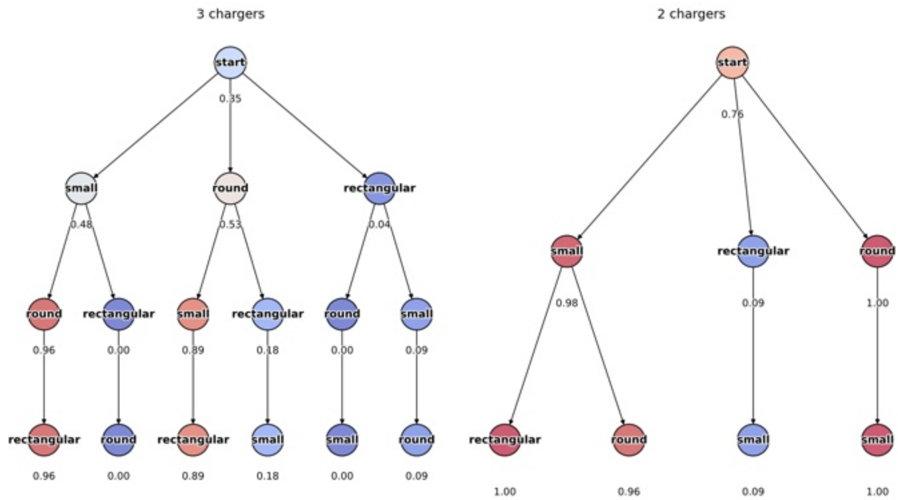


Figure B.1: Data Statistics from human demonstrations in plug insertion environment

stamped robot state (arm pose, gripper command), RGB images, and camera calibration.

Subgoal labels. From the logs we derive a minimal set of discrete subgoals tied to gripper state transitions and the target order:

- **Grasp-*x*:** `open` $\rightarrow$ `close` near plug *x*,
- **Insert-*x* in *y*:** `close` $\rightarrow$ `open` at socket *y*.

A trajectory thus becomes a chain of subgoals aligned to the instructed order (e.g., `grasp-small charger` $\rightarrow$ `insert-small charger in rightmost plug` $\rightarrow$ `grasp-round charger` $\rightarrow \cdots$).

Scene snapshots. Around each transition time, we extract a short image window and select the central frame as the *scene* for that subgoal.

Scene graph construction. For every scene, we build a 2D scene graph containing:

- **Gripper node:** 2D image location obtained by projecting the measured 3D gripper pose into the camera frame using known intrinsics/extrinsics.
- **Object nodes:** bounding–box proposals from *GroundingDINO [106]*; a VLM (*Gemini [2]*) assigns semantic names (e.g., `small`, `round`, `rectangular`) to crops via text prompts seeded by the task vocabulary.
- **Relations (edges):** coarse spatial relations between nodes (e.g., charger inside leftmost plug), inferred from box geometry and, when ambiguous, VLM judgments.

Output triplets. Each subgoal yields a {image, scene-graph, label} triple: the raw image, a graph with the gripper 2D position, object boxes, relations, and the symbolic subgoal induced by the gripper transition and the instructed order.

Data Statistics Figure B.1 visualizes the empirical *per–path success rate* for the plug–insertion task from human demonstrations. Each tree enumerates all candidate insertion paths (from the root `start` to a leaf), where a leaf corresponds to a *complete* order of socket types (`small`, `round`, `rectangular`). The number under each leaf is the estimated success probability $\widehat{p}_{\text{succ}} = \sharp\text{successes}/\sharp\text{trials}$ for that path, and node color encodes this value (red $\rightarrow$ high, blue $\rightarrow$ low). The left panel reports the 3-charger setting and the right panel the 2-charger setting. The distribution is skewed: in 3–3-charger, sequences such as `small` $\rightarrow$ `round` $\rightarrow$ `rectangular` are highly reliable (0.96), while any path involving `rectangular` early tends to fail (0.00–0.18). In a 2-charger, several paths are near–perfect (e.g., `round` $\rightarrow$ `small` and `small` $\rightarrow$ `rectangular` both at 1.00), whereas `rectangular` $\rightarrow$ `small` is weak (0.09).

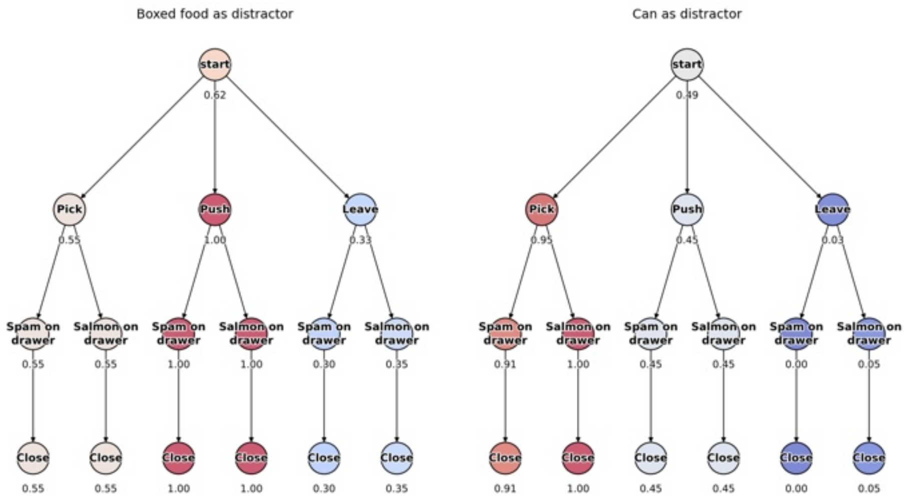


Figure B.2: Data Statistics from human demonstrations in the drawer packing environment

Drawer Packing Environment

Collection. We used teleoperation with a leader–follower arm setup to collect demonstrations of drawer packing. Four operators performed trajectories covering all admissible object–placement orders across different categories (`boxed food`, `cans`, etc.). For each run, the teleoperator followed a target guideline (e.g., `pick box` → `put on table` → `grasp canned spam` → `put in drawer`) ensuring coverage of *Pick*, *Push*, and *Leave* strategies. The initial scene includes two types of distractors—`boxed food` items and `cans` placed on (or near) the drawer surface. The task, conditioned on the language instruction, is to select the specified can (`spam` or `salmon`), place it inside the drawer, and then close the drawer. The dataset logs include time–stamped robot states (joint angles, gripper signal), RGB observations, object poses from the MuJoCo simulator, and synchronized language instructions.

Subgoal labels. From the logs we derive a minimal set of discrete subgoals aligned with gripper state transitions and the target order. Each trajectory is segmented into a chain of symbolic steps:

- **Pick-*x*:** grasp and lift a distractor object *x* from the table,
- **Push-*x*:** displace object *x* away to clear space,
- **Grasp-*y*:** grasp the target food item *y* (e.g., canned spam, salmon),
- **Put-*y* on drawer:** place *y* into the top drawer,
- **Close drawer:** close the top drawer with the gripper.

Thus, each demonstration becomes an ordered sequence of subgoals (e.g., `push box` → `grasp spam` → `put spam on drawer` → `close drawer`), which provides the symbolic scaffold for scene–graph based reasoning.

Scene graphs are constructed in the same method as a plug insertion environment.

Data Statistics Figure B.2 illustrates the empirical *per–path success rate* in the drawer–packing task. Each tree expands all candidate sequences from the root `start` to a leaf, where a leaf specifies a complete decision (e.g., placing `Spam` or `Salmon`). The numeric value under each node is the estimated success probability $\widehat{p}_{\text{succ}} = \sharp\text{successes}/\sharp\text{trials}$, and the node color encodes this value (red → high, blue → low).

In the `boxed food` setting (left), the distribution is uneven: `Push` actions are consistently reliable (1.00 for both items), while `Pick` yields moderate success (0.55) and `Leave` is weak (0.30–0.35). In contrast, the `can` setting (right) shows a sharper split: `Pick` is highly reliable (0.91–1.00), `Push` succeeds less often (0.45), and `Leave` almost always fails (0.00–0.05). This highlights how different object types bias subgoal feasibility—pushing works well for boxed food, while picking dominates for cans.

Simpler Environment

Collection and Setup. In this environment, we replace teleoperation with autonomous rollouts from a fine-tuned π_0 policy. Each trajectory is labeled with a binary success signal from the environment. We segment behavior into two edge types parameterized by the 2D gripper position: **grasp**$-x$ and **put** x **on top of** y. The grasping interval is annotated using the environment log flag `info["is_src_obj_grasped"]`.

Data Statistics. We collected 576 trajectories across four objects: `eggplant` (72), `carrot` (192), `spoon` (120), and `cube` (192). Table B.1 summarizes per-object grasp and success outcomes.

Table B.1: Simpler environment outcomes per object. Rates are ♯ / total (%).

Object	Total	Grasped	Success
eggplant	72	66 (91.7%)	60 (83.3%)
carrot	192	112 (58.3%)	48 (25.0%)
spoon	144	77 (53.4%)	56 (38.8%)
cube	192	128 (66.7%)	32 (16.7%)
Overall	**600**	**376 (65.3%)**	**195 (33.9%)**

Real-World Environment

Collection and Setup. We study a cabinet–packing task where the agent must place either a `sponge` or a `towel` into a lidded cabinet bin and close it. For `sponge` trials,

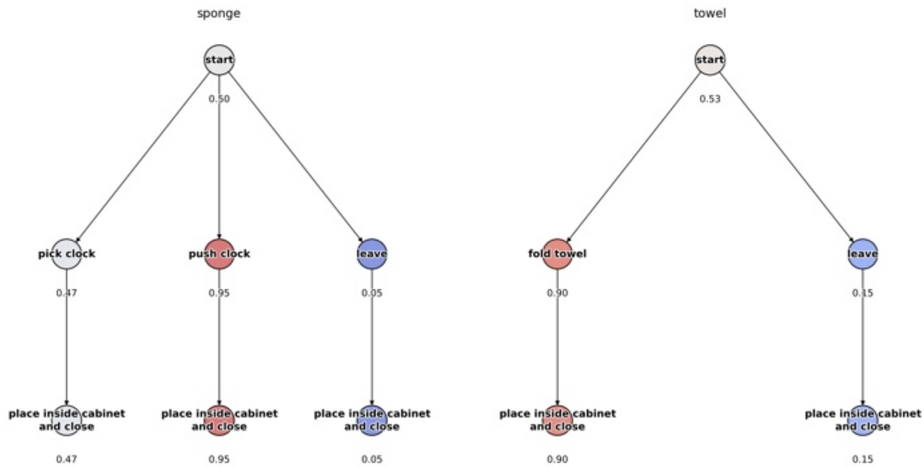


Figure B.3: Data Statistics from human demonstrations in the real-world environment

a `clock` distractor is present inside the bin; operators could (i) *push* the clock aside, (ii) *pick* it up and *place* it elsewhere, or (iii) *leave* it and insert the sponge. For `towel` trials, operators either *fold once* before insertion or *insert as is*. A rollout is marked as successful if the lid fully closes with no protrusion. We used an OpenManipulator-Y arm under leader–follower teleoperation, logging at 20HZ.

Subgoal labels. From the time–stamped logs (20 Hz) we derive a minimal set of discrete subgoals aligned with gripper state transitions and the target order. Each trajectory is segmented into a chain of symbolic steps:

- **Pick-*x*:** grasp and lift a distractor x (e.g., the `clock`) and relocate it,
- **Push-*x*:** displace x to clear space without grasping,
- **Fold-*towel*:** fold the towel once prior to insertion (towel trials only),
- **Place *y* in cabinet:** insert the target item y (e.g., `sponge`, `towel`) into the cabinet bin,

- **Close cabinet:** close the lid with no protrusions.

The *leave* strategy denotes omitting any distractor–handling step. Thus, each demonstration becomes an ordered subgoal sequence, e.g., `push clock` → `place sponge in cabinet` → `close cabinet` (sponge–push), `pick clock` → `place sponge in cabinet` → `close cabinet` (sponge–pick), `place sponge in cabinet` → `close cabinet` (sponge–leave), `fold towel` → `place towel in cabinet` → `close cabinet` (towel–fold), or `place towel in cabinet` → `close cabinet` (towel–leave), which provides the symbolic scaffold for downstream scene–graph–based reasoning and analysis.

Data Statistics. Figure B.3 visualizes the empirical *per–path success rate* for the cabinet–insertion task, with each root-to-leaf path representing a complete subgoal sequence. The numeric label under each node is the estimated success probability $\widehat{p}_{\mathrm{succ}} = \sharp\mathrm{successes}/\sharp\mathrm{trials}$, and node color encodes this value.

In `sponge` trials (with an internal `clock` distractor), **Push-clock** dominates (0.95, $20/21$), **Pick-clock** is moderate (0.47, $9/19$), and **Leave** almost always fails (0.05, $1/20$). In `towel` trials, **Fold** before insertion is highly reliable (0.90, $18/20$), whereas **Leave** is weak (0.15, $3/20$). Overall, handling the distractor (push/pick) or preparing the deformable item (fold) is crucial for success, while omitting these steps yields poor outcomes.

B.0.3 Ablation Studies

Tree Search Algorithm To validate our choice of a Monte Carlo Tree Search (MCTS)-like algorithm, we compare it against a standard Depth-First Search (DFS) baseline in the plug insertion task (Figure B.4). While both methods achieve an identical 80.0% success rate in the seen setting, their performance diverges in the more challenging unseen set-

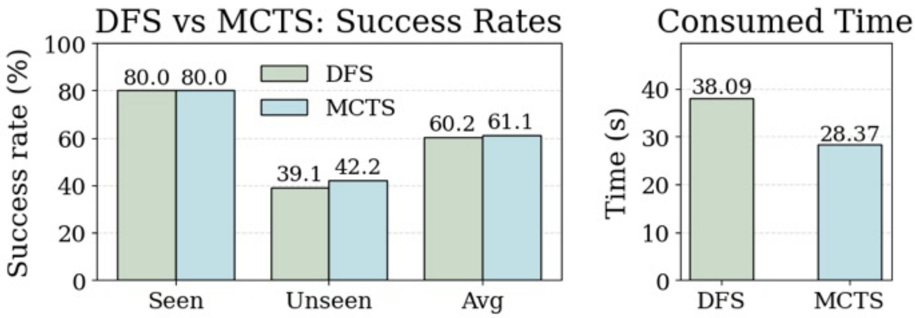


Figure B.4: Comparison between tree search algorithms. On plug insertion, MCTS's value-guided search yields a similar success rate and faster inference than fixed-order DFS.

Table B.2: Weight merging for language backbone in plug insertion environment. Interpolating shared layers between π_0 and PaliGemma via λ_m yields a clear optimum at $\lambda_m = 0.6$; heavier bias to either model reduces overall and unseen success.

λ_m	Seen SR.	Unseen SR.	Avg. SR.
0.5	0.756	0.356	0.555
0.6	**0.800**	**0.422**	**0.611**
0.7	0.733	0.289	0.511
0.8	0.422	0.244	0.333

ting. Here, our MCTS-like approach achieves a higher success rate (42.2% vs. 39.1% for DFS). More importantly, it is significantly more computationally efficient, reducing the average execution time by nearly 10 seconds (28.37s vs. 38.09s for DFS). This efficiency stems from MCTS's guided exploration, which uses the value function to prioritize more promising branches. In contrast, DFS explores paths in a fixed order, often wasting computation on unfeasible plans. This confirms our search strategy is not only more effective at generalizing but also makes better use of computational resources.

Weight Merging of Language Backbone We ablate backbone weight merging [109] between PaliGemma [7] and the π_0 [1] trunk. Specifically, we linearly interpolate the shared layers,

$$\theta_g^{\text{merge}}(\lambda_m) = \lambda_m \, \theta_g^{\pi_0} + (1 - \lambda_m) \, \theta_g^{\text{PG}},$$

while keeping non-overlapping heads separate and training LoRA [110] on top.

Table B.2 shows the effect of varying interpolation weight λ. Performance peaks at λ_m=0.6 , achieving our best results with a seen success rate of 0.800, an unseen success rate of 0.422, and an average of 0.611. Decreasing the weight to λ_m=0.5 (giving more weight to PaliGemma) maintains a high seen success rate (0.756) but harms generalization, with the unseen rate dropping to 0.356. Conversely, increasing the weight towards π_0 with λ_m=0.7 (Seen: 0.733, Unseen: 0.289) and λ_m=0.8 (Seen: 0.422, Unseen: 0.244) leads to a steep decline in performance across all metrics. This highlights the importance of finding a careful balance between the general linguistic knowledge from PaliGemma and the specialized action representations from the π_0 trunk to achieve both strong in-distribution performance and robust generalization.

Chapter C

Appendix 3: Persona-Based Interactive Robot System

This appendix provides supplementary materials supporting the main findings of the study. Section C.1 details the human-subject experiment protocols, including participant instructions, phase descriptions, and persona information used during the interactive sessions. Section C.2 presents additional quantitative and qualitative analyses, such as satisfaction comparisons among personas, the impact of facial expressions and motion cues, and representative post-study comments. Finally, Section C.3 outlines the structured prompt templates used throughout model-based motion and expression generation, describing the roles of motion selection, expression integration, state initialization, and state transition in building coherent persona-driven behaviors.

C.1 Human Subject Studies Protocols

This section describes the procedures, instructions, and materials used in our human-subject experiments. All interactions are non-contact and occur with the robot operating safely inside a transparent fence; participation is voluntary and responses are anonymized.

Hello, and thank you for taking the time to participate. A robot inside a glass fence will recognize and respond to the user's actions. The actions recognized by the robot are as follows: - Approaching - Looking at the robot / not looking - Raising one hand / raising both hands / not raising hands - Waving / not waving

Based on these given factors, you can communicate with the robot non-verbally. Please interact with the robot for about one minute and then fill out the questionnaire. No personal information is collected, and you may withdraw at any time without penalty.

Phase 1 In Survey Phase 1, the objective is to test whether users can identify the intended personality of the robot.

In Survey Phase 1, the objective is to test whether users can identify the intended personality of the robot. We embed one random personality in the robot, and participants focus on whether the robot feels *introverted vs. extroverted* and *agreeable vs. disagreeable*. Please base your judgment on the robot's motion and responsiveness rather than appearance. There are no "right" answers; use your best impression when completing the ratings (e.g., on a Likert scale).

Phase 2 In Survey Phase 2, the objective is to test whether users can recognize which film character the robot is trying to mimic.

In Survey Phase 2, the objective is to test whether users can recognize which film character the robot is trying to mimic. We embed one random character in the robot and test whether the user can identify it. To reduce bias, the survey first asks you to list a few keywords that describe the robot; on the next page, you will choose the closest character that matches the robot's personality. Please keep in mind that our robot does not mimic the character by appearance, but rather by personality and behavioral style. If you are unsure, select the option that best matches the traits you perceived during interaction.

Character-Based Persona Additional Guidelines To help participants better understand the objective of the study, we provide an example persona before starting the actual survey. This example is *illustrative only* and is not part of the evaluation; during the main task, rely on the observed behaviors rather than visual resemblance.

> Here is an example of mimicking a persona. In this case, the robot represents "Peter Parker" from *Spider-Man 1* (pre-hero persona). We are not representing the heroic characteristics of Spider-Man, but rather Peter himself: a nerdy yet friendly neighbor (we show a still image from the movie for context). As you see here, the robot reads a book when no one is engaging (a "nerdy" behavior), but actively greets you by waving or bowing when you initiate interaction (a "friendly neighbor" behavior). Note that in the real study, the robot's behaviors are stylized through motion and simple expressions; costume, voice, or props are not used.

Information of Characters To help the participants better understand the characters that is given, we provide an breif summary of those characters as follows:

> 1. **Spock** (from *Star Trek*): Stoic and logical; does not casually express emotions.
> 2. **Scrooge**: Greedy, grumpy, and miserly, especially before redemption.
> 3. **Captain America**: Embodies patriotism and virtue; principled and dependable.
> 4. **Cowardly Lion**: Characterized by timidity and self-doubt despite good intentions.
> 5. **Minions**: Mischievous, lively, spirited, and often brave in a playful way.
> 6. **Sloth (Priscilla)**: Relaxed temperament and slow, unhurried actions.
>
> During selection, use these brief descriptions as personality cues. If you are unfamiliar with a character, choose the one whose traits most closely match your impression of the robot.

C.2 Additional Results

Satisfaction among Character Personas Our objective is to compare overall user satisfaction across different character personas; Figure C.1 summarizes the distribution and

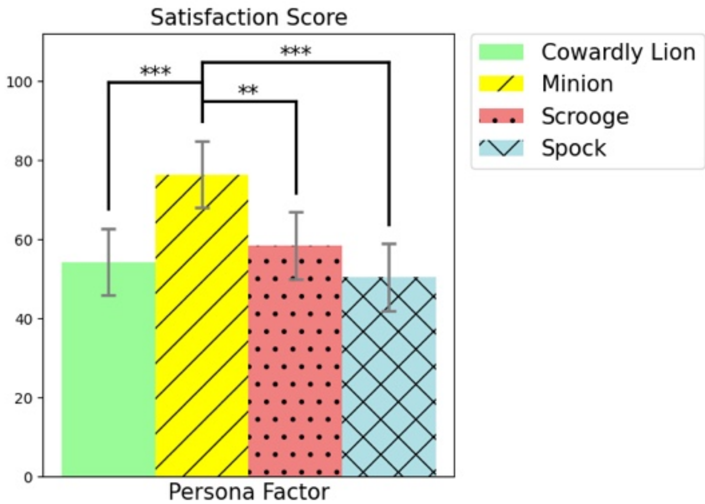


Figure C.1: Satisfaction among Character Personas

pairwise contrasts. We conducted a one-way ANOVA with *persona* as a fixed factor and satisfaction score as the dependent variable. The omnibus test showed a significant effect of persona on satisfaction, $F(3,104) = 7.30, p < .001$, with a medium effect size (partial $\eta^2 = .174$). Post-hoc pairwise comparisons (Tukey HSD) indicated that the *Minion* persona yielded higher satisfaction than each of the other characters: Minion vs. *Cowardly Lion* ($p < .001$), Minion vs. *Scrooge* ($p = .003$), and Minion vs. *Spock* ($p < .001$). No other pairwise differences were significant.

Post-study Comments After completing all survey phases, participants were invited to provide open-ended comments about their experience. This section aimed to collect qualitative feedback on the robot's behavior, perceived personality, and overall engagement level. Participants were encouraged to freely describe what aspects felt natural or confusing, as well as suggest any improvements for future studies. Their insights helped contextualize the quantitative results and provided valuable cues for refining the robot's expressive and interactive design.

Minions:

1. Judging by the constant peek-a-boo, it seemed like a friend with a strong sense of playfulness. Also, since it appeared to yawn when we didn't move, it seemed to get bored quickly if not engaged, representing mischief character.
2. When I see the Minions, it look like Minions, but their large size doesn't seem to match well with my perception.
3. At first, while being happy to meet for the first time with a mischievous expression, I chose the playful Minion because of its active interactions. However, from the middle, it seemed to get tired or started showing expressions of annoyance, which also seems to resemble the behavior of a Minion.
4. When I only saw the arm motions, I was confused, but after seeing the facial expressions on the pad, I could clearly understand.

Cowardly Lion

1. The response was slower than I expected, so it felt somewhat like a sloth, but it often made crying or scared expressions, which made me feel it was closer to the Cowardly Lion. It would have been more interesting if the response speed or the variety of scenarios had been more diverse.
2. The robot seemed to interpret the action of waving arms as an act of aggression, taking a defensive posture and shedding tears as if it was afraid of humans.
3. It seems to resemble the expressions seen on the screen, such as looking flustered, scared, or frightened. The hand gestures also appear to match the description as being defensive, but it might be difficult without the screen or hand gestures.

Scrooge

1. I was confused between Scrooge and the Minions, but it didn't seem to act lively and spirited, so I chose Scrooge haha. Even when the robot ignores or rejects my greeting, I feel a bit upset :(. It seems like it's because the robot appears to have its own personality!
2. It is always angry. Normally, it reads books.
3. It does seem to be an angry situation, but I wish there were more diverse actions! It would be nice if it responded to other actions besides just waving hands!

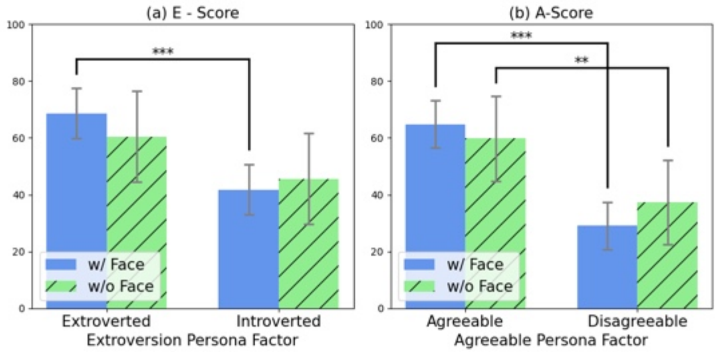


Figure C.2: Impact of Facial Expression (Personality Persona)

Spock

1. The difference in facial expressions is not pronounced, making it difficult to discern whether the object has been recognized. It's confusing what emotion making an X sign with the hands is supposed to express.
2. I chose Spock because it is indifferent to people and dislikes approaching them. However, the expression of indifference (making an X sign with the hands) was not very delicate, causing some confusion during the experiment.
3. It seems to be rejecting people and has its mouth tightly closed, so I thought it resembled Scrooge the most.
4. It is impossible to understand the message it is trying to convey, and I am not feeling the interaction.

Impact of Facial Expression (Personality Persona) Our objective here is to assess whether adding facial expressions improves persona identification for the *extroversion* and *agreeableness* dimensions; Figure C.2 summarizes the conditions and outcomes. We performed a two-way MANOVA to examine the effects of three key factors: the robot's persona (extroversion and agreeableness) and the presence or absence of facial expressions. The interaction effect between facial expression condition and the *ex-*

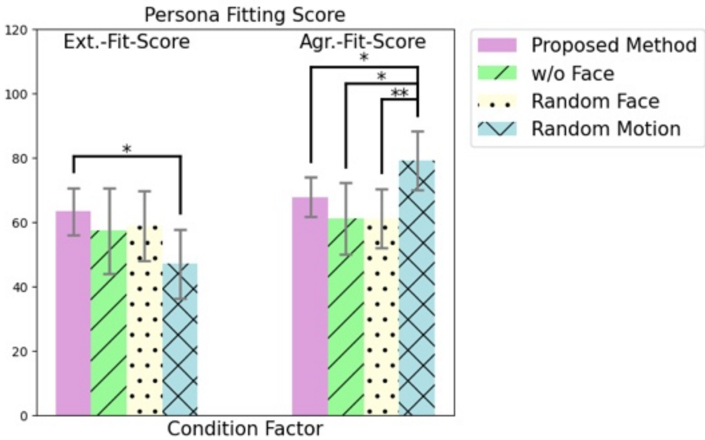


Figure C.3: Effect of Motions and Facial Expressions (Personality Persona)

troversion persona on extroversion scores was not statistically significant, $F(1,60) = 0.855, p = .329$, partial $\eta^2 = .014$. Similarly, the interaction effect between facial expression condition and the *agreeable* persona on agreeableness scores was not significant, $F(1,60) = 1.237, p = .270$, partial $\eta^2 = .020$. Therefore, no significant differences were found attributable to the presence or absence of facial expressions in persona identification.

Effect of Motions and Facial Expressions (Personality Persona) Our objective in this analysis is to quantify how *motion* and *facial expressions* contribute to correct persona identification; Figure C.3 summarizes the ablation conditions and outcomes. We define the fitting metrics as follows: *Ext.-Fitting-Score* equals the E-score when the persona is *extroverted* and 100−E when *introverted*; *Agr.-Fitting-Score* equals the A-score when the persona is *agreeable* and 100−A when *disagreeable*. We compared four ablation conditions (*Proposed*, *w/o Face*, *Random Face*, *Random Motion*). ANOVA revealed a significant effect of condition on *Agr.-Fitting-Score*, $F(3,112) = 3.129, p = .029$, partial

$\eta^2 = .052$. Post-hoc comparisons (Tukey) indicated that *Ext.-Fitting-Score* was higher for *Proposed* than for *Random Motion* ($p = .015$); for *Agr.-Fitting-Score*, significant differences were observed for *Proposed* vs. *Random Motion* ($p = .046$), *Random Motion* vs. *Random Face* ($p = .007$), and *Random Motion* vs. *w/o Face* ($p = .015$). Notably, agreeableness identification appeared better under randomized motion; given limited statistical power, these ablation results should be interpreted with caution. Overall, the pattern suggests that motion cues have a stronger influence on extroversion, whereas facial expressions may be more informative for agreeableness in our method.

C.3 Prompts

Motion Selection To generate motion responses that align with each persona type, we designed a set of structured prompts used to guide the model during the motion selection stage. These prompts aim to elicit behaviorally and semantically coherent motion outputs, reflecting the personality attributes defined in each condition. The following illustrates the exact prompt template used in the experiment:

System Prompt

I am [persona] robot that is interacting with users. I can interact with the users by motions, depending on the user's actions. I want to make a finite-state machine to interact with the user. I have a set of motions that I can do, and I have a set of observations that I can observe from the user. I can conduct the following actions: [motions]

We have 4 different observations:

1. hand up: The number of person's raised hand
2. distance: The distance between the person and the robot
3. gaze: The gaze of the person on the robot
4.hand velocity: The velocity of the person's hand. This would mean how fast the person's hand is waving.
5. approaching vel: The velocity of the person's approaching to the robot. This would mean how fast the person is approaching the robot.

User Prompt

You are [persona] robot that is interacting with users. Choose motions that you want to play, considering your persona. Choose at least 0.25[M] and at most 0.5[M] motions. Try to choose the minimum motion that only fits your persona. Please choose from the following motions: [motions]
Please answer in the following format: state 1, state 2, ... Try to choose different meanings of motions that can be used in different situations

Expression Integration To investigate how facial expressions contribute to perceived persona consistency, we constructed prompts that explicitly link emotional cues to the robot's behavioral context. These prompts guide the model to integrate facial expressions that align with both the ongoing motion and the underlying persona traits, ensuring coherent multimodal expression. The following presents the prompt template used for this stage:

System Prompt

I am [persona] robot that is interacting with users. I can interact with the users by motions, depending on the user's actions. I want to make a finite-state machine to interact with the user. I have a set of motions that I can do, and I have a set of observations that I can observe from the user. I can conduct the following actions: [selected motions]

User Prompt

For each state, we want to estimate the facial expression of the motion. The facial expression should be in the following labels: [facial expression set]

What would be the facial expressions for state current motion Please answer in following format:

speed: face Please adapt the robot persona, [persona] , and give explanation for your answer.

State Initialization Before generating any motion or expression, the model requires an initial contextual setup that defines the scene, task objective, and persona condition. We designed initialization prompts to establish this shared context, ensuring that subsequent motion and expression generations remain consistent with the intended role and environment. The following shows the prompt template used for initializing the state:

System Prompt

I am [persona] robot that is interacting with users. I can interact with the users by motions, depending on the user's actions. I want to make a finite-state machine to interact with the user. I have a set of motions that I can do, and I have a set of observations that I can observe from the user. I can conduct the following actions: [selected motions]

User Prompt

which state should the robot start with? Please adapt the robot persona, [persona] and the answer should be in the following format: state: , with the explanation

State Transition To enable coherent progression across interaction stages, we designed prompts that describe how the robot should transition from one state to another while maintaining task continuity and persona consistency. These prompts condition the model to generate context-aware intermediate behaviors, ensuring smooth and natural transitions between consecutive motion or expression states. The following provides the prompt template used for state transitions:

System Prompt

I am [persona] robot that is interacting with users. I can interact with the users by motions, depending on the user's actions. I want to make a finite-state machine to interact with the user. I have a set of motions that I can do, and I have a set of observations that I can observe from the user. I can conduct the following actions: [motions]

We have 4 different observations:

1. hand up: The number of person's raised hand

2. distance: The distance between the person and the robot

3. gaze: The gaze of the person on the robot

4.hand velocity: The velocity of the person's hand. This would mean how fast the person's hand is waving.

5. approaching vel: The velocity of the person's approaching to the robot. This would mean how fast the person is approaching the robot.

User Prompt

what would be the next state from [current state] ? The robot observations: [observation]

Try to answer in a way that fits the persona of the robot and the current state. In addition, just generate one candidate, and you can choose the same state as the next state. Try to generate diverse states that use all different motions and different motions from the different observations. Please generate the explanation for the answer first and choose one motion

Chapter D

Appendix 4: Unified Framework for Interactive Motion Reasoning and Generation

This appendix provides a comprehensive set of supplementary materials that reinforce the main findings of the research. The appendix begins with motion representation and motion token representation (Sec.D.1), followed by ablation studies on the pretraining method (Sec.D.2), along with ablation studies on the motion tokenizer (Sec.D.3), the demonstration for expansion to multiple human (≥ 3) motion generation (Sec.D.4), and illustrations of the data collection pipeline (Sec. D.5). More detailed results for traditional motion-related tasks are presented (Sec.D.6), limitations (Sec.D.7), and implementation details for the proposed methods (Sec.D.8) and baselines models trained for interactive motions (Sec.D.9). Task explanations cover motion editing and reasoning (Sec.D.10), with implementation details of two-stage baselines (Sec.D.11). The evaluation metrics for traditional motion-related tasks are presented in Sec. D.12. Further sections include templates for pre-training and instruction tuning (Sec.D.13), data visualization and statistics (Sec.D.14, Sec.D.15), qualitative results (Sec.D.16), user study protocols (Sec.D.17).

D.1 Motion Representation and Motion Token Representation

For two persons a and b, we denote the interactive motion as $\{\mathbf{m}_a, \mathbf{m}_b\}$, following non-canonical representation from [89]. Each timestep of the motion $\mathbf{m}^i = [\mathbf{j}_g^p, \mathbf{j}_g^v, \mathbf{j}^r, \mathbf{c}^f]$ is composed of global joint positions $\mathbf{j}_g^p \in \mathbb{R}^{3N_j}$, global joint velocities $\mathbf{j}_g^v \in \mathbb{R}^{3N_j}$, 6D representation of local rotations $\mathbf{j}^r \in \mathbb{R}^{6N_j}$, with the number of joints N_j, and binary ground contact features $\mathbf{c}^f \in \mathbb{R}^4$. This non-canonical representation is applied for both interactive motions and single-person motions. All the motions are represented in an SMPL-X [152] format.

Motion tokenizer encodes the interactive motion into discrete residual tokens in depth D, based on latent vector $\mathbf{z}$.

$$\mathscr{R}\mathscr{Q}(\mathbf{z}^i; \mathscr{C}, D) = (k_1^i, \cdots, k_D^i) \in [K]^D \tag{D.1}$$

where $\mathscr{C}$ is the codebook, $K = |C|$, D is a depth, and k_d^i is code of $\mathbf{z}$ at timestep i with depth d.

The interactive motion token sequence is represented as $X_m = \{k_{1:D}^{1;a}, k_{1:D}^{1;b}, \cdots, k_{1:D}^{L;a}, k_{1:D}^{L;b}\}$, where X_m is a sequence of motion represented in unified vocabulary and $k_{1:D}^{i;a} \in [K]^D$ is the i-th token of motion a. In particular, the motion token is represented as below:

$$
\begin{aligned}
X_m = \{ & \texttt{<motion\_token\_start>}, \\
& \texttt{<motion\_token\_a\_start>}, \quad k_1^{1;a}, \cdots, k_D^{1;a}, \quad \texttt{<motion\_token\_a\_end>}, \\
& \texttt{<motion\_token\_b\_start>}, \quad k_1^{1;b}, \cdots, k_D^{1;b}, \quad \texttt{<motion\_token\_b\_end>}, \\
& \cdots \\
& \texttt{<motion\_token\_a\_start>}, \quad k_1^{L;a}, \cdots, k_D^{L;a}, \quad \texttt{<motion\_token\_a\_end>}, \\
& \texttt{<motion\_token\_b\_start>}, \quad k_1^{L;b}, \cdots, k_D^{:;b}, \quad \texttt{<motion\_token\_b\_end>}, \\
& \texttt{<motion\_token\_end>} \}
\end{aligned}
$$

where `<motion_token_start>`, `<motion_token_a_start>`, `<motion_token_b_start>`, `<motion_token_a_end>`, `<motion_token_b_end>`, and `<motion_token_end>` is a special token added to the unified vocabulary. For modeling single-motion in pre-training we omitted the input string about `motion_token_b`.

D.2 Ablation Studies on Pretraining Method

Table D.1: Ablation studies in pertaining stage for three motion-related tasks on InterX and Interhuman dataset.

Methods	Data	Trainable Params	M2T R Top3 ↑	T2M R Top3 ↑	T2M FID ↓	Reaction Gen. MPJPE ↓	Reaction Gen. FID ↓
Real	-	-	0.867	0.869	0.00	-	0.00
MotionGPT*	InterX+H	248M	0.518	0.280	0.178	1.338	0.364
MoLaM-VQ	InterX+H	726M	0.709	**0.511**	0.181	1.750	0.181
MoLaM (Ours)	InterX+H	726M	0.721	0.427	**0.161**	1.494	0.157
MoLaM (Ours)	InterX+H + MotionX	726M	**0.729**	0.464	0.172	**1.236**	**0.131**

We conducted ablation studies on the pertaining method. All the baselines are pre-trained models, not including the fine-tuning stage. To evaluate the effectiveness of our pretraining approach, we conducted ablation studies comparing different methods on three motion-related tasks: Motion-to-Text (M2T), Text-to-Motion (T2M), and Reaction Generation. As shown in Table D.1, we compared our proposed method, MoLaM, against MotionGPT* and MoLaM-VQ, using the InterX [88] and Interhuman (H) datasets [89]. MotionGPT* serves as a baseline with 248M trainable parameters, achieving a retrieval Top3 score of 0.518 in M2T and 0.280 in T2M, with corresponding FID scores of 0.178 and 1.338 for T2M and Reaction Generation, respectively. MoLaM-VQ, with 726M parameters, improves the M2T retrieval Top3 to 0.709 and T2M retrieval Top3 to 0.511, while maintaining competitive FID scores.

Our method, MoLaM, further enhances performance by achieving a retrieval Top3 of 0.721 in M2T and reducing the T2M FID to 0.161, alongside an MPJPE of 1.494 and FID of 0.157 in Reaction Generation. Notably, when incorporating the additional MotionX [154] dataset, MoLaM achieves the highest M2T R Top3 of 0.729 and the lowest FID scores of 0.172 in T2M and 0.131 in Reaction Generation, demonstrating the substantial benefits of our comprehensive pretraining strategy. These results indicate that our approach not only outperforms existing models in generating accurate and high-quality motions but also effectively leverages additional data to enhance interactive motion understanding and generation. The ablation studies highlight the critical role of our pretraining methodology and the integration of diverse datasets in achieving superior performance across multiple interactive tasks.

Table D.2: Ablation Studies on motion tokenizer base model. We compared VQ-VAE-based tokenizer and the RQ-VAE-based model.

Methods	Reasoning			Editing		M2T	T2M		Reaction Gen.	
	Coh. ↑	Align. ↑	Nat.↑	MPJPE ↓	FID ↓	R Top3 ↑	R Top3 ↑	FID ↓	MPJPE ↓	FID ↓
MoLaM-VQ	5.004	4.256	6.915	0.892	0.128	0.861	**0.601**	0.101	1.109	0.055
MoLaM (Ours)	**5.252**	**4.511**	**6.981**	**0.758**	**0.064**	**0.901**	0.568	**0.059**	**0.691**	**0.019**

D.3 Ablation Studies on Motion Tokenizer

We conducted ablation studies comparing the VQ-VAE-based model with our RQ-VAE-based approach, as shown in Table D.2. The RQ-VAE-based motion tokenizer outperformed the VQ-VAE model in motion reasoning tasks, achieving higher scores in coherence, alignment, and naturalness. This improvement is attributed to reduced information loss, allowing our model to capture finer motion details while also enhancing its motion-to-text retrieval precision.

For generation and editing tasks, the VQ-VAE model achieved slightly better text-to-motion retrieval accuracy but performed worse in FID and MPJPE across editing, reaction generation, and T2M tasks, indicating degraded motion quality and less precise motion details. In contrast, our approach reduced MPJPE by 0.055 for reaction generation, preserving joint dynamics and producing more realistic and natural motions. VQ-VAE's limitations are especially problematic for modeling interactive motions, where precise relative positioning is crucial, making its information loss and reconstruction quality more evident.

We analyzed the effect of varying codebook size (128, 256, 512, 1024) and code depth (2, 3, 4, 5) on both reconstructed motion quality (measured by FID) and retrieval precision at top-3, and observed that increasing the codebook size from 128 to 512 reduced FID while simultaneously improving retrieval precision, indicating richer and more ac-

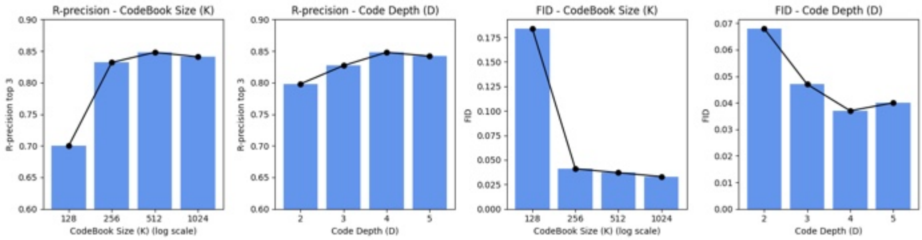


Figure D.1: Ablation Studies on codebook size and depth. We measured Top-3 retrieval-precision accuracy and FiD on the reconstructed motion.

curate motion representations, whereas moving to 1024 yielded diminishing returns at higher computational cost. Likewise, increasing the code depth from 2 to 4 provided better reconstruction quality and retrieval performance by allowing the model to capture more complex motion patterns, but further increasing the depth (e.g., to 5) showed marginal or even negative gains. Consequently, we selected 512 codes and a depth of 4 as the best trade-off between quality, retrieval accuracy, and efficiency.

D.4 Expansion to Multi-Human Motion (≥ 3) Generation

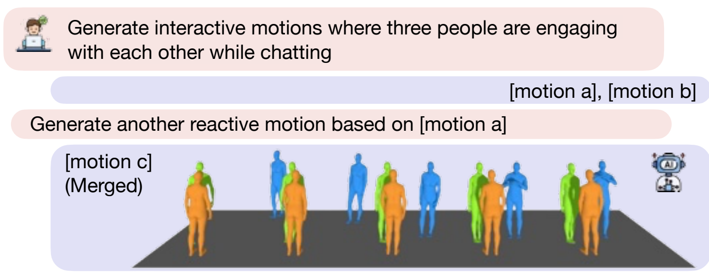


Figure D.2: Our method demonstrates its extendability by generating multi-person interactions (≥ 3 people) through iterative prompting. Despite being trained on two-person scenarios, our framework conditions new motions on prior interactions, enabling the synthesis of natural group dynamics from textual descriptions.

Our method demonstrates the capability to extend motion generation to multi-person interactions ($\geq$ 3 individuals) through iterative prompting, despite being trained exclusively on two-person motion scenarios. Specifically, we first generate motion for a pair of individuals based on the given text description, and then the motion of additional participants is synthesized while conditioning on the pre-existing interactions. This approach ensures that the newly generated motions remain coherent and contextually appropriate within the evolving group dynamics. We argue that this extendability is a key advantage of our versatile framework, as it allows for the scalable generation of complex human interactions without requiring additional multi-person training data. An example of this process is illustrated in Figure D.2, where the model successfully generates a realistic group conversation scene based on a textual prompt.

D.5 Illustration on Data Collection Pipeline

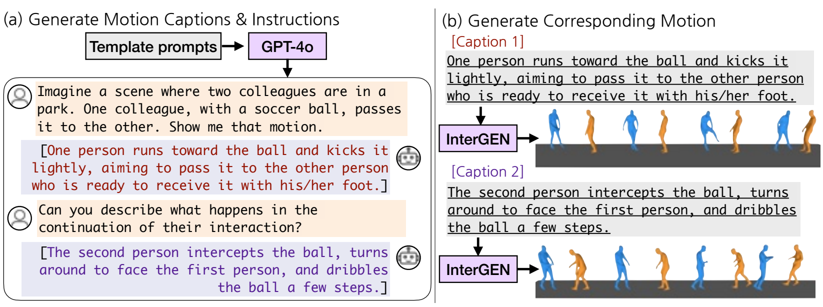


Figure D.3: Overview of synthetic data generation for multi-turn conversations with interactive motions. (a) Motion captions and instructions are generated using GPT-4o based on interactions between two characters, followed by (b) the corresponding motion being synthesized using the InterGEN.

We illustrate the data collection pipeline for generating synthetic multi-turn conversations paired with interactive motions. As shown in Figure D.3, GPT-4-based prompts are used to create captions and instructions, which are then converted into corresponding motions using InterGEN [89].

D.6 Additional Results for Traditional Motion Related Tasks

D.6.1 Motion to Text

Table D.3: Motion-to-Text

Methods	Ret. Precision			BLEU ↑	METEOR ↑	Rouge-L ↑
	Top1 ↑	Top2 ↑	Top3 ↑			
task-specific approach						
TM2T*	0.413	0.589	0.696	0.192	0.386	0.395
unified approach						
MotionGPT*	0.288	0.405	0.494	0.000	0.000	0.00
MotionGPT^{*}_{I}	0.282	0.423	0.503	0.000	0.000	0.00
MoLaM-w/o Inter-MT^2	**0.677**	0.831	0.894	0.220	0.433	0.412
MoLaM (Ours)	0.669	**0.842**	**0.903**	**0.230**	**0.441**	**0.420**

Table D.3 presents a comparative analysis of various methods on the motion-to-text generation task, focusing on retrieval precision and language evaluation metrics such as BLEU, METEOR, and Rouge-L. Among the task-specific approaches, TM2T* achieves moderate retrieval precision scores, with a Top1 precision of 0.413, and language metrics of 0.192 for BLEU and 0.386 for METEOR. In contrast, our proposed unified method, MoLaM, attains significantly higher retrieval precision scores, with a Top1 precision of 0.669, and surpasses TM2T* in all language metrics, achieving a BLEU score of 0.230 and a METEOR score of 0.441. This indicates that MoLaM not only narrows but effectively reverses the performance gap between task-specific and unified approaches in this task. The superior performance of MoLaM in both retrieval precision and language generation metrics demonstrates its effectiveness in generating accurate and descriptive textual captions from motion inputs. By achieving higher scores than the task-specific TM2T*, MoLaM showcases the potential of unified approaches to not only close but surpass the performance gap traditionally observed between task-specific and unified models in motion-to-text generation tasks. This advancement underscores the ability of MoLaM to balance motion understanding and language generation, leading to more coherent and relevant textual outputs.

D.6.2 Text to Motion

Table D.4: Text-to-Motion

Methods	Ret. Precision			FID ↓	Diversity →	MMDist ↓
	R Top1 ↑	R Top2 ↑	R Top3↑			
Real	0.649	0.807	0.878	0.00	0.988	1.072
task-specific approach						
TM2T*	0.276	0.437	0.534	0.300	0.676	1.130
MoMask*	0.402	0.535	<u>0.612</u>	**0.066**	**0.973**	<u>1.128</u>
ComMDM	0.090	0.122	0.201	0.302	0.578	1.201
InterGEN	**0.403**	**0.557**	**0.645**	0.078	<u>0.957</u>	**1.115**
unified approach						
MotionGPT*	0.180	0.262	0.328	0.123	0.898	1.167
MotionGPT$_I^*$	0.175	0.264	0.331	0.118	0.900	1.176
MoLaM-w/o Inter-MT^2	0.335	0.466	0.561	0.082	0.922	1.127
MoLaM(Ours)	0.318	0.469	0.568	<u>0.059</u>	0.945	1.126

Table D.4 provides a comparative analysis of various methods on the text-to-motion generation task, emphasizing the R Top3 retrieval precision metric. Among the task-specific approaches, MoMask* achieves the highest R Top3 score of 0.844, closely approaching the real data benchmark of 0.878, indicating its superior ability to retrieve relevant motions corresponding to textual inputs. InterGEN and TM2T* attain R Top3 scores of 0.645 and 0.534, respectively, showing moderate performance in capturing the top three relevant motions. In contrast, our proposed unified method, MoLaM, achieves an R Top3 score of 0.568, outperforming other unified methods like MotionGPT* and MotionGPT$^*_I$, which have lower R Top3 scores of 0.328 and 0.331, respectively. Although MoLaM does not surpass the task-specific MoMask in R Top3 precision, it narrows the performance gap between task-specific and unified approaches. Additionally, MoLaM maintains a favorable FID score of 0.059 and a high diversity of 0.945, suggesting that it effectively balances motion relevance with quality and variety.

D.6.3 Reaction Generation

Table D.5: Reaction Geneneration

Methods	MPJPE ↓	FID ↓	Ret. Precision R Top1 ↑	Ret. Precision R Top2 ↑	Ret. Precision R Top3 ↑	MMDist ↓
task-specific approach						
MoMask*	1.602	0.112	0.109	0.328	0.412	1.178
unified approach						
MotionGPT*	3.441	0.355	0.079	0.104	0.355	1.246
MotionGPT$^*_I$	1.486	0.106	0.059	0.128	0.106	1.215
MoLaM-w/o Inter-MT^2	<u>0.984</u>	<u>0.031</u>	<u>0.311</u>	<u>0.459</u>	<u>0.554</u>	<u>1.121</u>
MoLaM (Ours)	**0.690**	**0.019**	**0.381**	**0.537**	**0.625**	**1.110**

Table D.5 presents a comparative analysis of various methods on the reaction generation task, focusing on metrics such as MPJPE, FID, Retrieval Precision, and MMDist. Our proposed unified approach, MoLaM, achieves the best performance across all evaluated metrics. Specifically, MoLaM attains the lowest MPJPE of 0.690 and the lowest FID of 0.019, indicating highly accurate joint position predictions and high fidelity in generated motions, respectively. In terms of retrieval precision, MoLaM outperforms both the task-specific method MoMask* and other unified approaches, achieving R Top1, R Top2, and R Top3 scores of 0.381, 0.537, and 0.625. Additionally, MoLaM has the lowest MMDist of 1.110, suggesting that it generates motions closest to the real data distribution. The ablation model, MoLaM-w/o Inter-MT^2, also performs well but slightly lags behind MoLaM, highlighting the significance of the Inter-MT^2 component in enhancing performance. These results demonstrate that MoLaM not only narrows but effectively surpasses the performance gap between task-specific and unified approaches in reaction generation tasks, showcasing its effectiveness in generating accurate and realistic motion reactions.

D.7 Limitations and Impact Statement

The expressiveness of our models remains limited when handling complex or previously unseen actions, indicating a need for further diverse motion source data in its ability to generalize across diverse motion scenarios. In addition, the sequence length becomes excessively long as we flatten the residual motion tokens, which can impact efficiency and computational resources. Leveraging additional transformer models to predict the residual token can reduce this work. Lastly, our method faces challenges in personalization and interpretability, as motion is inherently ambiguous and users may interpret the same motion in different ways. Addressing this issue will require incorporating more tailored approaches that adapt to individual user preferences and expectations through further human-in-the-loop feedback and refinement processes.

Broader Impact Statement Our method opens up new possibilities for interactive motion modeling and understanding, potentially benefiting fields like robotics, virtual environments, and human-computer interaction. However, as the model evolves, careful consideration of ethical concerns, such as misinterpretation of motions or unintended behavioral biases, is crucial.

D.8 Implementation Details

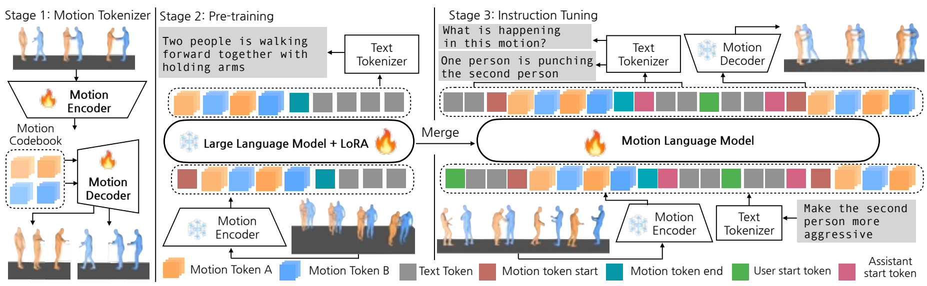


Figure D.4: Method Overview. Stage 1 involves training a motion tokenizer that encodes and decodes interactive motion data. In Stage 2, we pre-train the model by integrating motion and text data, allowing it to learn the alignment between text and motion. Stage 3 focuses on Instruction Tuning, fine-tuning the model to follow instructions and improve its responsiveness to conversational cues.

We set the codebook of the motion tokenizer as $K \in R^{512\times512}$ for most comparisons, with residual depth 4. The motion encoder $\mathscr{E}$ incorporates a temporal downsampling rate l of 4. We utilize LLaMA-3.1-8B [4] as the underlying architecture for our language model. During the pertaining, we train the large language model (LLM) using a low-rank adaptor (LoRA) [110], including the embedding layer and the decoder head. The rank was set as $r = 8$, $\alpha = 16$, with the dropout rate set as 0.05. During the instruction fine-tuning stage, we trained all the parameters. The learning rate was set as 0.0001, the warm-up ratio as 0.01, the learning rate scheduler with cosine decay, and the AdamW optimizer. For single-person modeling, we feed the encoder two identical copies of the motion. In other words, we concatenate the motion with itself so that the encoder, which expects two inputs, processes the same motion twice.

In this implementation, we first construct a motion tokenizer by training a motion encoder–decoder pair (Stage 1), where the encoder converts raw motion data into discrete tokens and the decoder can accurately reconstruct these motions from the tokens. Next, during pretraining (Stage 2), we fuse text tokens (from a large language model) with the motion tokens to jointly model their relationship, ensuring that the textual representations align with the motion embeddings. Finally, in the instruction-tuning phase (Stage 3), we refine the entire model using interactive text–motion tasks. Here, the model is trained to follow higher-level instructions by leveraging both the textual context and the learned motion tokens, enabling it to generate context-appropriate motions in response to user queries or commands. Figure D.4 illustrates the overall pipeline of the proposed method.

D.9 Implementation details for Baselines*

For training MotionGPT [78]*, TM2T [70]* and MoMASK [72]* in the interactive motion dataset, we have utilized the Flan-T5-base model [163] as a base large language model. We trained the model with Interhuman [89] and InterX [88] dataset, with the non-canonical representation, the same as the proposed method. To model the interactive motion, motion tokens of a person "A" and "B" are fed interleaved such as,

$$\texttt{<motion\_token\_start>}, k^{1;a}, k^{1;b}, k^{2;a} \quad \text{(D.2)}$$

, where $k^{i;a}$ is the i-th token of motion a. Although scaling up the model can improve the performance, we conducted the experiment with the same base model as the original paper from MotionGPT [78] and Motionchain [79]. The original paper reported that increasing the model size did not significantly improve the model's performance. We followed same motion token representation as MotionGPT* for both TM2T [70]* and MoMASK [72]*.

D.10 Detailed Task Explanations

Motion Editing Standard motion editing tasks typically involve modifying the motion of a single person based on physical descriptions, such as "raise higher" or "move faster." However, in this task, we focus on editing interactive motions involving two people based on their personas, such as emotions or relationships, by modifying just one person's persona. The primary challenge in motion editing for two people is that when the motion of one person changes, the motion of the second person, which is correlated, also needs to be adjusted. This requires more complex reasoning about social interactions. Specifically, we define the task as "`USER:{scene_information}, {reference_motion}. ASSISTANT: {motion_caption}. USER: {editing_command}. ASSISTANT: {edited_motion}.`" The editing command could be defined as asking the model to change the persona of a person, like "Make one person shy." We let our model generate motion caption in the middle to let the chain-of-thoughts technique improve the reasoning ability.

Motion Reasoning Motion reasoning involves predicting future motions or inferring past events based on the current motion context. This task requires understanding the sequence of motions and making logical inferences about the preceding or subsequent events. For instance, given a motion of an ongoing interaction between two individuals, the model needs to deduce what might have happened before this moment or predict what will likely occur next. This is crucial for applications requiring a temporal understanding of motions, such as surveillance analysis, animation, or human-robot interactions. We define the input sequence as follows: "`USER:{question_1}, {motion_1}. ASSISTANT: {answer_1}. USER: {question_2}, {motion_2}.`", where the model has to predict "`ASSISTANT: {answer_2}`". The inference question could involve queries like "Can you tell me what happened before?" or "What do you think will happen next in this scenario?". This task demands high-level reasoning and comprehension of motion sequences, enabling the model to generate plausible and coherent motion narratives based on the given context.

D.11 Detailed Explanation about Two-stage Baselines

In Section 5.2 and Section 5.3, we have compared the proposed method with two-stage models. In particular, we have utilized TM2T [70] for the motion captioner and InterGEN [89] for the text-to-motion generation model.

D.11.1 Motion Editing

In the motion editing task, the two-stage approach first uses the motion-to-text (TM2T; [70]) model to generate a caption from the source motion and append the editing command. Then, the text-to-motion (InterGen; [89]) model produces the edited motion based on this caption and command. In particular, the input for the text-to-motion model is "`[motion caption]. [editing command]`".

We first trained the TM2T model with the InterHuman dataset [89]and the InterX [88] dataset, similarly to MotionGPT*. Next, we trained the text-to-motion diffusion model, InterGEN for the second stage.

D.11.2 Motion Reasoning

In the motion reasoning task, the two-stage model integrates TM2T with large language models such as GPT-4o [150] and LLaMA-3.1-8B [4]. Here, the motion components in the conversational data are replaced with captions generated by TM2T, which are then fed into the LLM for reasoning and response generation. In particular, the original input for the motion-language model was "`USER:{question_1}, {motion_1}. ASSISTANT: {answer_1}. USER: {question_2}, {motion_2}.`", where the model has to predict "`ASSISTANT: {answer_2}`". We replaced the motion into motion caption obtained by motion captioner for the input for LLM like "`USER:{question_1}, {motion_caption_1}. ASSISTANT: {answer_1}. USER: {question_2}, {motion_caption_2}.`". Again, we utilized TM2T* for the motion captioner mentioned in the previous section.

D.12 More details about Evaluation Metric for Traditional Motion Related Tasks

Motion Quality The Fréchet Inception Distance (FID) is used to assess the similarity between the distributions of generated and real motions, utilizing an appropriate feature extractor tailored to each dataset. In addition, we use well-known motion capture metrics, MPJPE, to quantify global and local errors in meters.

Table D.6: Template for Pretraining

Task	Sequence	Label
Text-to-Motion	Generate caption from motion: [motion] [caption]	[caption]
Motion-to-Text	Generate motion from caption: [caption][motion]	[motion]
Reaction Generation	Generate reaction motion: [motion]	[motion B]
Motion Prediction	Predict motion: [motion]	[Last 75%motion]

Motion Diversity We have utilized diversity to evaluate the diversity of the motion following previous work [78, 151]. To evaluate Diversity, the generated motions are split into two equal-sized subsets, and the Diversity metric is calculated as the average distance between motions within these subsets.

Text-Motion Matching TMR [151] offers motion/text feature extractors that produce geometrically coherent features for aligned text-motion pairs and vice versa. In this feature space, we evaluate motion-retrieval precision (R Precision) by combining the generated motion with 31 mismatched motions and calculating the top-1/2/3 matching accuracy between the text and motion. Furthermore, we assess the Multi-modal Distance (MM Dist), which measures the distance between the generated motions and their corresponding text.

D.13 Template Forms for Pre-training and Instruction Tuning

We will detail the template forms utilized during the pre-training and instruction-tuning stages of our model development. Tables D.6 and D.7 illustrate the specific formats employed in each stage, providing a structured approach to aligning motion data with textual descriptions and enhancing the model's interactive capabilities. All the templates

Table D.7: Template for Instruction Tuning

Task	User	Assistant
Text-to-Motion	Demonstrate a sequence of movements that symbolizes the sentiment of [caption]	[motion]
	Please create a motion that represents the power of [caption]	The motion is [motion]
	I need a motion that represents the power of [caption]	Sure, [motion]
	Show me a gesture that conveys [caption]	
	Produce a motion that matches [caption]	
	...	
Motion-to-Text	Describe the motion represented by [motion]	[caption]
	Provide a summary of the action depicted in [motion]	
	Explain the motion shown in [motion]	
	Provide a text-based explanation of the action being shown in [motion]	
	Please provide a description of the motion in [motion]	
	...	
Motion Prediction	Predict motion: [first 25%motion]	[Last 75%motion]
	Do the motion prediction task for [first 25%motion]	

are from MotionGPT [78].

D.13.1 Pre-training Templates

During the pre-training stage, our objective is to align motion and language representations by leveraging large language models (LLMs). We design tasks such as Text-to-Motion, Motion-to-Text, Reaction Generation, and Motion Prediction using paired datasets like InterX [88] and Interhuman [89]. The pre-training templates involve generating captions from motion sequences, creating motions based on textual descriptions, producing reaction motions in response to initial motions, and predicting subsequent motions from partial sequences, as summarized in Table D.6. For single-person motion, we utilized text-to-motion, motion-to-text and motion prediction task during training.

D.13.2 Instruction-Tuning Templates

In the instruction-tuning stage, we enhance the model's ability to follow diverse instructions presented in a conversational format. Utilizing the INTER2-MT dataset alongside single-turn data from previous interactive motion datasets, we format user instructions and assistant responses to facilitate multi-turn interactions. Table D.7 outlines the templates used for tasks such as generating motions from user prompts, describing motions based on user queries, and predicting motion continuations. By structuring the interactions in this manner, the model becomes adept at understanding and responding to various motion-related commands, thereby improving its performance in interactive scenarios.

D.14 Data Sample Visualization

The samples from the synthesized dataset, Inter-MT^2, are illustrated in Figure D.5.

D.15 Inter-MT^2 Statistics

Table D.8: Statistics on Inter-MT^2.

	Total	Train	Val.	Test
# of Samples	82736	66194	4141	12401
# of Motions	317749	132388	8282	24802
From Dataset	56395	50258	3142	2995
Synthesized	96676	82130	5140	9406

We collected 82K samples of multi-turn conversational data, each involving interac-

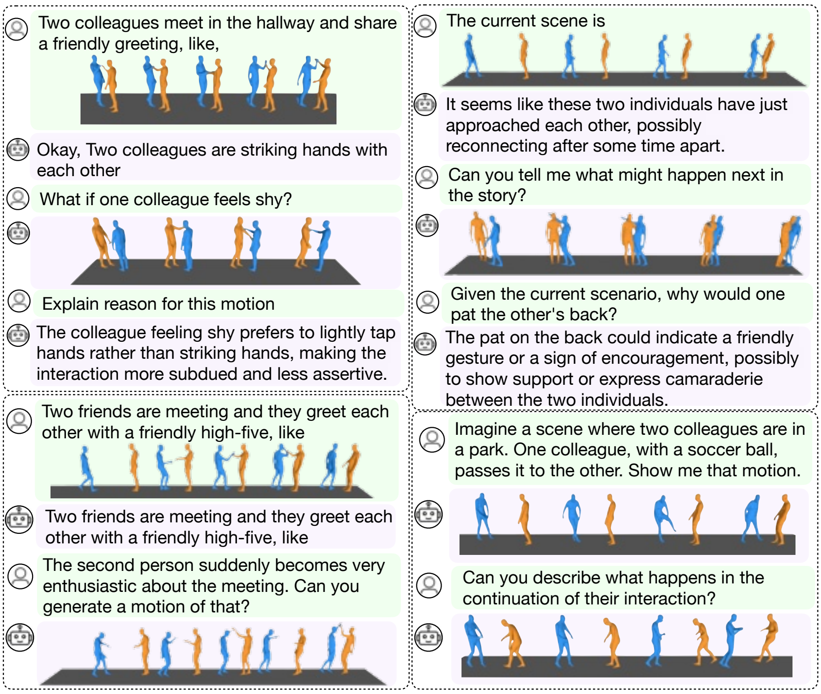


Figure D.5: Sample from Inter-MT^2 dataset. The left column visualizes samples of motion editing, and the right column shows examples from the motion reasoning task.

tive motions. Of these, 30K samples focus on motion editing, 30K on reasoning about past or future scenarios, and 12K on story generation. Each sample includes four to eight conversation turns and two distinct motions. The dataset contains 96K motions generated using a text-to-motion diffusion model, while 56K motions come from the original source dataset. The train-validation-test set is randomly split by the ratio 0.8:0.05:0.15.

D.16 Qualitative Results

We visualize our result gallery on motion editing in Figure D.6 and on motion reasoning in Figure D.7. Furthermore, the results for motion-to-text (Figure D.8), text-to-motion (Figure D.9), and reaction generation (Figure D.10) are demonstrated. In figure D.11, we demonstrated the generation ability of the proposed method in longer contexts with a failure case.

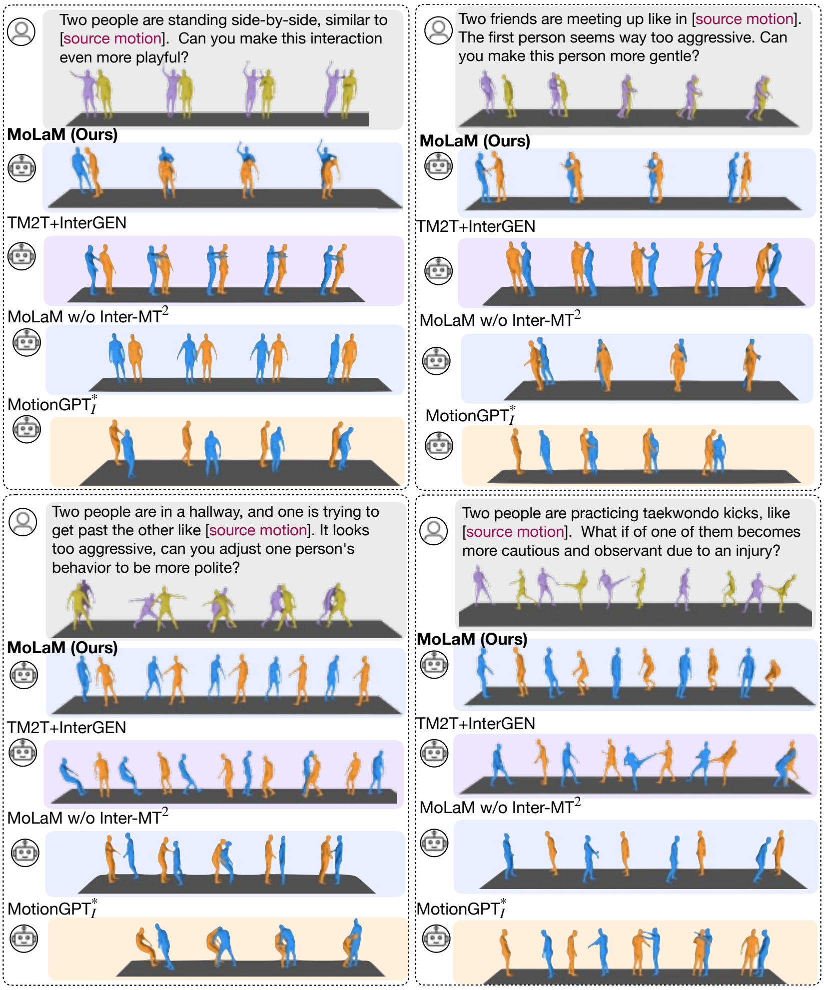


Figure D.6: Generated motion from source and editing command.

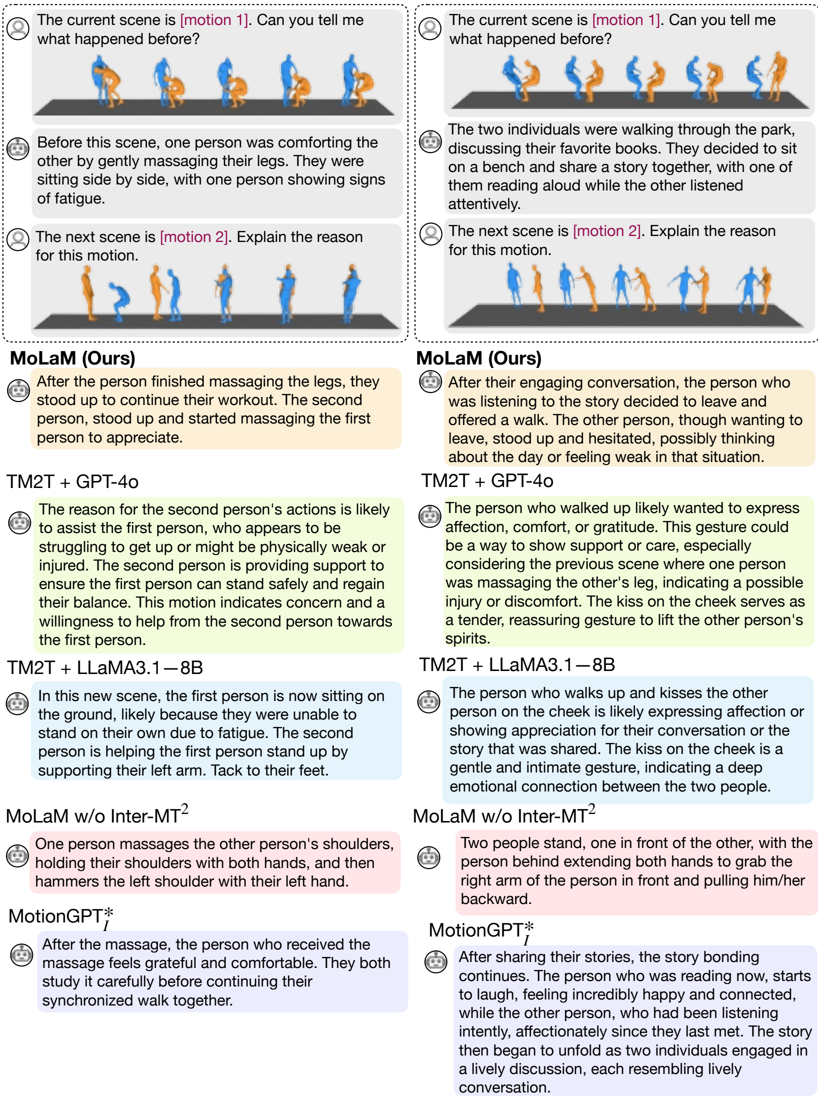


Figure D.7: Generated responses based on the previous conversations for motion reasoning task.

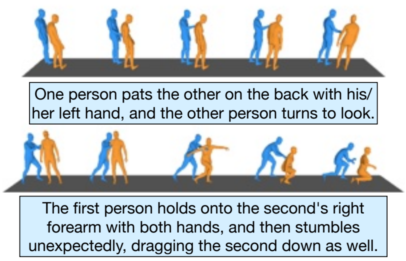


Figure D.8: Motion-to-text results. The blue part is generated motion captions from source motions.

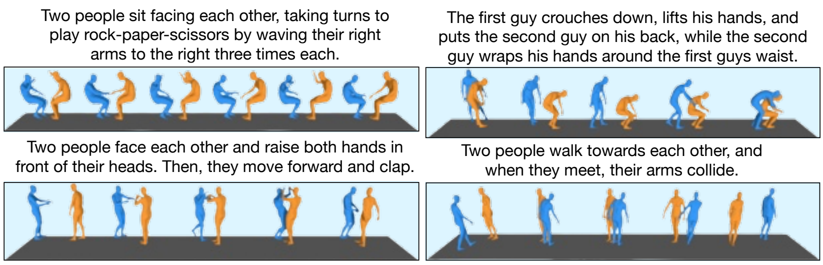


Figure D.9: Text-to-motion results. The blue part is generated motions from the motion caption.

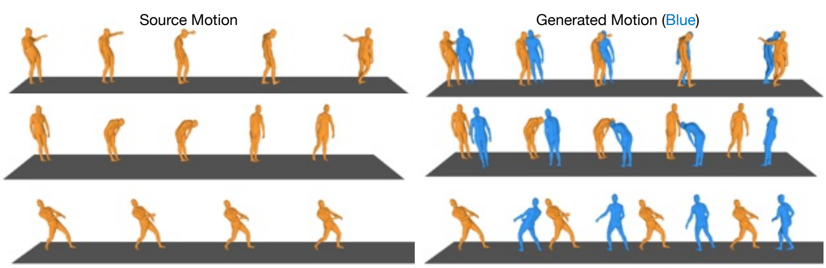


Figure D.10: Reaction Generation. The input motion is orange, while the generated reactive motion is colored blue.

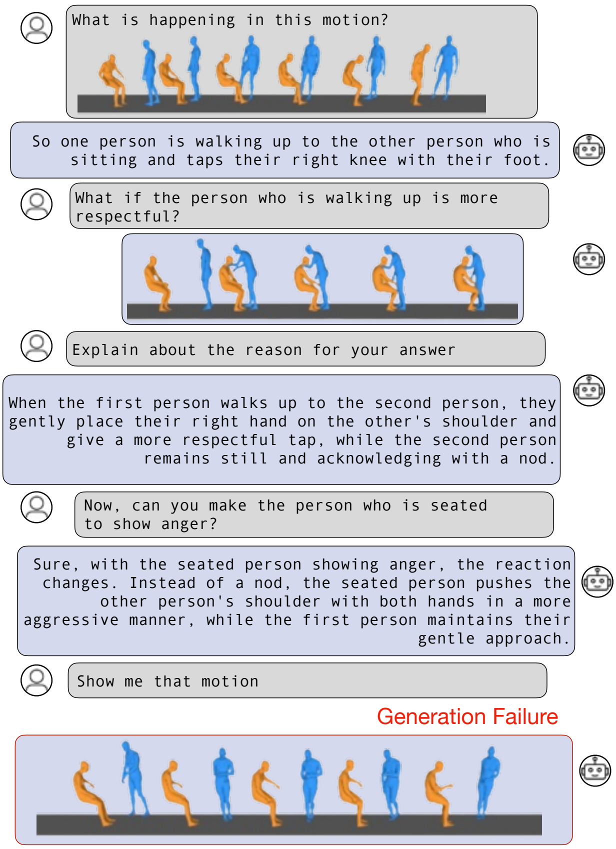


Figure D.11: We demonstrate the ability of the proposed method to generate responses in long conversations and also the failure cases.

D.17 User Subject Studies Protocols for motion editing

We conducted user subject studies using the platform on the Mechanical Turk service from AWS [164].

D.17.1 Instructions

The summary given to the user is as follows:

> We are conducting an academic survey about the quality of generated motions. We need to understand your opinion about the motion quality and ability to follow the editing commands. Please evaluate each motions based on the given criteria.
>
> You will be presented with multiple instruction samples. After completing the evaluations on each page, click ”Next” to proceed. On the last page, click ”Submit” to complete the survey.

The detailed instruction is as follows:

Objective: We are conducting a survey to evaluate how well AI-generated motions follow given instructions and how natural they appear. Your feedback is important to help us improve the AI's ability to create realistic movements that match specific editing commands.

Survey Overview: You will be shown a source motion and an edited motion. Your task is to evaluate both based on specific criteria. After evaluating a few examples, you will also rate multiple edited motions generated from the same source motion using different methods. The survey is divided into multiple pages, and you can move through the pages using ”Next” or ”Previous” buttons. You must complete all fields on each page before proceeding.

Evaluation Criteria: For each pair of videos (source and edited), you will be asked to rate them based on:

Content Similarity: Does the edited motion stay true to the original motion? Rating scale: 1 (Strongly Disagree) to 5 (Strongly Agree)

Alignment with Instruction: Does the edited motion follow the instructions given? Rating scale: 1 (Strongly Disagree) to 5 (Strongly Agree)

Motion Quality: Is the quality of the edited motion good, and does it look natural? Rating scale: 1 (Strongly Disagree) to 5 (Strongly Agree)

Survey Structure:

Evaluation of Pre-selected Motion Examples: In the first section, you will review hand-picked video pairs. Each page will show a source video and its edited version. You will rate how similar they are, how well the editing follows instructions and the overall quality of the motion.

Evaluation of Randomly Selected Motion Samples: In the second section, you will see five different edited motions for each scenario. These motions are created using different methods. You will rate each one based on content similarity, alignment with instructions, and motion quality.

Instructions:

Review the motion examples: Each page will show a description, editing instruction, and two videos (source and edited). Watch the videos and rate them using radio buttons based on the three criteria. Click ”Next” to move to the next example.

Evaluate random scenarios: You will be shown five edited motions per scenario. Review and rate them on the same criteria as before. Use ”Next” and ”Previous” to navigate.

Completion: Once all evaluations are finished, click ”Submit” to complete the survey.

Tips:

Watch both videos completely before deciding. If you're unsure, select ”Neutral.” All fields must be filled before you can move forward or submit the survey.

The examples of ratings given to the user are shown in Figure D.12.

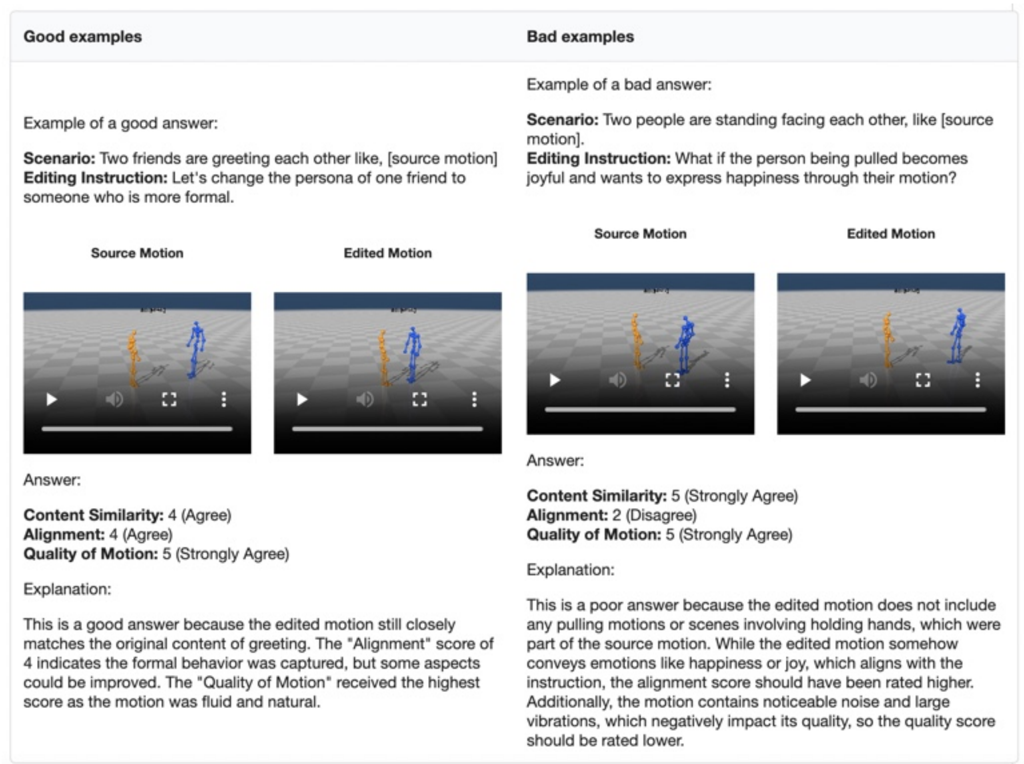


Figure D.12: The examples of ratings given to the user

D.17.2 Qualifying test

Before participating in the main user studies, all participants must pass a qualifying test to ensure they understand the evaluation criteria. In this test, participants are asked to assess four samples based on three metrics: Content Alignment, Fidelity of Motion, and Quality of Motion. Among the four samples, two are high-quality and derived from the ground-truth dataset, while the other two are low-quality—one is a mismatched motion with a single instruction, and the other is generated by the least effective model, MotionGPT*. Participants must rate the low-quality samples lower than the high-quality ones in each of the three metrics. If any of the low-quality samples receive ratings that are equal to or higher than the high-quality samples in Content Alignment, Fidelity, or Quality of Motion, the participant will receive an error message and will need to adjust their ratings accordingly. This ensures that only participants who can accurately distinguish between high and low-quality motions based on the defined metrics proceed to the main study. The example of the qualifying test is demonstrated in Figure D.13

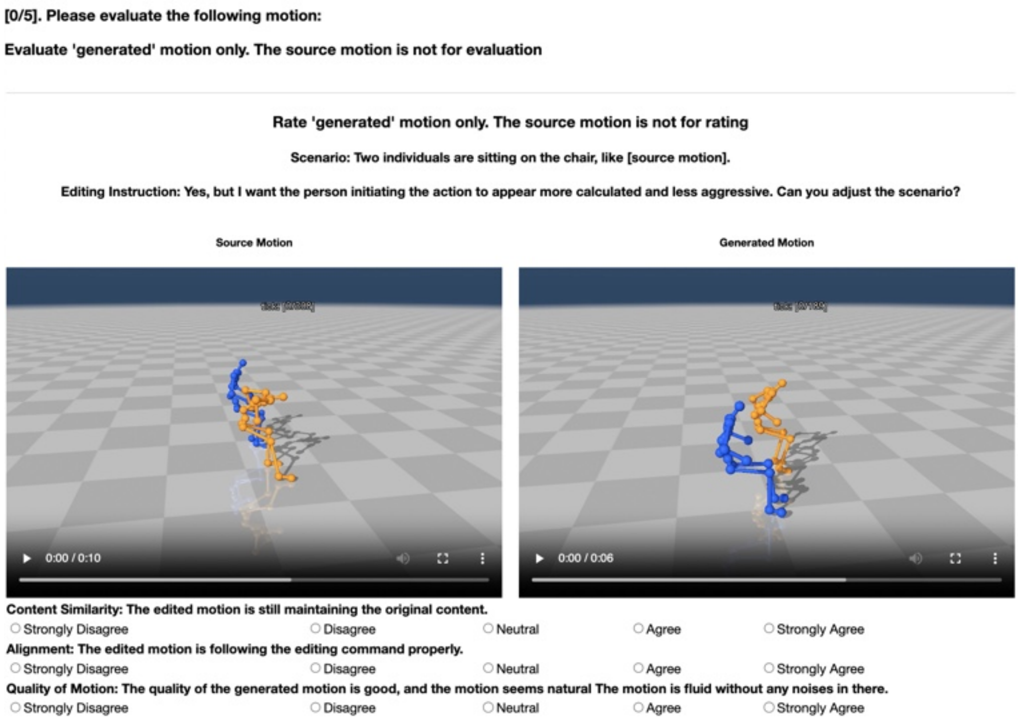


Figure D.13: Qualifying test in user subject studies

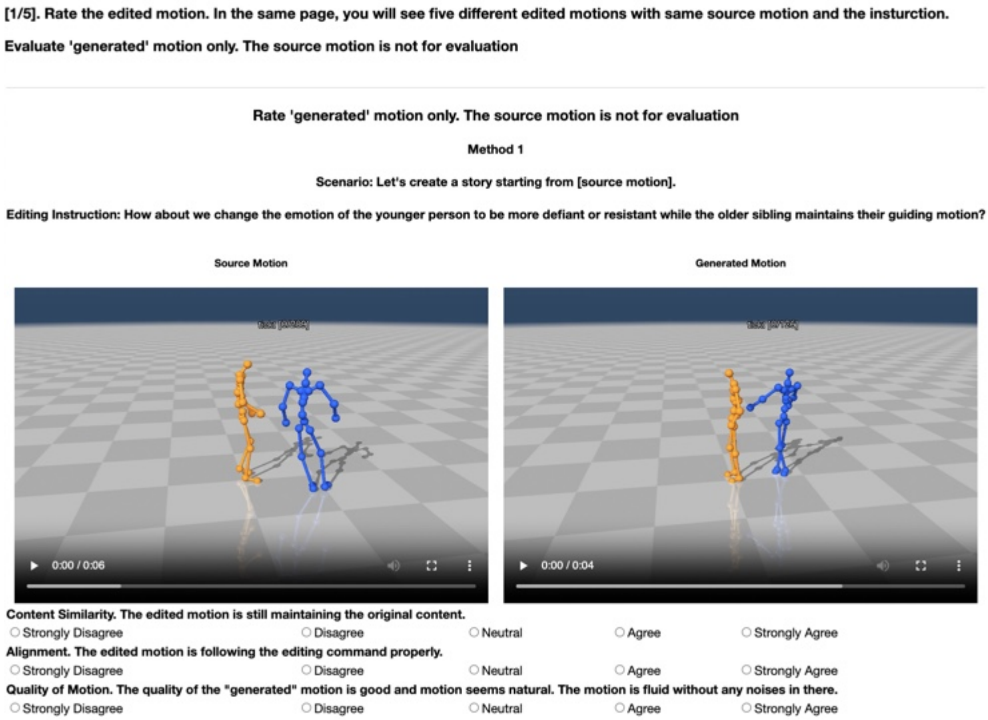


Figure D.14: Survey Example

D.17.3 Detailed Survey Format

Main Survey Structure In the main survey, each participant was randomly assigned 5 samples from a larger pool of 30 diverse motion sequences. This random sampling strategy was employed to ensure a broad and representative evaluation, minimizing any potential selection bias. For each of these selected samples, participants were asked to evaluate five baseline methods, including our proposed model (MoLaM), MoLaM w/o Inter-MT^2, MotionGPT*, MotionGPT$^*_I$, and two-stage model based on TM2T [70] and InterGEN [89]. To eliminate ordering effects and ensure that the evaluation was solely based on the quality of the motions rather than their presentation order, the order of the baseline methods was randomly shuffled for each participant. This randomization was crucial in preventing any unintended bias that might arise from the sequence in which the methods were presented.

Evaluation Metrics Participants assessed each motion sample using three evaluation metrics, which provided a multidimensional view of each model's performance:

- Content Similarity: The edited motion is still maintaining the original content.
- Alignment with Instruction: The edited motion is following the editing command properly.
- Motion Quality: The quality of the generated motion is good, and the motion seems natural The motion is fluid without any noises in there.

We leveraged a 5-scale Likert scale, 1 from strongly disagree to 5 for strongly agree.

Exclusion Criteria To maintain high data quality and ensure meaningful results, we implemented strict exclusion criteria. Participants who assigned the same rating across all evaluation metrics for every sample were excluded, as such uniformity indicated a lack of genuine engagement or understanding of the evaluation process. Additionally, those who provided identical ratings across all comparison methods for a given sample were also omitted. This approach ensured that only participants who thoughtfully differentiated between the methods based on their performance were included in the final analysis. These exclusion rules were essential in filtering out unreliable data and ensuring that the survey results accurately reflected the participants' true assessments of each model's performance.