Situated Language Grounding for Multimodal AI Assistant Modeling

by

Yichi Zhang

A dissertation submitted in partial fulfillment
of the requirements for the degree of
Doctor of Philosophy
(Computer Science and Engineering)
in The University of Michigan
2025

Doctoral Committee:

Professor Joyce Chai, Chair
Assistant Professor Maani Ghaffari
Professor Stella Yu
Associate Professor Zhou Yu

Yichi Zhang
zhangyic@umich.edu
ORCID iD: 0000-0003-3214-1070

© Yichi Zhang 2025

DEDICATION

To Mom, Dad, and Rui

ACKNOWLEDGEMENTS

First and foremost, I would like to express my deepest gratitude to my advisor, Professor Joyce Chai, for all the guidance and support she provided throughout my Ph.D. journey. I still remember the excitement of receiving her email inviting me to join her lab. At that time, I could not have imagined how profoundly her enthusiasm and passion for research would inspire me. Over the past five years, Joyce gives me so much freedom to pursue my research interests, and offers me the highest level of support and encouragement. I especially remember the countless hours she spent helping me revise our first co-authored paper—at a time when we had not yet met in person due to the pandemic. Beyond academia, her thoughtfulness and compassion have taught me invaluable life lessons. I am incredibly fortunate to have had her as my advisor.

I extend my sincere appreciation to my thesis committee members, Professors Maani Ghaffari, Stella Yu, and Zhou Yu, for their insightful comments and constructive feedback. In particular, I would like to give special thanks to Professor Zhou Yu, whom I've known since the summer of 2019, when I interned at her lab at the University of California, Davis. Zhou introduced me to the fascinating field of conversational AI and guided me through my first publication in this area. Her mentorship strongly influenced my decision to pursue research in situated dialogue and motivated me to participate in the Amazon Alexa Prize SimBot Challenge. I feel incredibly lucky to have her witness my growth as a researcher.

Special thanks go to all members of Team SEAGULL: Jianing Yang, Jiayi Pan, Yinpei Dai, Yuwei Bao, Dr. Shane Storks, Keunwoo Peter Yu, Nikhil Devraj, and Ziqiao Ma. Working with you—through moments of creativity, intensity, and occasional chaos—led us to the remarkable success of winning first place in the Alexa Prize SimBot Challenge, one of the most unforgettable highlights of my Ph.D. I also sincerely thank Rui for tirelessly testing our bot as its most devoted user.

I am deeply grateful to my collaborators: Run Peng, Jian Wang, Lingyun Wu, Luyang Hu, Andy Chung, Jung-Chun Liu, Xuweiyi Chen, Itamar Bar-Yossef, Alexander De La Iglesia, Megan Su, Nicholas Tomlin, Yifei Zhou, Dr. Sergey Levine, and Dr. Alane Suhr. I also thank my labmates in the Situated Language and Embodied Dialog (SLED) lab: Dr. Cristian-Paul Bara, Josue Torres-Fonseca, Dr. Shengyi Qian, Chimaobi Okite, and Jacob

Sansom. I'm especially thankful to my internship mentors and collaborators at Amazon AGI—Dr. Qiaozi Gao, Dr. Xiaofeng Gao, and Suhaila Shakiah—for their invaluable guidance and support. I also extend my sincere thanks to my internship mentors at Meta—Dr. Seungwhan Moon, Dr. Xin Luna Dong, Dr. Zhaojiang Lin, and Dr. Andrea Madotto—for the incredible opportunity to work on the AR/VR assistant project. This experience became the centerpiece of my dissertation and profoundly shaped my research direction. Furthermore, I appreciate the many anonymous reviewers whose constructive feedback has greatly improved my work.

On a personal note, I wish to express my heartfelt gratitude to my family and friends. I thank my parents for their unconditional love and support throughout my life. Due to visa restrictions, we endured a long separation before they could visit me in the U.S., and their presence meant the world to me. I am especially thankful to my partner, Rui Pan, whose love, patience, and support have carried me through all of life's highs and lows. She has profoundly influenced who I am today. Through countless trips between Ann Arbor and Chicago, I have always felt incredibly fortunate to have her by my side. I also want to thank my longtime friend Yue Meng, who has been by my side from middle school to Ph.D. To Yang, Han, Jaky, and Ji—I treasure the wonderful times we shared on our many trips together. To my friends—Hongzhe, Yixuan, Zijian, Jiading, Yuan, Shiyu, Yuhao, Tong, Zhexuan, Heng, Zhuo, and many others—I am truly grateful, and I've always missed you. I also want to thank the friends I met on the soccer field. Special thanks goes to the UMC team, for all these years of playing soccer together and winning those championships. Thanks to my teammate from Tsinghua Veterans, CalBlue, MetroTech, AAFC and JHU, for all the great games we played together.

Finally, I gratefully acknowledge the financial support provided by the National Science Foundation (grants IIS1949634 and SES2128623), the DARPA PTG program (grant HR00112220003), and the Amazon Alexa Prize Award.

TABLE OF CONTENTS

DEDICATION ii

ACKNOWLEDGEMENTS iii

LIST OF FIGURES ix

LIST OF TABLES xv

LIST OF APPENDICES xix

LIST OF ACRONYMS xx

ABSTRACT xxii

CHAPTER

1 Introduction **1**

1.1 Challenges in Situated Language Grounding 2
1.2 Dissertation Structure and Contributions 3

2 Related Work **7**

2.1 Multimodal Large Language Models 7
2.1.1 Evolution and Capabilities 7
2.1.2 Grounded MLLMs 8
2.1.3 Evaluating Visual Illusions in MLLMs 9
2.2 Embodied AI for Instruction Following 9
2.2.1 Simulation Environments 10
2.2.2 Key Techniques for Embodied Instruction Following 10
2.3 Interactive Visual Assistants and Situated Communication 11
2.3.1 Evolution and Challenges of Task Guidance Systems 11
2.3.2 Synthetic Data and Simulation 13

PART I: Language Grounding in Visual Perception **14**

3 Grounding Large Language Models to Holistic Segmentation **14**

3.1 Overview 14
3.2 GROUNDHOG Model Architecture 17
3.2.1 Building Entity Features from Masks 17

3.2.2 Language Grounding to Entity Segmentation . . . 18
3.2.3 Towards Holistic Entity Mask Proposals . . . 19
3.3 The M3G2 Dataset . . . 20
3.3.1 Grounded Image Captioning (GIC) . . . 21
3.3.2 Referring Expression Segmentation (RES) . . . 22
3.3.3 Grounded Visual Question Answering (GVQA) . . . 23
3.3.4 Referential Dialogue (RD) . . . 23
3.4 Experimental Results Analysis . . . 24
3.4.1 Generalist in Grounded Vision-Language Tasks . . . 24
3.4.2 Trustworthiness and Transparency . . . 27
3.4.3 Ablation Studies . . . 29
3.5 Discussion of Limitations . . . 30
3.6 Summary of Findings . . . 31

4 Language Grounding under Visual Illusions for Vision-Language Models 32

4.1 Overview . . . 32
4.2 The GVIL Benchmark . . . 34
4.2.1 Data Collection . . . 34
4.2.2 Benchmark Tasks . . . 36
4.3 Experimental Setup . . . 40
4.4 Results Analysis . . . 40
4.4.1 Illusion Recognition . . . 41
4.4.2 Communication Under Illusion . . . 42
4.4.3 Delving into Illusion Categories . . . 45
4.4.4 Understanding the Cause of Illusions . . . 45
4.5 Discussion of Limitations . . . 46
4.6 Summary of Findings . . . 47

PART II: Language Grounding in Embodied Planning . . . 49

5 Hierarchical Task Planning with Unified Transformers . . . 49

5.1 Overview . . . 49
5.2 Hierarchical Tasks via Unified Transformers . . . 52
5.2.1 Task Decomposition . . . 52
5.2.2 Unified Transformers . . . 53
5.2.3 Self-Monitoring and Backtracking . . . 55
5.3 Experiments . . . 57
5.3.1 Setting and Implementation . . . 57
5.3.2 Evaluation Results . . . 58
5.4 Discussion of Limitations . . . 63
5.5 Summary of Findings . . . 64

6 Deliberative Agent for Following Natural Language Instructions . . . 66

6.1 Overview . . . 66
6.2 Problem Definition . . . 67
6.3 A Neuro-Symbolic Deliberative Agent . . . 69

6.3.1 World Representation Construction . . . 70
6.3.2 Subgoal-Based Task Monitoring . . . 71
6.3.3 Online Symbolic Planning . . . 71
6.4 Experimental Results Analysis . . . 74
6.4.1 Experimental Setup . . . 74
6.4.2 Overall System Performance . . . 76
6.4.3 Generalization to New Tasks . . . 77
6.4.4 Analysis of Submodules . . . 78
6.4.5 Failure Analysis . . . 82
6.5 Discussion of Limitations . . . 83
6.6 Summary of Findings . . . 83

PART III: Language Grounding in Situated Communication . . . 85

7 Proactive Assistant Dialogue Generation from Streaming Egocentric Videos 85

7.1 Overview . . . 86
7.2 Proactive Assistant Dialogue Generation from Streaming Videos . . . 87
7.2.1 Problem Definition . . . 87
7.2.2 ProAssist: A Synthetic Dialogue Dataset . . . 88
7.3 Evaluation of Proactive Task Assistant . . . 89
7.4 Proactive Assistant Dialogue Modeling . . . 92
7.4.1 Feasibility Analysis of Existing Models . . . 92
7.4.2 Learning When to Speak under Imbalance . . . 92
7.4.3 Iterative Progress Summarization . . . 93
7.5 Experimental Results Analysis . . . 93
7.5.1 Experiment Setups . . . 93
7.5.2 Dialogue Quality of ProAssist . . . 95
7.5.3 Alignment Between Proposed Metrics and Human Judgment . . . 96
7.5.4 Result Analysis . . . 97
7.6 Discussion of Limitations . . . 99
7.7 Summary of Findings . . . 99

8 Bootstrapping Visual Assistant Development with Situated Interaction Simulation . . . 101

8.1 Overview . . . 101
8.2 Problem Definition . . . 103
8.3 The BASIS Framework . . . 104
8.3.1 Stage 1: Situated Interaction Simulation . . . 104
8.3.2 Stage 2: Autonomous Assistant Model Development . . . 106
8.3.3 Stage 3: Real-User Validation . . . 107
8.4 Instantiation of BASIS in Alexa Arena . . . 107
8.4.1 Alexa Arena Environment . . . 107
8.4.2 Implementation of BASIS Components . . . 109
8.5 Experimental Results Analysis . . . 110
8.5.1 Experiment Setups . . . 110

8.5.2 Result Analysis 111
8.6 Discussion of Limitations 115
8.7 Summary of Findings 115
9 Conclusion 117
9.1 Broader Implications 119
9.2 Limitations and Future Directions 119
9.3 Concluding Remarks 120

APPENDICES 122

BIBLIOGRAPHY 196

LIST OF FIGURES

FIGURE

3.1 We propose GROUNDHOG, a multimodal large language model that enhances its text output with pixel-level phrase grounding across diverse semantic granularities. The figure demonstrates outputs from our model on the four task types we considered in this work. 15

3.2 The model architecture of GROUNDHOG model. Given a set of class-agnostic entity mask proposals, the masked feature extractor first extracts the feature of each entity as the visual input of the multi-modal large language model (left). The output hidden states of the grounding tokens are averaged and used to retrieve the entities to ground, which will be merged into a single grounding mask for the phrase. Modules are colored by their trainability: parameter-free operators (grey), frozen (blue), trainable (orange), and partially trainable (mix). 16

3.3 GROUNDHOG can take arbitrary spatial prompts that can be resolved by an interactive segmentation model, such as SAM. The placeholder pointer token `<PTR>` will be replaced by the extracted entity features and fed as input to the model. 18

3.4 Illustration of the four grounding task types in the M3G2 dataset. 22

3.5 Examples of GROUNDHOG's performance in grounded image captioning. 26

3.6 Region caption using the best match proposal from Mask2Former+ versus from SAM. Mask2Former+ fails to propose the exact mask of the spire, leading to a less precise caption. 27

3.7 Illustration of a partially correct grounding. The grounding phrase and the ground truth mask are shown on the left. The top-4 mask proposals are presented, with highly-scored masks (green) selected for the merged mask, and low-scored masks (red) excluded. This illustrates the failure to recognize the word "KWIK" by the MLLM, despite its successful proposal. 29

4.1 The checker shadow illusion [3]. 33

4.2 Data augmentation examples for the Ebbinghaus illusion. 37

4.3 Illustration of the *SameDiffQA* setup. For each instance, the model is asked about its perception of an object property across two images, one illusion-free and one illusion-induced. For valid illusion evaluation, the model must initially identify identical properties in the illusion-free image. 38

4.4 Illustration of the *RefQA*, *AttrQA* and *RefLoc* setups. We flip the illusion image wherein the grounding outcome should also be inverted, to create a pair of images for each test. Model success requires accurate identification in both original and flipped versions to align with human responses. Matching human answers signals the model's capability to interpret illusions in a humanlike way, while a mismatch indicates otherwise. 39
4.5 Results of *SameDiffQA*. The number shows the percentage of the answers. Each cluster represents the distribution over humanlike, no-illusion and N/A answers from a model. The green and red line correspond to the linear regression of humanlike rate and no-illusion rate across all the model sizes. Except for OFA-Large, Unified-IO-Large, InstructBLIP-13B, the differences between the humanlike rate and the no-illusion rate are statistically significant $P < 0.005$. Details are in Table B.1 Appendix A. 42
4.6 Humanlike rate on *RefQA*, *AttrQA* and *RefLoc*. Each bar represents a different model size, arranged in ascending order from left to right. Note that LLaVA and InstructBLIP cannot predict object bounding boxes thus do not have the *RefLoc* results. 43
4.7 Humanlike rates of the largest model of each family, with finer-grained humanlikeness scores on each illusion category. 44
4.8 Attention weight distribution of OFA models for the *RefLoc* task. 45
4.9 Visualization of the attention maps generated by the OFA-Large model for the *RefLoc* task. In each row, the input image is shown on the left, and the attention map for the referential expression "smaller orange ball" is shown on the right. The attention maps surrounding the object of interest are highlighted for enhanced visibility. 46

5.1 An example task in ALFRED. 51
5.2 The structure of HiTUT. 54
5.3 Overview of HiTUT where unified transformers for sub-programs are integrated together. 56
5.4 Illustration of self-monitoring and backtracking. 57
5.5 Step-by-step prediction accuracies given the golden sub-goal/action history w.r.t. the proportion of training data on the unseen validation set. Each solid line corresponds to a specific input configuration. Dashed lines are the scores obtained using 100% data. 62

6.1 An example task in TEACh. 68

6.2 Illustration of our agent's reasoning process behind a single decision step. After receiving the current observation, the agent first updates its internal representation (left), then checks the current subgoal progress (upper right), and plans for the next steps (lower right). Finally the first action in the plan is popped out and grounds to the agent's ego-centric view for execution. In the pop-up boxes we show example object instances with their instance ids, states and positions in the 3D map. New instances and state changes are labeled in green. The status for each subgoal and action is labeled in front of it, where the arrows denote status transitions. 69
6.3 Overview of the subgoal learning process. Figure (a) shows the model architecture. Figure (b) shows the different input/output configurations we experiment with. Figure (c) illustrates the decoding process. We first predict completed subgoals and then predict the future subgoals conditioned on them, where we use different prompts to distinguish different types of subgoals. 72
6.4 Decision trees for handling exceptions encountered during plan execution. We term pre-execution errors as infeasible actions that cannot be instantiated in the situation, and post-execution errors as those when no outcome is observed after the action is executed. Red boxes denote situations based on the agent's belief. Green boxes denote exception handling strategies. 74
6.5 Comparison between HET and DANLI of relative success trajectory path lengths against ground-truth human trajectory lengths. The red dashed line denotes the same length as human trajectories, thus area to its left corresponds to the proportion of model predictions that achieved better efficiency than human performance. 76
6.6 Distribution of execution exceptions encountered and self-handled by our agent. The proportions of self-handled exceptions are in parentheses. The results are based on 2708 exceptions encountered in 1078 sessions from the validation unseen split. 81
6.7 Distribution of unrecoverable errors identified by our agent that cannot be recovered. 82

7.1 An example conversation between user and their task assistant. The assistant receives real-time video streams from the user's perspective, and provides proactive guidance to assist with the task. Images excerpted from Ego4D [90]. 86
7.2 A cooking task example from ProAssist, derived from EpicKitchen. The task goal, recpie and dialogue are generated through our synthetic data curation pipeline (dialogue partially shown due to space constraints). 88
7.3 **Left**: Streaming video-to-dialogue generation using VideoLLM-Online. The model processes live video frames and optional textual inputs to decide whether to speak or remain silent at designated decision points (yellow stars), and autogressively generates assistant responses as needed. Learning when to speak faces significant class imbalance due to the sparsity of speaking frames. **Right**: Illustration of iterative progress summarization. When approaching its context length limit, the model generates a concise task progress summary, then restarts generation with this summary incorporated into a new system prompt. 90

7.4 Model performance under different speaking decision threshold. The trade-off between precision and recall exists across both tasks. 94

8.1 **(Left)** Visual assistant interaction framework: The assistant and user share the same egocentric view but have complementary capabilities. The assistant draws on task knowledge to provide guidance through language communication, while the user follows the guidance and physically acts in the environment to achieve a specific goal. **(Right)** Example situated interactions where the assistant proactively offers guidance. 102
8.2 BASIS presents a three-stage pipeline for visual assistant development: (1) *Situated Interaction Simulation* generates contextually grounded data through conversational and physical interactions between an oracle assistant and simulated users in a virtual environment; (2) *Autonomous Model Development* leverages this data to train assistant models while using the simulated user for automated model evaluation; and (3) *Real-User Validation* assesses the trained assistants with human users to measure real-world usability. 105
8.3 Different types of environment observation and object info used in agent simulation. In partial object info, we use the fuzzy name to replace the precise name of uncommon objects (e.g. "machine" for "time portal generator") and hide the object state to simulate missing knowledge in object identification and state estimation. 108
8.4 Illustration of the oracle assistant (left), the simulated user (middle), and visual assistant modeling (right). 108
8.5 Intention differences between GPT-4o simulated users and real humans, comparing fuzzy vs. precise object naming. Positive/negative values indicate more/fewer instances than humans. Fuzzy naming better aligns with overall human behaviors (lower absolute differences) and better captures human uncertainty (more asking-related intentions). 112
8.6 Stepwise error analysis comparing the assistant model versus the oracle assistant during interactions with real human users. Speak errors occur when the assistant provides guidance containing inaccuracies or mistakes, while silent errors indicate instances when the assistant fails to provide guidance despite it being needed. . 114

A.1 Illustrations of the merged segmentation annotations from COCO Panoptic, LVIS, and PACO datasets. 123
A.2 Additional qualitative examples for short grounded captioning. 129
A.3 Additional qualitative examples for detailed grounded captioning. 129
A.4 Additional qualitative examples for referring expression segmentation. 130
A.5 Additional qualitative examples for grounded VQA of visual text. 130
A.6 Additional qualitative examples for referential dialogue with pointer input. . . . 131
A.7 Additional qualitative examples for referential dialogue with pointers as multiple choices input. 132

D.1 Overview of our neuro-symbolic system, which learns a rich, visually-informed semantic representation of the surrounding environment, and captures hierarchical tasks in two phases. First, it uses a sequence-to-sequence model to predict the sequence of high-level subgoals, both completed and upcoming, and additionally extracts some spatial relations from the dialog to supplement the world representation. For each subgoal, we apply a specialized planner (either for navigation or manipulation) over the world representation to accurately execute the low-level actions and achieve the subgoal. Best viewed in color. 138
D.2 Illustration of the 3D voxel map construction and interaction process of a making toast task. The agent is labeled in black, and the agent path is labeled in blue. Current navigation target in the agent's plan is labeled in red. 142
D.3 An example of subgoal annotations. 143
D.4 Demo of the Graphical User Interface (GUI) created for visualizing DANLI's internal representations. 147
D.5 Architecture of the hierarchical ET model. 148

E.1 ProAssistdataset statistics overview including task domain distribution, dialogue structure, duration variability, and assistant interaction patterns. Mean values for each distribution are provided in the legend. 165
E.2 Example of a cooking task in ProAssist. 166
E.3 Example of a cooking task in ProAssist. 167
E.4 Example of a cooking task in ProAssist. 168
E.5 Example of a assembly task in ProAssist. 169
E.6 Example of a laboratory task in ProAssist. 170
E.7 Example of a cooking task in ProAssist. 171

F.1 **Game Video Introduction (Tutorial 1/6)**: Your goal is to complete the task assigned to you with the help of an AI assistant. You can move around, interact with objects, and talk to the assistant. The assistant shares the same view as you do, and will provide you with instructions and guidance throughout the task. **Mission Goals:** Displays the goal of the current task. This is only visible to you but not the assistant. **Map & Current Room:** Shows your position in the world and the current room you are in. Score can be ignored for now. **Current Room Status:** Indicates the operational status of the room, such as whether it is powered or not. **Holding Item:** Displays the object you are holding. This area will be empty if you are not holding an item. **Note**: You can only hold one item at a time, so you need to place the current item down before picking up a new one. 188

F.2 **Action Control Panel (Tutorial 2/6)**: The **Action Control** panel lets you move around and interact with objects. Use **Movement Actions** for navigation, and **Object Manipulation** for actions like picking up or placing items, toggling machines and computers, and opening/closing doors. Hover over the question mark icon for details of each action. **Go to Object:** Quickly navigate to specific objects in view. **Go to Room:** Directly move to a specific room. **Note**: Go to Object/Room is much more efficient than moving around manually, so it is recommended to use them whenever possible. 189
F.3 **Show Observation (Tutorial 3/6)**: A pop-up window displaying all the interactable objects in the current view will show up after clicking the **Show Observation** button. It can help you better understand the objects and plan your actions. Objects that are not common in daily life are displayed with their base category (e.g., "machine") instead of their specific type (e.g., what type of machine it is). The assistant may help you identify them if they are relevant to the current task. 190
F.4 **Event History Intro (Tutorial 4/6)**: The **Event History** panel shows: (1) Chat history between you and the assistant; (2) Actions you have taken; (3) Action execution feedback when an action has failed (starting with [!]); (4) System messages (in grey) indicating the start and end of a task. You can use the input field at the bottom to send a message to the assistant. 190
F.5 **Blocking Page Intro (Tutorial 5/6)**: After you select an action or send a message, a blocking overlay will be shown while the system processes your request, which normally takes a few seconds. You can only continue after the overlay disappears. 191
F.6 **Give Up or Task Completion (Tutorial 6/6)**: The task will end if (1) you successfully complete the task; (2) you click the **Give Up** button; (3) max steps (50 actions) is reached. **Note**: the assistant is not perfect, so it is normal that the assistant's instructions are not always correct. You may choose to give up on a task at any time, when you feel getting stuck and the assistant is not helpful. After a task is finished, the system will automatically advance you to the next task. You will work on a total of 5 tasks in this session. 191
F.7 Assistant interface for task monitoring and debugging. 192
F.8 Annotation interface for assistant error anntation. 192
F.9 Illustration of the asymmetric task setting and key events during the interaction, where the oracle assistant provides assistance for the user to normalize a hammer using the embiggenator and use it to break a plate. 193
F.10 Illustration of interactions between the **trained model assistant** and a human user where the task is to fire the freeze ray. Two assistant errors are highlighted in Steps 5 and 8. 194

LIST OF TABLES

TABLE

3.1 The average Any-IoU of the proposals on each dataset. The vanilla Mask2Former is trained on the COCO-Panoptic dataset and our Mask2Former+ is trained on our combined dataset. Mask2Former+ obtains a consistent improvement in all scenarios, especially in non-COCO domains. 20
3.2 Summary of datasets included in M3G2. We show the availability of Grounding Annotations (Box, Mask, and Pointer inputs), the Semantic Granularity (Stuff, Things, Parts, Groups, and Text), and the number of text-image pairs for training. 21
3.3 Results on 7 Referring Expression Segmentation (RES) benchmarks with single instance queries [116, 178], multi-/null instance queries [151, 275] and reasoning-based queries [133, 210]. We report cIoU for RefCOCO/+/g and mIoU for other benchmarks, respectively. 24
3.4 Top-1 box recall results on Flickr30K-Entity [203]. 24
3.5 Phrase grounding results on PNG [84]. 24
3.6 Results on TextVQA-X [237] dataset. 24
3.7 Grounded Captioning on Flickr30K-Entity [203]. 25
3.8 Results on $\text{PointQA}_{\text{Twice}}$ [176] and V7W [317] test sets. 26
3.9 Object hallucination results on the POPE [147] benchmark. 28
3.10 Ablation study on model design choices and evaluation setups. Models are trained on RefCOCO+, Flickr30K, TextVQA-X and tested on corresponding validation sets. 30

4.1 Example illusion from each category and the corresponding explanations. . . . 35
4.2 Dataset statistics. 37
4.3 Pearson's correlation analysis between the humanlike rate and model size. Statistically significant results with $p < 0.05$ and $p < 0.1$ are marked with ** and *, respectively. 42

5.1 Statistics of data distribution in ALFRED. The number of annotations is equivalent to the number of tasks in each split. 57
5.2 Task and Goal-Condition success rates. The path length weighted version is in parentheses. The highest values per column are in **bold**. "-" denotes scores that are not reported. *G only* denotes only using the goal directive during evaluation without any sub-goal instructions. 59
5.3 Success rates of manipulation sub-goals on validation sets. The highest values per fold are in **bold**. 59

5.4 Success rates w.r.t. the allowed maximum backtracking number (#BT). 60
5.5 Success rate of the navigation sub-goal `Goto` with backtracking 60
5.6 Success rates of HiTUT with different parts of predictions replaced by oracle operations with expert demonstrations. N, M, SG and GR denote oracle navigation actions, manipulation actions, sub-goals and object grounding (i.e., mask generation) respectively. 61

6.1 Caption for LOF 75
6.2 Per task performance on the unseen split. Tasks are sorted in a decreasing order w.r.t. the number of average goal conditions. 77
6.3 Results on the TfD subtask of TEACh. 78
6.4 Performance of our panoptic segmentation model on valid seen and unseen split. PQ, SQ, RQ are panoptic quality metric, segmentation quality metric and recognition quality metric originally defined in [122]. 78
6.5 Accuracy of our state estimation model on both validation seen and unseen split. `isDirty`, `isFilledWithWater` and `isToggled` are the physical states that our system needs. 79
6.6 Subgoal prediction accuracy of DANLI on the validation unseen split, compared to ablations that do not condition on the action history, language forms of the actions, and self-predicted completed subgoals. 79
6.7 Ablation study of components with respect to the end task performance. *Last subgoal only* refers to directly working on the last predicted subgoal instead of all predicted subgoals one after another. *Without scene pruning* refers to using the planner without search space pruning. *Without replanning* refers to disabling replanning from the action failure recovery strategy. 80
6.8 PDDL planning success rate and average runtime. We set the overall time-out for domain translation and solution searching to be 120s in our experiments. . 81

7.1 Data statistics of ProAssistfor train/validation/test splits. More statistics in Appendix E.1.3. 90
7.2 Human evaluation of the generated dialogue quality. For HoloAssist and WTaG where human-collected dialogues are available, we evaluate them using the same approach for a side-by-side comparison with our generated dialogues. 95
7.3 Pearson and Spearman coefficient between our metrics and human judgment (*: $p < 0.05$, **: $p < 0.01$). 96
7.4 Match rate between human and metric-based selection of the best θ. 96
7.5 Model evaluation results. Comparisons can be made across different tasks (Action Narration vs. Dialogue Generation), model variants (I=1, 5, or 10), and knowledge access setups (w/ or w/o knowledge). P, R, and F1 are precision, recall, and F1 score, respectively. Corr., Prompt., Effic., and Overall are correctness, promptness, efficiency, and overall score, respectively. 96
7.6 Improvement from negative frame sub-sampling under different sub-sampling ratios. 97
7.7 Comparison between inference-time context management method for long video processing. 97

7.8 Per-subset performance across domains. 98

8.1 Performance comparison of different user variants paired with our oracle assistant, under different setups: with/without vision access (✓/✗) and precise/fuzzy naming (✓/✗) of uncommon objects. † indicates the simulated user selected for the following stages. 112
8.2 Performance comparison of different assistant models with our simulated user. All models are fine-tuned from Qwen2.5-VL 3B/7B. "GT Obj." denotes using ground-truth object information. † indicates the best assistant model selected for Stage 3. 113
8.3 In Stage 3 human evaluations, our best-performing assistant, trained entirely on synthetic data and without access to privileged information, achieves 82.3% of the oracle assistant's performance in SR@1, demonstrating strong generalization to real users. 114

A.1 Summary of the training datasets for Mask2Former+. Entity includes both thing and stuff categories. 123
A.2 The full list of datasets used in M3G2. 124
A.3 Instruction templates for the GIC task. 124
A.4 Templates used for the RES task. 125
A.5 Templates used for the GVQA task. 126
A.6 Templates used for the RD task. 127

B.1 Chi-square test for the SameDiff Task (Figure 5). #HL and #NI denote the number of humanlike illusionary answers and no-illusion answers, respectively. Statistically significant results with $p < 0.001$ and $p < 0.05$ are marked with *** and **. respectively. 134

C.1 Success rates across 7 task types on the validation sets. Highest values per fold are **bold**. 136
C.2 The validation success rates for models pretrained and trained from scratch with different allowed maximum number of backtrackings. 137

D.1 EDH results on the original validation splits. 148

E.1 Summary of egocentric video datasets used in ProAssist. The statistics presented are as originally reported in the corresponding papers before filtering. The number of tasks indicates the types of tasks in each dataset, except for Assembly101, where it represents the unique number of toys. Label abbreviations are as follows: C (coarse action labels), F (fine-grained action labels), M (mistake and correction labels), D (human-collected assistant-user dialogues), and R (ground-truth recipes). 149
E.2 Detailed training data statistics for different model variants under the maximum sequence length of $L = 4096$. We report the upsampling ratio ($\times$), the original dataset size after splitting and packing under different I, the final dataset size after upsampling, and the proportion of each data source in the final mixture. The final data size grows with the number of tokens used to encode each frame/image. 160

E.3 Selected speaking threshold θ for each model on each subset. We evaluate a series of θ values (0.1, 0.2, ...) for each setup and select the optimal threshold based on the appearance of a local maximum in F1 score as θ increases. 161

LIST OF APPENDICES

A Appendix for Chapter 3 . . . . . **122**

B Appendix for Chapter 4 . . . . . **133**

C Appendix for Chapter 5 . . . . . **135**

D Appendix for Chapter 6 . . . . . **138**

E Appendix for Chapter 7 . . . . . **149**

F Appendix for Chapter 8 . . . . . **172**

LIST OF ACRONYMS

AI Artificial Intelligence

LLM Large Language Model

MLLM Multimodal Large Language Model

VLM Vision-Language Model

M3G2 Multi-Modal Multi-Grained Grounding

SAM Segment Anything Model

GIC Grounded Image Captioning

GVQA Grounded Visual Question Answering

RES Referring Expression Segmentation

RD Referential Dialog

PNG Panoptic Narrative Grounding

CPG Caption Phrase Grounding

REG Referring Expression Generation

QA Question Answering

COCO Common Objects in Context

LoRA Low-Rank Adaptation

RGB Red Green Blue

DANLI Deliberative Agent for Natural Language Instructions

TEACh Task-driven Embodied Agents that Chat

EDH Execution from Dialog History

PDDL Planning Domain Definition Language

PLW Path Length Weighted

SR Success Rate

GC Goal Condition

HET Hierarchical Episodic Transformer

SGC Subgoal Completion

GUI Graphical User Interface

VIN Value Iteration Network

TfD Trajectory from Dialog

ProAssist Proactive Assistant dialogue dataset

FPS Frames Per Second

CoT Chain-of-Thought

NFS Negative Frame Sub-sampling

IPS Iterative Progress Summarization

WTaG Watch, Talk and Guide

IRB Institutional Review Board

RAG Retrieval-Augmented Generation

VQA Visual Question Answering

BASIS Bootstrapping Assistant development with Situated Interaction Simulation

ABSTRACT

Building multimodal AI assistants that can perceive the physical world, communicate naturally with humans, and help with real-world tasks is a cornerstone of AI research. Situated language grounding—the ability to connect language to rich, multimodal contexts—is fundamental for developing assistants that can interpret human instructions, reason about their environment, and provide timely responses. Despite advances in Large Language Models (LLMs) and their multimodal extensions (MLLMs) for processing multimodal inputs, effectively grounding language within the dynamic physical world to facilitate seamless human-AI interaction remains a significant challenge. This dissertation addresses this challenge by investigating situated language grounding across multiple dimensions and developing novel approaches for creating more contextually aware AI assistants.

In Part I, we explore language grounding in visual perception. We first introduce GROUNDHOG, a multimodal large language model designed for holistic visual segmentation across various semantic granularities. By leveraging innovative architectural designs and a curated multi-granularity dataset, this model outperforms previous task-specific approaches on a range of grounding tasks using a single model. We then examine the behavior of vision-language models under a unique setup of visual illusions. We propose GVIL, a systematic benchmark to test whether models are able to perceive visual illusions similarly to humans or represent reality faithfully, and demonstrate critical insights into shared and misaligned patterns between human and machines. These contributions advance our understanding of how multimodal models link language to vision in both standard and challenging perceptual scenarios.

In Part II, we extend our investigation to language grounding in embodied planning within 3D environments. We propose two neural-symbolic approaches to enable assistants to follow natural language instructions in such settings. First, HiTUT introduces a unified transformer model for hierarchical task planning, allowing the use of the same model for both high-level task decomposition and low-level action prediction. Second, DANLI constructs semantic representations of the physical environment and integrates a symbolic planner for transparent and efficient task execution. These complementary approaches demonstrate how assistants can leverage grounded understanding to interact effectively within complex

environments.

In Part III, we focus on grounding language in dynamic and interactive settings, where assistants must process continuous visual input from a user's first-person perspective and engage proactively. We propose two complementary approaches. First, ProAssist provides a framework for generating realistic user-assistant dialogues from egocentric videos, harnessing large-scale real-world video data to train proactive assistants. Second, BASIS presents a simulation-based pipeline that supports scalable assistant development and evaluation through situated interactions in virtual environments, reducing the need for costly human intervention. These works provide foundations of scalable development for interactive, perceptual egocentric assistant systems.

Together, these contributions advance assistant modeling by introducing novel tasks, datasets, and modeling strategies for grounded, context-aware human-AI interaction.

CHAPTER 1

Introduction

The pursuit of artificial intelligence (AI) systems capable of seamlessly collaborating with humans in the physical world has driven significant progress across the domains of language, vision, and robotics. Among these efforts, multimodal AI assistants—agents that perceive, interpret, act and communicate with humans using complex sensory inputs—represent a particularly promising direction. Such assistants must go beyond text understanding and learn to ground language in the rich, dynamic contexts that characterize human environments.

A central capability enabling this vision is *situated language grounding*—the process of linking linguistic expressions to percepts, objects, actions, and events within an environment. As early as Stevan Harnad's seminal work on the *symbol grounding problem* [95], it was argued that linguistic symbols remain semantically hollow unless they are anchored in perceptual and sensorimotor experiences. In other words, the meaning of a word is not inherent in its symbolic form but arises from its connection to observable phenomena. This insight laid the foundation for subsequent research in vision-and-language, where early efforts focused on mapping words to visual objects, attributes, and actions [130, 203, 75, 231, 167]. More recently, vision-language models (VLMs) and multimodal large language models (MLLMs) have scaled up learning of vision-language alignment from massive amounts of text and image data from the internet, resulting in effective grounding of visual content to language tokens in the latent space of neural networks [213, 264, 156, 142, 9, 315, 249, 2, 165].

While symbol grounding is a necessary first step, it is not sufficient for robust language understanding and reasoning. The next layer, which we refer to as *experience grounding*, argues that language is truly meaningful only when connected to the agent's own sensorimotor interactions. As highlighted by Bisk et al. [23], language understanding emerges not merely from textual patterns but from actively engaging with the world—observing, acting, and experiencing the consequences of those actions. In human development, words are learned in the context of doing and perceiving; similarly, AI systems must incorporate sensorimotor experience to resolve ambiguities in instructions and infer causality or spatial relationships in

dynamic settings [99, 231, 112].

Yet, both perceptual and experiential grounding fall short of capturing the full spectrum of human communication. In real-world settings, meaning is co-constructed through interaction. This brings us to the third level: *communicative grounding*. Drawing from theories of common ground in psycholinguistics [46] and developed in works such as Chai et al. [30] for human-machine collaboration, communicative grounding emphasizes the dynamic, interactive process by which communication partners iteratively negotiate shared knowledge, beliefs, and assumptions. Whether through clarifying questions, feedback, or adaptive dialogue, achieving mutual understanding is essential for an AI assistant to provide timely and effective assistance to humans in the dynamic and complex physical environments we live in [25, 152, 29, 193]

Together, these three conceptual layers—symbol grounding, experience grounding, and communicative grounding—form a comprehensive theoretical framework for situated language grounding. They illustrate how language can be initially anchored in perceptual representations, then enriched through sensorimotor experience, and finally refined via dynamic interaction. This framework not only justifies the need for multimodal approaches in AI but also guides the design of systems that are both robust and context-aware to communicate effectively with humans.

1.1 Challenges in Situated Language Grounding

Despite recent breakthroughs in large-scale pretraining, grounding language in dynamic, physical environments presents several key challenges, spanning all three layers of the grounding framework:

- **Fine-grained Visual Grounding:** Existing multimodal models often rely on bounding boxes to localize visual elements [301, 199, 38, 33, 308], which fall short in representing amorphous objects, object parts, or overlapping entities. There is a need for pixel-level grounding across semantic granularities to enable more precise and interpretable connections between language and vision. This challenge echoes the fundamental issue of symbol grounding—ensuring that words are accurately tied to the rich detail of visual percepts.

- **Grounding under Perceptual Ambiguity and Adversarial Conditions:** Standard benchmarks often test grounding in clear, unambiguous visual scenes [116, 14]. However, human perception is complex, noisy, and sometimes misleading under challenging perceptual scenaiors, such as visual illusions [57, 27]. Understanding how AI models ground language in such scenarios—whether they align with human-like perception,

which can be misled by illusions, or maintain a faithful representation of reality—is critical for building robust assistants.

- **Grounding Actions in 3D Embodied Environments:** To follow instructions in 3D spaces, agents must construct internal semantic representations and reason about spatial, temporal, and causal dynamics. This requires decomposing tasks, managing partial observability, and planning under uncertainty—capabilities where current MLLMs often falter [257, 41, 279, 112, 285, 306]. This challenge highlights the need for experience grounding, where language is integrated with sensorimotor interactions to enable effective decision-making and robust action planning in physical environments.

- **Grounding Communication in Dynamic Perceptual Contexts:** Next-generation egocentric visual assistant systems take in the video streams from the user's perspective (e.g., through smart glasses), and must process the signals in real-time and and engage in interactive dialogues with the users. Learning to interpret ongoing activities, anticipate user needs, and provide timely assistance requires adapting dialogue to the evolving context—a capability underexplored in current systems [21, 36]. This challenge highlights the need for developing more capable models that go beyond static multimodal reasoning, towards dynamic perceptual understanding and communicative grounding.

- **Scalable Data Collection and Evaluation for Situated Interaction:** Training models for complex interaction requires vast amounts of interaction data that capture diverse scenarios, user behaviors, and environmental dynamics. Furthermore, evaluating the effectiveness of assistance in interactive settings is inherently complex. Traditional approaches relying on human data collection and human-in-the-loop evaluation [21, 267, 241, 136, 218, 177, 193] are extremely costly, time-consuming, and difficult to scale. Overcoming this bottleneck requires developing novel, scalable methodologies for data generation and evaluation, to accelerate progress in building truly interactive and helpful AI assistants.

1.2 Dissertation Structure and Contributions

This dissertation addresses the question: *How can we develop multimodal AI assistants that ground language within visual and embodied contexts to enable robust perception, efficient planning and proactive interaction in dynamic, situated environments?* Tackling this question expands the frontier of multimodal learning and contributes to building AI assistant systems that are transparent, adaptive, and capable of supporting users in complex tasks. In Chapter 2,

we briefly review the relevant background and related work before detailing our contributions, which are organized into three interconnected parts:

Part I: Language Grounding in Visual Perception (Chapters 3 and 4). This part tackles the challenge of symbol grounding for fine-grained vision-language alignment, and under challenging perceptual conditions of visual illusions. We first introducing GROUNDHOG, a multimodal large language model (MLLM) designed to achieve pixel-level grounding across diverse semantic granularities. GROUNDHOG employs a mask proposal network [42] to extract visual entities from the input image, and uses the per-entity features as input to the MLLM backbone [156]. The model is then trained for grounded language generation, where texts are generated jointly with the corresponding visual entity selected from the proposals. To train GROUNDHOG, we curate the M3G2 dataset, a comprehensive collection of 2.5M text-image pairs covering four grounding tasks: Grounded Image Captioning (GIC), Referring Expression Segmentation (RES), Grounded Visual Question Answering (GVQA), and Referential Dialogue (RD). By precisely aligning linguistic symbols with regions of various semantic granularities, GROUNDHOG addresses the limitation of bounding-box-based models, reduces object hallucination, and provides interpretable diagnostics for failure cases.

While models like GROUNDHOG advance fine-grained grounding, a crucial test of their perceptual understanding involves scenarios where human perception deviates from physical reality. Therefore, we subsequently introduce GVIL, the first benchmark designed to systematically test language grounding of MLLMs under visual illusions. This benchmark assesses whether models exhibit human-like understanding when visual illusions present, or if they remain veridical, faithfully representing the underlying physical properties despite misleading cues. By comparing model predictions from 4 state-of-the-art MLLMs [264, 165, 48, 156] to human perceptual norms, GVIL uncovers fundamental shared patterns and divergences in how machines and humans interpret ambiguous visual information.

Part II: Language Grounding in Embodied Action Planning (Chapters 5 and 6). Building upon the foundational visual grounding explored in Part I, this part shifts the focus to grounding language in embodied action and planning within complex 3D environments. We propose two complementary neural-symbolic approaches that enable AI assistants to follow natural language instructions for complex, multi-step tasks. First, in Chapter 5, we introduce HiTUT, an hierachical task learning approach that decomposes complex tasks into three sub-problems: sub-goal planning, scene navigation, and object manipulation. HiTUT employs a unified Transformer architecture [258] capable of handling all three sub-tasks within a single model, enabling the model to learn task structures and action policy

jointly. This hierarchical representation facilitates self-monitoring and backtracking during execution, leading to significantly improved generalization and task success rates on the ALFRED benchmark [234], particularly in unseen environments.

Next, complementing the learned predictive policy of HiTUT, we present DANLI (Chapter 6), a deliberative neuro-symbolic agent that emphasizes explicit world modeling and symbolic planning. DANLI constructs detailed semantic representations of the physical environment from visual and spatial inputs. It leverages language models [137] to predict intermediate subgoals from high-level natural language instructions and employs a classical symbolic planner [4] to translate these abstract goals into concrete, executable action sequences. Evaluated on the TEACh benchmark [193], DANLI demonstrates how explicit world modeling and symbolic planning can enhance the effectiveness, efficiency, and interpretability of embodied agents. Together, HiTUT and DANLI showcase different, yet synergistic, ways to integrate language understanding with planning for effective interaction in complex environments.

Part III: Language Grounding in Situated Communication (Chapters 7 and 8). Extending beyond static perception and single-agent environments, the final part addresses communicative grounding, enabling multimodal AI assistants to engage proactively with users in situated, task-oriented interactions. We focus specifically on scenarios where users perform tasks while wearing smart glasses, which provide the assistant with a shared, continuous egocentric video stream. Based on this real-time visual input, the assistant must interpret the user's surroundings and ongoing activities to offer timely and relevant task assistance through natural language. This part introduces two complementary contributions aimed at developing scalable methods for training and evaluating such interactive, egocentric assistants. In Chapter 7, we present ProAssist, a large-scale synthetic dataset featuring situated assistant-user dialogues grounded in continuous streaming video from diverse sources and task domains. Leveraging ProAssist, we propose novel metrics specifically for evaluating proactive dialogue generation and develop an end-to-end MLLM trained to jointly learn when and what to communicate based on visually observed user activities.

Complementing these data generation efforts, Chapter 8 introduces BASIS, a simulation-based pipeline designed to bootstrap the development and evaluation of visual assistants via situated interaction simulations. This framework facilitates scalable training by simulating interactions between MLLM-powered assistants and simulated users within virtual environments, significantly reducing the need for expensive and often limited human-in-the-loop data collection and evaluation. Our experiments validate this approach, demonstrating that models trained solely on synthetic interaction data generated by BASIS can effectively transfer their learned capabilities to guide real users through complex tasks in the physical world, thereby

paving the way for more efficient and scalable development cycles for interactive AI assistants.

Finally, we conclude the dissertation by summarizing the contributions of each part and discussing potential future directions towards the goal of building robustly grounded, context-aware, and interactive multimodal AI assistants (Chapter 9).

CHAPTER 2

Related Work

This thesis builds upon and contributes to several key research areas centered on enabling multimodal AI assistants to robustly ground language in situated environments. This chapter provides an overview of the relevant background and prior work, contextualizing the specific contributions made in subsequent chapters. We organize this review around the core challenges addressed by this dissertation:

- Achieving robust visual grounding in Multimodal Large Language Models (MLLMs), including fine-grained pixel-level understanding and evaluating perception under challenging conditions like visual illusions (Section 2.1, relevant to Chapter 3 and Chapter 4).
- Enabling robust embodied action planning based on grounded language instructions, exploring both learned hierarchical policies and explicit neuro-symbolic approaches (Section 2.2, relevant to Chapter 5 and Chapter 6).
- Facilitating dynamic, situated communication for interactive task guidance, including addressing proactivity, data scarcity, and scalable evaluation challenges (Section 2.3, relevant to Chapters 7 and 8).

2.1 Multimodal Large Language Models

The foundation for visually-aware AI assistants lies in Multimodal Large Language Models (MLLMs), which extend the capabilities of Large Language Models (LLMs) to process and reason about visual information alongside text.

2.1.1 Evolution and Capabilities

The rapid progress in LLMs has catalyzed the development of MLLMs [256, 9]. Techniques like visual instruction tuning have enabled models such as MiniGPT4 [315], LLaVA [154, 153],

InstructBLIP [48], and others [83, 289, 266, 138] to perform a wide range of vision-language tasks with impressive fluency, often trained with relatively modest resources. However, despite these advances, a critical challenge remains: *object hallucination*, where models generate text referring to objects not present in the visual input [115, 49, 147]. This issue highlights the need for stronger mechanisms to anchor language firmly in visual perception, aligning with Harnad's concept of symbol grounding [95].

2.1.2 Grounded MLLMs

Explicitly connecting language to visual referents—grounding—is crucial for reliable communication about the physical world [144, 171]. Several MLLMs have incorporated grounding mechanisms, often by representing visual regions (typically bounding boxes) as discrete tokens within the input sequence [288, 166, 264]. Examples include GPT4ROI [301], Kosmos-2 [199], Shikra [38], PVIT [33], BuboGPT [308], Qwen-VL [17], and Ferret [292]. While demonstrating the value of grounding, these models predominantly rely on bounding boxes. This reliance presents a significant limitation: bounding boxes are ill-suited for representing amorphous entities (e.g., 'a pile of sand'), object parts (e.g., 'the handle of the mug'), or precisely delineating overlapping objects. Consequently, these models struggle with fine-grained visual understanding and lack pixel-level precision, hindering their applicability in scenarios demanding detailed spatial awareness and limiting diagnostic capabilities when failures occur. Research in language-guided semantic localization, which aims to segment regions based on referring expressions [293, 179, 294, 201, 133], highlights the need for pixel-level understanding but often focuses on specific tasks rather than general-purpose MLLM capabilities.

Our Contribution: This thesis addresses the limitation of coarse, bounding-box-based grounding in Chapter 3 by introducing **GROUNDHOG**. GROUNDHOG is specifically designed for fine-grained, pixel-level visual grounding across diverse semantic granularities. Its modular architecture, utilizing plug-and-play mask proposals, allows it to precisely link linguistic expressions to corresponding pixels, overcoming the limitations of bounding boxes. Trained on the comprehensive M3G2 dataset, GROUNDHOG demonstrates enhanced capabilities in tasks requiring precise localization (GIC, RES, GVQA, RD). Our contribution lies in developing a universally applicable grounded MLLM that improves grounding precision, reduces object hallucination through tighter visual anchoring, and enhances interpretability by providing mask-level outputs, thus establishing a stronger foundation for symbol grounding.

2.1.3 Evaluating Visual Illusions in MLLMs

Human visual perception is known to diverge from physical reality under certain conditions, creating visual illusions where properties like color or size are perceived differently from their objective measurements [27, 92]. These perceptual discrepancies are not mere curiosities; they can influence human action and language use [79, 28], highlighting that human understanding is shaped by both physical stimuli and cognitive interpretation. While prior research explored machine understanding of visual illusions, it primarily focused on vision-only models like CNNs [80, 22, 82], and typically by assessing the model's internal representations [162, 5]. However, these efforts largely overlook the critical role of language in describing and grounding these perceptual experiences. As a result, how MLLMs, which integrate vision and language, perceive and communicate about such illusory stimuli remained largely unexplored, leaving a gap in understanding whether their grounding mechanisms reflect objective reality or subjective human perception.

Our Contribution: This thesis addresses the gap in evaluating MLLM perception under illusionary conditions in Chapter 4 by introducing the **GVIL** benchmark. GVIL is the first benchmark specifically designed to systematically assess whether MLLMs perceive visual illusions similarly to humans or maintain fidelity to objective reality, evaluated through language-based interaction. Using a curated dataset covering five illusion categories and four tasks (SameDiffQA, RefQA, AttrQA, RefLoc) grounded in natural language, GVIL probes the alignment between model behavior and human subjective perception when faced with misleading visual cues. Our contribution lies in the development of GVIL and the presentation of initial findings from evaluating 4 state-of-the-art MLLMs on this benchmark. This work offers critical insights into the nature of visual grounding in MLLMs when faced with discrepancies between physical reality and subjective human perception.

2.2 Embodied AI for Instruction Following

Moving beyond static image understanding, embodied AI agents operate within 3D environments, perceiving, reasoning, and acting to accomplish tasks, often guided by language instructions. This necessitates grounding language not just in perception (symbol grounding) but also in the agent's sensorimotor interactions and experiences within the environment (experience grounding).

2.2.1 Simulation Environments

The development of embodied agents has been greatly facilitated by simulation platforms like AI2-THOR [125], Habitat [225, 248], iGibson [233, 139], and others [206, 60, 74, 76]. These environments allow for scalable training and testing, often incorporating procedural generation [61, 286] to improve robustness and aid sim-to-real transfer [252, 60, 68]. Some simulators even model human behavior [206, 77, 207]. However, while physical interaction is well-simulated, realistic simulation of situated verbal communication remains a gap, which we address later (Section 2.3).

2.2.2 Key Techniques for Embodied Instruction Following

Successfully following instructions in an embodied setting relies on several key capabilities:

- **Semantic Scene Representation:** Agents require internal models and representations of their surroundings. Techniques range from vectorized encodings capturing implicit spatial and semantic information [197, 244, 245, 118] to more structured symbolic scene graphs [15, 120, 145, 7]. Recent work has focused on semantic maps [32, 223, 24, 182], which explicitly integrate spatial layouts with object semantics to facilitate planning. A limitation of many existing map representations is the lack of detailed instance-level object information, including physical states such as object openness, position, or orientation, which are crucial for interacting with symbolic planners.

- **Hierarchical Policies:** Long-horizon tasks involving complex sequences of actions are often tackled using hierarchical frameworks [64, 247, 180, 127, 52, 131, 16]. These approaches typically decompose a high-level goal or instruction, often derived from language, into a sequence of simpler subgoals or sub-tasks [230, 313, 24, 309, 56]. Explicitly modeling task structure can enable better planning and potential recovery mechanisms [170, 117]. However, a common limitation in some hierarchical systems is the reliance on purely reactive policies for low-level action execution, which can lack foresight and struggle with complex interactions or recovery scenarios compared to more deliberative planning approaches.

- **Symbolic Planning:** Classical planning techniques [4, 71, 114, 72] offer systematicity, interpretability, and the ability to reason over long horizons. There is growing interest in bridging these symbolic methods with learned representations derived from language [231, 296, 236, 175] or vision [134, 282]. However, challenges remain in effectively combining high-level goal understanding (e.g., from LLMs) with detailed, perception-grounded world states and robust, multi-level planning algorithms specifically tailored

for embodied interaction challenges like object search, partial observability, and failure recovery.

Our Contributions: Part II of this dissertation presents two complementary approaches to address the challenges of experience grounding and instruction following in embodied agents. In Chapter 5, we introduce **HiTUT**, addressing the problem by learning implicit hierarchical task structures. It departs from opaque end-to-end models by decomposing tasks and using a unified transformer to predict sub-goals and actions. This explicit hierarchical modeling facilitates self-monitoring and backtracking, leading to state-of-the-art performance and significantly improved generalization on the ALFRED benchmark [234]. HiTUT demonstrates the power of learned hierarchical representations for robust task execution.

Complementing this, Chapter 6 introduces **DANLI**, a deliberative neuro-symbolic agent emphasizing explicit world modeling and reasoning. DANLI constructs detailed semantic maps capturing object instances and states, overcoming limitations of less informative representations. It integrates language models for high-level understanding with multi-level symbolic planning for robust, interpretable navigation and interaction, addressing the limitations of purely reactive policies. We enhance the planner with capabilities for object search and recovery. Together, HiTUT and DANLI advance the state-of-the-art by demonstrating synergistic ways—through learned predicitive policies (HiTUT) and explicit symbolic planning (DANLI)—to achieve more effective, robust, and interpretable grounding of language instructions in complex embodied actions.

2.3 Interactive Visual Assistants and Situated Communication

The ultimate goal for many multimodal systems is to act as helpful assistants, collaborating with humans in real-world tasks. This requires moving beyond static grounding to dynamic, interactive *communicative grounding*, where the assistant understands the user's context from potentially streaming visual input (e.g., egocentric video) and provides timely, relevant guidance.

2.3.1 Evolution and Challenges of Task Guidance Systems

Task guidance systems have progressed from rule-based approaches [191, 190] to perception-enabled solutions [136, 218, 168, 268, 224, 21], significantly boosted recently by powerful MLLMs [9, 2, 156, 18, 263, 249, 21, 267, 36]. Much research focuses on component skills like

understanding [273, 107], behavior analysis [50, 90, 104], or mistake detection [227, 135, 198]. However, several challenges hinder the development of truly interactive, general-purpose assistants:

- **Proactive and Timely Interaction:** Assistants need to decide when and how to intervene based on streaming input, anticipating user needs rather than just reacting to completed actions or errors. Effective proactivity requires understanding the user's progress within a task and predicting potential difficulties or next steps to offer seamless guidance[181, 186]. Models processing full offline videos [143, 173, 300] are unsuitable for this real-time interaction dynamic. While online models like VideoLLM-Online [36] represent progress, achieving sophisticated, context-aware proactive decision-making, especially for complex, long-horizon task guidance, remains an underdeveloped capability.

- **Data Scarcity:** Training robust interactive assistants requires vast amounts of situated, conversational data that captures the nuances of real-world collaboration grounded in shared visual context. While the collection of egocentric video data is growing [90, 227, 91, 50, 104], the data are mainly for capturing the action and activity from the user, while interaction between the assistant and the user is absent. Recent work relies on Wizard-of-Oz (WoZ) collection methods [172, 21, 267] to collect such dataset. This approach typically involves configuring tasks in lab environments and recruiting human participants to play both roles: the user performs the task while the assistant, provided with instructions (e.g., a recipe), guides the user via a microphone. This process is prohibitively expensive and slow to scale, making it difficult to cover the diversity of tasks, environments, and interaction styles needed.

- **Scalable Evaluation:** A major challenge in evaluating assistants lies in assessing their performance within dynamic, user-driven interactive settings. Existing static benchmarks are typically based on pre-recorded videos, and only focus on isolated sub-tasks such as environment understanding [273, 107], behavior analysis [50, 90, 104], or mistake detection [227, 135, 198]. While valuable, these benchmarks are insufficient for evaluating the overall effectiveness and helpfulness of assistants, particularly in terms of dialogue quality. More critically, they suffer from a fundamental limitation: the absence of user interaction makes it impossible to incorporate user feedback or assess how the assistant's guidance influences user behavior. As a result, developing scalable and reliable evaluation methodologies that capture the quality and impact of the entire interactive process remains a pressing open challenge.

2.3.2 Synthetic Data and Simulation

To address the data bottleneck, synthetic dialogue generation using LLMs has shown promise [228, 216, 196, 148, 1, 39], including generation grounded in visual content [128, 276, 298, 184, 156, 173, 169, 36]. However, generating realistic, proactive assistant-user dialogues specifically for egocentric task guidance is less explored. Furthermore, while embodied simulators excel at physics, the simulation of rich, situated verbal interactions within these environments is nascent [77, 194, 200], leaving a gap in tools for scalable development and evaluation of communicative grounding.

Our Contribution: Part III of this thesis explores two complementary directions to tackle the challenges of scalable data collection and evaluation for interactive visual assistants: (1) synthesizing situated dialogue grounded in real-world video, and (2) simulating interactive behaviors in virtual environments.

In Chapter 7, we introduce the **ProAssist**, a large-scale dataset of proactive assistant-user dialogues generated from diverse egocentric videos using an LLM-driven pipeline. We also propose novel metrics specifically for end-to-end evaluation of proactive dialogue generation from streaming video. We develop an end-to-end MLLM trained on ProAssist that learns when and how to speak based on streaming video input. Our contribution here is advancing the modeling and evaluation of proactive communication for visual task guidance.

In Chapter 8, we present **BASIS**, a simulation-based framework that enables scalable generation of situated, interactive training data and automated evaluation of visual assistants. BASIS simulates interactions between MLLM-powered assistants and users within 3D environments, directly addressing the data and evaluation bottlenecks. We show that models trained solely on BASIS's synthetic data can effectively guide real users. Our contribution is a novel methodology and platform that systematically bootstraps the development cycle of interactive visual assistants, making the pursuit of communicative grounding more feasible and scalable.

PART I: Language Grounding in Visual Perception

CHAPTER 3

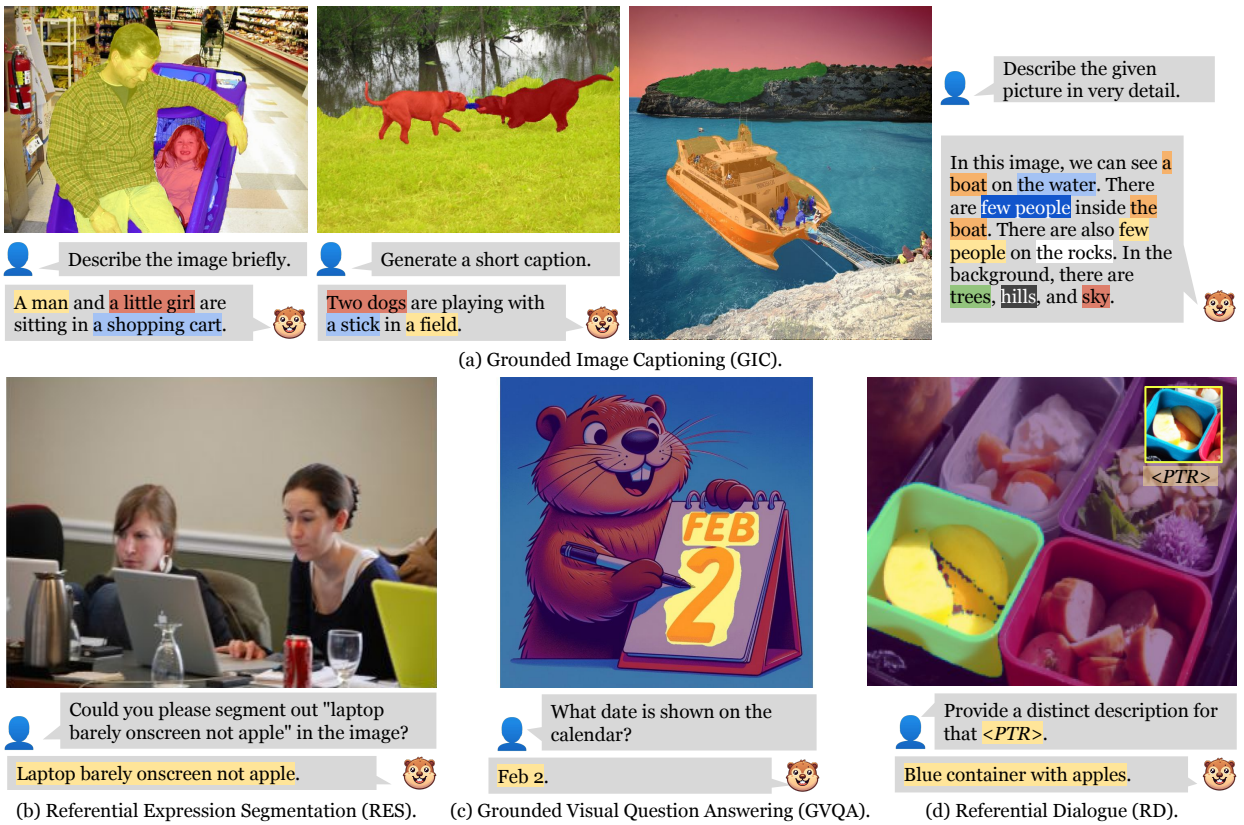
Grounding Large Language Models to Holistic Segmentation

Building on the foundational motivation for situated language grounding introduced in Chapter 1, Part I of this dissertation begins by addressing the core challenge of symbol grounding: connecting linguistic expressions to the rich perceptual details of the visual world. While recent multimodal large language models (MLLMs) have shown impressive capabilities, their reliance on coarse bounding boxes often constrains grounding precision. This limitation hampers fine-grained understanding and interpretability, particularly for amorphous objects, object parts, or overlapping entities. To bridge this gap, this chapter focuses on advancing pixel-level visual grounding across diverse semantic granularities, enabling more precise and holistic understanding. The content of this chapter is based on the work published in CVPR 2024[1].

3.1 Overview

Multimodal large language models (MLLMs) have received an increasing amount of attention to address tasks that necessitate non-linguistic knowledge, e.g., perception and reasoning about the visual world [291, 140]. For fine-grained visual understanding, grounded MLLMs often learn language-to-object grounding by causal language modeling, where grounded

[1]Yichi Zhang, Ziqiao Ma, Xiaofeng Gao, Suhaila Shakiah, Qiaozi Gao, and Joyce Chai. "GROUNDHOG: Grounding large language models to holistic segmentation." Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2024 [303].



(a) Grounded Image Captioning (GIC).

(b) Referential Expression Segmentation (RES). (c) Grounded Visual Question Answering (GVQA). (d) Referential Dialogue (RD).

Figure 3.1: We propose GROUNDHOG, a multimodal large language model that enhances its text output with pixel-level phrase grounding across diverse semantic granularities. The figure demonstrates outputs from our model on the four task types we considered in this work.

objects are captured by bounding boxes as sequences of location tokens. However, bounding boxes are often inadequate for capturing amorphous entities [26], semantic parts of objects [86], finer-grained regions with irregular shapes [94], or groups of instances at the same time. As a result, a single bounding box can often include other irrelevant semantics in order to engulf the target entities, leading to ambiguity in detection. In addition, the generated box coordinate lacks interpretability. When the model hallucinates, such as incorrectly predicting the association between objects and language, it is hard to diagnose whether the problem is due to the model's failure to detect the object, or its incorrect alignment of the object with language.

To address these issues, in this work, we introduce GROUNDHOG, an MLLM developed by grounding Large Language Models to holistic segmentation. Our goal of language grounding is to connect text spans that refer to or can be deduced from visual information, termed as *groundable phrases* [171], to their corresponding regions of visual entities (Figure 3.1). GROUNDHOG incorporates a masked feature extractor that takes an input image and a

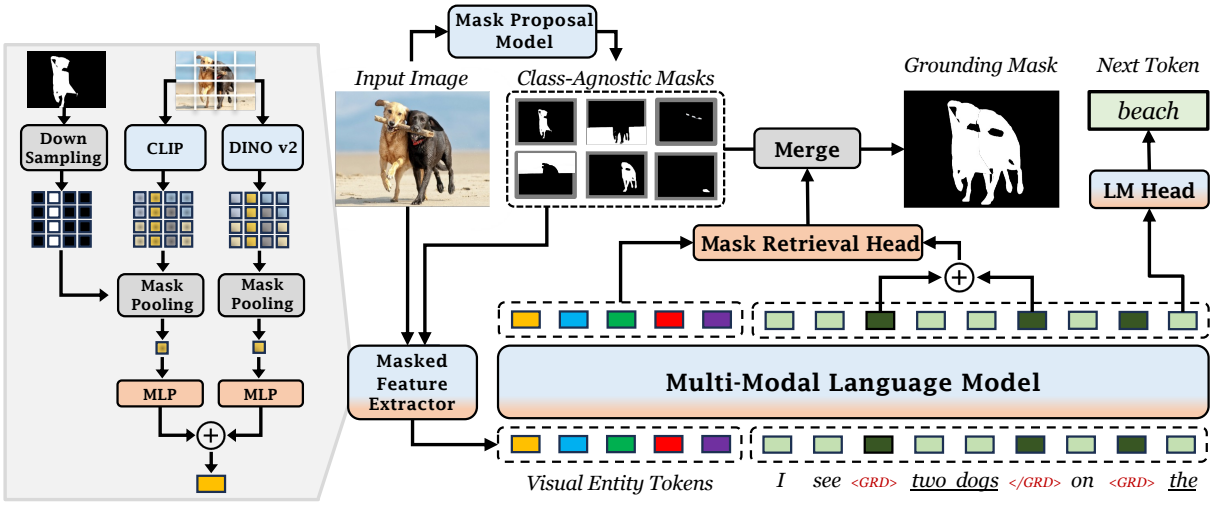


Figure 3.2: The model architecture of GROUNDHOG model. Given a set of class-agnostic entity mask proposals, the masked feature extractor first extracts the feature of each entity as the visual input of the multi-modal large language model (left). The output hidden states of the grounding tokens are averaged and used to retrieve the entities to ground, which will be merged into a single grounding mask for the phrase. Modules are colored by their trainability: parameter-free operators (grey), frozen (blue), trainable (orange), and partially trainable (mix).

set of class-agnostic entity mask proposals, and converts each mask's features into visual entity tokens for an MLLM backbone. This MLLM then connects groundable phrases to unified grounding masks by retrieving and merging the entity masks. Compared to previous grounded MLLMs, GROUNDHOG unlocks unprecedented pixel-level vision-language alignment. It naturally supports visual pointers as input, and can plug-in-and-play with any choice of mask proposal networks, e.g., Segment Anything Model (SAM) [123], domain-specific semantic segmentation models, or user-provided mask candidates. We introduce an enhanced Mask2Former [42] as our default mask proposal network, which detects regions at multiple granularities, e.g., instances (things and stuff), semantic parts, and visual text, leading to a holistic coverage of visual semantics.

To train GROUNDHOG, we curated a Multi-Modal Multi-Grained Grounding (M3G2) dataset consisting of 2.5M text-image pairs for visually grounded instruction tuning, consisting of 36 sub-problems derived and augmented from 27 existing datasets. We present extensive experiments on vision-language tasks that require grounding, including grounded language generation with minimal object hallucination, language-guided segmentation, visual question answering with answer grounding, and referential dialog with spatial pointer inputs (Figure 3.1). Our empirical results show that GROUNDHOG, without task-specific fine-tuning, can achieve superior or comparable performance with previous models that either require fine-

tuning or are specialized only for that dataset. In addition, GROUNDHOG has supports easy-to-understand diagnosis when grounding fails.

3.2 GROUNDHOG Model Architecture

The language grounding task can be succinctly delineated into two fundamental components: *localization* and *recognition*, as established in the literature [239, 312, 171]. Such categorization not only aids in the identification of object presence (objectness) without reliance on specific object classes, but also sets the stage for models to be robust in open-vocabulary settings. Building upon this framework, we formulate the grounding process as an *entity segment selection* problem, which involves (1) proposing entity segmentation masks where the masks encapsulate regions with discernible semantic content, and (2) recognizing the retrieved entities through the understanding of both visual and language context. Concurrently performing both tasks is where MLLMs bring a distinct advantage. This decoupled design of entity mask proposal and language-guided grounding brings several advantages. First, it allows independent improvement of the mask proposal model and MLLM, where specialized data, training, and inference setups can be applied. Second, by decoupling language grounding, it becomes straightforward to determine if a failure is due to the model's inability to propose the entity segment, or its misalignment of the object with the language, thus improving the interpretability of the whole framework. Third, as shown later, when connecting the two parts to work in tandem in a model-independent manner, the MLLM can benefit from multiple different vision specialist models in a plug-and-play fashion. In the remainder of this section, we give details of our model design.

3.2.1 Building Entity Features from Masks

Our approach assumes the availability of a mask proposal model, which is capable of generating a set of class-agnostic entity masks from an image with high coverage. In contrast to prior studies that relied on low-level features [154, 199, 38, 48], GROUNDHOG interprets the image as a collection of entities. The primary challenge then becomes the derivation of effective visual features to accurately represent these entities. To achieve a complete decoupling of the MLLM from the mask proposal model responsible for providing the masks, we propose to condition the entity features solely on the binary masks without using any embeddings from the mask proposal model. Specifically, the mask corresponding to each entity is employed to extract patch features from pretrained vision foundation models, such as CLIP [211] and DINOv2 [192], through a convolutional mask pooling layer [47]. Given that the feature map

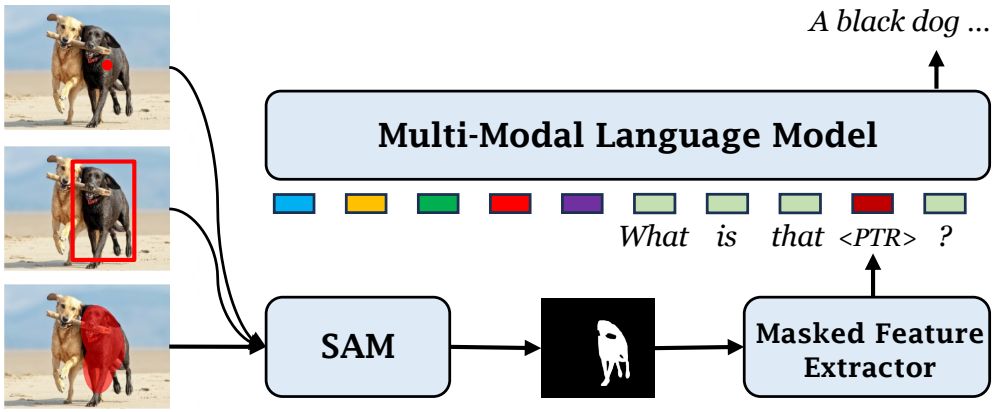


Figure 3.3: GROUNDHOG can take arbitrary spatial prompts that can be resolved by an interactive segmentation model, such as SAM. The placeholder pointer token `<PTR>` will be replaced by the extracted entity features and fed as input to the model.

dimensions are usually smaller than those of the mask proposals, we resize the masks to match the size of the feature maps prior to pooling. The pooled features are then fed into a Multi-Layer Perceptron (MLP) network to align with the input embeddings of the MLLM. We empirically find the combination of CLIP and DINOv2 features yields the best result, and these features are added to obtain the final input visual entity tokens to the MLLM.

Spatial Prompts Furthermore, for grounded MLLMs to be more broadly applicable, they must be capable of interpreting multi-modal user inputs, including spatial prompts. Thanks to the mask model agnostic design, GROUNDHOGcan seamlessly support such inputs. As demonstrated in Figure 3.3, by applying an interactive segmentation model such as Segment-Anything (SAM) [123], arbitrary spatial prompts can be translated into binary masks and processed by the same masked feature extractor we just introduced. This extracted feature for the pointed entity will replace the pointer token `<PTR>` placeholder in the textual input.

3.2.2 Language Grounding to Entity Segmentation

Existing box-grounded MLLMs typically append location tokens after the groundable phrases [40, 199, 38, 292]. However, this method is not readily interpretable. To alleviate this disconnect, we introduce a pair of grounding tokens `<GRD>` and `</GRD>` to indicate the start and end of groundable phrases, with the assumption that grounding these phrases requires mapping to certain representations of visual entities irrespective of the visual modality. In Figure 3.2, a sentence can be represented as `I see <GRD> two dogs </GRD> on <GRD> the beach </GRD>`, with two distinct visual entities grounded. The representation of

each groundable phrase, termed as the *grounding query*, is obtained by adding `<GRD>` and `</GRD>`'s output embedding from the last transformer layer of the MLLM. The representation is then used to retrieve the entities that the phrase should be grounded to. In particular, we concatenate the grounding query with the last layer output of each visual entity token, and use an MLP to predict a scalar score for each entity. Finally, we merge all the mask proposals into one single mask with pixel-wise maximization:

$$\mathcal{M}_{h,w} = \max_{q} \left(\mathcal{S}_q \cdot \widehat{\mathcal{M}}_{q,h,w} \right)$$

where $\mathcal{S}_q$ is the normalized score of the q-th mask ranging from 0 to 1, and $\widehat{\mathcal{M}}_{q,h,w}$ denotes the pixel probability at position (h, w) for the q-th mask. Note that a phrase may ground to multiple entities, thus multiple mask proposals may get a high score simultaneously and be selected in conjunction. One of the primary benefits of this decoupled design is its transparency in the selection of entities. Users can easily visualize both the mask proposals and their respective scores, providing a clear understanding of how a grounding mask is predicted. This level of clarity and interpretability is a significant advantage, offering users a tangible insight into the model's grounding process.

3.2.3 Towards Holistic Entity Mask Proposals

In order to support holistic language grounding to arbitrary segmentations, the entity proposal should have two essential properties. First, the proposals should strike a delicate balance in terms of semantic atomicity. While it is possible to merge multiple proposals later to form multi-entity segmentations, the reverse, i.e., dividing a single proposal into smaller segments, is not feasible. Therefore, instance segmentation is generally preferred over semantic segmentation. However, the segmentation should not be excessively fine-grained to the extent that it compromises basic semantic integrity. Over-segmentation can lead to a loss of the coherent concept of an entity, which is detrimental to the grounding process. Second, the entity proposals should have a high coverage of entities, encompassing a diverse range of granularities. This includes not only tangible objects (things) and amorphous concepts (stuff) but also extends to sub-components of objects (parts of things) and structured regions such as areas containing visual text. The ability to propose entities across this spectrum of granularity is pivotal, as it directly determines the upper bound of the grounding capability of MLLM.

We initiated our study with a Mask2Former model pre-trained on the COCO panoptic segmentation dataset. However, preliminary experiments revealed its limitations in semantic coverage and adaptability to open-world scenarios. To enhance this, we developed

Model	RefCOCO+	PhraseCut	ReasonSeg	TextVQA-X
Mask2Former	0.867	0.563	0.602	0.137
Mask2Former+	**0.873**	**0.624**	**0.745**	**0.446**

Table 3.1: The average Any-IoU of the proposals on each dataset. The vanilla Mask2Former is trained on the COCO-Panoptic dataset and our Mask2Former+ is trained on our combined dataset. Mask2Former+ obtains a consistent improvement in all scenarios, especially in non-COCO domains.

Mask2Former+, an upgraded version designed for multi-grained segmentation. This upgrade involved creating a diverse dataset by merging annotations from various sources, including COCO [26], LVIS [93], Entity-v2 [208], Pascal [58], PACO [214] (Figure A.1); MHP-v2 [141] for human part parsing; and TextOCR [238] for text segmentation. We expanded the model's capabilities by adding 50 expert queries each for semantic parts and visual text regions, alongside the original 200 entity queries. We assessed Mask2Former+'s performance on 1000 images from validation splits from 4 grounding benchmarks, RefCOCO+ [293], PhraseCut [275], ReasonSeg [133], and TextVQA-X [237]. We use the Any-IoU [111] metric for evaluation, i.e., for each ground truth mask, we extract the most overlapped mask proposals and compute the IoU, then take the average. As Table 3.1 demonstrates, Mask2Former+ shows consistent improvements across all domains, particularly in those significantly divergent from COCO. This highlights its enhanced adaptability and precision in a broader range of segmentation challenges, providing a good mask proposal model for GROUNDHOG. We refer to Appendix A.1 for more details of the model and data.

3.3 The M3G2 Dataset

In this section, we introduce M3G2, a Multi-Modal Multi-Grained Grounding dataset consisting of 2.5M text-image pairs for visually grounded instruction tuning, consisting of 36 sub-problems derived and augmented from 27 existing datasets. We re-organize and augment public datasets of language grounding, visual question answering, referring expression segmentation, and referring expression generation into various forms of visually grounded dialogue for grounded instruction tuning, outlined briefly in Table 3.2. The dataset is categorized into four main types: (1) Grounded Image Captioning (GIC), (2) Referential Expression Segmentation (RES), (3) Grounded Visual Question Answering (GVQA), and (4) Referential Dialog (RD). We provide illustrated descriptions of our prompt design, accompanied by examples of each task type as depicted in Figure 3.4. We detail the task schema in the

Task	Dataset	Gr. Ann.			Sem. Gran.					# Pairs
		M	B	Po	S	Th	Pa	G	Tx	Train
GCAP	PNG	✓	✓		✓	✓		✓		132k
	Flickr30K-Entity		✓		✓	✓	✓	✓		149k
RES	RefCOCO	✓	✓		✓					113k
	RefCOCO+	✓	✓		✓					112k
	RefCOCOg	✓	✓		✓					80k
	RefCLEF	✓	✓		✓					105k
	gRefCOCO	✓	✓		✓					194k
	PhraseCut	✓	✓		✓	✓	✓	✓		85k
	D-Cube	✓	✓		✓			✓		10k
	ReasonSeg	✓	✓		✓	✓	✓	✓		1k
	RIO	✓	✓		✓			✓		28k
	SK-VG		✓		✓					23k
GVQA	VizWiz-G	✓	✓		✓	✓			✓	6k
	TextVQA-X	✓	✓						✓	15k
	GQA		✓		✓	✓	✓	✓		302k
	VQS		✓		✓					20k
	Shikra-BinaryQA		✓		✓	✓	✓	✓		4k
	EntityCount	✓	✓		✓	✓	✓	✓		11k
	FoodSeg-QA	✓	✓		✓			✓		7k
	LVIS-QA	✓	✓		✓	✓		✓		95k
RD	RefCOCO-REG	✓	✓	✓	✓					17k
	RefCOCO+-REG	✓	✓	✓	✓					17k
	RefCOCOg-REG	✓	✓	✓	✓					22k
	gRefCOCO-REG	✓	✓	✓	✓					20k
	VG-SpotCap		✓	✓	✓	✓	✓	✓		247k
	V7W		✓	✓	✓					23k
	PointQA			✓	✓					64k
	VCR		✓	✓	✓					156k
	ShikraRD		✓	✓	✓	✓	✓	✓		2k
	SVIT-RD		✓	✓	✓	✓	✓	✓		33k
	Guesswhat	✓	✓	✓	✓					193k
	VG-RefMatch		✓	✓	✓	✓	✓	✓		247k
	HierText	✓	✓	✓					✓	6k
M3G2 (Total)										2.5M

Table 3.2: Summary of datasets included in M3G2. We show the availability of Grounding Annotations (Box, Mask, and Pointer inputs), the Semantic Granularity (Stuff, Things, Parts, Groups, and Text), and the number of text-image pairs for training.

following sections and provide the complete sets of templates in Appendix A.2.

3.3.1 Grounded Image Captioning (GIC)

The task of *grounded image captioning* requires the model to produce a narrative for the visual scene, and accurately identify and associate the groundable phrases with their respective binary segmentation masks. The objective of this task is to empower the model to articulate the scene while acknowledging various visual elements and their spatial interrelations. We incorporate the Panoptic Narrative Grounding (PNG) [122] and Flickr30K-Entity [203] datasets. PNG, derived from Localize Narrative [204] and COCO Segmentation [26], provides

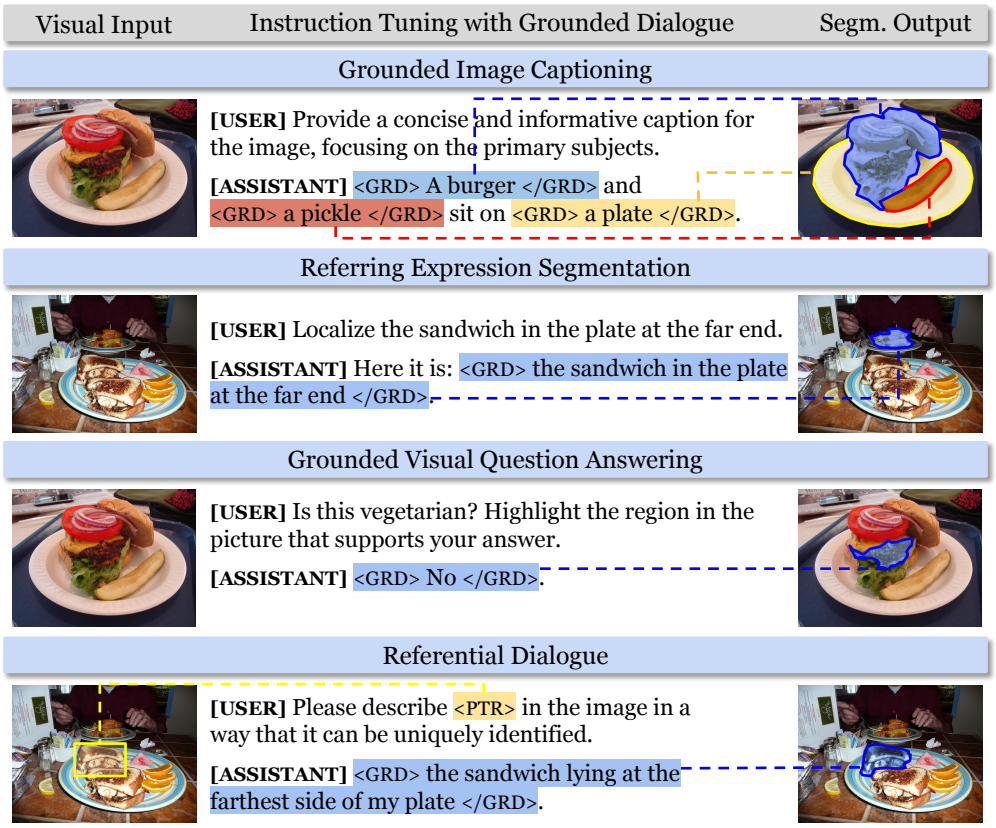


Figure 3.4: Illustration of the four grounding task types in the M3G2 dataset.

long and detailed narratives with an average of 36.5 words per description. These narratives are rich in detail, offering a high coverage of the visual content including the background. Flickr30K-Entity, offering concise captions with box annotations, complements PNG with its larger vocabulary and finer granularity. The example instruction templates used to construct the conversation are listed in Table A.3, where we use key words such as "short/briefly" and "in detail" to distinguish between short and long captioning.

3.3.2 Referring Expression Segmentation (RES)

In contrast to previous tasks, the *referring expression segmentation* task requires that the model generates a segmentation mask based on a given referring expression. Besides the RefCOCO series [293, 179, 151], we have further leveraged existing RES benchmarks [220, 275, 281, 277, 133] for this purpose. We employ the widely used RefCOCO/+/g datasets [116, 178] and RefCLEF [220] for single-object RES. gRefCOCO [151] is employed for multi-object and negative queries. To enhance the visual diversity, we also incorporate PhraseCut [275] and D-Cube [281] that use an image source different than COCO. Additionally, ReasonSeg [133],

RIO [210], and SK-VG [277] are included, where a textual context is given and the models need to not only understand that context, but also equips with a certain degree of commonsense knowledge to successfully solve the query. To improve the learning efficiency and enhance contextual understanding, we format queries from the same image into a simulated multi-turn dialogue (Table A.4).

3.3.3 Grounded Visual Question Answering (GVQA)

The task of *grounded visual question answering* requires the model to comprehend a question (with optional pointers) and to produce an answer that is grounded to a binary segmentation mask that justifies the answer. The goal of this task is to enable natural QA-based interaction with users and reasoning in the model with grounded explanations. We include 8 datasets for the grounded VQA task in M3G2. First, we collect and organize some existing datasets that can directly fit into our grounded vision-language task framework, including VizWiz-Grounding [34], TextVQA-X [215], GQA [105], VQA [73] and Shikra-BinaryQA [38]. To further improve the data scale and visual concept coverage, we enlarge the GVQA collection by re-purposing existing panoptic segmentation datasets with templated instructions and model responses. Specifically, based on the annotations from LVIS [93] and EntityV2 [208], we design questions about object presence, object counting, and segment query with a possibly negative request (i.e. the target object does not exist in the image), for the model to learn to recognize a diverse set of concepts more faithfully. See example question templates used in Table A.5.

3.3.4 Referential Dialogue (RD)

The task of *referential dialogue* requires the model to conduct dialogue communication with users, especially when conditioned on user-provided spatial prompts. We include various types of RD in our dataset and the templates used are listed in Table A.6. First, we add several existing RD datasets such as V7W [317], PointQA [176], VCR [295], ShikraRD [38] and SVIT [307] without much modifications. We then revisit the RefCOCO series [293, 179, 151] for referential expression generation, where the referred object is given and the goal is to generate a unique description that leads to that object. We use the region caption annotations from the VG dataset [129] for region captioning and a region-matching game. We select a set of region pointers and several descriptions to provide to the model, and the goal is to match the pointed regions with the descriptions. We repurpose the GuessWhat dataset [59] to make it fit into our RD formulation. We also construct a referred text reading task based on the HierText [161] dataset and enhance the model's capability of text recognition.

Model	Single Instance								Multi-/No Instance		Reasoning		
	RefCOCO			RefCOCO+			RefCOCOg		gRefCOCO	PhraseCut	ReasonSeg	RIO	
	val	test-A	test-B	val	test-A	test-B	val-u	test-u	val	test	val	test-c	test-u
Specialist													
MDETR [111]	-	-	-	-	-	-	-	-	-	53.7	-	44.1	22.0
CRIS [270]	70.5	73.2	66.1	62.3	68.1	53.7	59.9	60.4	55.3	-	-	-	-
LAVT [287]	72.7	75.8	68.8	62.1	68.4	55.1	61.2	62.1	58.4	-	-	-	-
ReLA [151]	73.8	76.5	70.2	66.0	71.0	57.7	65.0	66.0	63.6	-	22.4	-	-
PolyFormer [157]	76.0	78.3	73.3	69.3	74.6	61.9	69.2	70.2	-	-	-	48.8	26.8
UNINEXT-H [283]	82.2	83.4	81.3	72.5	76.4	66.2	74.7	76.4	-	-	-	-	-
Generalist													
LISA_{7B} [133]	74.1	76.5	71.1	62.4	67.4	56.5	66.4	68.5	-	-	44.0	-	-
LISA_{7B} (FT) [133]	74.9	79.1	72.3	65.1	70.8	58.1	67.9	70.6	-	-	52.9	-	-
🐻_{7B}	78.5	79.9	75.7	70.5	75.0	64.9	74.1	74.6	66.7	54.5	56.2	57.9	33.9

Table 3.3: Results on 7 Referring Expression Segmentation (RES) benchmarks with single instance queries [116, 178], multi-/null instance queries [151, 275] and reasoning-based queries [133, 210]. We report cIoU for RefCOCO/+/g and mIoU for other benchmarks, respectively.

Model	Flickr30K-E	
	R@1_{val}	R@1_{test}
Shikra_{13B}	77.4	78.4
Ferret_{13B}	81.1	84.8
Shikra_{7B}	75.8	76.5
Ferret_{7B}	80.4	82.2
🐻_{7B}	79.2	79.8

Table 3.4: Top-1 box recall results on Flickr30K-Entity [203].

Model	PNG				
	AR	AR_{th}	AR_{st}	AR_{s}	AR_{p}
PiGLET	65.9	64.0	68.6	67.2	54.5
🐻_{7B}	66.8	65.0	69.4	70.4	57.7

Table 3.5: Phrase grounding results on PNG [84].

Model	TextVQA-X [mIoU]
SAB	29.0
🐻_{7B}	39.8

Table 3.6: Results on TextVQA-X [237] dataset.

3.4 Experimental Results Analysis

We adopt the LLaMA2-7B model [255] as our base LLM, and initialized the weight from LLaVA-1.5 [153]. For the vision encoders, we use the OpenAI CLIP@336 [211] model and DINOv2-L/14-reg [54] pretrained checkpoints. Please refer to Appendix A.3 for more implementation details.

3.4.1 Generalist in Grounded Vision-Language Tasks

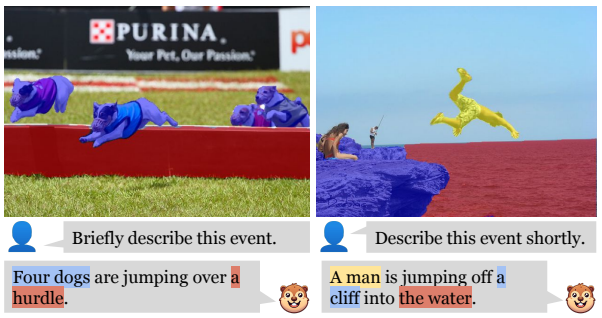
We first demonstrate GROUNDHOG's capabilities as a generalist model for three different types of grounded vision-language tasks. It's worth noting that, unlike previous work that needs dataset-specific fine-tuning on each of the tasks, GROUNDHOG can achieve comparable performance on all the tasks directly after training on M3G2, i.e., all the reported results from our model are from a single set of weights without any dataset-specific fine-tuning.

Model	Bleu-4	METEOR	CIDEr	SPICE	$\mathbf{F1_{all}}$
Shikra_{13B}	-	-	73.9	-	-
Ferret_{13B}	37.0	25.5	76.1	18.3	15.1
Ferret_{7B}	35.1	24.6	74.8	18.0	15.0
🐻_{7B}	36.7	26.5	91.3	20.4	32.1

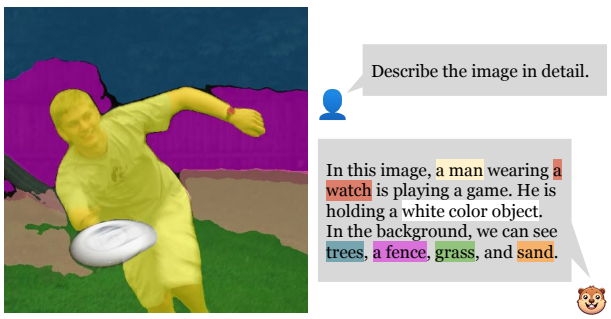
Table 3.7: Grounded Captioning on Flickr30K-Entity [203].

Language Grounding To Segmentation. We start by evaluating the model on language grounding tasks, which takes text as input and generates segmentation masks as output. We assess GROUNDHOG on Referential Expression Segmentation (RES) [116] and Caption Phrase Grounding (CPG) tasks. While traditional RES benchmarks [116, 178] focus on single-instance referents requiring primarily visual understanding, we expanded our evaluation to include complex scenarios involving multi-instance or negative queries [275, 151], and those necessitating common sense reasoning [133, 210]. The results in Table 3.3 show GROUNDHOG outperforming the generalist model LISA across all benchmarks and achieving significant improvements over specialist models in multi-instance, null, and reasoning-based RES tasks. It also performs comparably on the competitive RefCOCO series. For CPG tasks, which involve grounding all phrases in a caption and demand a deep understanding of the context for coreference resolution, we first evaluated GROUNDHOG on the Flickr30K-Entity dataset [203]. Since this dataset only has box annotations, we convert the mask predictions of our model to box and compute the top-1 box recall following the merged-box protocol [111]. Despite not specializing in predicting boxes, GROUNDHOG still outperforms Shikra 7B/13B [38] and is on par with Ferret-7B [292] in a concurrent work (Table 3.4). Additionally, on the PNG dataset [122] which tests phrase grounding in longer narratives, GROUNDHOG surpasses the previous state-of-the-art model, PiGLET [85], in all metrics including average recall of grounding masks and detailed scores for things, stuffs, and singular and plural entities (Table 3.5).

Grounded Language Generation. Our model excels in generating language that accurately grounds to segmentation masks during user conversations. Quantitatively, we assess grounded captioning on the Flickr30K-Entity dataset [203], employing standard text generation metrics such as Bleu-4 [195], METEOR [20], CIDEr [259], and SPICE [10] for language quality; and the F1_{all} score for grounding accuracy following [292]. As shown in Table 3.7, GROUNDHOG significantly surpasses existing box-based grounded MLLMs, even their 13B versions, in both language quality and grounding accuracy. This improvement



(a) Grounded short caption generation on Flickr30K-Entity. While only box supervisions are available for this dataset, GROUNDHOG generalize to pixel-level grounding after joint training on M3G2.



(b) Grounded detailed narrative generation on PNG. GROUNDHOG successfully generalize to grounding a novel category *watch* in the generated caption, which is not included in the 80 categories of PNG annotation.

Figure 3.5: Examples of GROUNDHOG's performance in grounded image captioning.

Model	$\text{PointQA}_{\text{Twice}}$ Acc	V7W Acc
Shikra_{13B}	70.3	85.3
GPT4RoI_{13B}	-	84.8
Shikra_{7B}	-	-
GPT4RoI_{7B}	-	81.8
🧸_{7B}	72.4	85.5

Table 3.8: Results on $\text{PointQA}_{\text{Twice}}$ [176] and V7W [317] test sets.

is hypothesized to stem from the diverse task distribution in our M3G2 dataset. We show some generated captions in Figure 3.5, with a highlight of box-to-pixel generalization (Figure 3.5a) and novel category grounding (Figure 3.5b). Specially, Figure 3.5a highlights that our model generalize to pixel-level grounding while trained only with box supervision on that dataset, and successfully generate and ground novel categories unseen during training. See the Appendix for more examples.

For groundable question answering, we evaluate on the TextVQA-X benchmark [215]. GROUNDHOG outperforms the state-of-the-art specialist model SAB [119] by a significant margin, as measured by the mean IoU of the predicted mask (Table 3.6).

Spatial Prompt Understanding. For grounded MLLMs, accurately interpreting multi-modal instructions is essential, particularly in interactive tasks. We evaluated its performance on two pointer-based QA benchmarks, $\text{PointerQA}_{\text{Twice}}$ [176] and V7W [317], which require

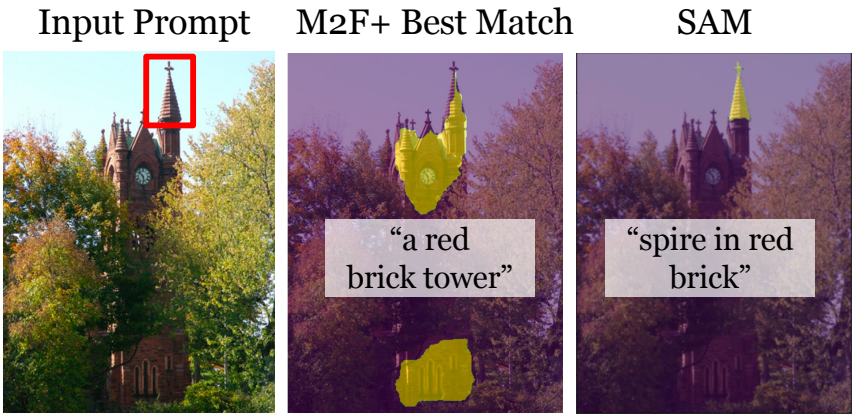


Figure 3.6: Region caption using the best match proposal from Mask2Former+ versus from SAM. Mask2Former+ fails to propose the exact mask of the spire, leading to a less precise caption.

the model to answer questions guided by spatial prompts, such as bounding boxes. The model is tasked to generate free-form textual answers in $\text{PointerQA}_{\text{Twice}}$, and selects from multiple-choice options in V7W. GROUNDHOG demonstrates superior performance in these benchmarks, outperforming previous models as shown in Table 3.8. This highlights its effectiveness in spatial understanding and response accuracy. To further demonstrate the effectiveness of using SAM for the pointer-to-mask conversion, we show the best-matched mask proposal from our Mask2Former+ model in comparison to the mask from SAM in Figure 3.6. While the best match proposal from the Mask2Former+ model includes a broader area, the SAM-generated mask offers a more precise representation of the specified region, potentially leading to a more accurate caption.

3.4.2 Trustworthiness and Transparency

Beyond its superior performance as a grounding generalist, we highlight two key improvements for creating a more trustworthy and transparent agent.

Reduced Object Hallucination. Thanks to the varied task distribution and the inclusion of negative question-answering samples in M3G2 dataset, GROUNDHOG significantly reduces object hallucination. We assessed this using the POPE [147] benchmark, which includes binary questions about object existence across three splits, each with a different object distribution (with an order of difficulty *Random* < *Popular* < *Adversarial*). Remarkably, GROUNDHOG consistently outperforms other models in both accuracy and F1 score across all splits, particularly on the more challenging ones. It shows an absolute improvement of

Model	Accuracy	Precision	Recall	F1 Score	Yes (%)
Random					
mPLUG-Owl	53.30	51.71	99.53	68.06	96.23
LLaVA	54.43	52.32	99.80	68.65	95.37
MultiModal-GPT	50.03	50.02	100.00	66.68	99.97
MiniGPT-4	77.83	75.38	82.67	78.86	54.83
InstructBLIP	88.73	85.08	93.93	89.29	55.20
Shikra-13B	86.90	94.40	79.26	86.19	43.26
Ferret-13B	90.24	97.72	83.00	89.76	43.26
🐻$_{7B}$	91.03	85.80	96.40	90.79	45.88
Popular					
mPLUG-Owl	50.63	50.32	99.27	66.79	98.63
LLaVA	52.43	51.25	99.80	67.72	97.37
MultiModal-GPT	50.00	50.00	100.00	66.67	100.00
MiniGPT-4	68.30	64.27	82.40	72.21	64.10
InstructBLIP	81.37	75.07	93.93	83.45	62.57
Shikra-13B	83.97	87.55	79.20	83.16	45.23
Ferret-13B	84.90	88.24	80.53	84.21	45.63
🐻$_{7B}$	90.13	85.93	93.81	89.70	45.80
Adversarial					
mPLUG-Owl	50.67	50.34	99.33	66.82	98.67
LLaVA	50.77	50.39	99.87	66.98	99.10
MultiModal-GPT	50.00	50.00	100.00	66.67	100.00
MiniGPT-4	66.60	62.45	83.27	71.37	66.67
InstructBLIP	74.37	67.67	93.33	78.45	68.97
Shikra-13B	83.10	85.60	79.60	82.49	46.50
Ferret-13B	82.36	83.60	80.53	82.00	48.18
🐻$_{7B}$	86.33	85.93	86.63	86.28	49.60

Table 3.9: Object hallucination results on the POPE [147] benchmark.

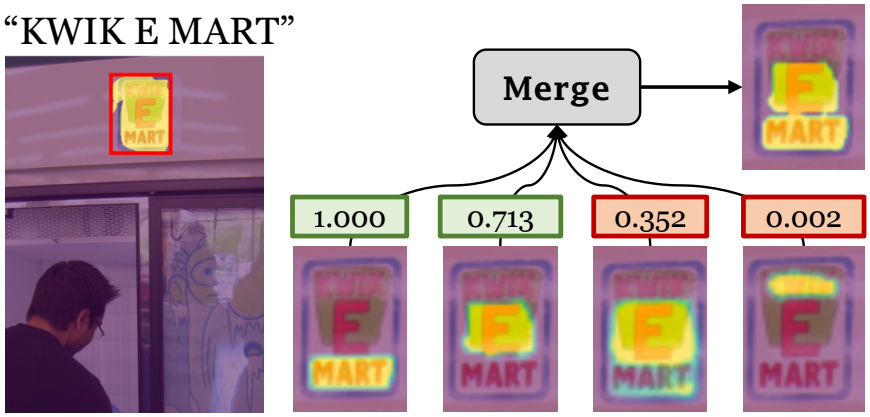


Figure 3.7: Illustration of a partially correct grounding. The grounding phrase and the ground truth mask are shown on the left. The top-4 mask proposals are presented, with highly-scored masks (green) selected for the merged mask, and low-scored masks (red) excluded. This illustrates the failure to recognize the word “KWIK” by the MLLM, despite its successful proposal.

5.2% in accuracy for *Popular* and 4.0% for *Adversarial* over the previously best-performing model. This suggests that our model's enhanced grounding capability plays a significant role in mitigating the object hallucination problem.

Explainability and Diagnosability. Another important highlight of GROUNDHOG is its enhancement of explainability through the decoupled design of entity proposal and selection, as outlined earlier in section 3.2.2. This is exemplified in the case study illustrated in Figure 3.7, which illustrates the mask proposal scoring and selective merging process of our model. We show the top-4 masks, where the higher-score masks are labeled in green while the lower-score masks are labeled in red. Users can easily interpret that the failure is due to the incapability of MLLM to recognize the word “KWIK”, despite it being successfully localized and proposed as an entity candidate.

3.4.3 Ablation Studies

We performed ablation studies to validate our design decisions, training, and evaluating a subset of the M3G2 dataset that includes RefCOCO+, Flickr30K, and TextVQA. These cover a range of visual entities from various image sources and granularities. We start by comparing our Mask2Former+ with the original Mask2Former for mask proposal effectiveness. As indicated in Table 3.10, the original Mask2Former performs slightly better on RefCOCO, as it is developed specificity on COCO object categories. However, Mask2Former+ significantly surpasses the original in domains with non-COCO entities. Our second set of experiments

Setups	RefCOCO+	Flickr30K	TextVQA-X
Mask Proposal Models			
Mask2Former	**67.1**	69.0	9.8
Mask2Former+	66.6	**77.2**	**34.0**
Entity Features			
CLIP	59.8	75.0	32.0
DINOv2	62.3	76.3	28.4
CLIP+DINOv2	**66.6**	**77.2**	**34.0**
Grounding Query			
`<GRD>` only	64.4	67.5	**34.2**
`</GRD>` only	64.4	**77.2**	33.5
Sum	**66.6**	**77.2**	34.0
Eval Input Resolution			
224–480	54.7	67.2	27.6
480–640	65.5	76.7	27.6
800–1024	**66.6**	**77.2**	**34.0**

Table 3.10: Ablation study on model design choices and evaluation setups. Models are trained on RefCOCO+, Flickr30K, TextVQA-X and tested on corresponding validation sets.

examined the choice of visual entity features. Although using either CLIP or DINOv2 features alone shows advantages in specific datasets, their combination consistently yields the best results across all datasets. To obtain a robust grounding query representation, we experimented with using the output embedding of the `<GRD>` token, the `</GRD>` token, and their sum. We found that the latter approach achieves the best overall results. Finally, we demonstrate that our decoupling design of the mask proposal model and MLLM allows for training at a lower resolution (320px) to expedite grounding training, while scaling up the resolution during evaluation enhances performance.

3.5 Discussion of Limitations

This work, while exciting, has several limitations that we acknowledge and aim to address in future research. Firstly, the datasets utilized to develop M3G2 consist of a blend of existing academic datasets. The quality of annotations in these datasets varies significantly, and they often lack comprehensive coverage of concepts. To enhance training efficiency, applying data filtering methods could help reduce the size of the dataset without compromising its effectiveness. Additionally, expanding the vision-language grounding data to a web-scale could significantly improve the comprehensiveness of grounding learning.

Secondly, our current model is limited to processing only single images. Although the entity-centric approach we adopted could theoretically extend to other modalities like 3D or video, this potential has not yet been empirically validated. Testing and validating our model on datasets relevant to these modalities would be a valuable direction for future research. This step is crucial to understanding the model's effectiveness across different types of application scenarios and further improving its usefulness.

3.6 Summary of Findings

This chapter addressed the challenge of fine-grained, pixel-level visual grounding of holistic semantic concepts for Multimodal Large Language Models (MLLMs). We introduced GROUNDHOG, a novel MLLM architecture designed for segmentation-based grounding, leveraging entity mask proposals and a masked feature extraction mechanism to connect language phrases directly to pixel regions. To facilitate this, we curated the M3G2 dataset, comprising 2.5M image-text pairs across 36 sub-problems, unifying diverse grounding tasks like Grounded Image Captioning (GIC), Referring Expression Segmentation (RES), Grounded Visual Question Answering (GVQA), and Referential Dialogue (RD) under a single framework with rich mask annotations.

Our experiments demonstrated several key findings:

- **Generalist Grounding Capability:** GROUNDHOG achieved strong performance across a variety of grounding benchmarks without task-specific fine-tuning, often exceeding prior specialist and generalist models, particularly in complex scenarios involving multiple instances, reasoning, or novel objects.
- **Improved Precision and Reduced Hallucination:** By grounding to segmentation masks, GROUNDHOG provided more precise localization compared to box-based methods and significantly reduced object hallucination as validated on the POPE benchmark [147], showing the benefit of fine-grained grounding.
- **Enhanced Interpretability:** The decoupled design of entity proposal and selection allows for clearer diagnosis of grounding failures, distinguishing between localization issues (missing proposals) and recognition/alignment errors (incorrect grounding), as illustrated in our case studies (e.g., Figure 3.7).

In conclusion, this chapter demonstrates the effectiveness of grounding MLLMs to holistic segmentation for achieving fine-grained, interpretable, and robust symbol grounding. This work represents a significant step toward bridging language with detailed visual perception in multimodal systems.

CHAPTER 4

Language Grounding under Visual Illusions for Vision-Language Models

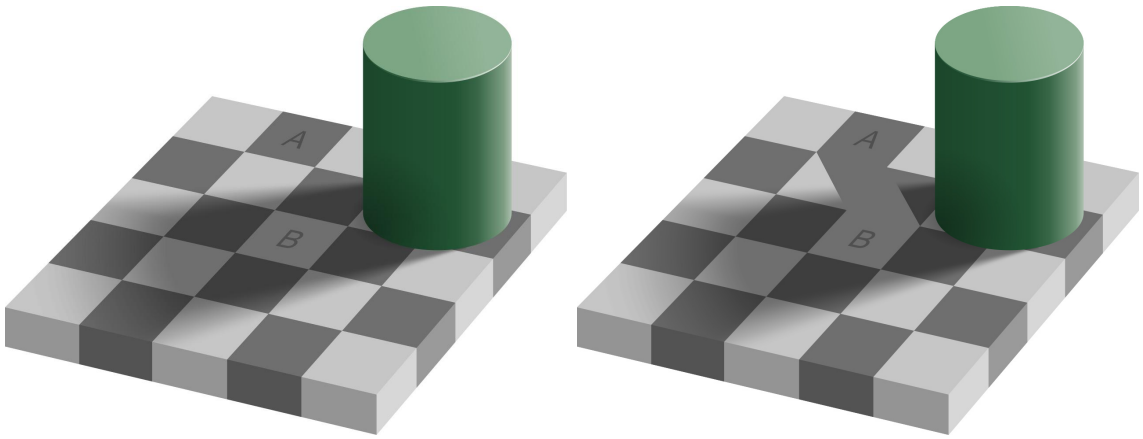
Following our exploration of achieving fine-grained, pixel-level visual grounding with GROUNDHOG in the previous chapter, we now turn to a deeper investigation into the nature of this grounding. While establishing accurate alignment between language and visual referents under standard conditions is crucial, a fundamental question remains: how do these models perceive the world when visual information is ambiguous or potentially misleading, particularly in ways that consistently affect human perception? Human vision is susceptible to various illusions, where subjective experience deviates systematically from objective reality. This chapter delves into this phenomenon within the context of MLLMs[1], asking whether these models exhibit human-like perceptual biases or maintain strict fidelity to the physical input. The content of this chapter is based on the work published in EMNLP 2023[2].

4.1 Overview

It's well established that human perceptual systems are susceptible to visual illusions, which are defined as "consistent and persistent discrepancies between a physical state of affairs and its representation in consciousness" [57]. Figure 4.1 shows a well-known example - the checker shadow illusion [3]. Here, a cylinder on the checker board creates a shadow on the board. Human viewers are directed to look at the two squares A and B as shown in Figure 4.1a. To most normal-sighted people, they will perceive that square A is darker than square B. However, the reality is, the color pixels of A and B are exactly the same, as shown in Figure 1(b). This example demonstrates that while the physical attributes of A and B are the same,

[1]In this chapter, we will use the term VLM (Vision-Language Model) interchangeably with MLLM, as the models evaluated here are typically referred to as VLM in the literature.

[2]Yichi Zhang, Jiayi Pan, Yuchen Zhou, Rui Pan, and Joyce Chai. "Grounding Visual Illusions in Language: Do Vision-Language Models Perceive Illusions Like Humans?". Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing. 2023 [304].



(a) (b)

Figure 4.1: The checker shadow illusion [3].

from humans' eyes, they may look different, which may further influence how language is used to describe these objects.

Motivated by human visual illusion phenomena, recent years have seen an increasing amount of work in machine visual illusions [81, 82, 100, 246, 162]. These previous works were solely based on vision, for example, evaluating how the internal representation from a computer vision model can be used as a proxy of stimulus compared to human's stimulus shift under illusions. Most previous experiments were conducted in a case-by-case manner, without addressing general behaviors through a systematic investigation.

Different from these previous works, this paper studies visual illusion from a new angle, in the context of language communication. Language comprehension and language production are tightly linked to how we perceive the visual world. Back to Figure 1(a), when two people are observing the figure together, due to their likely shared illusion, the expression "the darker square" will lead to the same reference of square A. But when a human communicates with a machine, will the machine also interpret "the darker square" as square A? Given the rise of large vision-language models (VLM), it's important to understand whether these VLM models have a similar tendency to visual illusions, and to what degree they may align with human vision. The answers to these questions will further impact the alignment of the grounded language communication between humans and machines.

To address these questions, we created a new visual illusion dataset covering five different categories from the cognitive literature. Based on the dataset, we created a benchmark, Grounding Visual Illusion in Language (GVIL), which consists of four subtasks: Same-Difference Question Answering (*SameDiffQA*), Referential Question Answering (*RefQA*), Attribute Question Answering (*AttrQA*), and Referential Localization (*RefLoc*) to assess

machines' alignment with the human under visual illusions.

We specifically evaluated four state-of-the-art vision-language models: Unified-IO [166], OFA [264], LLaVA [155] and InstructBLIP [48]. Our results have shown that these four models mostly do not align with human vision illusions, especially for QA-based tasks. However, for the *RefLoc* task, these models (especially ones with larger parameters) have demonstrated an impressive alignment with humans.

To our knowledge, this is the first work that takes language into consideration to study machine visual illusion. There are two main contributions of this work. First, this investigation provides an initial understanding of the alignment between human and machine visual illusions. Such understanding will enable techniques for a better communicative grounding in situated language communication and help develop more reliable embodied agents in the future. Second, unlike using internal representations to explain illusions, language can be used as a proxy to demonstrate whether and how machine illusions match or mis-match with the human illusion. This benchmark will not only facilitate computational models for better human agent alignment, but also provide tools for scientific understanding of visual illusion in both humans and machines.

4.2 The GVIL Benchmark

To facilitate our investigation, we created a benchmark for evaluating machine visual illusions in the context of grounded communication. This benchmark is built upon a set of images with visual illusions. Each image consists of two objects which may look different to humans but are actually identical in their pixels. This setup has two advantages. First, the definition of illusion is clear and non-ambiguous, thus it is easy to measure whether the machine has a similar illusion as humans. Secondly, the multi-object setup naturally supports the evaluation of language grounding, such as evaluating whether the machine can select the object an expression grounds to under the illusion (i.e., square A is what "the darker square" grounds to in Figure4.1a).

4.2.1 Data Collection

Our dataset encapsulates five distinct types of illusions, each reflecting different elements of human physiological and cognitive processes [92, 124, 219]. Table 4.1 displays a sample of each illusion type, along with a detailed description.

These illusions can be categorized into two broad areas: color and geometric illusions. For color illusions, we adopt the classifications of color constancy, assimilation, and simultaneous

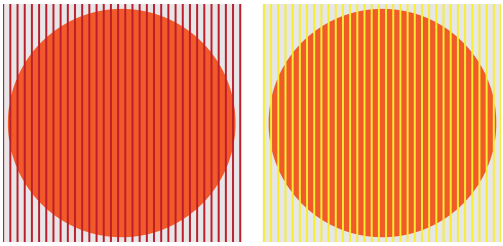
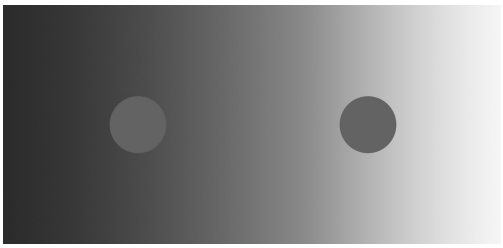
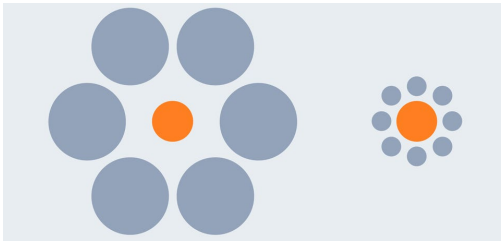
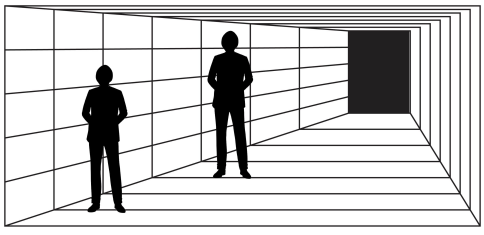
Color Constancy	
	The red ship on the left still looks red after applying a blue filter, the blue ship on the right still looks blue after applying a red filter, even though the RGB colors of both ships are the same.
Color Assimilation	
	The two circles have the same color, while the one on the left looks red (due to its neighbor/foreground) and the one on the right looks orange.
Color Contrast	
	The two grey circles have the same color, while the one on the left looks lighter and the one on the right looks darker.
Geometrical Relativity	
	The two orange circles have the same size, while the one on the left looks smaller and the one on the right looks bigger.
Geometrical Perspective	
	The two people have the same height, while the one on the left looks shorter and the one on the right looks taller.

Table 4.1: Example illusion from each category and the corresponding explanations.

contrast [174]. In terms of geometric illusions, we only included distortions among the four categories in Robinson's illusion classification in order to fix the need for a comparative assessment. The illusions we used to generate our data include Delboeuf [62], Ebbinghaus, and Jastrow illusions [109] for relativity, and Müller-Lyer [187] and Ponzo illusions [205] for perspective distortion. The following explanations give an overview of the human perception phenomena underlying each category:

- **Color Constancy** refers to phenomenon where the color of an object remains constant perceptually, despite changes in lighting conditions.
- **Color Assimilation** shows how two adjacent color patches influence each other's perceptual appearance, reducing the perceived color difference.
- **Color Contrast.** The perceived color of an object shifts opposite to the color of its surroundings
- **Geometrical Relativity** refers to the distortion in the perceived shape or size of an object due to the influence of surrounding oobjects.
- **Geometrical Perspective** reflects the tendency of our visual system to perceive perceptually distant objects as larger than nearby ones of the same size.

For each illusion type, we first collected several root images from the literature [253] and online resources[3]. We manually identify attributes that can be changed without impacting the effect of illusion (e.g., the color of objects in geometric illusions, or the position of objects in color illusions), and edit them to create more illusion instances of the same type, to enrich the number of images in our dataset. We show some augmentation examples in Figure 4.2.

The statistics of our dataset is shown in Table 4.2. Note that since this dataset is only used for the evaluation purpose, i.e., to assess machine's alignment with human in visual illusion, we chose quality over quantity. The dataset is modest in size as each instance is carefully selected (or augmented) based on cognitive literature. Nonetheless, our infrastructure is also designed to foster the continuous development and augmentation of this dataset, allowing for community contributions, for instance. It will become an important resource to not only support a better understanding of machine/human visual illusion, but also facilitate the adaptation of computational models to visual illusions.

4.2.2 Benchmark Tasks

We define four vision-language tasks targeting different model capabilities.

[3] https://michaelbach.de/ot/

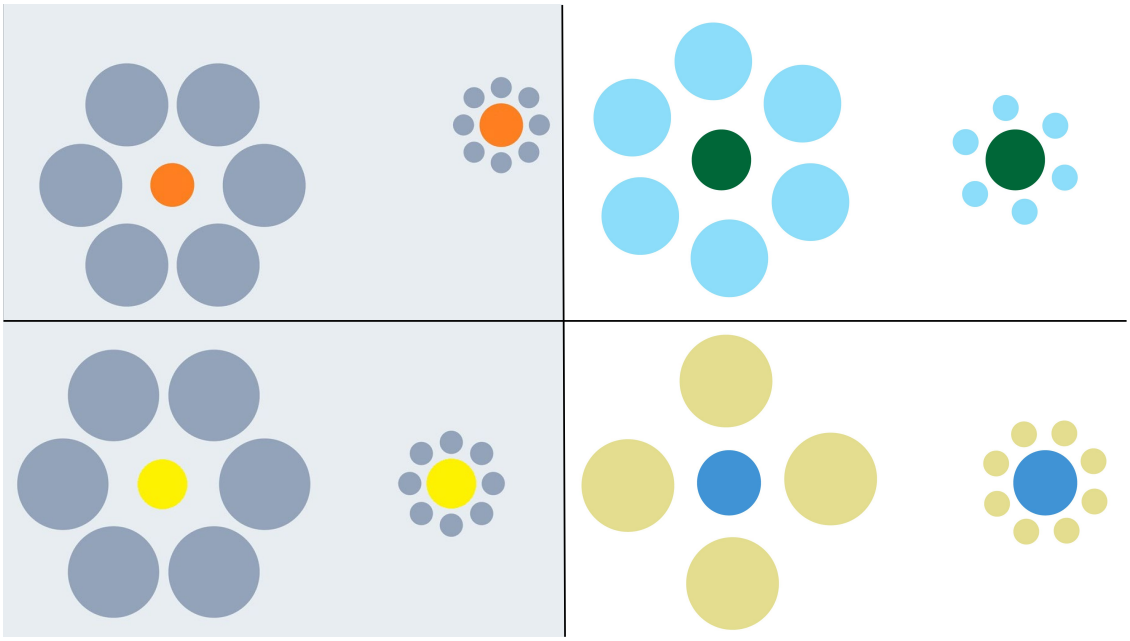


Figure 4.2: Data augmentation examples for the Ebbinghaus illusion.

Category	#Root	#Image	#Instance
Color Constancy	3	6	96
Color Assimilation	5	34	544
Color Contrast	3	30	480
Geometrical Relativity	3	20	320
Geometrical Perspective	2	10	160
Total	16	100	1600

Table 4.2: Dataset statistics.

Same-Different Question Answering (*SameDiffQA*) aims at evaluating the ability of recognizing illusions. As shown in Figure 4.3, each question concerns a pair of images (`IMG1` and `IMG2`). One image (`IMG1`) is illusion-free where two objects (blue balls) are identical in color. The other image (`IMG2`) is induced with an effect of illusion where two balls appear in different colors (blue on the left and green on the right) although their pixels are the same as in `IMG1`. The model is tasked to answer whether two objects are the same color for each of the images. From a human's perspective, the answer would be "Same" to `IMG1` and "Different" to `IMG2` due to the visual illusion. If the model gives the answer 'Same" to `IMG1` and "Different" to `IMG2`, then the answers align with human's answers and therefore the model is considered "human-like". If the model gives "Same" to both images, it implies that the model is faithful to reality and does not perceive the same illusion as humans do. If the

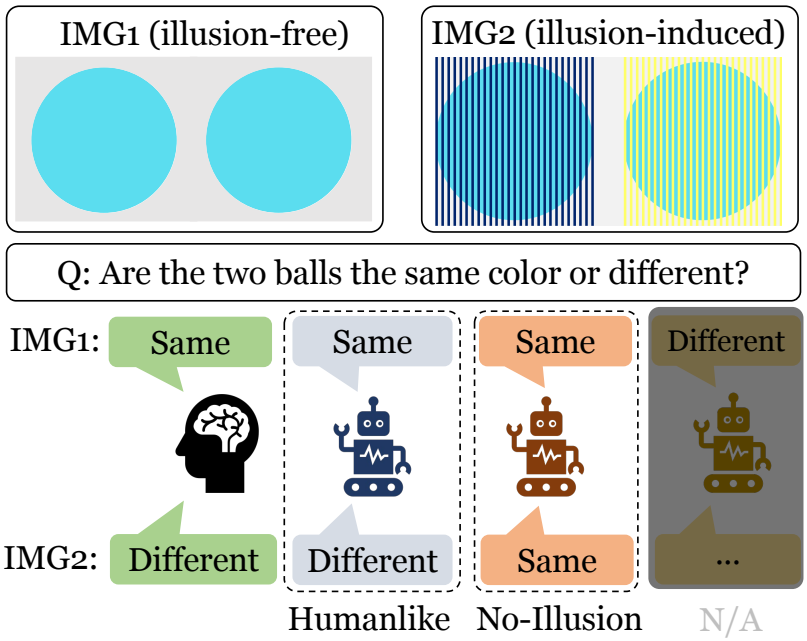


Figure 4.3: Illustration of the *SameDiffQA* setup. For each instance, the model is asked about its perception of an object property across two images, one illusion-free and one illusion-induced. For valid illusion evaluation, the model must initially identify identical properties in the illusion-free image.

model answers "Different" to `IMG1`, it means it lacks basic ability to correctly perceive reality and these cases are considered not applicable to our illusion evaluation.

While *SameDiffQA* focuses on the detection of the presence of illusions, we design three tasks to examine how well do machines align with humans when communication happens under the influence of illusions. Since it is reported that models tend to take shortcut by giving an answer purely based on the text question without looking at the image [89], we propose a paired test to reduce the evaluation bias. As shown in Figure 4.4, each instance comes with two images: one original illusion image (`IMG1`) and one image `IMG2` that flips the objects from the original image (`IMG1`) in a way that will also invert the answer to the question.

Specifically, we evaluate the following three aspects:

Referential Question Answering (*RefQA*) tests the human-likeness of referring to objects under the illusion. In the question, the object of interest is referred to by a property affected by the illusion, and the machine is asked to select the object from two options, e.g., select either left or right for the ball that looks blue, in `IMG1` and `IMG2`.

Attribute Question Answering (*AttrQA*) tests the human-likeness to describe the attribute of objects under the illusion. The question is about describing a visual attribute

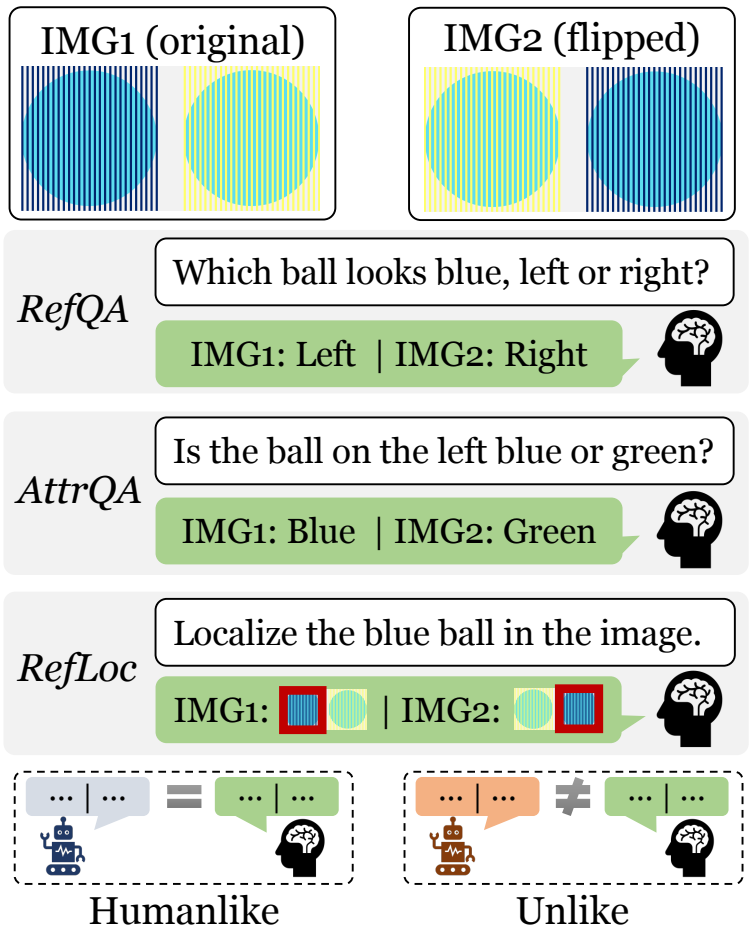


Figure 4.4: Illustration of the *RefQA*, *AttrQA* and *RefLoc* setups. We flip the illusion image wherein the grounding outcome should also be inverted, to create a pair of images for each test. Model success requires accurate identification in both original and flipped versions to align with human responses. Matching human answers signals the model's capability to interpret illusions in a humanlike way, while a mismatch indicates otherwise.

(e.g. color) influenced by the illusion of a selected object, and we provide two answer options to select from.

Referential Localization (*RefLoc*) tests the human-likeness of localizing the referred object under the illusion. Given a referential expression that makes sense to humans under the effect of illusion (but may not be faithful to reality), the model needs to predict a bounding box for the object the expression is referring to. For each referential query, we consider the machine's response to be humanlike only when the pair of responses from the two images both match with those from human's. This enforces that a humanlike response from machines has to be grounded in the illusion image.

To create this benchmark, we annotate the collected illusion images with natural language

questions and the corresponding answers that humans will give under illusions. To support the study of language grounding, we also annotate the referring expressions for each of the objects with the corresponding bounding box, where the expressions are formed under illusions. We provide several paraphrases for all the language annotations to help the evaluation be more robust to models that are sensitive to language forms. All the images and corresponding annotations are verified by at least three human annotators from our team.

4.3 Experimental Setup

Vision-Language Models. To be evaluated on all of the four tasks in GVIL, the model has to be equipped with the visual question-answering skill and the object localization skill simultaneously. Among a few candidates, we choose two state-of-the-art models, the Unified-IO [166] and OFA [264], both of which are trained on a wide range of vision-language tasks, and achieve impressive performance with a strong zero-shot inference capability on unseen data. Additionally, we select two recent works that adapt large language models to understand visual images: the LLaVA [155] and InstructBLIP [48]. These models are interesting to inspect as they have shown a highly conceptual understanding of the nuance in images, such as the capability of interpreting jokes, which may also be useful in interpreting illusions. For each of the aforementioned models, there exists a range of variants in different sizes: OFA-{Tiny, Base, Large, Huge}, Unified-IO-{Small, Base, Large, XL}, LLaVA-{Vicuna-7B, Vicuna-13B}, InstructBLIP-{Vicuna-7B, Vicuna-13B}. This allows us to study the impact of size variations on model's understanding of visual illusions.

Metrics. Through the experiments, we keep track of the **Humanlike** rate to measure the alignment between humans and VLMs, which is the percentage of examples where the machine gives exactly the same answers as humans. For the *SameDiffQA* task, we also compute the **No-Illusion** rate, which corresponds to the percentage of examples where the machine consistently considers the objects as the same under both illusion and illusion-free settings. For examples where the model fails to identify the objects as the same in the illusion-free image or produces nonsense answers to the questions, we mark them as **Not Applicable (N/A)** and exclude them from the illusion recognition assessment.

4.4 Results Analysis

From our experiments, we are interested in investigating the following research questions:

- RQ1: to what extent do VLMs recognize the presence of illusions similar to humans?

- RQ2: how much do VLMs align with humans when communication happens under the influence of illusions?
- RQ3: does the degree of alignment between VLMs and human responses vary across different categories of illusions?

We highlight several of our findings across this three questions in below.

4.4.1 Illusion Recognition

The results of *SameDiffQA* are shown in Figure 4.5. Relative proportions of "humanlike," "no-illusion," and "not applicable (N/A)" responses are represented as green, orange, and grey bars respectively for each model, which all together account for 100%. First of all, we notice a large percentage of responses, across all models, fall under the N/A category. This suggests that these models often cannot even tell that the objects are identical in the illusion-free image, underscoring the need for improvement in standard vision-language reasoning capabilities beyond the scope of illusion contexts.

Given the high proportion of N/A responses, one might question the benchmark's adequacy in reliably reflecting a model's tendency towards either "humanlike" or "no-illusion". Excluding the N/A responses, we employed a χ^2-test and found that 9 out of 12 models would reject the null hypothesis which posits that the "humanlike" or "no-illusion" responses are uniformly distributed. In other words, these models do not behave randomly. Refer to Appendix A for more details. Such findings indicate that, despite certain limitations in their capabilities, our dataset and experimental design effectively gauge illusion recognition in the assessed VLMs.

When examining cases where responses are applicable for testing illusion recognition, we observe that the majority of models are more likely to fail in recognizing illusions (35.4% on average) than producing humanlike responses (15.6% on average). This discrepancy is most pronounced in the case of InstructBLIP, where the model predominantly offers 'no-illusion' answers. Conversely, the Unified-IO XL model stands out as the only model exhibiting a significant propensity towards humanlike illusion recognition. A further investigation of the underlying reason that causes this discrepancy would be interesting further work.

To illustrate how the scores evolve with model size, we plot regression lines of "humanlike" (green) and "no-illusion" (red) rates, respectively. An emerging trend reveals that "humanlike" scores tend to increase as the model scales, whereas "no-illusion" responses tend to decline. This finding suggests a positive correlation between model scale and human-machine alignment under illusions. We hypothesize that this observed trend could be attributed to the enhanced pattern-recognition capabilities of larger models. These models, arguably, are better suited

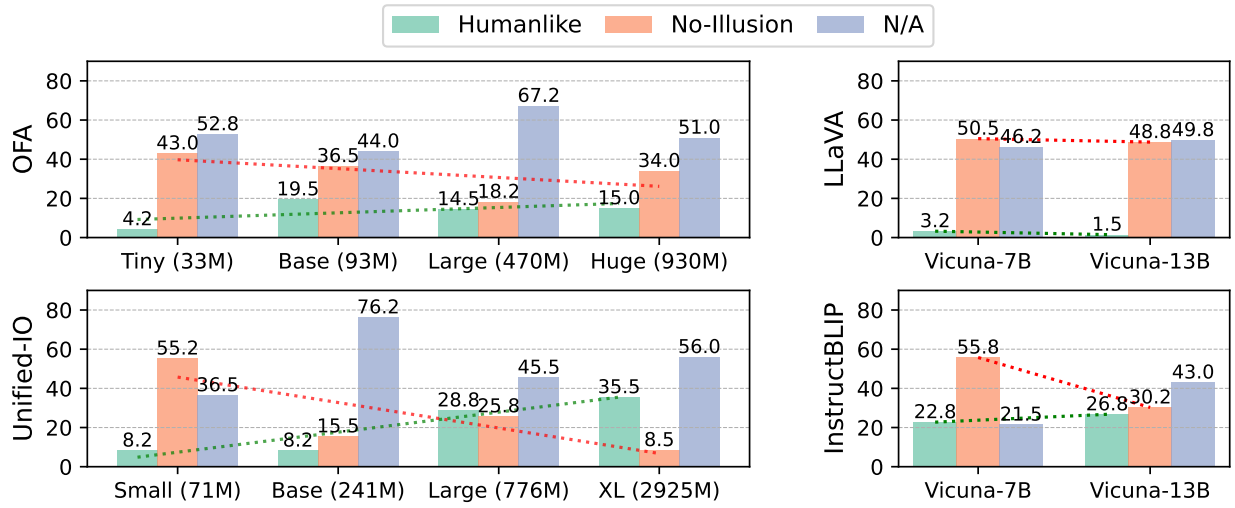


Figure 4.5: Results of *SameDiffQA*. The number shows the percentage of the answers. Each cluster represents the distribution over humanlike, no-illusion and N/A answers from a model. The green and red line correspond to the linear regression of humanlike rate and no-illusion rate across all the model sizes. Except for OFA-Large, Unified-IO-Large, InstructBLIP-13B, the differences between the humanlike rate and the no-illusion rate are statistically significant $P < 0.005$. Details are in Table B.1 Appendix A.

to discern and assimilate patterns present in data generated by humans, which may have been shaped by the influence of illusions. Consequently, it's plausible to assume that these models are progressing towards a more humanlike comprehension of illusions.

Task	Model	Pearson coeff.	p-value
SameDiffQA	OFA	0.689	0.311
	UnifiedIO	0.940	0.059*
RefQA	OFA	0.946	0.054*
	UnifiedIO	0.977	0.022**
AttrQA	OFA	0.957	0.043**
	UnifiedIO	0.853	0.146
RefLoc	OFA	0.933	0.066*
	UnifiedIO	0.960	0.039**

Table 4.3: Pearson's correlation analysis between the humanlike rate and model size. Statistically significant results with $p < 0.05$ and $p < 0.1$ are marked with ** and *, respectively.

4.4.2 Communication Under Illusion

The results of *RefQA*, *AttrQA*, and *RefLoc* experiments are shown in Figure 4.6, offering insights into the alignment between machine and human responses under the influence of

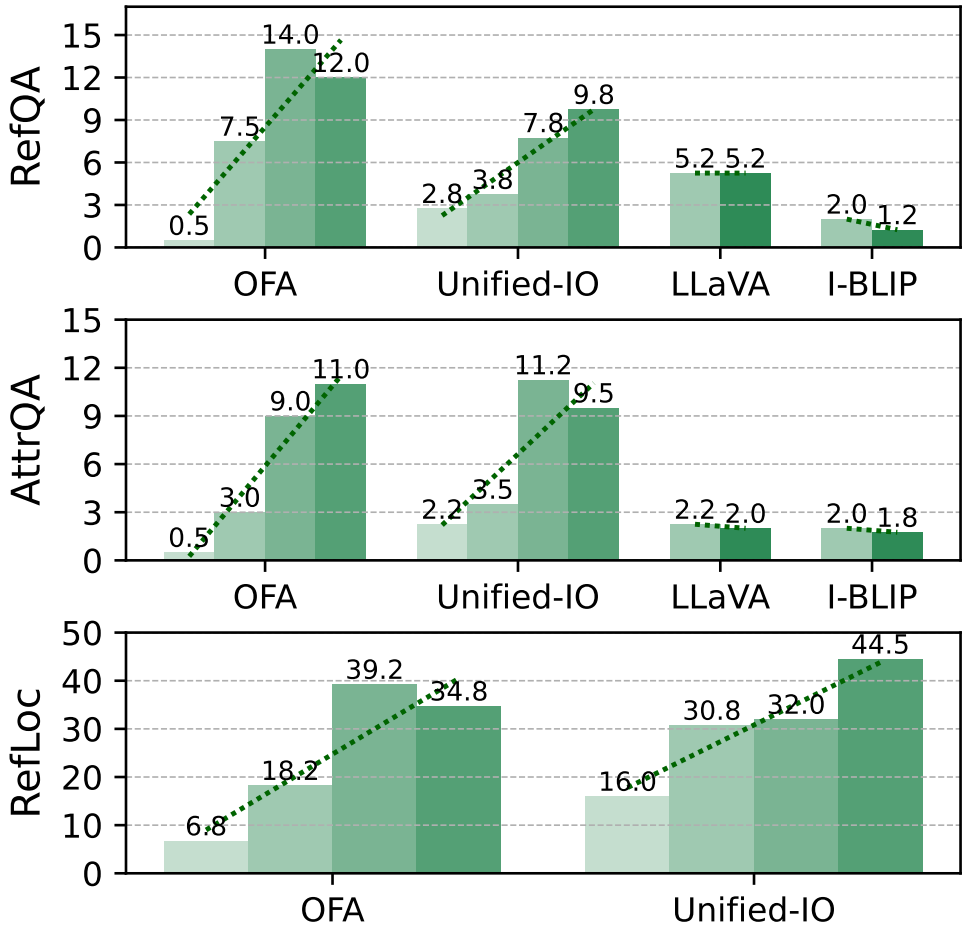


Figure 4.6: Humanlike rate on *RefQA*, *AttrQA* and *RefLoc*. Each bar represents a different model size, arranged in ascending order from left to right. Note that LLaVA and InstructBLIP cannot predict object bounding boxes thus do not have the *RefLoc* results.

visual illusions. We find that all the VLMs encounter significant challenges in responding to questions presented under the influence of visual illusions in both *RefQA* and *AttrQA*. As a result, the models obtain a maximum humanlike response rate of only 14.0% and 11.2% for *RefQA* and *AttrQA*, respectively. Interestingly, models exhibit much stronger alignment in the localization task, with the highest alignment of 44.5% achieved by Unified-IO XL. This indicates that the learned object localization skill aligns better with humans under illusions compared to the visual question answering skills. Research into the underlying reason behind this difference might be an interesting future direction.

Notably, we find a positive correlation between scaling up models and the increase of humanlike rate across different models and tasks, which echoes our earlier observations from the *SameDiffQA* experiment. To verify the statistical significance, we conducted Pearson's

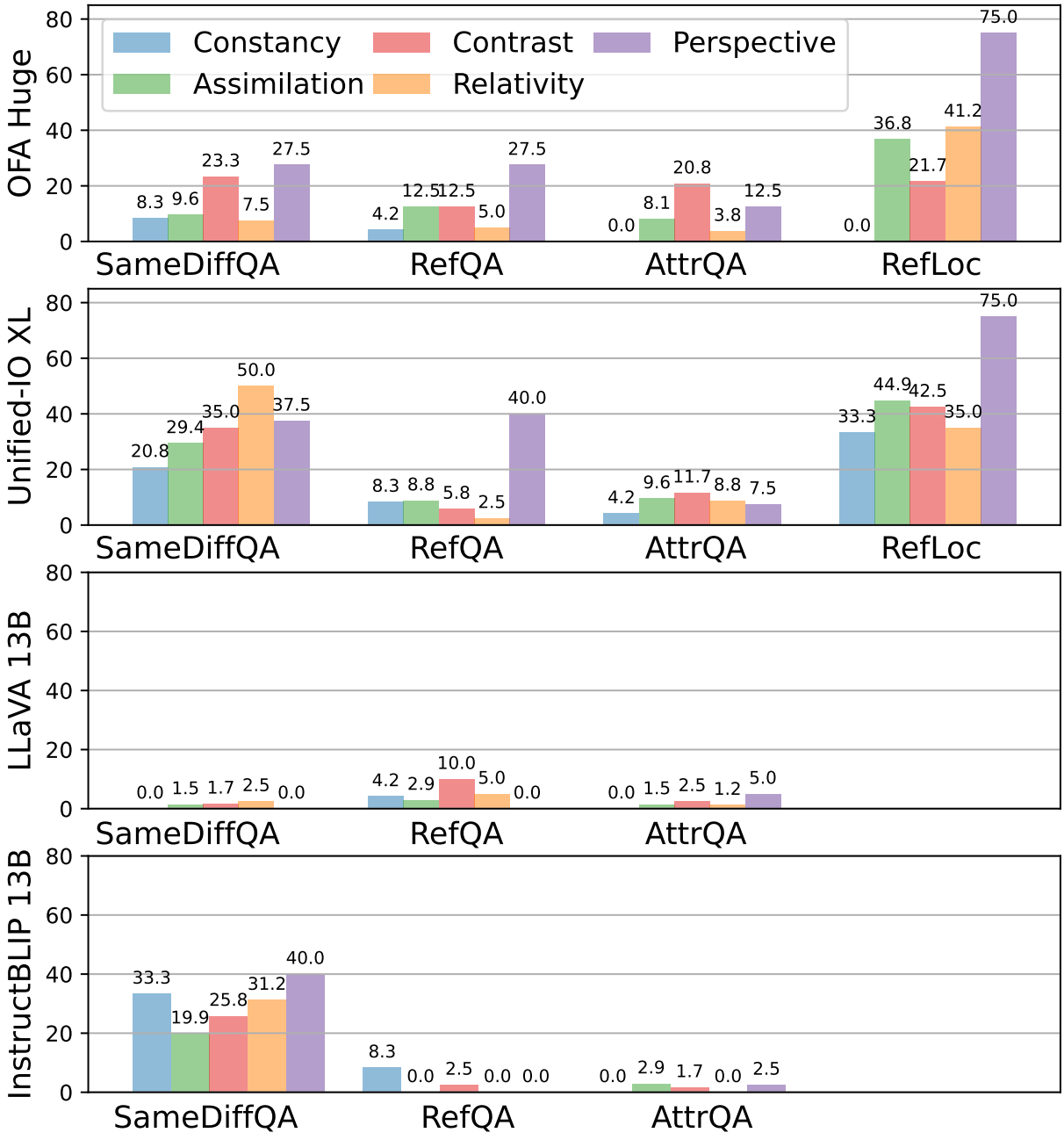


Figure 4.7: Humanlike rates of the largest model of each family, with finer-grained human-likeness scores on each illusion category.

correlation analysis for OFA and UnifiedIO models[4]. As shown in Table 4.3, 6 of the 8 tested scenarios showed significant or moderately significant positive correlations, with Pearson coefficients exceeding 0.9. Such results underscore the potential of larger models to enhance the human-machine alignment of responses across different tasks and illusion contexts.

[4]InstructBLIP and LLaVA were excluded since at least three data points are needed for the test.

4.4.3 Delving into Illusion Categories

We provide a more granular analysis by examining each type of illusion, presenting the humanlike rates for each category in Figure 4.7. The results depicted here are sourced from the largest model within each model family, namely Unified-IO Huge, OFA Huge, LLaVA Vicuna-13B, and InstructBLIP Vicuna-13B. Our observation reveals that the *perspective* category demonstrates the highest degree of alignment between machines and humans. On the contrary, color constancy illusions emerge as the category with the least congruity in relation to human responses.

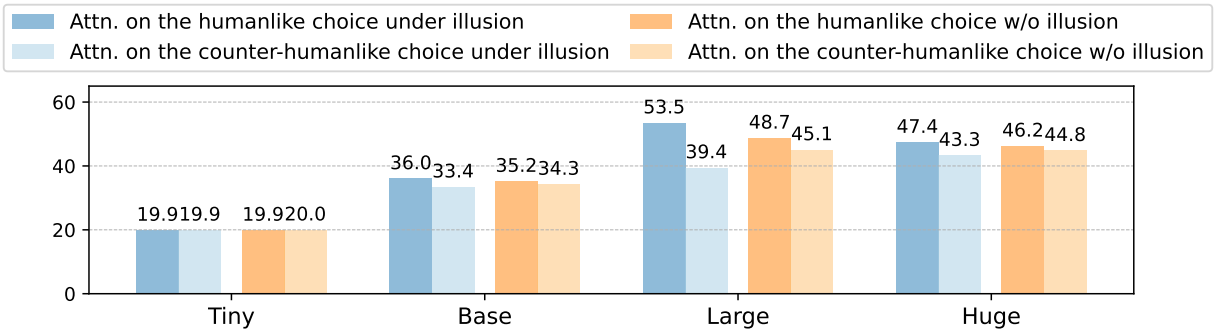


Figure 4.8: Attention weight distribution of OFA models for the *RefLoc* task.

4.4.4 Understanding the Cause of Illusions

To gain insight into model predictions under the influence of illusions, we analyze the attention distributions of OFA models in the *RefLoc* task. Specifically, we compute the attention weight from the localization query (e.g., "the smaller orange ball") to the object representation of either a "humanlike" or "counter-humanlike" perception under illusions. As depicted by the dark blue and light blue bars in Figure 4.8, as the model size increases, attention weights lean more towards the humanlike selection. This trend is consistent with the humanlike rate observed for the *RefLoc* task in Figure 4.6. To determine if this bias stems from the illusion, we also calculate attention weights for images without the illusion inducer (represented by orange bars). These weights are nearly equally distributed across both objects, suggesting that the illusion does indeed influence the model's attention and biases its predictions similarly to human perceptions.

Figure 4.9 shows an example using the attention visualization tool [6]. The first image displays the original illusion image, with two orange balls of identical size while the left ball seems smaller. The second image is devoid of the illusion inducer, while the third image artificially enlarges the right orange ball. Attention maps corresponding to the "smaller

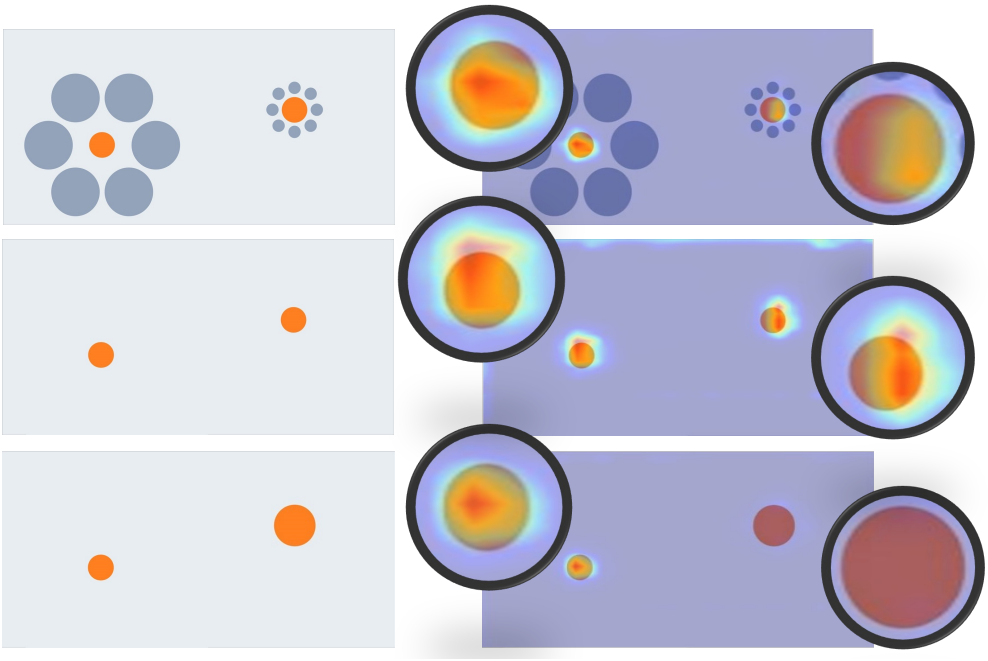


Figure 4.9: Visualization of the attention maps generated by the OFA-Large model for the *RefLoc* task. In each row, the input image is shown on the left, and the attention map for the referential expression "smaller orange ball" is shown on the right. The attention maps surrounding the object of interest are highlighted for enhanced visibility.

orange ball" query[5] are shown adjacent to each image. In the original illusion, the model predominantly focuses on the left ball, aligning with human observations. Without the illusion inducer, the query becomes ambiguous, leading to a dispersed attention map. However, when an actual size difference is present, the model's attention decisively shifts to the correctly perceived smaller ball on the left. A comparison of these attention maps highlights that while illusions can steer the model's attention similarly to humans, its effect is less pronounced than when a real disparity exists.

4.5 Discussion of Limitations

This work is only the initial attempt to the question and there are many limitations which we think of as exciting future research directions. First of all, although our experiments yields some interesting empirical findings, it is not clear why different forms of tasks (e.g., QA-based tasks vs. RefLoc) lead to a stark contrast in the results. As these findings may have implications in future technology that adapt to visual illusions, more in-depth understanding

[5]We use the second last layer of the OFA large model, as the overall attention score of this layer is the highest. Attentions from all the heads are averaged.

of these behaviors will be needed in the future. Second, our benchmark is currently small in size. It lays an infrastructure for this work. Future efforts to collect more data to form a centralized repository will be desired for studying visual illusions in both humans and machines. Third, our investigation is only based on a manually collected dataset for our intellectual curiosity. The construction of this dataset has the limitations that the effect of visual illusions are not validated by a wider range of human subjects other than the authors. While it has motivation in improving situated language communication with embodied agents in the physical world, how visual illusions play in perceiving and communicating about the real physical world remains an interesting question.

4.6 Summary of Findings

In this chapter, we investigated the alignment between human and Vision-Language Model (VLM) perception in the context of visual illusions, focusing on how these perceptual differences impact grounded language communication. We developed the **Grounding Visual Illusion in Language (GVIL)** benchmark, the first benchmark for systematic evaluation of machine visual illusion via natural language, covering five distinct illusion types (Color Constancy, Assimilation, Contrast, Geometrical Relativity, Perspective) and four tasks (SameDiffQA, RefQA, AttrQA, RefLoc).

Our experiments demonstrated several key findings:

- **Limited Humanlike Illusion Perception:** Overall alignment between VLMs and human perception under visual illusions was found to be low. On the SameDiffQA task designed for illusion recognition, models more often responded based on physical reality ('No-Illusion') than mimicking human perception ('Humanlike').

- **Positive Scaling Effects:** A consistent trend was observed where larger VLM models (with more parameters) exhibited higher 'Humanlike' rates and lower 'No-Illusion' rates across different tasks. This suggests model scale positively correlates with alignment to human perceptual tendencies, potentially due to better capturing latent patterns related to human perception in large-scale training data.

- **Task-Dependent Alignment:** Alignment varied significantly by task. Models achieved considerably higher 'Humanlike' scores on the Referential Localization (RefLoc) task (up to 44.5% for the best model) compared to Question Answering tasks (RefQA $< 14\%$, AttrQA $< 11.2\%$). This indicates that visual grounding mechanisms currently align better with human illusory perception than language-based reasoning about attributes or references under illusion.

- **Variability Across Illusion Types:** The degree of human-VLM alignment differed depending on the category of visual illusion. Geometrical perspective illusions yielded generally higher alignment compared to color-based illusions like color constancy, highlighting that different underlying perceptual phenomena are replicated to varying degrees by current VLMs.

Through these findings, we hope to raise awareness of the need to assess VLMs beyond standard benchmarks, particularly concerning phenomena like visual illusions which are fundamental to human perception yet often overlooked. Failing to account for these differences risks misalignment between human intentions and AI interpretations during real-world interactions. While our results hint at improved alignment with larger models, variations in architecture and training data necessitate further investigation, for which the GVIL benchmark provides a dedicated resource. Studying model responses to illusions offers a unique lens into their internal representations and reasoning, treating these perceptual discrepancies as a tangible case study for the broader challenge of aligning AI with implicit human biases and shared experiences, akin to aligning with societal values.

PART II: Language Grounding in Embodied Planning

CHAPTER 5

Hierarchical Task Planning with Unified Transformers

Having established the foundations of symbol grounding in static images in Part I, we now turn to the second part of this thesis, which addresses the challenge of experience grounding—connecting language to sequences of actions within 3D embodied environments. As emphasized in Chapter 1, understanding language in context often requires agents not only to perceive, but also to reason, plan, and act based on sensorimotor interactions. Inspired by the inherently hierarchical nature of human task planning, we propose a unified transformer model that learns to infer hierarchical task structures from language instructions and demonstrations, and to ground decomposed sub-goals into executable primitive actions. The work presented in this chapter is based on our publication in the Findings of ACL-IJCNLP 2021[1].

5.1 Overview

As physical agents (e.g., robots) start to emerge as our assistants and partners, it has become increasingly important to empower these agents with an ability to learn new tasks by following human language instructions. Many benchmarks have been developed to study the agent's ability to follow natural language instructions in various domains including navigation [12, 35],

[1]Yichi Zhang and Joyce Chai. "Hierarchical Task Learning from Language Instructions with Unified Transformers and Self-Monitoring." Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021.

object manipulation [183, 316] and embodied reasoning [55, 87]. Despite recent progress, learning new tasks through language instructions remains an extremely challenging problem as it touches upon almost every aspect of AI from perception, reasoning, to planning and actions. For example, on the ALFRED benchmark for task learning [234], the state-of-the-art system only achieves less than 10% task success rate in an unseen environment [240], compared to the human performance of over 90%. Most previous works apply an end-to-end neural architecture [234, 240, 243] which attempt to map language instructions and visual inputs directly to actions. While striving to top the leader board for end task performance, these models are opaque, making it difficult to understand the nature of the problem and the ability of the agent.

To address this issue, this paper takes a closer look at task learning using the ALFRED benchmark. In a departure from an end-to-end architecture, we have developed an approach to learn the hierarchical structure of task compositions from language instructions. As shown in Figure 5.1, a high-level goal directive ("place a clean mug in the coffee machine") can be decomposed to a sequence of sub-goals. Some sub-goals involve navigation in space (e.g., `Goto(Mug)`, `Goto(Sink)`) and others require manipulation of objects (e.g., `Pickup(Mug)`, `Clean(Mug)`). These sub-goals can be further decomposed into navigation actions such as `RotateLeft` and `MoveAhead`, and manipulation actions such as `Put(Mug, Sink)`, `TurnOn(Faucet)`. In fact, such hierarchical structure is similar to Hierarchical Task Network (HTN) widely used in AI planning [69]. While this hierarchical structure is explicit and has several advantages in planning and making models transparent, how to effectively learn such structure remains a key challenge.

Motivated by recent work in multi-task learning [159], we decomposed task learning in ALFRED into three sub-problems: sub-goal planning, scene navigation, and object manipulation; and developed a model called HiTUT (stands for Hierarchical Tasks via Unified Transformers) that addresses each sub-problem in a unified manner to learn a hierarchical task structure. On the ALFRED benchmark, HiTUT has achieved the best performance with a remarkably higher generalization ability. In the unseen environment, HiTUT achieves over 160% performance gain in success rate compared to the previous state of the art.

The contributions of this work lie in the following two aspects.

An explainable model achieving the new state-of-the-art performance. By explicitly modeling a hierarchical structure, our model offers explainability and allows the agent to monitor its own behaviors during task execution (e.g., what sub-goals are completed and what to accomplish next). When a failed attempt occurs, the agent can backtrack to previous sub-goals for alternative plans to execute. This ability of self-monitoring and backtracking

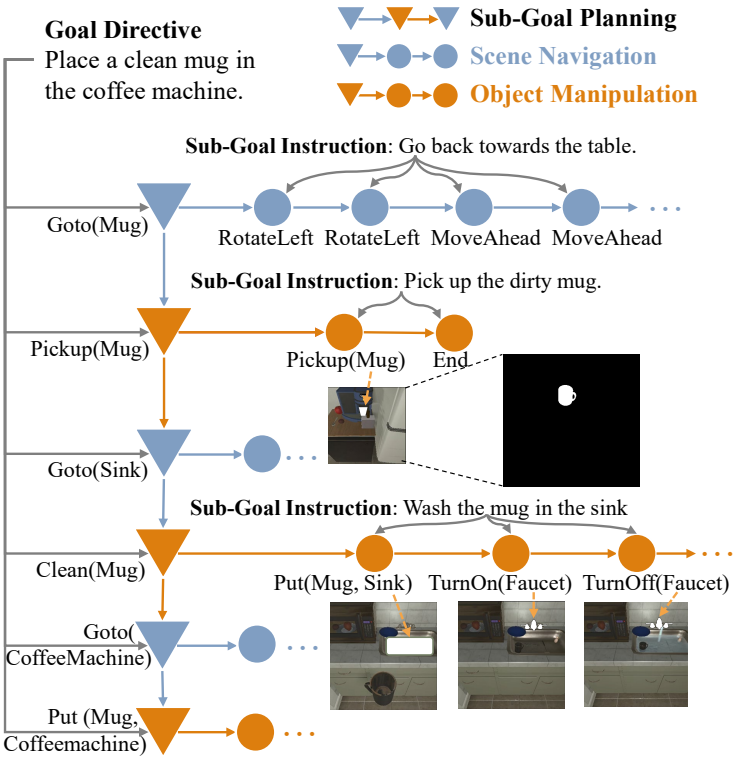


Figure 5.1: An example task in ALFRED.

offers flexibility to dynamically update sub-goal planning at the inference time to cope with exceptions and new situations. It has led to a significantly higher generalization ability in unseen environments.

A de-composable platform to support more in-depth evaluation and analysis. The decomposition of task learning into sub-problems not only makes it easier for an agent to learn, but also provides a tool for an in-depth analysis of task complexity and the agent's ability. For example, one of our observations from the ALFRED benchmark is that the agent's inability to navigate is a major bottleneck in task completion. Navigation actions are harder to learn than sub-goal planning and manipulation actions. For manipulation actions, the agent can learn action types and action arguments predominantly based on sub-goals and the history of actions, while language instructions do not contribute significantly to learning. The success of manipulation actions also largely depends on the agent's ability in detecting and grounding action arguments to corresponding objects in the environment. These findings allow a better understanding of the nature of the tasks in ALFRED and provide insight to address future opportunities and challenges in task learning.

5.2 Hierarchical Tasks via Unified Transformers

As discussed in Section 5.1, task structures are inherently hierarchical, which compose of goals and sub-goals. Different sub-goals involve tasks of different nature. For example, navigation focuses on path planning and movement trajectories, while manipulation concerns more about interactions with concrete objects. Instead of end-to-end mapping from language instructions to primitive actions [234, 240, 243], we decomposed task learning into three separate but connected sub-problems: sub-goal planning, scene navigation, and object manipulation, and developed a model called HiTUT (stands for Hierarchical Tasks via Unified Transformers) to tie these sub-problems together to form a hierarchical task structure.

5.2.1 Task Decomposition

We first introduce some notations to describe the task and the model. There are three types of information:

- Language ($\mathcal{L}$). We use G to denote a high-level goal directive, e.g., "place a clean mug in the coffee machine" and I_i to refer to a specific low-level language instruction.
- Vision ($\mathcal{V}$). It captures the visual representation of the environment.
- Predicates ($\mathcal{P}$). Symbolic representations are defined to capture three types of predicates: sub-goals (sg), navigation actions (a^n), and manipulation actions (a^m). Each sg has two parts (sg^{type}, sg^{arg}) where sg^{type} is the type (e.g., `Goto`) and sg^{arg} is the argument (e.g., `Knife`). Each a^n specifies a type (a^{n_type}) of action, from `{RotateLeft, RotateRight, MoveAhead, LookUp, LookDown}`. Each a^m has also two parts (a^{m_type}, a^{m_arg}) where a^{m_type} is the action type (e.g., `TurnOn`); a^{m_arg} is the action argument (e.g., `Faucet`).

Sub-Goal Planning. Sub-goal planning acquires a sequence of sub-goals $sg_1, \cdots, sg_n$ to accomplish the high-level goal G.

We predict the type sg_i^{type} and argument sg_i^{arg} separately to avoid the combinatorial expansion of the output space. Previous work [108] models sub-goal planning merely from high-level goal directives without visual grounding. These plans are fixed and thus not robust to potential failures during execution and variations of the visual environment. To overcome these drawbacks, our sub-goal planning is done on the fly after the previous sub-goal is executed in the environment. More specifically, our sub-goal planning objective is to learn a model (M_{sg}) that takes the visual observation at the current step (v_t), the high-level goal directive (G), and a complete sub-goal history prior to the current step ($sg_{<i}$) to predict the

current sub-goal as follows:

$$sg_i \triangleq (sg_i^{type}, sg_i^{arg}) = M_{sg}(v_t, G, sg_{<i})$$

The predicted sub-goals serve as a bridge between the high-level goal and the low-level predictions of navigation actions and/or manipulation actions.

Scene Navigation. Navigation sub-goals only require predictions for the types of navigation actions. The objective is to learn a model for navigation (M_n) which takes the current visual observation (v_t), current sub-goal (sg_i), language instruction (I_i), and the navigation action history up to the current step ($a^n_{<j}$) to predict the next navigation action:

$$a^n_j \triangleq a^{n_type}_j = M_n(v_t, I_i, sg_i, a^n_{<j})$$

Object Manipulation. For a manipulation sub-goal, in addition to the type and argument of the action, the model (M_m) also needs to generate a segmentation mask (m_j) on the current visual observation to indicate which object to interact with (i.e., which object the argument is grounded to):

$$\begin{aligned}(a^m_j, m_j) &\triangleq (a^{m_type}_j, a^{m_arg}_j, m_j) \\ &= M_m(v_t, I_i, sg_i, a^m_{<j})\end{aligned}$$

The mask prediction is crucial because the action will not be successfully executed with an incorrect grounding even if a^m_j is correctly predicted.

As described above, although the context of the three sub-problems varies, each model has similar input components from the space of $\langle \mathcal{V}, \mathcal{L}, \mathcal{P} \rangle$. This similarity inspires us to design an unified model to solve three sub-problems simultaneously.

5.2.2 Unified Transformers

We leverage the effective self-attention based model [258] to capture the correspondence of different input sources as shown in Figure 5.2. We first project the input from different modalities into the language embedding space, and adopt a transformer to integrate the information together. Multiple prediction heads are constructed on top of the transformer encoder to make predictions for the sub-goal type and argument, the action type and argument, and object masks respectively. As the three sub-problems share the similar input form, we solve them all together using a unified model based on multi-task learning [159].

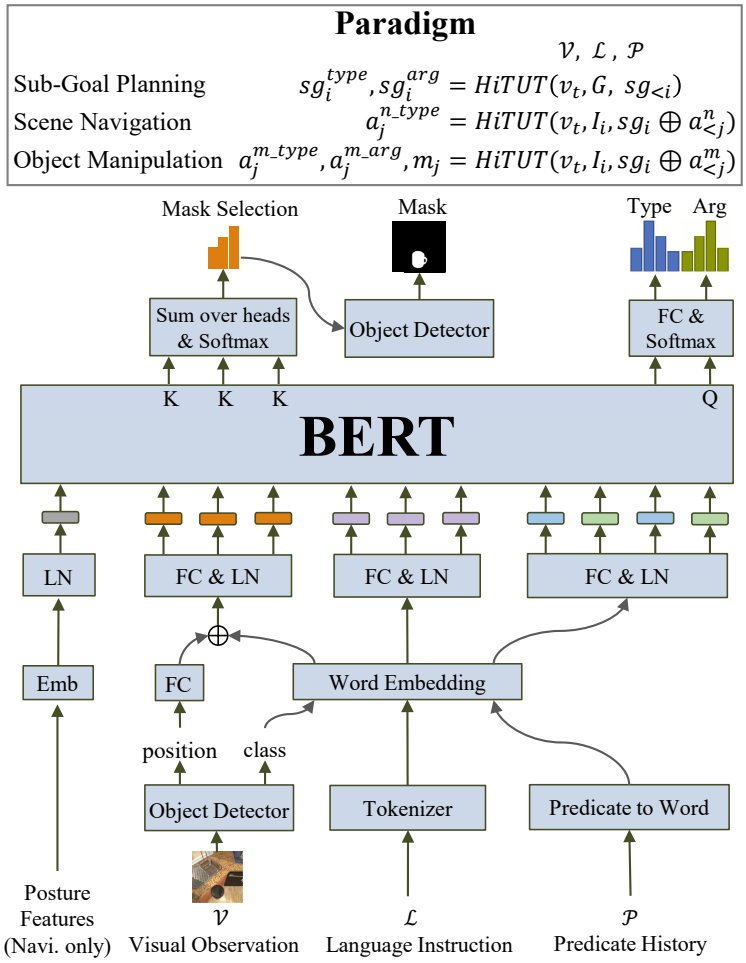


Figure 5.2: The structure of HiTUT.

Our model differs from previous works [234, 240] in the following aspects. First, we do not apply recurrent state transitions, but feed the prediction history as the input to each subsequent prediction. This may help better capture correlations between predicates and other modalities. Second, we do not use dense visual features from the scene, but rather the object detection results. By doing this, we map different modalities to the word embedding space before feeding them into the transformer encoder, thus taking advantage of the pre-trained language models. Third, we use a predicate embedding to share linguistic knowledge between predicate symbols and word embeddings.

Predicate Embedding. We use the term predicates to refer to symbolic representations including sub-goal types, action types, and their arguments. We map symbols to their corresponding natural language phrases (e.g., `AppleSliced` is mapped to *a sliced apple*). We then tokenize and embed the tokens using word embeddings, and take the sum of the

embeddings to obtain the representation of each predicate.

Vision Encoding. We use a pre-trained object detector (Mask R-CNN [97]) to encode visual information. Instead of dense features, we simply use the detection results (class labels, bounding box coordinates and confidence scores) as visual features. Specifically, we use the top K detected objects with a confidence score higher than 0.4 to form the visual features. The object class labels share the same space with object arguments, thus can be embedded into the same space. The position information of an object is encoded by a 7-dimensional vector consisting of its coordinates, width and height of the bounding box and its confidential score. This vector is first mapped to the same dimension as word embeddings by a liner transformation, then added to the class embedding to form the final object representation.

Object Grounding. HiTUT does not generate masks by itself. Instead it chooses an object from the K input objects and uses the corresponding mask generated by the object detector. This method makes use of the strong prior learned from object detection pre-training, so the model can focus on learning the grounding task. A drawback is that the object detector cannot be improved during training, and the performance of the detector determines the upper bound of our model's grounding ability. We leave the exploration of more robust grounding method for future work.

Posture Feature We use an additional posture feature to assist scene navigation, which includes the agent's rotation (N, S, E, W) and its angle of sight horizon (discretized by 15 degree). The positions are embedded and summed up to form the posture feature representation. The agent maintains its own posture in the form of a relative change to its initial posture instead of the absolute posture in the environment, thus avoid using additional sensory data.

5.2.3 Self-Monitoring and Backtracking

These unified transformers trained for sub-problems are integrated together as shown in Figure 5.3. One important advantage of intermediate sub-goal representations is to facilitate self-monitoring and backtracking which allows the agent to dynamically adjust the plan to cope with failures during execution. As shown in Section 5.3, this feature brings out the most remarkable performance gain compared to the state of the art.

Self-Monitoring. The world is full of uncertainties, and mistakes are inevitable. Based on the learned model, the agent should be able to monitor its own behaviors and dynamically

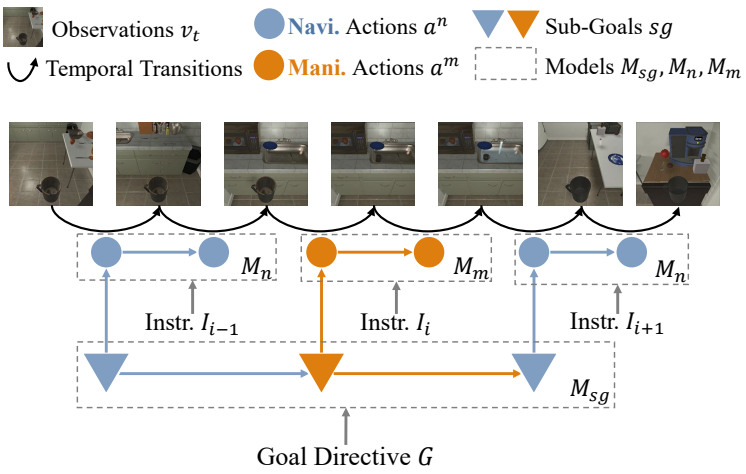


Figure 5.3: Overview of HiTUT where unified transformers for sub-programs are integrated together.

update its plan when the situation arises. Our explicit representation of sub-goals allows the agent to self-check whether some sub-goals are accomplished. Particularly for manipulation sub-goals, it is feasible for the agent to detect their failures by simply monitoring whether all the manipulation actions are successfully executed. For example, `Clean(Mug)` cannot succeed if any of the actions along the path `Put(Mug, Sink), TurnOn(Facuet), TurnOff(Facuet), Pickup(Mug)` fail. When the agent detects the failure of a subgoal, for example, as shown in Figure 5.4 the manipulation sub-goal `Pickup(Mug)` fails, it can reason about whether the previous sub-goal (i.e., `Goto(Mug)`) is successfully achieved.

Backtracking. In classical AI, backtracking is the technique to go back and try an alternative path that can potentially lead to the goal. As shown in Figure 5.4, when `Pickup(Mug)` fails, the agent backtracks to `Goto(Mug)` and tries a different sequence of primitive actions to accomplish this sub-goal. In ALFRED, only based on the visual information without other sensory information (e.g., only observing a mug without knowing how far it is), is it difficult to check whether a navigation sub-goal is successfully achieved (e.g. whether a `Mug` is reachable). So every time after trying a different path for `Goto(Mug)`, the agent will check whether the subsequent manipulation action `Pickup(Mug)` is successful. If it's successful, the agent will move on to the next sub-goal; otherwise the agent will continue to backtrack until a limit on the maximum number of attempts is reached. Our explicit representation of sub-goals makes this backtracking possible and has led to a significant performance gain in unseen environments.

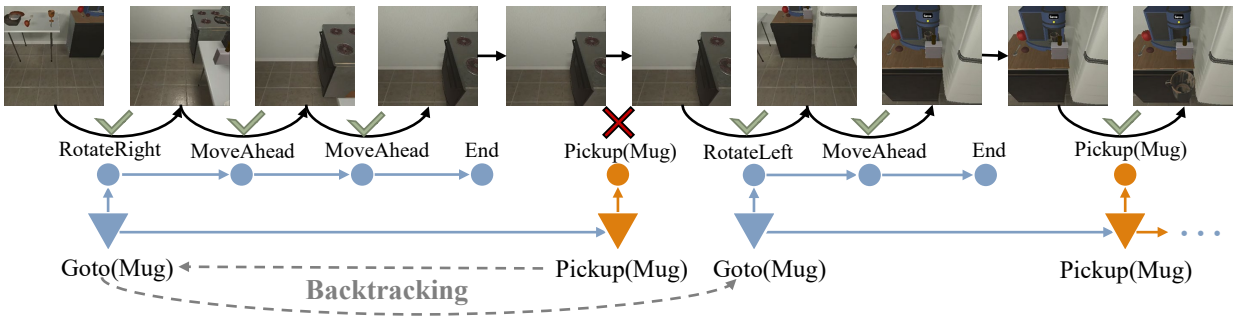


Figure 5.4: Illustration of self-monitoring and backtracking.

5.3 Experiments

5.3.1 Setting and Implementation

Dataset. We follow the train/validation/tests data partition proposed in ALFRED, where validation and test sets are further split into *seen* and *unseen* based on whether the scene is shown to the model during training. Each sub-goal planning step or a primitive prediction step forms a data instance for the corresponding sub-problem. The number of data instances are shown in Table 5.1.

	Train	Validation		Test	
		Seen	Unseen	Seen	Unseen
#Scenes	108	88	4	107	8
#Demonstrations	6,574	251	255	483	488
#Annotations	21,023	820	821	1,533	1,529
#Sub-goals	162k	6.4k	6.0k	-	-
#Navi. Actions	983k	39k	35k	-	-
#Mani. Actions	209k	8.3k	8.1k	-	-

Table 5.1: Statistics of data distribution in ALFRED. The number of annotations is equivalent to the number of tasks in each split.

Pre-training. We employ the pre-training followed by fine-tuning paradigm for both the object detector and the main model. For the object detector, we use a Mask R-CNN [97] model pre-trained on MSCOCO [150], and fine-tune it on 50K images collected by replaying the expert trajectories in the ALFRED train split. As we observe that the model struggles on detecting small objects together with large receptacles, we train two networks to detect movable objects and big receptacles separately. We use the pre-trained RoBERTa [160] model

to initialize the transformer encoder.

Training. We perform imitation learning (supervised learning) on the expert demonstrations. The ground-truth labels of sub-goals and primitive actions are obtained from the metadata. Different input and output labels are organized for each sub-problem respectively as described in Section 5.2. We use the mask proposal that overlaps the most with the ground truth mask as the mask selection label if the intersection-of-union is above 50%. If there is no valid mask proposals, the label is assigned to 0 as an indicator of non-valid grounding. We optimize the cross-entropy loss between model predictions and the ground truth. We follow the multi-task training schema in [159] where for each iteration, a batch is randomly sampled among all the sub-problems, and the model is updated according to the corresponding objective. More details are in Appendix.

Evaluation Metrics. ALFRED leverages an interactive evaluation in the AI2-THOR environment [126]. A task is considered successful if all the goal conditions (e.g. the target object is placed on a correct receptacle and in a requested state such as heated or cleaned etc.) are met. Three measures are used: (1) success rate (the ratio of successfully completed tasks), (2) goal-condition rate (ratio of completed goal conditions), and (3) a weight version of these two rates which takes into account of the length difference between the predicted action sequence and the expert demonstrated action sequence [234].

Baselines. We compare HiTUT to: (1) Seq2Seq - an LSTM-based baseline model with progress monitoring proposed in [234]; (2) HAM - a hierarchical attention model over enriched visual inputs [189], and (3) MOCA - a modular approach which also uses a Mask R-CNN for mask generation [240] and achieved previous state-of-the-art performance.

5.3.2 Evaluation Results

5.3.2.1 Overall Performance of HiTUT

We first evaluate the overall performance of the proposed framework as shown in Table 5.2. On the testing data reported by the leader board, in seen environments, HiTUT achieves comparable performance as MOCA. However in unseen environments, HiTUT outperforms MOCA by over 160% on success rate. This demonstrates our hierarchical task modeling approach has higher generalization ability compared to end-to-end models. Self-monitoring and backtracking enabled by hierarchical task structures allows the agent to better handle new situations. Remarkably, only based on high-level goal directives (i.e., HiTUT (G Only))

Model	Validation Seen		Validation Unseen		Test Seen		Test Unseen	
	Success	Goal-Cond	Success	Goal-Cond	Success	Goal-Cond	Success	Goal-Cond
Seq2Seq	3.70 (2.10)	10.00 (7.00)	0.00 (0.00)	6.90 (5.10)	3.98 (2.02)	9.42 (6.27)	0.39 (0.80)	7.03 (4.26)
HAM	-	-	-	-	12.40 (8.20)	20.68 (18.79)	4.50 (2.24)	12.34 (9.44)
MOCA	19.15 (**13.60**)	28.50 (**22.30**)	3.78 (2.00)	13.40 (8.30)	**22.05** (**15.10**)	28.29 (**22.05**)	5.30 (2.72)	14.28 (9.99)
HiTUT	**25.24** (12.20)	**34.85** (18.52)	**12.44** (**6.85**)	**23.71** (**11.98**)	21.27 (11.10)	**29.97** (17.41)	**13.87** (**5.86**)	**20.31** (**11.51**)
HiTUT (*G* only)	18.41 (7.59)	25.27 (12.55)	10.23(4.54)	20.71 (9.56)	13.63 (5.57)	21.11 (11.00)	11.12 (4.50)	17.89 (9.77)
Human	-	-	-	-	-	-	91.00 (85.80)	94.50 (87.60)

Table 5.2: Task and Goal-Condition success rates. The path length weighted version is in parentheses. The highest values per column are in **bold**. "-" denotes scores that are not reported. *G only* denotes only using the goal directive during evaluation without any sub-goal instructions.

	Model	*Pick*	*Put*	*Cool*	*Heat*	*Clean*	*Slice*	*Toggle*	Avg.
Seen	Seq2Seq	32	**81**	88	85	81	25	**100**	70
	MOCA	53	62	87	84	79	51	93	73
	HiTUT	**81**	77	**95**	**100**	**83**	**81**	97	**88**
Unseen	Seq2Seq	21	46	92	89	57	12	32	50
	MOCA	44	39	38	86	71	55	11	49
	HiTUT	**71**	**69**	**100**	**97**	**91**	**78**	**58**	**81**

Table 5.3: Success rates of manipulation sub-goals on validation sets. The highest values per fold are in **bold**.

without using any sub-goal instructions, is HiTUT able to obtain a success rate of 11% in unseen environment, achieving 110% performance gain compared to MOCA. This result indicates that HiTUT can learn prior task knowledge from the hierarchical modeling process and apply that directly in new environment with some success. Nevertheless, our results are far from human performance and there is still huge room for future improvement.

To have a better understanding of the problem, we also conduct evaluations on sub-goals. The agent is positioned at the starting point of each sub-goal by following the expert demonstration and the success rate of accomplishing the sub-goal is measured. HiTUT predicts first a symbolic sub-goal representation and then the action sequence to complete the sub-goal. As shown in Table 5.3, HiTUT outperforms previous models on almost all of the manipulation sub-goals by a large margin. The performance gain is particularly significant in unseen environment, which demonstrates the advantage of our explicit hierarchical task modeling in low-level action planning.

	Valid Seen		Valid Unseen	
#BT	Success	Goal-Cond	Success	Goal-Cond
No	10.5 (6.0)	18.4 (13.8)	5.2 (3.0)	13.5 (11.1)
2	18.9 (9.9)	27.6 (18.0)	10.2 (5.9)	20.2 (13.6)
4	23.1 (11.3)	32.9 (18.6)	12.9 (7.0)	22.7 (12.9)
6	25.6 (12.0)	35.1 (18.5)	14.5 (7.4)	24.3 (12.3)
8	27.2 (12.5)	37.0 (18.5)	16.2 (7.8)	25.9 (12.1)

Table 5.4: Success rates w.r.t. the allowed maximum backtracking number (#BT).

	Seq2Seq	MOCA	Our model with different #backtracks					
			no	1	2	4	6	8
Seen	51	54	35	48	56	64	68	70
Unseen	22	32	31	45	53	60	63	65

Table 5.5: Success rate of the navigation sub-goal `Goto` with backtracking .

5.3.2.2 The Role of Backtracking

We conduct experiments to better understand the role of self-monitoring and backtracking. We repeat the task-solving evaluation with different limits on the allowed maximum number of backtracking. The agent only stops when the model predicts to stop (i.e., predicts `End`) or it reaches the backtracking limit. As shown in Table 5.4, as the limit increases, the task/goal-condition success rate increases accordingly. One thing notable is that the gap between success rates (weighted and unweighted) become larger when more backtrack attempts are allowed. This is within our expectation because backtracking deviates from instruction following navigation to goal-oriented exploration, which usually takes more steps than the expert demonstration.

Since backtracking is particularly targeted to navigation sub-goals `Goto` (see Section 5.2.3), we further examine the role of number of re-tries (i.e. backtracks) in completing the sub-goal. As shown in Table 5.5, HiTUT reaches more targets when given more opportunities to backtrack. The backtracking is most beneficial in unseen environment.

5.3.2.3 Complexity of Tasks

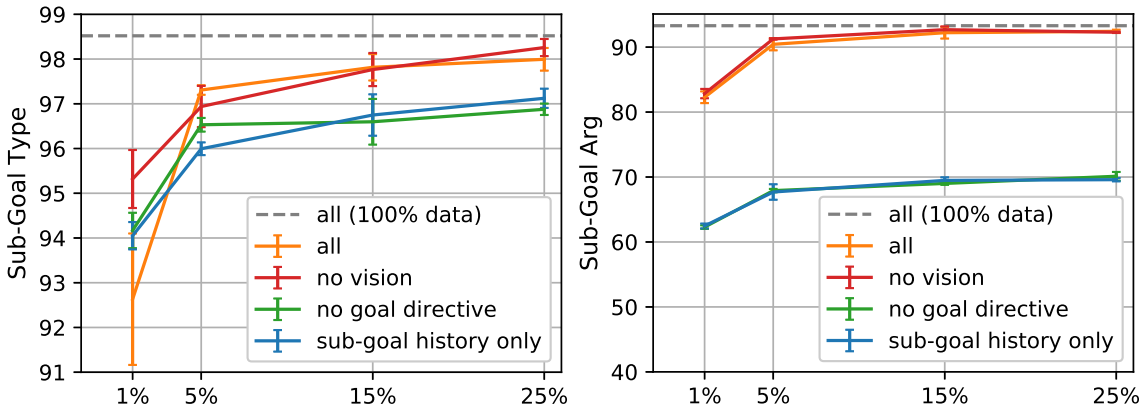
Task decomposition provides a tool to enable better understanding of task complexity and agent's ability. To do that, we replace different part of model predictions by the corresponding oracle sub-goals, actions, or masks, as shown in Table 5.6.

Method	Valid Seen		Valid Unseen	
	Success	Goal-Cond	Success	Goal-Cond
HiTUT	25.2 (12.2)	34.8 (18.5)	12.4 (6.8)	23.7 (12.0)
+ Oracle				
SG	29.0 (15.6)	39.1 (21.3)	14.0 (7.6)	25.6 (12.7)
N	75.0 (72.7)	78.0 (77.4)	57.9 (60.0)	67.7 (65.2)
SG+N	79.2 (77.8)	84.0 (81.3)	64.2 (64.2)	72.0 (68.1)
SG+N+M	89.0 (100)	90.0 (100)	80.5 (100)	83.7 (100)
SG+N+GR	99.3 (99.0)	99.4 (99.1)	99.4 (99.3)	99.6 (99.6)

Table 5.6: Success rates of HiTUT with different parts of predictions replaced by oracle operations with expert demonstrations. N, M, SG and GR denote oracle navigation actions, manipulation actions, sub-goals and object grounding (i.e., mask generation) respectively.

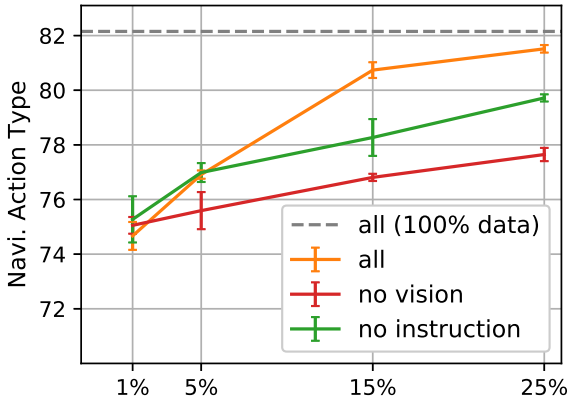
Using oracle sub-goals improves the success rate for 2%-6% (line SG), showing sub-goal planning is a relatively easy problem and the agent can perform reasonably well. After using the oracle navigation actions, the seen and unseen success rates are boosted by an absolute gain of 50% and 46% respectively (line N), indicating that navigating to reach target objects is a particularly hard problem and the agent performs poorly. When oracle sub-goals, navigation actions, and manipulation actions (only symbolic representations) are given (line SG+N+M), the task success is bounded by the performance of the pre-trained object mask generator (i.e., visual grounding of the object). When oracle object masks are given together with oracle sub-goals and navigation actions (line SG+N+GR) and the agent only needs to predict symbolic representation of manipulation actions, the performance is near perfect. These last two lines indicate that predicting the type and the argument of a manipulation action is a rather simple problem in the ALFRED benchmark while grounding action arguments to the visual environment remains a challenging task.

We further examine the complexity of learning to solve sub-problems by evaluating the next-step prediction accuracy given the golden history under different conditions as shown in Figure 5.5. The models are trained and evaluated with different combinations of input and different amount of training data. We observe that excluding the visual input does not hurt performance for sub-goal prediction and manipulation action prediction (shown by a,b,d,e). This indicates that in ALFRED, pure symbolic planning is often independent from visual understanding, which is consistent with the findings in [235]. However, this could be an oversimplification brought by the bias in the dataset rather than a true reflection of the physical world. For example, next action prediction can be made by remembering the correlation of predicates instead of reasoning over vision and language, due to the lack

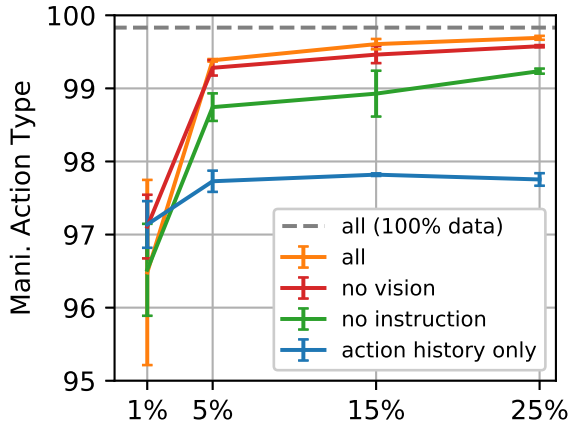


(a) Sub-Goal Type

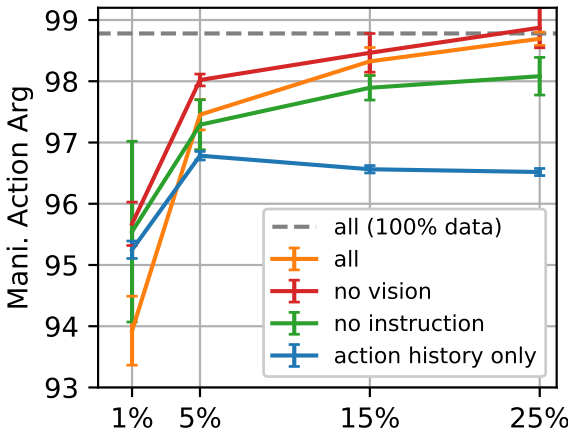
(b) Sub-Goal Argument



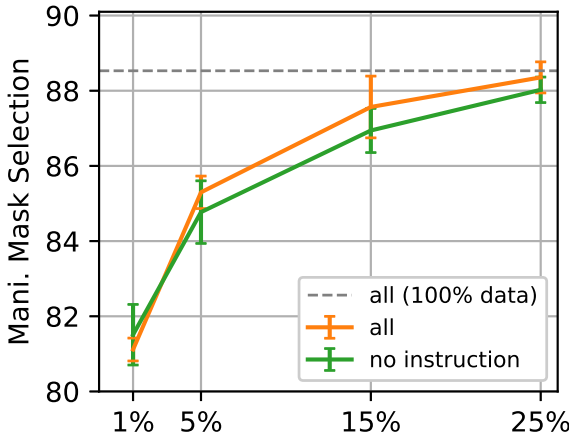
(c) Navigation Action Type



(d) Manipulation Action Type



(e) Manipulation Action Argument



(f) Manipulation Mask Selection

Figure 5.5: Step-by-step prediction accuracies given the golden sub-goal/action history w.r.t. the proportion of training data on the unseen validation set. Each solid line corresponds to a specific input configuration. Dashed lines are the scores obtained using 100% data.

of diversity of the task environments. Removing language instructions causes a minimal performance drop of 1%-2% on action prediction tasks, which brings up the question about the usefulness of language instructions in this benchmark. Furthermore, the prediction accuracy is above 90% and 98% with only 5% training data for sub-goal and manipulation planning respectively, while the navigation accuracy is only 82% given all the data. This again supports the finding that planning and performing navigation actions is a much harder problem than sub-goal planning and manipulation actions in ALFRED.

5.4 Discussion of Limitations

Despite the significant advancements demonstrated by HiTUT in hierarchical task learning for embodied instruction following, several limitations remain.

Performance Gap. While HiTUT achieved state-of-the-art results on ALFRED at the time, its absolute success rates, particularly in unseen environments (around 14-20% depending on backtracking), remain substantially lower than human performance (over 90%). This highlights the difficulty of the task and indicates significant room for improvement across all aspects of perception, planning, and execution.

Navigation Bottleneck. Our analysis (Table 5.6) identified navigation as a primary bottleneck limiting overall task success. Developing more effective navigation modules with richer spatial representations or specialized architectures is crucial for future improvement.

Grounding Dependency and Limitations. HiTUT relies on a pre-trained object detector (Mask R-CNN) for visual perception and object grounding. The grounding mechanism simply selects the best-matching mask proposal from this detector. The overall performance is inherently capped by the accuracy and robustness of the fixed object detector, which cannot be improved during HiTUT's training. Enable end-to-end training of grounding mechanism is a promising direction to mitigate this issue.

Potential Dataset Biases and Interpretation. The analysis presented in Figure 5.5 raised questions about the ALFRED benchmark itself. The findings suggest that sub-goal and manipulation action prediction might be less dependent on visual input than expected, potentially learnable through symbolic correlations alone. Furthermore, the impact of low-level language instructions on predicting the next action appeared minimal. While HiTUT effectively leverages the hierarchical structure, these observations suggest that

success on ALFRED might partially stem from exploiting dataset regularities rather than deep, visually-grounded language understanding for every step. This highlights the need for careful benchmark design and diverse evaluation scenarios to ensure models are developing truly generalizable grounding capabilities.

5.5 Summary of Findings

This chapter introduced HiTUT (Hierarchical Tasks via Unified Transformers), an approach designed to improve the ability of embodied agents to learn complex tasks from language instructions by explicitly modeling hierarchical task structures. The key findings and contributions of this work are:

- **Hierarchical Modeling with Unified Transformer:** We successfully decomposed the complex task learning problem on ALFRED into three interconnected sub-problems: sub-goal planning, scene navigation, and object manipulation, and showed that a unified transformer was able to address all three sub-problems within a single model, enabling joint learning of task structures and action policy.
- **Effectiveness of Self-Monitoring and Backtracking:** The explicit hierarchical structure enabled the implementation of self-monitoring (tracking sub-goal completion) and backtracking mechanisms. Allowing the agent to re-attempt failed sub-goals, particularly navigation, substantially boosted task success rates, especially in novel environments where initial plans are more likely to fail.
- **State-of-the-Art Performance and Generalization:** HiTUT achieved state-of-the-art performance on the ALFRED benchmark, significantly outperforming previous approaches. Notably, it demonstrated remarkably stronger generalization ability, achieving over a 160% relative improvement in task success rate in unseen environments compared to the prior best model (MOCA).
- **Analytical Insights:** The decomposition allowed for in-depth analysis, revealing that navigation is a major performance bottleneck in ALFRED. It also indicated that predicting symbolic manipulation actions and arguments is relatively easier than grounding these arguments to specific objects via mask selection in the visual scene.

In summary, HiTUT demonstrated that moving away from purely end-to-end approaches towards models that explicitly represent and learn hierarchical task structures can lead to

significant improvements in performance, generalization, and interpretability for embodied instruction following. The ability to monitor progress and backtrack provides crucial robustness for operating in complex, partially predictable environments.

CHAPTER 6

Deliberative Agent for Following Natural Language Instructions

In the previous chapter, we demonstrated that hierarchical task learning can significantly improve the ability to learn complex tasks in challenging embodied environments. While promising, end-to-end neural models that predict primitive actions directly from raw language and visual inputs often struggle with long-horizon tasks that demand sustained reasoning and spatial memory. To address this limitation, this chapter explores a neuro-symbolic approach that integrates 3D spatial-semantic world modeling with symbolic planning, enabling agents to perform deliberate action planning within an internal symbolic representation of their environment. This framework supports robust, efficient, and interpretable instruction following through explicit reasoning about both the environment and task structure. The work presented in this chapter is based on our publication at EMNLP 2022[1].

6.1 Overview

Natural language instruction following with embodied AI agents [30, 11, 250, 209, 234, 193] is a notoriously difficult problem, where an agent must interpret human language commands to perform actions in the physical world and achieve a goal. Especially challenging is the hierarchical nature of everyday tasks, for example, *making breakfast* may require preparing one or more dishes (e.g., toast and coffee), each of which requires several sub-tasks of navigating through the environment and manipulating objects (e.g., finding a knife, slicing bread, cooking it in the toaster), and even more fine-grained primitive actions entailed by them (e.g., walk forward, pick up knife). Effectively following such instructions often requires reasoning about subgoals and reconciling them with the world state and overall goal. However, despite recent

[1]Yichi Zhang, Jianing Yang, Jiayi Pan, Shane Storks, Nikhil Devraj, Ziqiao Ma, Keunwoo Yu, Yuwei Bao, and Joyce Chai. "DANLI: Deliberative Agent for Following Natural Language Instructions." Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing, 2022 [305].

progress, past approaches are typically *reactive* [274] in their execution of actions: conditioned on the rich, multimodal inputs from the environment, they perform actions directly without using an explicit representation of the world to facilitate grounded reasoning and planning [197, 302, 230]. Such an approach is inefficient, as natural language instructions often omit trivial steps that a human may be assumed to already know [313]. Besides, the lack of any explicit symbolic component makes such approaches hard to interpret, especially when the agent makes errors.

Inspired by previous work toward *deliberative* agents in robotic task planning, which apply long-term action planning over known world and goal states [232, 7, 242, 261], we introduce DANLI, a neuro-symbolic Deliberative Agent for following Natural Language Instructions. DANLI combines learned symbolic representations of task subgoals and the surrounding environment with a robust symbolic planning algorithm to execute tasks. First, we build a uniquely rich semantic spatial representation (Section 6.3.1), acquired online from the surrounding environment and language descriptions to capture symbolic information about object instances and their physical states. To capture the highest level of hierarchy in tasks, we propose a neural task monitor (Section 6.3.2) that learns to extract symbolic information about task progress and upcoming subgoals from the dialog and action history. Using these elements as a planning domain, we lastly apply an online planning algorithm (Section 6.3.3) to plan low-level actions for subgoals in the environment, taking advantage of DANLI's transparent reasoning and planning pipeline to detect and recover from errors.

Our empirical results demonstrate that our deliberative DANLI agent outperforms reactive approaches with better success rates and overwhelmingly more efficient policies on the challenging Task-driven Embodied Agents that Chat (TEACh) benchmark [193]. Importantly, due to its interpretable symbolic representation and explicit reasoning mechanisms, our approach offers detailed insights into the agent's planning, manipulation, and navigation capabilities. This gives the agent a unique self awareness about the kind of exceptions that have occurred, and therefore makes it possible to adapt strategies to cope with exceptions and continually strengthen the system.

6.2 Problem Definition

The challenge of hierarchical tasks is prominent in the recent Task-driven Embodied Agents that Chat (TEACh) benchmark for this problem [193]. Here, language instructions are instantiated as a task-oriented dialog between the agent and a commander (who has comprehensive knowledge about the task and environment, but cannot perform any actions), with varying granularity and completeness of guidance given. We focus on the Execution from Dialog

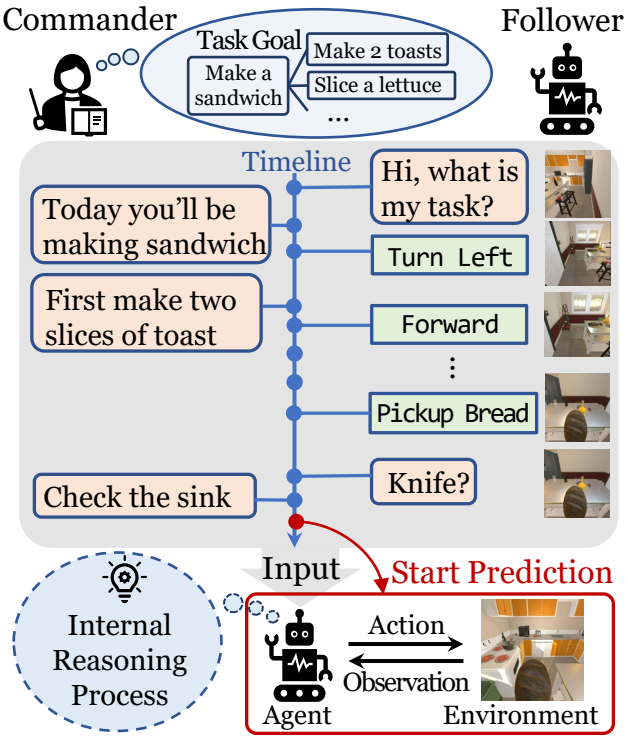


Figure 6.1: An example task in TEACh.

History (EDH) setting in TEACh, where the agent is given a dialog history as input, and is expected to execute a sequence of actions and achieve the goal set out by the commander. This setting allows us to abstract away the problem of dialog generation, and focus on the already difficult problem of instruction following from task-oriented dialog.

As shown in Figure 6.1, a *task*, e.g., *Make a Sandwich*, may have several subtasks that the agent must achieve in order to satisfy the overall task goal. The success of a task/subtask is achieved by meeting a set of *goal conditions*, such as slicing a bread and toasting two pieces. At each timestep, the agent receives an egocentric visual observation of the world and the full dialog history up to that time, and may execute a single low-level *action*. Actions can either involve navigation, e.g., to step forward, or manipulation, e.g., to pick up an object. Manipulation actions additionally require the agent to identify the action's target object by specifying a pixel in its field of view to highlight the object. The execution continues until the agent predicts a `Stop` action, otherwise the session will terminate after 1000 timesteps or 30 failed actions. At this time, we can evaluate the agent's completion of the task.

It is worth noting that while we focus on TEACh, our approach is largely transferable between benchmark datasets and simulation environments, albeit requiring retraining of some components.

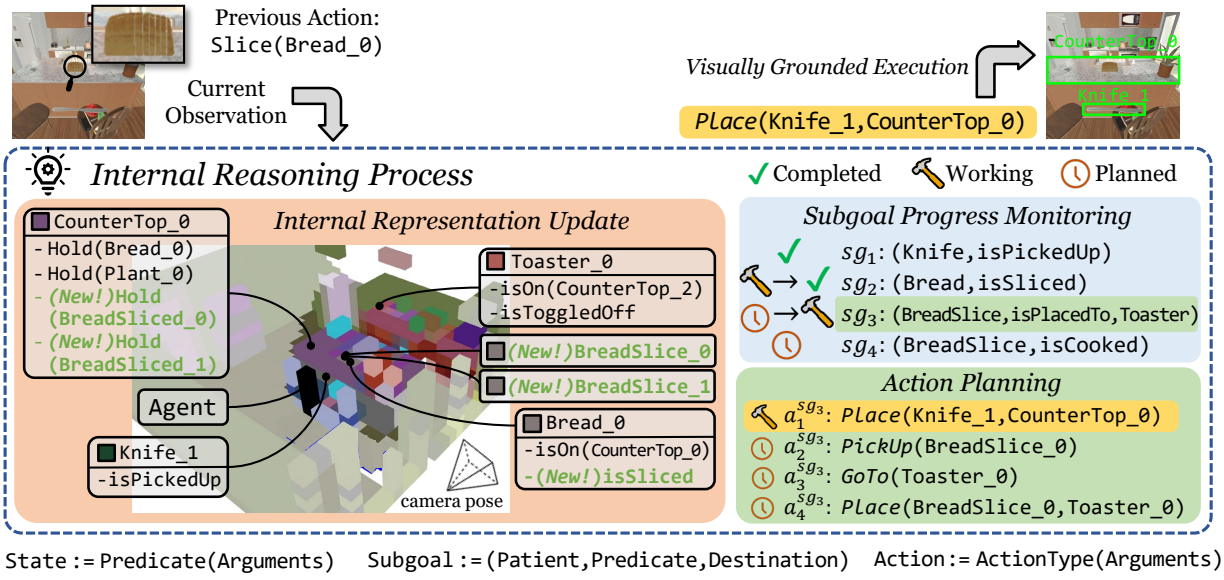


Figure 6.2: Illustration of our agent's reasoning process behind a single decision step. After receiving the current observation, the agent first updates its internal representation (left), then checks the current subgoal progress (upper right), and plans for the next steps (lower right). Finally the first action in the plan is popped out and grounds to the agent's ego-centric view for execution. In the pop-up boxes we show example object instances with their instance ids, states and positions in the 3D map. New instances and state changes are labeled in green. The status for each subgoal and action is labeled in front of it, where the arrows denote status transitions.

6.3 A Neuro-Symbolic Deliberative Agent

An overview of our neuro-symbolic deliberative agent is shown in Figure 6.2. We first introduce the symbolic notions used in our system. We use the object-oriented representation [65] to represent the symbolic world state. Each *object instance* is assigned an instance ID consisting of its canonicalized class name and ordinal. We define a *state* in the form of `Predicate(Arguments)` as an object's physical state or a relation to another object. We define *subgoals* as particular states that the agent should achieve while completing the task, represented by a symbolic form `(Patient, Predicate, Destination)`[2], where the `Patient` and `Destination` are object classes, and the `Predicate` is a state that can be applied to the `Patient`. We define an *action* in the agent's plan as `ActionType(Arguments)` where each argument is an object instance.

To complete tasks, our agent reasons over a learned spatial-symbolic map representation (Section 6.3.1) to generate a hierarchical plan. At the high level, it applies a neural language model to the dialog and action history to predict the complete sequence of completed and

[2] `isPlacedTo` is the only predicate with a `Destination`.

future subgoals in symbolic form (Section 6.3.2). For each predicted subgoal, it then plans a sequence of low-level actions using both the symbolic subgoal and world representations online, with robustness to various types of planning and execution failures (Section 6.3.3). Next, we describe how each component works and highlight our key innovations.

6.3.1 World Representation Construction

The reasoning process of an embodied AI agent relies heavily on a strong internal representation of the world. As shown in Figure 6.2, we implement the internal representation as a semantic map incorporating rich symbolic information about object instances and their physical states. We introduce our methods for the construction of this representation in this section.

3D Semantic Voxel Map As the agent moves through the environment while completing a task, it constructs a 3D semantic voxel map to model its spatial layout. Following [24], we use a depth estimator to project the pixels of egocentric observation images and detected objects to 3D point cloud and bin the points into 0.25m^3 voxels. The resulting map can help symbolic planner (Section 6.3.3) break down high-level navigation actions, such as `GOTO Knife_0`, to atomic navigation actions such as `Forward, TurnLeft, LookUp`.[3]

Object Instance Lookup Table Everyday tasks can involve multiple instances of the same object, and thus modeling only object class information may be insufficient.[4] As shown in the internal representation update part of Figure 6.2, we store object instance information for a single task episode in a symbolic lookup table, where each instance in the environment is assigned a unique ID once observed. These symbols in the lookup table become the planning domain of the symbolic planner (Section 6.3.3). To collect this symbolic lookup table, we use a panoptic segmentation model[5] to detect all object instances in the current 2D egocentric visual frame. These 2D instance detections are then projected into the 3D map, and we use each instance's 3D centroid and size information to match and update existing object instances' information in the lookup table[6]. As the agent moves through the scene and receives more visual observations, the symbolic lookup table becomes more complete and accurate.

[3]See Appendix D.1.2 for more details on path planning.

[4]For example, when making a sandwich, the agent will likely need to distinguish the top and bottom pieces of bread to make the sandwich complete.

[5]As opposed to a semantic segmentation model as used in prior work [32, 182, 24], which can only detect object class information.

[6]To perform this update and decide whether a newly detected instance should be merged with an existing instance or added as a new one, we use a matching algorithm described in Appendix D.1.3.

Physical State Prediction Additionally, tasks can hinge upon the physical states of particular object instances. For example, when making coffee, the agent should disambiguate dirty and clean coffee mugs and make sure to use the clean mug. To recognize the physical state of each object instance, we propose a physical state classification model where inputs include the image region of a detected object instance and its class identifier, and the output is physical state labels for the instance. As classifying the physical state from visual observation alone can introduce errors, we also incorporate the effect of the agent's actions into physical state classifications. For example, the `isToggledOn` attribute is automatically modified after the agent applies the `ToggleOn` action, overriding the classifier's prediction.

6.3.2 Subgoal-Based Task Monitoring

Due to the hierarchical nature of tasks, natural language instructions may express a mix of high-level and low-level instructions. In order to monitor and control the completion of a long-horizon task given such complex inputs, we first model the sequence of high-level *subgoals*, i.e., key intermediate steps necessary to complete it.

As shown in Figure 6.3, we apply a sequence-to-sequence approach powered by language models to learn subgoals from the dialog and action history. At the beginning of each session, our agent uses these inputs to predict the sequence of all subgoals. Our key insight is that to better predict subgoals-to-do, it is also important to infer *what has been done*. As such, we propose to additionally predict the completed subgoals, and include the agent's action history as an input to support the prediction.

To take advantage of the power of pre-trained language models for this type of problem, all inputs and outputs are translated into language form. First, we convert the agent's action history into synthetic language (e.g. `PickUp(Cup)` → *"get cup"*), and feed it together with the history of dialog utterances into the encoder. We then decode language expressions for subgoals one by one in an autoregressive manner. As the raw outputs from the decoder can often be noisy due to language expression ambiguity or incompleteness, we add a *natural-in, structure-out* decoder which learns to classify each of the subgoal components into its symbolic form, and transform them to a language phrase as decoder input to predict the next subgoals.

6.3.3 Online Symbolic Planning

Symbolic planners excel at generating reliable and interpretable plans. Given predicted subgoals and a constructed spatial-symbolic representation, PDDL [4] planning algorithms can be applied to generate a plan for each subgoal.[7] These short-horizon planning problems

[7]See Appendix D.1.5.1 for more details on PDDL planning.

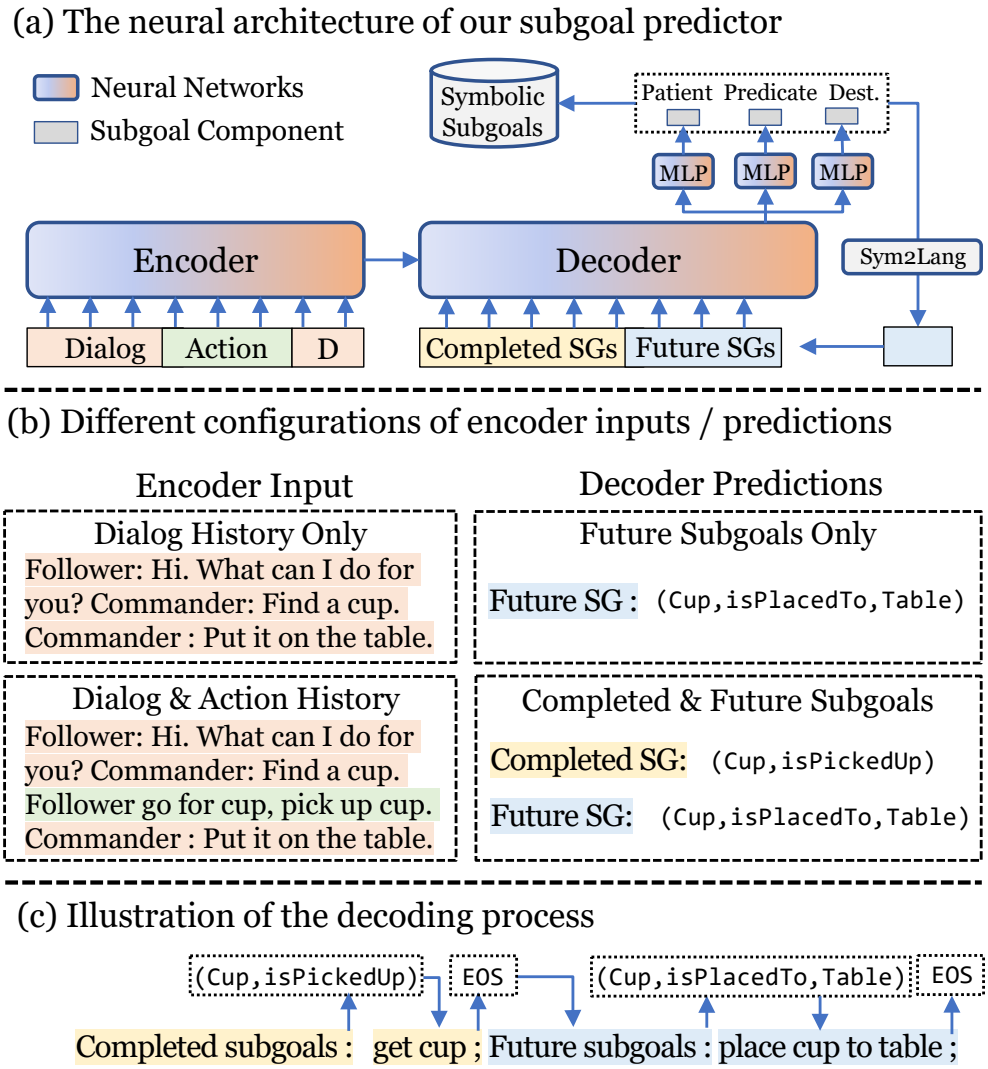


Figure 6.3: Overview of the subgoal learning process. Figure (a) shows the model architecture. Figure (b) shows the different input/output configurations we experiment with. Figure (c) illustrates the decoding process. We first predict completed subgoals and then predict the future subgoals conditioned on them, where we use different prompts to distinguish different types of subgoals.

reduce the chance of drifting from the plan during execution. Nonetheless, failures are bound to happen during execution. A notable advantage of our approach is the transparency of its reasoning process, which not only allows us to examine the world representation and plan, but also gives the agent some awareness about potential exceptions and enables the development of mechanisms for replanning. In this section, we introduce several new mechanisms to make online symbolic planning feasible and robust in a dynamic physical world.

Finding Unobserved Objects The agent's partial observability of the environment may cause a situation where in order to complete a subgoal, the agent needs an object that has not been observed yet. In this case, a traditional symbolic planner cannot propose a plan, and thus will fail the task. To circumvent this shortcoming, we extend the planner by letting the agent search for the missing object(s). Specifically, during planning, our agent assumes that all objects relevant to subgoals exist, mocking full observability. If it has not observed any instance of an object required to satisfy the subgoal, then the agent plans with an assumed dummy object instance, which is assigned an additional state predicate `unobserved`. By incorporating these dummy objects, a plan can still be produced. For example, if the subgoal is `(Mug,isPickedUp)` but the agent has not observed a mug, the plan will become `Search(Mug), GoTo(Mug), PickUp(Mug)`. During execution, the agent is equipped with a dedicated `Search` action, which lets the agent explore the environment until an instance of the target object class has been observed and added to the world representation.[8]

Search Space Pruning Large object and state search space slows symbolic planner down and creates unbearable latency to the system, as also found in [7]. We propose to solve this problem by pruning the search space to only include the relevant object instances. First, the planner finds all relevant object types for a subgoal predicate. For example, if the subgoal is to cook an object, then all object classes that can be used to cook (e.g., Microwave, Stove Burner, Toaster, etc.) are seen as relevant, even if the class is not explicitly stated in the subgoal. Once relevant object classes are determined, the agent collects relevant instances of those classes. If there are other instances tied to the state of an instance, they are also deemed relevant. For example, if an object is inside of a receptacle, that receptacle object instance is also deemed relevant in the search space. All irrelevant instances are then discarded to speed up planning.

Action Failure Recovery The ability to detect exceptions is critical, as the agent's actions may fail for unexpected reasons. For example, an agent may try to place an object in a receptacle, but the action fails because the receptacle is full. We can address this by applying exception-specific strategies; in this case, the agent should try to clear out the receptacle first. Figure 6.4 shows a decision tree of recovery mechanisms that can be used to handle various types of these exceptions. If recovery fails, this replanning process repeats until a step limit threshold is hit, at which point the agent gives up and moves on to the next subgoal. While we manually define recovery strategies as an initial step, other ways to acquire or learn these strategies can be explored in the future.

[8] Details on object search can be found in Appendix D.1.2.

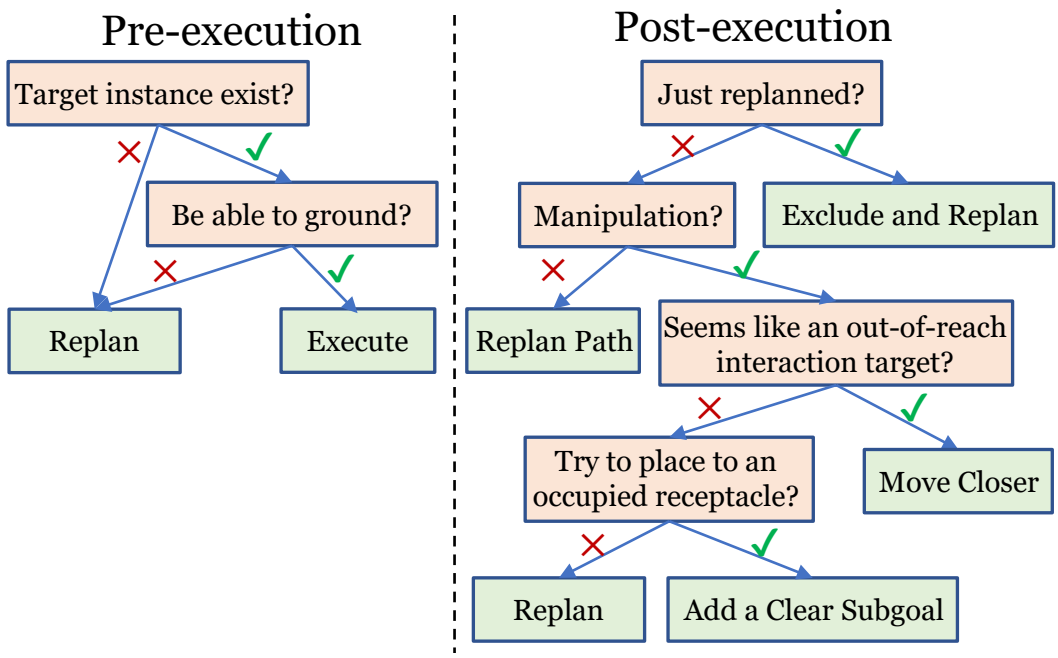


Figure 6.4: Decision trees for handling exceptions encountered during plan execution. We term pre-execution errors as infeasible actions that cannot be instantiated in the situation, and post-execution errors as those when no outcome is observed after the action is executed. Red boxes denote situations based on the agent's belief. Green boxes denote exception handling strategies.

6.4 Experimental Results Analysis

6.4.1 Experimental Setup

We follow the evaluation methods originally proposed for TEACh, reporting metrics for task completion success and efficiency. First, *task success rate* is the proportion of task sessions for which the agent met all goal conditions. *Goal condition success rate* is more lenient, measuring the average proportion of goal conditions that were met in each session. Each metric has a path length weighted (PLW) counterpart, where the agent is penalized if it takes a longer sequence of actions than the human-annotated reference path. Given a metric M, the PLW version of M is calculated by $M_{PLW} = \frac{\hat{L}}{max(\hat{L},L_*)} * M$, where $\hat{L}$ and L_* are the lengths of the predicted and reference path, respectively. There are also seen and unseen divisions for each split, where seen and unseen indicate whether the rooms in that division appear in the train set.

Implementations Our sequence-to-sequence subgoal predictor is implemented by the BART-Large language model [137]. For depth estimation, we fine-tune the depth model from

Model	Val Seen		Val Unseen		Test Seen		Test Unseen	
	Success	Goal-Cond	Success	Goal-Cond	Success	Goal-Cond	Success	Goal-Cond
Reactive								
ET	8.3 (1.1)	8.7 (3.8)	8.4 (0.9)	6.3 (3.7)	8.8 (0.3)	9.5 (3.0)	7.4 (1.0)	6.1 (3.2)
HET	8.0 (1.2)	10.6 (5.7)	8.6 (1.8)	9.5 (5.6)	9.5 (0.9)	9.4 (4.9)	10.0 (1.7)	9.5 (5.1)
HET-ON	12.6 (3.4)	16.9 (10.2)	12.5 (3.7)	15.1 (9.5)	11.4 (3.5)	16.1 (9.7)	12.3 (3.1)	14.8 (9.3)
Deliberative								
DANLI-PaP	13.9 (8.2)	21.7 (19.5)	14.9 (6.0)	19.8 (17.0)	17.3 (7.4)	19.1 (17.4)	13.5 (6.2)	18.8 (18.2)
DANLI (Ours)	**16.9 (9.1)**	**25.1 (22.6)**	**17.3 (7.2)**	**23.9 (19.4)**	**18.6 (9.4)**	**24.8 (21.9)**	**16.7 (7.3)**	**23.0 (20.6)**

Table 6.1: Task and Goal-Condition success rates on the TEACh EDH benchmark. The path-length-weighted version of metrics are in (parentheses). The highest values per column are in **bold**.

[24] on TEACh. For object instance detection, we use mask2former [42] as our panoptic segmentation model and fine-tune CLIP [213] to classify physical states. Lastly, we use Fast Downward [98] for symbolic planning. More details can be found in the Appendix.

Baselines We compare DANLI to several baselines:

- **Episodic Transformer (ET)**: End-to-end multimodal transformer model that encodes both dialog and visual observation histories, and predicts the next action and object argument based on the hidden state [197]. We reproduced the results based on the official code.[9]
- **Hierarchical ET (HET)**: A hierarchical version of ET where low-level navigation actions are abstracted out by a navigation goal (e.g. `GoTo Mug`), then predicted by a separate transformer.
- **Oracle Navigator (HET-ON)**: HET with the navigator replaced by an oracle navigator that can obtain the ground truth target object location from the simulator's metadata.
- **DANLI-PaP**: A version of DANLI where symbolic planning is replaced by procedures as programs (PaP), i.e., manually defined rule-based policies following [313].
- **JARVIS** [309]: A neuro-symbolic model similar to DANLI in spirit that also leverages a language model for subgoal planning and constructs maps for navigation. However, low-level plans for each subgoal are manually defined instead of being generated by a symbolic planner.

[9] https://github.com/alexpashevich/E.T.

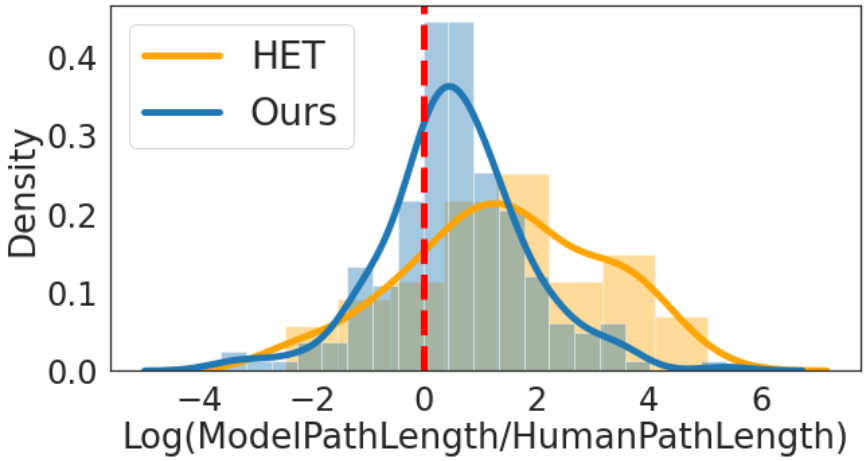


Figure 6.5: Comparison between HET and DANLI of relative success trajectory path lengths against ground-truth human trajectory lengths. The red dashed line denotes the same length as human trajectories, thus area to its left corresponds to the proportion of model predictions that achieved better efficiency than human performance.

6.4.2 Overall System Performance

We compare our agent with baseline models on the EDH task from TEACh in Table 6.1.[10]

Success Rate DANLI consistently outperforms all baseline models across both splits in every task and goal condition success rate metric. It outperforms HET-ON, the best reactive baseline equipped with an oracle navigator, by a sharp margin, demonstrating the advantage of our world representation and navigation planning. It also outperforms the deliberative DANLI -PaP baseline, showing the value of employing existing mature automatic planning tools in the system rather than handcrafting procedural knowledge for the agent.

As shown in Table 6.2, there is a high variance in the success rate of different task types, ranging from 10.6% up to 57.4%. The inverse correlation between the task success rate and the average number of task goal conditions indicates that longer-horizon tasks are more challenging, as expected.

Efficiency For task completion efficiency, we find that DANLI again outperforms all baselines in PLW metrics, including HET-ON by over 10% in PLW goal condition success rate. We further compare the predicted trajectory lengths of our agent with the HET baseline and

[10]The results are based on the new validation/test split as described in the official TEACh repository (https://github.com/alexa/teach#teach-edh-offline-evaluation), as the original test sets are not publicly available. Note that JARVIS is not listed here because their results are based on the original data split, which is not directly comparable to ours. A comparison with JARVIS on the original validation set is listed in Table D.1 in Appendix D.2.1.

Task Name	#Session	Avg. #GCs	DANLI	
			SR	GC
WaterPlant	61	1.0	57.4	59.3
PutInOne	95	1.9	26.3	38.1
Boil	82	2.2	19.5	29.8
PutAll	90	2.2	26.7	33.0
Coffee	122	2.8	27.9	41.9
CleanAll	114	2.8	12.3	27.3
Toast	201	7.1	15.4	27.8
Slices	200	7.1	15.5	20.8
CookSlices	243	7.5	11.9	30.2
Breakfast	380	8.7	14.7	21.7
Sandwich	287	9.0	13.9	22.9
Salad	274	10.9	10.6	15.8

Table 6.2: Per task performance on the unseen split. Tasks are sorted in a decreasing order w.r.t. the number of average goal conditions.

ground truth human data in Figure 6.5. DANLI outperforms HET in efficiency by a large margin, with 26% of tasks surpassing human efficiency. These results show the superiority of the deliberative agents in creating both accurate and efficient plans for task completion.

6.4.3 Generalization to New Tasks

To demonstrate the generalizability of our approach, we additionally applied DANLI to the trajectory from dialog (TfD) subtask of TEACh without re-training any modules. The major difference between EDH and TfD is the form of inputs given at each timestep. In EDH, the agent receives the contextualized interaction history (including dialog and physical actions) up to the current time, while in TfD, the agent is given the full session dialog without context, and must infer actions to complete the task induced by the dialog (analogous to instruction following). As shown in Table 6.3, DANLI significantly outperformed both the ET baseline and JARVIS, achieving success rates up to about 8%, compared to a maximum of 1.8% from baselines which are specifically trained on the TfD data.

Model	Valid Seen		Valid Unseen	
	SR	GC	SR	GC
ET	1.02 (0.17)	1.42 (4.82)	0.48 (0.12)	0.35 (0.59)
JARVIS	1.70 (0.20)	5.40 (4.50)	1.80 (0.30)	3.10 (1.60)
DANLI (Ours)	**4.97 (1.86)**	**10.50 (10.27)**	**7.98 (3.20)**	**6.79 (6.57)**

Table 6.3: Results on the TfD subtask of TEACh.

Object Type	Valid Seen			Valid Unseen		
	PQ	SQ	RQ	PQ	SQ	RQ
All	70.65	84.68	80.17	52.91	71.90	60.25
Things	71.36	85.11	81.40	56.37	75.10	64.37
Stuff	65.43	81.77	71.71	32.95	53.46	36.49

Table 6.4: Performance of our panoptic segmentation model on valid seen and unseen split. PQ, SQ, RQ are panoptic quality metric, segmentation quality metric and recognition quality metric originally defined in [122].

6.4.4 Analysis of Submodules

6.4.4.1 Perception Modules

We first perform standard evaluation on the panoptic segmentation and physical state estimation modules. Note that we do not perform a dedicated evaluation for the depth estimation model as it is directly adopted from [24] with minimal change.

Panoptic Segmentation In Table 6.4, we report the panoptic segmentation performance of our model on both validation seen and unseen splits. The significant performance drop on PQ of about 20% when transferring from seen to unseen room setup indicates the difficulty in building a more generalized model, which we leave future works to explore.

Physical State Estimation In Table 6.5, we report the performance of our model on both validation seen and unseen splits. It is worth noting that although our model can perform well on `isDirty` state with an accuracy of over 95%, `isToggled` state only has an accuracy of 77.0 % and 74.7%. After sampling and inspecting data points and the model's corresponding predictions, we argue that the performance gap between different states mainly comes from the difference in the states' visual saliency. For example, while `isDirty` state affects almost

State Name	Valid Seen	Valid Unseen
	Accuracy	Accuracy
All	85.5	85.6
`isDirty`	95.8	96.5
`isFilledWithWater`	82.7	86.2
`isToggled`	77.0	74.7

Table 6.5: Accuracy of our state estimation model on both validation seen and unseen split. `isDirty`, `isFilledWithWater` and `isToggled` are the physical states that our system needs.

Model	Patient	Predicate	Destination
DANLI	**81.1**	**65.6**	**76.4**
w/o Action History	71.4	60.1	69.5
w/o Actions in Language	73.6	57.4	74.2
w/o Pred. Completed SG	77.8	63.8	76.3

Table 6.6: Subgoal prediction accuracy of DANLI on the validation unseen split, compared to ablations that do not condition on the action history, language forms of the actions, and self-predicted completed subgoals.

every part of the object, the effect of `isToggled` is much more inconspicuous, e.g. `isToggled` only affects the color of a small LED light on the `CoffeeMachine`.

6.4.4.2 Subgoal-Based Task Monitoring

We conduct ablations on our subgoal-based task monitor from the perspective of subgoal prediction accuracy and overall task completion.

Subgoal Prediction Accuracy In Table 6.6, we examine the subgoal prediction accuracy of our subgoal prediction module (shown in Figure 6.3). First, we find conditioning on action history in addition to the dialog history achieves much higher prediction accuracy, suggesting this is a strong signal for the model to infer the progress and predict which subgoals still need to be completed in the task. Further, we find that converting actions from unnatural symbols into language improves the prediction performance, showing the advantage of translating symbols to natural language inputs when using pre-trained language models. Finally, we show that conditioning on the model's predicted completed subgoals also improves the performance (as opposed to inferring them implicitly).

Model	Valid Seen		Valid Unseen	
	SR	GC	SR	GC
DANLI	**16.9** (9.1)	**25.1 (22.6)**	**17.3 (7.2)**	**23.9 (19.4)**
last subgoal only	**16.9** (**12.2**)	19.0 (21.4)	12.4 (5.6)	12.9 (14.6)
w/o scene pruning	14.9 (8.2)	20.2 (20.3)	11.8 (5.9)	14.2 (15.6)
w/o replanning	14.9 (8.2)	22.5 (20.4)	14.9 (6.9)	19.0 (17.5)

Table 6.7: Ablation study of components with respect to the end task performance. *Last subgoal only* refers to directly working on the last predicted subgoal instead of all predicted subgoals one after another. *Without scene pruning* refers to using the planner without search space pruning. *Without replanning* refers to disabling replanning from the action failure recovery strategy.

Task Completion In the second row of Table 6.7, we see the impact of monitoring all subgoals rather than the current one on the end task performance. In 6 out of 8 metrics, DANLI , which learns to predict all subgoals throughout the task session, achieves a significant improvement. A possible explanation of the improvement is that some subgoals can only be addressed when a previous one is completed,[11] while not all subgoals are strictly conditioned on previous ones.[12]

6.4.4.3 Online Symbolic Planning

We also conduct ablations for our online planner, specifically in its capabilities for search space pruning and action failure recovery.

Finding Unobserved Objects The planner depends heavily on the constructed spatial-symbolic map, and errors here may cause errors in scene navigation, especially when target objects are hidden in the environment. As 17.6% of failures are caused by not finding the target object, we made a further analysis on our agent's object search performance. We found that DANLI performs at least one object search action in 11.8% of subgoals across 21.7% of the game sessions. Among these sessions, the success rate is 3.7%, which is far below the overall success rate of 17.3%. This indicates object search is indeed a big performance bottleneck. To improve performance, future work may develop stronger methods to locate missing objects.

[11]For example, `(BreadSlice, isPlacedTo, Plate)` requires `(BreadSlice, isSliced)` to be achieved first for bread slices to exist in the domain.

[12]For example, `isSliced(Bread)` entails addressing `isPickedUp(Bread)` along the way.

Method	Plan Search Fail Rate	Avg. Runtime
w/o scene pruning	24.3%	94.64s
w/ scene pruning	**9.1%**	**4.37s**

Table 6.8: PDDL planning success rate and average runtime. We set the overall time-out for domain translation and solution searching to be 120s in our experiments.

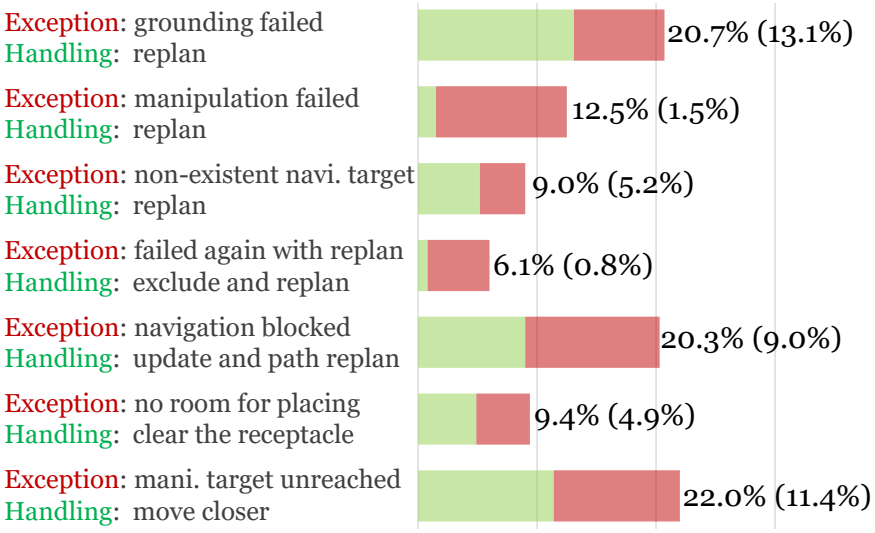


Figure 6.6: Distribution of execution exceptions encountered and self-handled by our agent. The proportions of self-handled exceptions are in parentheses. The results are based on 2708 exceptions encountered in 1078 sessions from the validation unseen split.

Search Space Pruning As seen in Table 6.8, scene pruning sharply improves both planning success (i.e., the rate at which formed PDDL problems had a solution generated for them) and the average runtime of the planner, which greatly speeds up online execution. Table 6.7 also depicts task and goal-condition success rates for our model without scene pruning, showing that scene pruning contributes to DANLI's success, and that improving PDDL plan search complexity improves overall task success.

Action Failure Recovery DANLI's exception handling mechanisms enable it to identify and recover from exceptions to prevent some types of potential task failures. These mechanisms work in symbolic space, and are fully interpretable. In Figure 6.6, we show the distribution of exception types the agent encountered and how it attempted to recover. A major source of exceptions is the mismatch between the agent's perceived internal world state to the real situation. This is because the internal state is updated per timestep, while a generated plan remains unchanged unless it is not applicable anymore. Typical cases include failure to

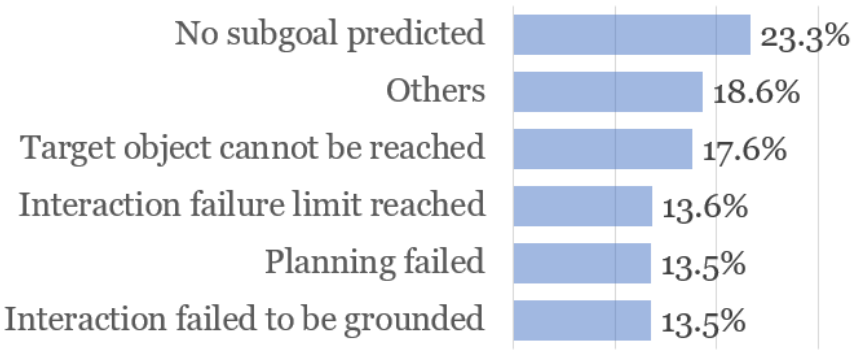


Figure 6.7: Distribution of unrecoverable errors identified by our agent that cannot be recovered.

ground to the interaction target, being blocked by an obstacle due to the change of estimated destination, or attempting to navigate to an object that no longer exists. We find that replanning can handle about half of these failure cases. Within these cases, our proposed mechanisms for addressing exceptions with receptacle occupancy and object reachability contribute to around 15% of recoveries. Meanwhile, when a manipulation fails (i.e., no effect observed) and replanning does not help, our agent can only handle about 12% of these cases. Future work can apply further analysis and exception handling techniques to mitigate this type of failure. As shown in the last line of Table 6.7, replanning contributes up to 2.4% of the overall task success rate.

6.4.5 Failure Analysis

Finally, thanks to the transparent logging of the agent's internal state and reasoning process, we can perform a detailed analysis of the agent's failures. As shown in Figure 6.7, Subgoal prediction is a major bottleneck (accounting for 23.3% of failures), since the agent only takes actions for subgoals that are predicted by this module. Most remaining errors are related to planning and exception handling. Inability to find a plan accounts for 13.5% of failures, since the agent can only execute actions that it has planned. Meanwhile, 17.6% of failures are caused by not finding an instance of a planned target object after searching for it. Interaction and grounding failures, where interaction with the virtual environment fails and the planner's exception handling mechanisms fail to resolve the issue, account for a respective 13.6% and 13.5% of errors. Other types of failures are categorized as *others*, which are typically due to inaccuracies in the agent's world representation. We hope the error analysis can help point out future research directions to improve the agent's robustness.

6.5 Discussion of Limitations

Despite the strong results, we are still far away from building embodied agents capable of following language instructions, and we identify the following major bottlenecks that will motivate our future work.

As in many embodied agents which apply planning algorithms, DANLI operates in a closed domain of objects and actions, which requires manually specifying object affordances as well as state changes caused by actions. This means that if faced with a task involving new objects and/or predicates, the agent will fail without updating the knowledge base and re-training perception modules. Moreover, the agent is tied to its action space in the Thor simulator, and cannot easily acquire new actions. As recent years have seen an exciting application of large language models for acquiring action plans [8], future work can address this by exploring the possibility to acquire new objects and actions in this space, including object affordances and action effects.

Second, the evaluation applied to DANLI is completely automatic and only measures task completion information. We do not evaluate the safety of the agent's movements, or the disruption of its actions to the environment. This evaluation cannot capture a human user's trust and confidence toward the agent's behavior. Before such agents can be brought from the simulator space into the real world, extensive study will be required to ensure the safety and respect of human users.

Finally, as most of the work currently is done in the simulated environment, whether the learned models can be successfully transferred to the physical environment remains a big question. Sim2real transfer has not been well explored in this line of work on embodied AI. Furthermore, since real world environment has more uncertainty and noise, it is also important to study how to build more robust and flexible agents that can handle exceptions and adapt to new situations. While DANLI is able to handle exceptions to some extent, it is still limited by the predefined strategies. Integrating dialog-powered exception handling policies is our next step to extend DANLI.

6.6 Summary of Findings

This chapter introduces DANLI, a deliberative neuro-symbolic agent designed for following natural language instructions in complex, long-horizon embodied tasks. Motivated by the limitations of purely reactive approaches, DANLI integrates neural perception and language understanding with explicit symbolic reasoning and planning. It consists of three key components: a neural subgoal predictor that decomposes instructions into symbolic subgoals, neural

perception models that construct a 3D semantic voxel map for world state representation, and an online PDDL planner that generates low-level action sequences. Evaluated on the challenging TEACh benchmark, DANLI demonstrated significant advancements:

- **Superior Performance:** DANLI substantially outperformed reactive baselines (ET, HET) and oracle-enhanced reactive baselines (HET-ON) in both task success rate and goal completion, particularly on the unseen split, showcasing the benefits of its deliberative approach. It also proved more effective than a procedural baseline (DANLI-PaP), highlighting the value of automated planning over handcrafted rules.
- **Improved Efficiency:** The agent achieved higher path length weighted (PLW) scores, indicating more efficient task completion, with a notable portion of trajectories being more efficient than human references.
- **Generalizability:** DANLI showed strong zero-shot generalization to the related TfD subtask, significantly outperforming baselines trained specifically for it.
- **Interpretability and Diagnosability:** The neuro-symbolic framework enabled detailed analysis of submodule performance and failure modes. Key bottlenecks identified included subgoal prediction errors, failures in finding unobserved objects during search, and interaction/grounding failures not resolved by the recovery mechanisms.

In essence, this chapter demonstrated that integrating explicit symbolic world modeling, subgoal decomposition, and robust planning provides a powerful mechanism for experience grounding. DANLI's success highlights the importance of deliberation and reasoning for enabling embodied agents to effectively understand and execute complex, language-guided tasks, paving the way for more capable and transparent AI collaborators. The identified bottlenecks also highlight critical areas for future improvement in perception, planning, and error recovery.

PART III: Language Grounding in Situated Communication

CHAPTER 7

Proactive Assistant Dialogue Generation from Streaming Egocentric Videos

While the previous parts of this thesis have focused on grounding language in perception (symbol grounding) and embodied planning (experience grounding), we now turn to the third level introduced in Chapter 1: communicative grounding in interactive scenarios. Starting with this chapter, we address the challenge of building proactive AI assistants capable of providing timely, context-aware guidance to a human user performing physical tasks, based on a shared egocentric view from the user's perspective. This problem requires not only understanding the visual scene and user actions, but also inferring user intent, tracking task progress, anticipating future needs, and deciding when and what to communicate in real time. As discussed earlier, the development of such systems is hindered by the lack of suitable training data and scalable evaluation methodologies. To address this, we explore the use of large language models (LLMs) to synthesize situated dialogues from annotated egocentric videos, enabling scalable data curation for training and evaluation. We further propose novel evaluation metrics and an end-to-end modeling approach to tackle the unique challenges of proactive dialogue generation from continuous visual streams. The work presented in this chapter is based on our submission to ACL 2025[1].

[1]Yichi Zhang, Xin Luna Dong, Zhaojiang Lin, Andrea Madotto, Anuj Kumar, Babak Damavandi, Joyce Chai, Seungwhan Moon. "Proactive Assistant Dialogue Generation from Streaming Egocentric Videos." In submission to ACL 2025.

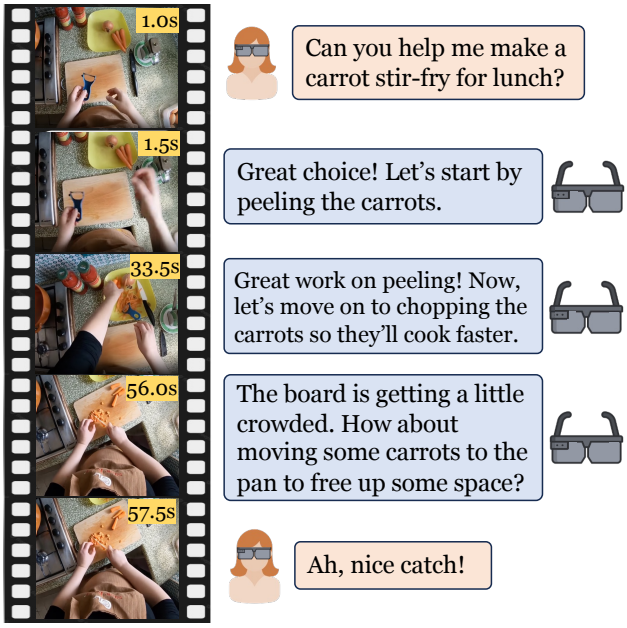


Figure 7.1: An example conversation between user and their task assistant. The assistant receives real-time video streams from the user's perspective, and provides proactive guidance to assist with the task. Images excerpted from Ego4D [90].

7.1 Overview

Recent advances in multimodal language models have transformed various aspects of human-AI interaction [2, 249, 67]. However, developing AI systems capable of providing real-time, interactive guidance for physical tasks remains a significant challenge [21]. Ideally, as illustrated in Figure 7.1, such an assistant should proactively guide users through each step of a task based on a high-level goal, determining both when and how to communicate through continuous processing of the environment and understanding of task objectives. This requires the system to handle streaming video inputs while simultaneously managing diverse user interactions including requests, questions, and comments, and offering timely guidance upon detecting the completion of each task step. The dual challenge of determining both appropriate response timing and content through real-time processing of long-horizon video inputs makes task guidance a particularly complex problem.

Despite significant progress in component technology such as action recognition [50, 90], mistake detection [227, 267, 135], and question answering [273, 107], we are still far from enabling the holistic ability to generate appropriate dialogue responses for task guidance. Two major challenges have hindered progress toward this goal: the lack of large-scale and diverse training data, as existing datasets are primarily constrained by labor-intensive Wizard-of-Oz setups limited to single domains [21, 267]; and the lack of scalable evaluation frameworks for

assistant dialogue generation that could serve as efficient proxies for human evaluation to support rapid and reproducible model comparisons during system development.

To address these limitations, we propose a new problem of proactive assistant dialogue generation from streaming videos and develop comprehensive resources to approach this problem. We introduce an automated approach for synthesizing task-oriented dialogues from well-annotated egocentric video datasets [241, 104, 91, 50, 227], resulting in ProAssist– a large-scale synthetic dialogue dataset containing 30,135 dialogues across 479 hours of video in cooking, object manipulation, assembly, and laboratory domains. Our method leverages state-of-the-art large language models [2, 13, 67] to generate realistic assistant-user interactions, using detailed timestamped video descriptions to maintain temporal alignment. For systematic evaluation, we propose two complementary automatic metrics: a pairwise approach based on sentence matching and an end-to-end approach utilizing LLM-as-a-Judge [310]. Through extensive human studies, we validate both the quality of our synthetic data and the alignment between our proposed metrics and human judgment, establishing ProAssistas a reliable benchmarking resource.

Based on ProAssist, we develop an end-to-end multimodal large language model (MLLM) for generating contextually appropriate responses from streaming video inputs. Building upon the VideoLLM-Online architecture [36], we introduce two key innovations: negative frame sub-sampling to improve response timing decisions, and iterative progress summarization to enable efficient processing of long video sequences. Our experimental results demonstrate the effectiveness of these modeling techniques while providing valuable insights into the complexities of perceptual task guidance.

7.2 Proactive Assistant Dialogue Generation from Streaming Videos

7.2.1 Problem Definition

The goal of a proactive assistant system is to generate prompt, appropriate, and helpful guidance from egocentric video streams in real time. We formulate this problem as a streaming video-to-dialogue generation task. Given a video stream of T frames, our objective is to generate a sequence of assistant responses $s_{1:T}$ that maximizes the conditional probability:

$$s_{1:T} = \arg\max \prod_t P(s_t | v_{1:t}, s_{1:t-1}, k) \tag{7.1}$$

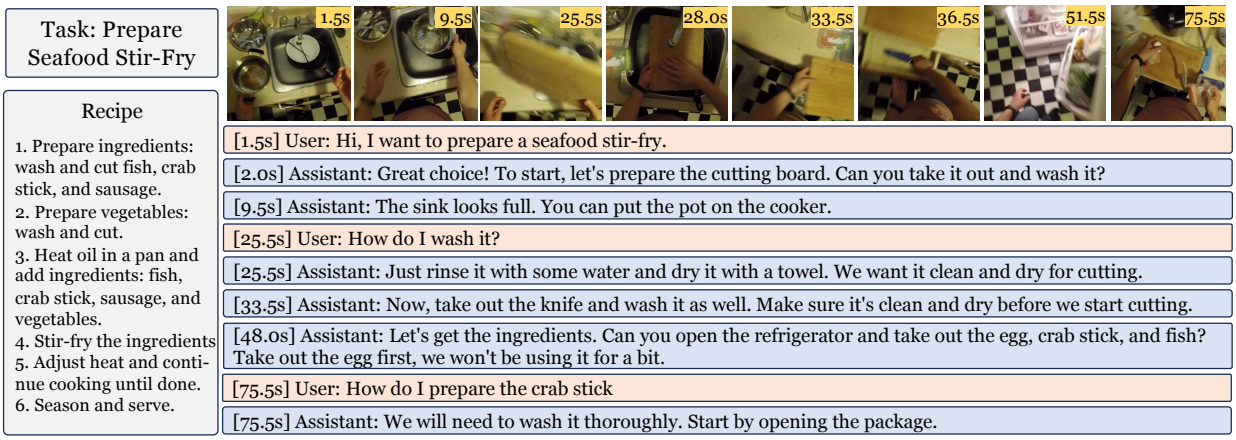


Figure 7.2: A cooking task example from ProAssist, derived from EpicKitchen. The task goal, recpie and dialogue are generated through our synthetic data curation pipeline (dialogue partially shown due to space constraints).

where s_t represents the assistant's response at time step t (either a textual message to user or $\varnothing$ for keeping silence), v_t denotes the multimodal input including the video frame and optional user utterance input, and k represents optional task knowledge (e.g., a recipe). The task begins when the user provides a goal through text input. When k is provided, we term this knowledge-conditioned evaluation, reflecting a realistic retrieval-augmented setup for real-world systems. This formulation requires the assistant to determine both when to speak and what to say based on current visual context, dialogue history, and task understanding.

7.2.2 ProAssist: A Synthetic Dialogue Dataset

We now describe our approach for the creation of ProAssist. We first collect egocentric videos that are extensively annotated with timestamped user action descriptions from six public dataset: Ego4D-Goalstep [90, 241], EpicKitchen [50], HoloAssist [267], Assembly101 [227], EgoExoLearn [104], and WTaG [21]. The annotations are processed into a standardized format, `[t]<description>`, where `[t]` represents a timestamp or time span. Additional annotations such as high-level task step and error correction labels are similarly formatted and inserted in chronological order whenever available. This unified representation enables LLMs to effectively understand ongoing activities at each time step.

Building on these annotations, we design a data curation pipeline consisting of the following steps:

1. Task Goal and Recipe Generation: We first prompt the LLM to summarize the task goal and generate a task recipe based on the video descriptions. This skip will be skipped if the dataset already includes these elements (e.g., WTaG). The generated goal serves as

the initial user input describing the task, while the recipe supports knowledge-conditioned evaluation (§7.5.1).

2. Video Pre-Filtering: Non-procedural, multi-tasking, or incompletely annotated videos are filtered out to ensure dataset quality.

3. Multi-Round Dialogue Generation: Dialogues are generated using three types of user behavior: *no talk* (i.e., silent except for giving the goal), *talk some* (occasional task-related questions), and *talk more* (frequent conversational interactions). Inputs include the goal, video descriptions, and user behavior type. To handle long video descriptions, we adopt a multi-round generation approach, dividing videos into chunks and generating dialogues incrementally to stay within the LLM's context window. Afterward, we prompt the LLM for a refinement pass to improve dialogue naturalness and coherence. At this step, we generate 10 dialogues per video, distributed across user types in a 2:4:4 ratio.

4. Dialogue Annotation: The generated dialogues are then labeled by LLM, including assistant intent (instruction, mistake correction, feedback) and response type (responsive or proactive). Additionally, we also generate a summary of progress at each assistant turn for the user's progress so far, which will be used to support the iterative progress summarization approach (§7.4.3).

5. Automatic Quality Evaluation and Post-Filtering: We perform automatic evaluations to ensure the dialogues meet high standards, assessing timing precision, task step coverage, and assistant responsiveness. Low-quality dialogues are filtered out from the training set. For the validation split, we retain only the highest-scoring dialogue per user type, splitting them evenly into validation and test sets. This process removes approximately 25% of dialogues and 41 hours of video. Final data statistics are shown in Table 7.1.

We leverage LLaMA-3.1-70B-Instruct [67] as the LLM for all the aforementioned steps. An example dialogue generated through this pipeline is shown in Figure 7.2. More details including the prompt for each step, data distributions and dialogue examples are available in the Appendix.

7.3 Evaluation of Proactive Task Assistant

Evaluating interactive dialogue systems is inherently challenging [63], particularly for proactive task guidance where both response timing and content must be assessed. While direct human evaluation through system interaction would be ideal, the high cost makes it impractical for large-scale benchmarking, especially during rapid development cycles. We therefore propose an offline evaluation framework that enables efficient, automatic dataset-based assessment.

Our framework aims to measure the overall helpfulness of a system's sequential predictions

Subset	Video Hour	#Videos	#Dialogues
Ego4D	136.6 / 11.6 / 13.2	382 / 32 / 33	3182 / 96 / 99
HoloAssist	107.0 / 7.6 / 6.8	1436 / 97 / 97	7052 / 291 / 291
EgoExoLearn	68.5 / 8.8 / 9.4	321 / 41 / 41	3210 / 123 / 123
Assembly101	43.1 / 6.9 /7.2	756 / 112 / 112	7492 / 336 / 336
EpicKitchens	34.0 / 4.2 / 4.1	320 / 50 / 50	6376 / 150 / 150
WTaG	7.1 / 1.2 / 1.3	40 / 7 / 7	786 / 21 / 21
Total	478.7	3934	30135

Table 7.1: Data statistics of ProAssistfor train/validation/test splits. More statistics in Appendix E.1.3.

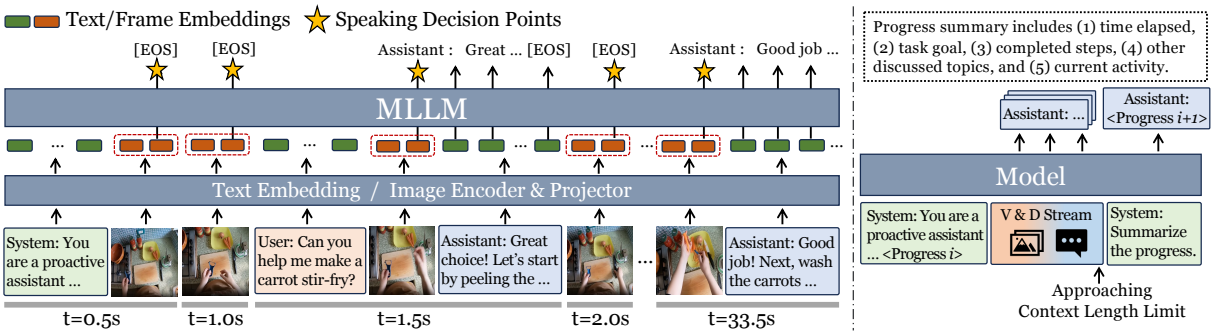


Figure 7.3: **Left**: Streaming video-to-dialogue generation using VideoLLM-Online. The model processes live video frames and optional textual inputs to decide whether to speak or remain silent at designated decision points (yellow stars), and autogressively generates assistant responses as needed. Learning when to speak faces significant class imbalance due to the sparsity of speaking frames. **Right**: Illustration of iterative progress summarization. When approaching its context length limit, the model generates a concise task progress summary, then restarts generation with this summary incorporated into a new system prompt.

$s_{1:T}$ for a video stream $v_{1:T}$ (as defined in Eq.7.1) by comparing them against ground-truth dialogue $\hat{s}_{1:T}$. This comparison is challenging due to the potential long task horizon and the need to evaluate predictions that may differ from ground-truth in both timing and content. Below, we introduce two evaluation metrics designed to address these challenges.

Pairwise Evaluation via Sentence Matching Our first metric evaluates system performance by matching each predicted utterance with semantically similar and temporally aligned reference utterances. Based on these matches, we compute three metrics: *precision* (matched predictions over total predictions), *recall* (matched predictions over total references), and *F1* (their harmonic mean). To identify optimal matches, we apply bipartite matching based on

a cost matrix combining both semantic and temporal alignment costs. The semantic cost between predictions s_i and references $\hat{s}_j$ is defined as:

$$s(i,j) = \begin{cases} 1 & \text{if } \hat{s}_j = \varnothing \\ 1 - sim(e(s_i), e(\hat{s}_j)) & \text{else} \end{cases}$$

where $e(\cdot)$ is the sentence embedding function and $sim(\cdot)$ denotes cosine similarity. The temporal cost encourages matching with temporally proximate messages:

$$d(i,j) = \begin{cases} |i-j|^p & \text{if } i-j \in [-L, R] \\ \infty & \text{else} \end{cases}$$

where p controls the cost increase rate with time difference, while R and L define maximum allowable time differences for predictions preceding or following references. We set $R < L$ to favor earlier predictions, preventing the model from exploiting future frame information. The final matches are computed using the LAPJVsp algorithm [110] with a weighted sum of both costs. More details are provided in the Appendix.

End-to-End Evaluation via LLM-as-a-Judge Our ultimate goal is to evaluate the assistant's overall usefulness to the user. While the pairwise matching approach approximates this through prediction-reference similarity, it cannot capture the flexibility of different guidance strategies. Drawing inspiration from recent LLM-based evaluation approaches [310, 173], we propose using LLMs to directly assess the quality of the overall assistance experience. Given timestamped predictions and reference dialogues, we prompt an LLM to evaluate system performance across four dimensions: *correctness* of guidance and feedback, *appropriateness* of response timing, *efficiency* of information delivery, and *overall* helpfulness. Each aspect is rated on a 5-point Likert scale from "very poor" to "excellent". For reliability, we average scores from three independent runs. The complete evaluation prompt is provided in the Appendix.

Note that our metrics are generally applicable beyond ProAssistto any dataset with ground-truth dialogues. Moreover, the pairwise evaluation can be applied to other streaming video-to-text tasks requiring joint assessment of timing and content, such as online action narration (see §7.5.1).

7.4 Proactive Assistant Dialogue Modeling

Next, we present our exploration into developing a functional proactive assistant dialogue generation model. We begin with an analysis of existing models to assess their feasibility in addressing our problem. Then, we describe how we enhance a baseline model with two novel techniques to enable it to tackle the unique challenges involved.

7.4.1 Feasibility Analysis of Existing Models

Streaming video-to-dialogue generation poses unique modeling challenges to real-time video processing and online text generation. Most existing MLLMs [149, 173, 300, 146, 185, 96, 271, 269] are designed for offline scenarios where the complete video is available beforehand, making them unsuitable for our setup. While state-of-the-art proprietary MLLMs [2, 13, 249] can process interleaved image-text inputs, they suffer from high API latency and often struggle to determine appropriate response timing [36].

VideoLLM-Online [36] offers a viable baseline for our task, as it specifically handles live-streamed video inputs. As shown in Figure 7.3, the model processes interleaved video frames and textual inputs by encoding frames into visual tokens through a frozen pretrained image encoder and a tunable projector layer. For each frame, it predicts whether to respond at the last visual token position, generating `[EOS]` to remain silent or initiating response generation otherwise. To enable real-time interaction, it employs a compact frame representation of 1-10 tokens, significantly fewer than mainstream MLLMs.

However, VideoLLM-Online faces two key limitations in our task guidance scenario: the difficulty of learning when to speak due to the sparsity of speaking moments, and the inability to handle long-horizon tasks due to context window constraints. In the following sections, we present our enhanced model that addresses these challenges through two novel techniques.

7.4.2 Learning When to Speak under Imbalance

Learning when to speak can be framed as a sequence of binary decisions, where at each step the model must choose between speaking and remaining silent. We denote frames requiring responses as positive samples and those requiring silence as negative samples. As shown in Figure 7.3 (left), the speaking decision points (yellow stars) demonstrate a significant imbalance, with far more negative samples (predicting `[EOS]`) than positive ones (predicting `Assistant`). This imbalance creates a challenging learning problem, as directly optimizing cross-entropy on the original distribution leads to a classifier biased toward silence.

We propose Negative Frame Sub-sampling (NFS) to address this challenge. During training,

we compute gradients only for positive frames and a uniformly sampled subset of negative frames, comprising a proportion ρ of total negative samples. The loss remains unchanged for non-decision positions to maintain response generation capability. This approach can be efficiently implemented by adjusting the gradient computation mask without modifying model inputs. Furthermore, dynamically resampling negative samples each epoch ensures all positions can potentially contribute to learning over time, enhancing model robustness.

7.4.3 Iterative Progress Summarization

Processing long-horizon tasks (e.g., hour-long videos) poses significant challenges for both training and inference, as the model must track user goals and task progress over long time period. During training, hardware constraints (e.g., GPU memory) necessitate processing data in fixed-length sequences of length L, forcing truncation of longer samples[2]. This truncation leads to substantial information loss and impairs the model to learn long-horizon task progressions. During inference, context length limitations prevent the model from handling tasks that exceed its training context window.

To address these challenges, we introduce Iterative Progress Summarization (IPS), which enables continuous task tracking through dynamic memory compression. As illustrated in Figure 7.3 (right), when approaching context length limits, we prompt the model to generate a concise progress summary that captures key task-relevant information. Generation then resumes with this summary incorporated into the initial system prompt for the next processing segment. During training, we preprocess long videos into chunks that fit in the context limit, with each segment's progress summary carried forward. Unlike alternative iterative memory methods that require specialized training protocols [265, 43], IPS integrates seamlessly with standard LLM training while enabling our model to process potentially infinite-length video streams without losing track of task goals and progress.

7.5 Experimental Results Analysis

7.5.1 Experiment Setups

Baseline Task. While we primarily evaluate our model on proactive assistant dialogue generation, we also include egocentric action narration as a baseline task, where the model describes the camera wearer's actions in real-time. Action narration serves as a simpler variant of streaming video-to-text generation that mainly requires visual perception capabilities. By

[2]For example, 86% of samples from the Ego4D subset of ProAssistmust be truncated when $L = 4096$ and each frame is encoded as 10 tokens.

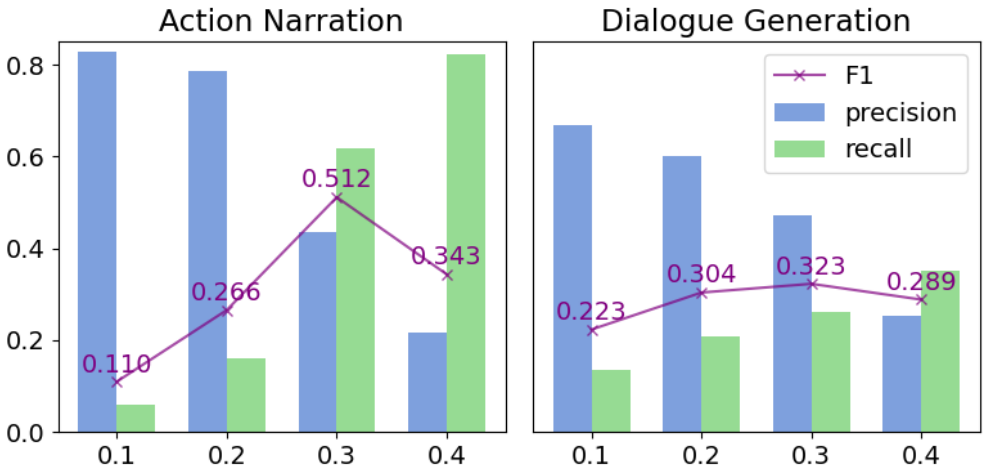


Figure 7.4: Model performance under different speaking decision threshold. The trade-off between precision and recall exists across both tasks.

comparing performance between action narration and dialogue generation, we can better understand how well our model handles capabilities beyond visual perception, such as situational reasoning and progress tracking, which are essential for effective task guidance.

Knowledge-Conditioned Evaluation. We introduce a knowledge-conditioned evaluation setup where the model receives task-specific instructions (e.g., recipes) as a system prompt after the user states their goal. This setup mirrors real-world scenarios where assistants access user-provided recipes or retrieved knowledge to offer guidance.

Speaking Decision Threshold. At inference time, we convert the model's probabilistic token predictions into binary decisions using a threshold θ: the model remains silent if the probability of `[EOS]` exceeds θ. Our experiments show that model performance is highly sensitive to θ, with a clear precision-recall tradeoff (Figure 7.4). We use the θ with the highest validation F1 score [3] for testing.

Model Variants. We implement three variants of VideoLLM-Online, with different number of visual tokens per frame: $I = 1, 5, 10$. The model is intuitively better at visual perception with more tokens, with a cost of computationally more expensive.

Implementation Details. We use LLaMA-3.1-70B-Instruct [67] for LLM-based evaluation. We use LLaMA-3.1-8B-Instruct as the backbone and SigLIP-SO400M-14-384 [297] as the frame encoder for VideoLLM-Online. For training, we adopt a single stage training following [113, 36], on mixed data of dialogues both with and without knowledge from ProAssist, Ego4D online action narration, and some auxiliary vision-language datasets [31, 88, 314]. As a result, we obtain a single model that can be tested with different setup and tasks. More

[3] We opt for F1 over LLM metrics due to the computational high cost of running LLM-based evaluations.

	Correctness	Helpfulness	Alignment	Naturalness
All	3.27 ± 0.79	3.46 ± 0.77	2.91 ± 1.00	3.54 ± 0.70
- No Talk	3.27 ± 0.70	3.47 ± 0.74	2.75 ± 0.96	3.32 ± 0.79
- Talk Some	3.23 ± 0.86	3.37 ± 0.80	2.93 ± 1.01	3.50 ± 0.74
- Talk More	3.32 ± 0.79	3.53 ± 0.76	3.05 ± 0.99	3.80 ± 0.44
HoloAssist-Gen	3.15 ± 0.91	3.40 ± 0.49	2.65 ± 0.91	3.60 ± 0.73
HoloAssist-Human	2.88 ± 1.05	2.62 ± 1.11	2.75 ± 1.09	2.50 ± 1.32
WTaG-Gen	3.50 ± 0.59	3.50 ± 0.92	3.15 ± 1.11	3.65 ± 0.73
WTaG-Human	3.60 ± 0.49	3.60 ± 0.66	3.60 ± 0.66	3.60 ± 0.66

Table 7.2: Human evaluation of the generated dialogue quality. For HoloAssist and WTaG where human-collected dialogues are available, we evaluate them using the same approach for a side-by-side comparison with our generated dialogues.

details in the Appendix.

7.5.2 Dialogue Quality of ProAssist

To validate ProAssistas a reliable resource for studying proactive assistant dialogue generation, we conducted a comprehensive human evaluation of the synthetic dialogues. We uniformly sampled 100 dialogues from the test split across all six data subsets, covering three user types (i.e., *no talk*, *talk some*, and *talk more*). Two annotators evaluated each dialogue along four dimensions using a 4-point Likert scale (1=bad, 2=fair, 3=good, 4=excellent): *correctness* of guidance, *helpfulness* of assistance, *alignment* with video content, and *naturalness* of dialogue (detailed rubrics in Appendix). The evaluation achieved a weighted inter-rater agreement of 81%, indicating strong consensus. As shown in Table 7.2 (top), the synthetic dialogues demonstrate consistently high quality, with average scores exceeding 3 across all dimensions. Notably, dialogue quality correlates with user interaction frequency, with more interactive dialogues scoring higher, particularly in naturalness.

Direct Comparison to Human Dialogues. To contextualize these results, we additionally evaluated human dialogues from HoloAssist and WTaG on the same samples, enabling direct comparison between synthetic and human dialogues. Table 7.2 (middle and bottom) shows that ProAssist's synthetic dialogues match or outperform their human-collected counterparts across multiple dimensions. This advantage is particularly notable in the HoloAssist subset, where our generated dialogues achieve significantly higher scores in helpfulness, correctness and naturalness. Qualitative analysis reveals that human-collected dialogues often contain

Metric	P	S
F1 vs Human	0.35^{**}	0.32^{*}
Overall vs Human	0.47^{**}	0.44^{**}
Overall vs F1	0.67^{**}	0.64^{**}

Table 7.3: Pearson and Spearman coefficient between our metrics and human judgment (*: $p < 0.05$, **: $p < 0.01$).

Metric	A.N.	D.G.
F1	0.80	0.67
Precision	0.53	0.42
Recall	0.47	0.63

Table 7.4: Match rate between human and metric-based selection of the best θ.

Model	Action Narration			Dialogue Generation						
	P	R	F1	P	R	F1	Corr.	Prompt.	Effic.	Overall
I=1	43.61	61.86	51.16	51.26	24.72	32.55	2.15	2.47	2.11	2.11
I=1 (w/ klg)	-	-	-	49.57	28.58	35.43	2.46	2.78	2.31	2.36
I=5	62.81	61.12	61.96	44.41	26.36	32.62	2.13	2.46	2.09	2.10
I=5 (w/ klg)	-	-	-	44.24	31.52	36.25	2.50	2.78	2.34	2.41
I=10	66.17	65.08	65.62	36.97	30.04	32.77	2.19	2.50	2.11	2.15
I=10 (w/ klg)	-	-	-	37.54	34.93	36.07	2.53	2.83	2.31	2.42

Table 7.5: Model evaluation results. Comparisons can be made across different tasks (Action Narration vs. Dialogue Generation), model variants (I=1, 5, or 10), and knowledge access setups (w/ or w/o knowledge). P, R, and F1 are precision, recall, and F1 score, respectively. Corr., Prompt., Effic., and Overall are correctness, promptness, efficiency, and overall score, respectively.

artifacts from Wizard-of-Oz collection setups, where untrained individuals acting as assistants may not maintain consistent professional standards. In contrast, ProAssistdialogues are designed to emulate standardized, professional assistant interactions, resulting in more consistent and helpful guidance. These results validate the effectiveness of our data curation pipeline in producing high-quality synthetic dialogues.

7.5.3 Alignment Between Proposed Metrics and Human Judgment

To measure whether our proposed metrics align with human judgment for model assessment, we conducted two human evaluations.

Correlation with human preference in model ranking. We selected 50 random tasks and collected predictions from three model variants, differing in tokens per frame and access to ground-truth recipes. Annotators ranked these predictions from best to worst (allowing ties), for comparison with rankings from our pairwise F1 score and LLM overall helpfulness score. Table 7.3 shows that both metrics correlate positively with human judgment, with

Model	Action Narration			Dialogue Generation		
	Precision	Recall	F1	Precision	Recall	F1
Baseline	20.5	56.6	30.1	49.6	25.9	32.9
ρ=0.2	56.4	40.6	47.2	48.5	26.7	33.5
ρ=0.1	51.3	68.6	**58.7**	48.0	27.5	**34.4**
ρ=0.01	58.5	52.9	55.5	35.9	33.0	34.2

Table 7.6: Improvement from negative frame sub-sampling under different sub-sampling ratios.

Methods	Precision	Recall	F1
Drop-Middle	30.4	25.9	25.7
IPS (Ours)	49.6	25.9	**32.9**

Table 7.7: Comparison between inference-time context management method for long video processing.

LLM scoring showing stronger alignment. We note that these correlation scores match those of previous automatic dialogue evaluation metrics [290, 299], despite our additional challenge of measuring response timing. These results establish a baseline for developing metrics with better human correlation.

Validation of speaking threshold selection. The speaking threshold θ is a crucial hyperparameter that controls the model's balance between conservative and impulsive talking styles. To validate our use of validation F1 score for selecting θ, we compared human preferences across different thresholds with our metric-based selections. The F1 score demonstrated the highest alignment with human preferences compared to precision and recall, achieving agreement rates of 0.8 for action narration (A.N.) and 0.67 for dialogue generation (D.G.), confirming its effectiveness as a selection criterion.

7.5.4 Result Analysis

We analyze our experimental findings to understand the challenges of proactive task guidance and evaluate the effectiveness of our proposed techniques.

Limited gains from improved perception in dialogue generation. Table 7.5 shows that while increasing tokens per frame (I) substantially improves action narration performance, it provides minimal benefits for dialogue generation. This indicates that effective task guidance requires more than just better visual perception. While the model becomes better

Subset	Correctness	Promptness	Efficiency	Overall
Ego4D	2.07	2.32	2.06	2.02
HoloAssist	2.13	2.55	2.13	2.08
EgoExoLearn	1.90	2.27	1.95	1.93
Assembly101	1.94	2.24	1.98	1.93
EpicKitchens	2.07	2.26	2.04	2.02
WTaG	2.79	3.16	2.51	2.67

Table 7.8: Per-subset performance across domains.

at recognizing user actions, it still needs additional capabilities–such as long-horizon progress tracking, situational reasoning, and knowledge application–to provide meaningful guidance. These results highlight the fundamental challenges in our new problem formulation and emphasize the importance of higher-level reasoning capabilities.

Benefits of task-specific knowledge. Table 7.5 also shows that providing the model with ground-truth knowledge (e.g., recipes) significantly improves guidance quality across all metrics. This improvement yields two key insights. First, access to recipes enables the model to align its guidance strategy with the specific plan shown in the video. This is critical for evaluation with pre-recorded demonstrations where multiple valid solutions exist but only one is shown. Without such knowledge, the model might be penalized for suggesting equally valid alternatives. We therefore recommend the knowledge-conditioned setup as the standard configuration for our evaluation framework. Second, these findings suggest that incorporating retrieval-augmented generation (RAG) to access task-relevant knowledge could substantially improve real-world proactive assistant systems.

Effectiveness of Negative Frame Sub-sampling (NFS). We apply NFS with different sampling ratios ρ. As shown in Table 7.6, training with NFS consistently improves the model's response timing decisions, with higher F1 scores across both tasks. The optimal performance is achieved at $\rho = 0.1$, which we adopt for all subsequent experiments.

Effectiveness of Iterative Progress Summarization (IPS). Direct ablation of IPS is infeasible, as the evaluation cannot complete for videos exceeding the model's training context length. We instead compare against a modified version of StreamingLLM [280]–a context management approach that handles memory constraints by dropping middle tokens while preserving initial task goals. Table 7.7 shows that IPS significantly outperforms this baseline, demonstrating its effectiveness in long-term task progress tracking.

Task familiarity impacts performance. Table 7.8 reveals significant performance variation across ProAssistsubsets. The model performs notably better on WTaG tasks, which contain

only three unique tasks that appear in training (albeit in different environments during evaluation). In contrast, performance drops substantially for EgoExoLearn and Assembly101 tasks, due to relatively less training samples available for laboratory and assembly domains. These results highlight the need to improve generalization to new tasks and domains.

7.6 Discussion of Limitations

Our dialogue synthesis pipeline, while carefully designed, has room for improvement in quality control. As shown in Table 7.2, the alignment between dialogues and video content requires enhancement. Future work could leverage more advanced LLMs, refined prompt engineering, or incorporate multimodal models to increase synthesis quality.

The dataset's reliance on pre-existing video annotations limits its scalability, as such annotations are expensive and time-consuming to obtain. Recent advances in multimodal LLMs [2, 13, 249] open the possibility of generating dialogues directly from raw videos, which could make data synthesis more efficient and scalable.

While our automatic evaluation metrics show promise, their validation is limited to our current experimental setup. These metrics need broader testing across diverse models, performance levels, and related tasks. Additionally, our text-only evaluation approach could be enhanced by incorporating multimodal metrics that consider video content, to establish more robust benchmarks for interactive assistant systems.

Finally, while our proposed proactive assistant model is the first to tackle this challenge, its performance remains suboptimal. As shown in Table 7.8, even on the best-performing domain, the model falls below acceptable thresholds in LLM evaluation (overall scores below 3 out of 5). Notably, it struggles with response timing, dialogue consistency, and delivering detailed guidance that demands fine-grained perception. These limitations underscore the need for better modeling of speaking time, stronger visual-language alignment, and enhanced visual understanding in streaming dialogue generation.

7.7 Summary of Findings

This chapter addressed the challenges of data scarcity and evaluation in developing proactive visual task assistants. We formulated the novel problem of proactive assistant dialogue generation from streaming egocentric videos and made the following contributions:

- **ProAssist Dataset:** We developed an automated pipeline leveraging LLMs to generate ProAssist, a large-scale (30k dialogues, 479 video hours) synthetic dataset of

assistant-user dialogues grounded in diverse egocentric videos (cooking, assembly, etc.). Human evaluations confirmed the high quality and realism of these synthetic dialogues, often matching or exceeding human-collected counterparts from WoZ studies.

- **Scalable Evaluation Metrics:** We proposed two complementary automatic evaluation frameworks: a pairwise metric based on temporally-aware semantic matching (computing Precision, Recall, F1) and an end-to-end LLM-as-a-Judge approach assessing correctness, promptness, efficiency, and overall helpfulness. To the best of our knowledge, this is the first attempt to evaluate proactive dialogue generation that measures both the timing and correctness of the generated dialogue. Human studies validated that these metrics, particularly the LLM-based scores, correlate reasonably well with human judgments of model quality and preference.

- **End-to-End Modeling:** We developed an MLLM based on VideoLLM-Online [36], incorporating two novel techniques: Negative Frame Sub-sampling (NFS) to address the class imbalance inherent in learning when to speak, significantly improving response timing (F1 score); and Iterative Progress Summarization (IPS) to handle long video streams beyond fixed context windows, enabling continuous task tracking.

Key insights from our experiments include the limited benefit of purely improving visual perception (more tokens/frame) for the complex task of dialogue guidance, highlighting the need for better reasoning and progress tracking. Conversely, providing task-specific knowledge (recipes) dramatically improved performance, underscoring the potential of RAG for real-world systems. While ProAssist and our modeling efforts establish a strong foundation, results also indicate remaining challenges in response timing, consistency, and fine-grained guidance, pointing towards necessary future work in advancing real-time communicative grounding.

CHAPTER 8

Bootstrapping Visual Assistant Development with Situated Interaction Simulation

Building upon the work in Chapter 7 which highlighted the critical need for scalable data and evaluation for proactive assistants, this chapter introduces a complementary approach to tackle these bottlenecks through interaction simulation. While Chapter 7 focused on generating dialogue data from existing videos, it still relied on pre-recorded, non-interactive data. Truly capturing the dynamics of communicative grounding requires evaluating and training assistants within interactive loops where assistant guidance influences user actions, and user feedback shapes subsequent assistance. However, collecting such interactive data with real humans is prohibitively expensive and slow. To overcome this, we present a novel framework that simulates both users and assistants powered by MLLMs within 3D environments. This enables scalable generation of synthetic interaction data, automatic evaluation using simulated users, and sim-to-real transfer for real-world applicability—thus bootstrapping the development cycle of capable visual assistants with minimal human supervision. The work presented in this chapter is based on our submission to COLM 2025[1].

8.1 Overview

Visual assistants [188, 222, 260] are an emerging direction of AI systems designed to work alongside humans by understanding their goals, observing their actions, and providing real-time guidance in physical environments. Enabled by wearable devices such as smart glasses, these assistants foster a synergistic interaction in which the AI perceives the world from the user's egocentric perspective, understands their intent and progress, and offers context-aware support through natural language [202]. As illustrated in Figure 8.1, these assistants operate in a unique interaction paradigm where the AI and human share the same physical space

[1]Yichi Zhang, Run Peng, Yinpei Dai, Lingyun Wu, Xuweiyi Chen, Qiaozi Gao, Joyce Chai. "Bootstrapping Visual Assistant Modeling with Situated Interaction Simulation." In submission to COLM 2025.

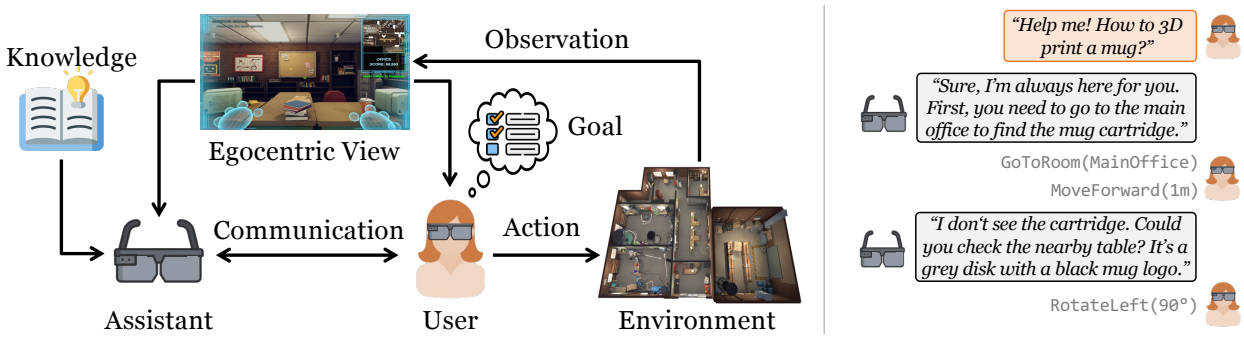


Figure 8.1: **(Left)** Visual assistant interaction framework: The assistant and user share the same egocentric view but have complementary capabilities. The assistant draws on task knowledge to provide guidance through language communication, while the user follows the guidance and physically acts in the environment to achieve a specific goal. **(Right)** Example situated interactions where the assistant proactively offers guidance.

but possess asymmetric knowledge and capabilities. The assistant leverages task-specific knowledge to interpret human actions and communicates actionable guidance, while the human exercises physical agency to complete the task at hand.

Despite significant advances in AI, developing effective visual assistants faces many challenges. The first challenge comes from the lack of diverse interaction data. Traditional approaches rely heavily on wizard-of-oz methods where human experts act as assistants during data collection [21, 267], making the process prohibitively expensive and limiting the variety of communicative dynamics that may occur during the interaction. Furthermore, previous work typically focuses on benchmarking with static video data [104, 90, 227]. It does not address interactive evaluation with real-time user feedback, which is difficult but essential for measuring the effectiveness of the assistant.

To address these challenges, we present BASIS (Bootstrapping Assistant modeling with Situated Interactive Simulation), a novel framework that rethinks the visual assistant development cycle by leveraging simulation to replace labor-intensive human data collection. The core idea is to construct highly-realistic, training-free *simulated users* and *oracle assistants* using powerful multimodal large language models (MLLMs) [2, 249] for role-playing and interaction. BASIS progresses through three interconnected stages: (1) **Situated Interaction Simulation**: in this stage, MLLM-powered oracle assistants interact with simulated users in a virtual environment to generate contextually grounded interaction trajectories. The oracle assistant has access to privileged information (e.g., ground-truth object positions and labels) to enable accurate language guidance generation. In contrast, simulated users are intentionally designed with asymmetrical, incomplete knowledge to mimic real-world user characteristics, such as ambiguous intent and suboptimal behavior, capturing the complexity

of human-assistant interactions. This stage lays the foundation for training assistant models in the next phase. (2) **Autonomous Model Development**: Assistant models are trained using the synthetic interaction data from Stage 1 under more realistic perceptual constraints (i.e., replying on visual observations and no access to privileged information). The assistant's perception is aligned with the user's egocentric view. The simulated users from Stage 1 are also reused to automatically evaluate assistant performance, enabling rapid, low-cost iteration. This stage is crucial for developing and refining assistant models at scale with minimal human intervention. (3) **Real-User Validation**: the refined assistant models are evaluated with real human users to validate that the skills learned in simulation transfer effectively to real-world scenarios. This stage ensures practical usability and grounds simulation-driven development in actual user needs and expectations.

We implement and evaluate BASIS within Alexa Arena, a rich 3D simulator supporting complex object interactions and diverse tasks [76]. Our experiments demonstrate the framework's effectiveness: a visual assistant model trained *entirely* on synthetic data achieves a 72.9% success rate when guiding real human users. This strong performance reaches 82.3% of an oracle assistant's success rate (88.6%) despite operating under realistic perceptual constraints and without access to privileged ground-truth information, demonstrating the efficacy of our sim-to-real transfer approach. Further analysis, including ablation studies and detailed error breakdowns, highlights the strengths of BASIS and offers valuable insights for future research on perceptual assistant systems.

8.2 Problem Definition

We formally define the visual assistance problem as an interactive process between a user and an assistant who share visual observations but possess asymmetric knowledge and capabilities. This interaction occurs in an environment where the user has physical agency while the assistant provides guidance through language. See Figure 8.1 for an illustration.

Formulation. The interaction is defined as a tuple $\mathcal{I} = (\mathcal{E}, \mathcal{O}, \mathcal{A}_u, \mathcal{A}_a, \mathcal{K}_u, \mathcal{K}_a)$, where:

- $\mathcal{E}$ represents the environment state space
- $\mathcal{O}$ denotes the shared egocentric observation space
- $\mathcal{A}_u$ is the user's action space (physical and verbal)
- $\mathcal{A}_a$ is the assistant's action space (verbal guidance only)
- $\mathcal{K}_u$ represents the user's knowledge base

- $\mathcal{K}_a$ represents the assistant's knowledge base

Observation and Actions. At each timestep t, both agents receive the same egocentric observation $o_t \in \mathcal{O}$, consisting of visual input from the user's perspective and the dialogue history. This shared visual context grounds their communication about the environment.

The user can perform two types of actions: physical actions in the environment (navigation, object manipulation) and verbal communication (questions, requests). In contrast, the assistant's actions are limited exclusively to verbal guidance through text, including instructions, explanations, and alerting the user to relevant information.

Knowledge Asymmetry and Interaction Dynamics. The interaction is characterized by complementary knowledge and capabilities. The assistant typically possesses superior task-specific knowledge ($\mathcal{K}_a \supset \mathcal{K}_u$), while the user has direct environmental agency that the assistant lacks. The interaction proceeds through a collaborative cycle where:

- The user observes the environment and makes a decision of speaking or acting
- The environment changes in response to the user's actions
- The assistant processes the updated observation and provides relevant guidance

This fundamental asymmetry necessitates collaboration, where the assistant's guidance enables the user to achieve goals neither could accomplish alone.

8.3 The BASIS Framework

Developing capable visual assistants traditionally requires costly human involvement, such as Wizard-of-Oz data collection and iterative testing with real users [21, 267], which hinders the rapid iteration for system development. BASIS addresses this bottleneck by minimizing human participation, leveraging MLLM-driven simulation within a three-stage pipeline (Figure 8.2), as detailed below.

8.3.1 Stage 1: Situated Interaction Simulation

This stage aims to bootstrap user-assistant interaction data for assistant model training by constructing capable *oracle assistants* and *simulated users*. Real-world data collection is costly and safety-critical. Therefore, we leverage simulation as a flexible and powerful tool, which can provide full controllability and reliable ground-truth information labels, enabling fast and scalable data generation over diverse scenarios.

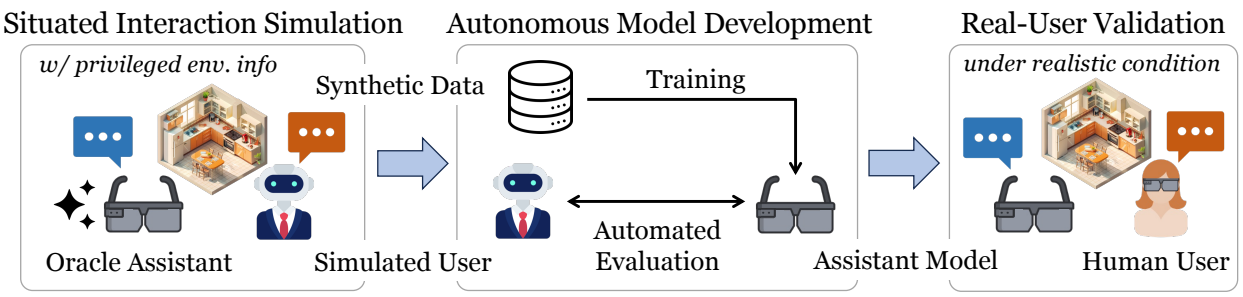


Figure 8.2: BASIS presents a three-stage pipeline for visual assistant development: (1) *Situated Interaction Simulation* generates contextually grounded data through conversational and physical interactions between an oracle assistant and simulated users in a virtual environment; (2) *Autonomous Model Development* leverages this data to train assistant models while using the simulated user for automated model evaluation; and (3) *Real-User Validation* assesses the trained assistants with human users to measure real-world usability.

Oracle Assistant Simulation. We simulate an *oracle assistant* that provides high-quality, near-optimal task guidance. Rather than training new models, we use in-context learning to prompt existing MLLMs for effective behavior. The oracle is granted *privileged access* to ground-truth information from the simulator, including object categories, locations, states, and the causal effects of user actions taken in the environment. This allows the model to reason with perfect information and generate precise, context-aware instructions. Despite relying solely on prompt engineering, the oracle achieves strong performance, as validated through human user studies (§8.5). The resulting textual guidance, paired with visual observations, forms the training data for practical assistant models (§8.3.2), which must learn to operate from visual inputs alone, without access to any privileged information.

User Simulation. Using the same in-context learning technique as the oracle, we construct a *simulated user* with a distinct design tailored for realistic interaction and evaluation. Unlike the oracle, the user model is crafted to exhibit three key properties: (1) responsiveness to assistant guidance, (2) limited task-solving capability, and (3) natural interaction behaviors, such as uncertainty, clarification-seeking, and occasional mistakes. Responsiveness is ensured by prompting the user to follow instructions generated by the validated oracle assistant. To induce reliance on the assistant and simulate realistic communication, we deliberately *restrict* the user's access to environmental information (e.g., providing fuzzy object names like "machine" instead of specific types, hiding object states/locations) and limit its task knowledge. These constraints naturally lead to uncertainty and encourage collaborative problem-solving with assistant. To further enrich behavioral diversity, the user is conditioned on different personas [229, 262], representing varied expertise levels and communication styles.

Beyond data generation, the simulated user also serves as an automatic evaluator in Stage 2, providing a consistent and scalable way to assess assistant model performance.

Synthetic Data Generation. We generate large-scale synthetic datasets by simulating interactions between the oracle assistant and diverse simulated users within controlled environments. Each interaction produces embodied trajectories paired with situated dialogue, capturing rich, multimodal user-assistant exchanges. Simulation offers key advantages over real-world data collection: it allows full control over environmental conditions, enabling systematic coverage of diverse tasks, behaviors, and edge cases that are difficult to capture manually. Additionally, the simulator provides automatic ground-truth annotations for both perceptual events (e.g., object appearances, state transitions) and cognitive elements (e.g., user intent, assistant reasoning steps), resulting in highly informative data for training robust assistant models.

8.3.2 Stage 2: Autonomous Assistant Model Development

The second stage leverages the synthetic data from Stage 1 to develop practical assistant models. Unlike the oracle, these models operate under realistic constraints, relying solely on visual input and interaction history rather than privileged simulator information.

Training from Synthetic Data. We train assistant models using the interaction data generated via situated interaction simulation. The core task is to learn effective guidance strategies based only on visual perception and dialogue history. Training focuses on developing key capabilities: environmental understanding from visual input, proactive decision-making on when and what to communicate, and effective guidance generation using multimodal reasoning. This process functions as imitation learning [106, 45, 68], transferring the oracle's effective decision-making (based on privileged information) to a practical model operating under perceptual constraints.

Automated Evaluation with Simulated Users. A key strength of BASIS is the use of simulated users built in Stage 1 for automated, continuous evaluation during model development. Unlike static benchmarks [90, 227, 104], these users provide interactive feedback in dynamic scenarios, enabling assessment of critical skills such as responding to user input, handling misunderstandings, and completing tasks end-to-end. By automatically tracking metrics like task success, interaction efficiency, and guidance quality across diverse scenarios, we can rapidly iterate on assistant models without constant human testing, significantly accelerating the development cycle.

8.3.3 Stage 3: Real-User Validation

While simulated users provide valuable feedback for iterative development, the ultimate measure of an assistant's effectiveness is its performance with real human users. This stage conducts systematic human evaluations to validate the assistant's ability with real users.

Bridging Simulation and Reality. Our framework primarily focuses on bridging the gap between *simulated* and *real* users, both of whom interact with assistant models in the same simulated environment. The designed evaluation protocol (see §8.5.1) enables systematic assessment of core capabilities such as multimodal perception, reasoning, and communication. While we do not address sim-to-real *environment* transfer in this work, it is relatively straightforward in our setting since the assistant model operates purely based on visual perception, leaving action execution to human users. Therefore, transferring to real environments mainly involves adapting the visual inputs, which can be handled through domain adaptation or real-world fine-tuning [252]. This makes environment transfer largely orthogonal to our core contribution: building a scalable, interaction-centric data pipeline for training and evaluating assistive agents.

8.4 Instantiation of BASIS in Alexa Arena

This section details how we instantiate the BASIS framework using Alexa Arena [76] as our simulation testbed. We first describe the key environmental features of Alexa Arena that enable realistic assistant evaluation, then explain the implementation of each BASIS component, and finally outline our human evaluation setup.

8.4.1 Alexa Arena Environment

Alexa Arena offers a realistic multi-room 3D household environment with interactive objects and diverse task scenarios. The environment includes several key features that makes it an ideal testbed for evaluating our framework.

Rich Object Interactions and Task Complexity. Arena includes over 300 objects, some fantastical, with diverse properties enabling complex state changes and interactions (e.g., using a time machine to repair a broken bowl). This facilitates tasks with intricate dependencies and preconditions (e.g., powering on a computer before use), demanding the assistant to adapt guidance strategy based on the perceived environment. The environment

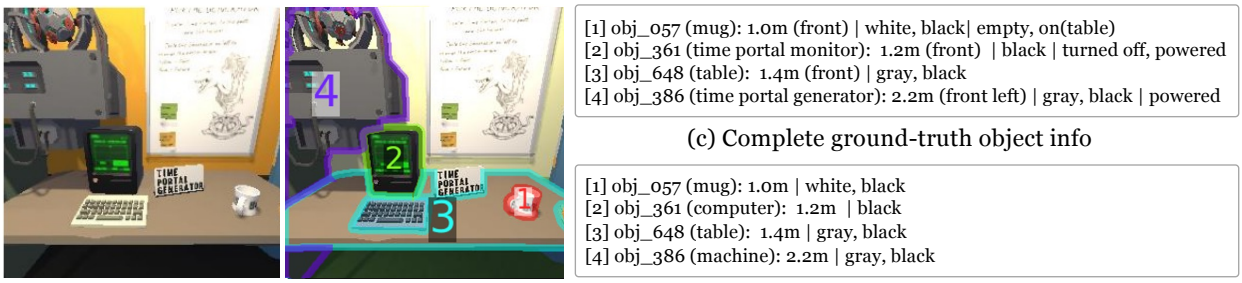


Figure 8.3: Different types of environment observation and object info used in agent simulation. In partial object info, we use the fuzzy name to replace the precise name of uncommon objects (e.g. "machine" for "time portal generator") and hide the object state to simulate missing knowledge in object identification and state estimation.

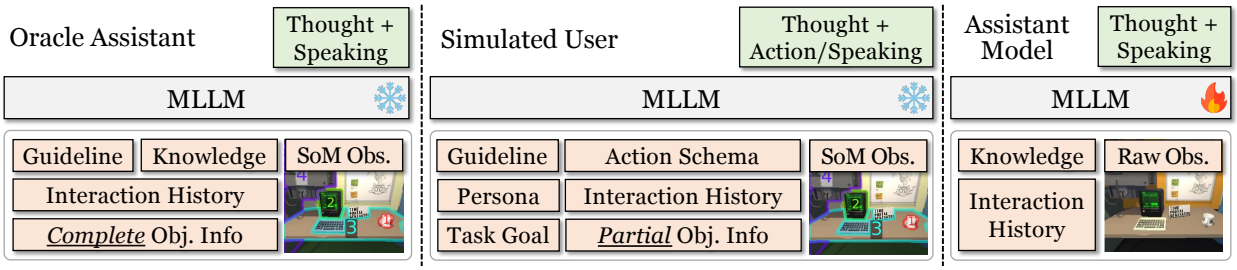


Figure 8.4: Illustration of the oracle assistant (left), the simulated user (middle), and visual assistant modeling (right).

also incorporates elements like irreversible actions and safety considerations (e.g., dangerous objects), allowing for the creation and evaluation of diverse and complex scenarios.

Intuitive Action Space. The agent can navigate through both low-level (e.g., `Rotate`, `MoveForward`) and high-level actions (e.g., `GoToRoom`, `GoToObject`), and perform object-centric interactions (e.g., `PickUp`, `Place`, `Toggle`), abstracting away low-level complexities while supporting necessary interactions. A full list is in Appendix F.2.1.

Environmental Observation. Arena provides RGB images, instance segmentation masks, and symbolic object metadata (ID, type, distance, state, etc.). As illustrated in Figure 8.3, we utilize set-of-mark (SoM) annotations [284] to ground object references visually. We leverage the object metadata to generate ailored descriptions for different roles: the oracle receives complete information (Figure 8.3c), while simulated users receive partial or fuzzy descriptions (Figure 8.3d). This effectively simulates different levels of environmental familiarity and enforcing the knowledge asymmetry crucial to BASIS.

8.4.2 Implementation of BASIS Components

Figure 8.4 provides an overview of the inputs and outputs for the different agents. We will explain each component in the following. More details about prompt and web interface design can be found in the Appendix F and F.4.

Oracle Assistant. The oracle assistant is implemented by prompting MLLMs. The prompt defines its role, objectives, and interaction behaviors, using three specialized templates for proactive guidance, response handling, and error correction. As input, the oracle receives the full interaction history and complete object information from simulator (Figure 8.3c), along with structured knowledge bases covering general task procedures (*task knowledge*), current object states and locations (*environment knowledge*), and object properties (*object knowledge*).

Simulated User. The simulated user is also implemented via a prompted MLLM, with a focus on producing valid, executable actions. We include an action schema in the prompt and use parsers to ensure outputs map correctly to simulator commands. Unlike the oracle, the user is given a task goal but only *partial* object information (Figure 8.3d), mimicking real-world uncertainty and encouraging rich assistant-user collaboration. Personas further modulate user behavior (e.g., help-seeking frequency) to enhance realism and diversity.

Synthetic Data Collection. We generated a synthetic dataset based on 70 unique task variants, derived from 22 base tasks and 10 preconditions, all requiring expert-level guidance (details in Appendix F.2.2). For instance, the task "Heat up a mug using the laser cannon" might necessitate assembling or powering the laser. To enhance diversity, we varied task parameters and randomized scenes across over 7,000 generated scenarios. We sampled 20 scenarios per variant for a balanced dataset of 1,440 scenarios. The final dataset comprises 1,381 successful trajectories obtained through simulated interactions (up to 3 attempts) between an oracle assistant and our best simulated user model.

Visual Assistant Modeling. We fine-tune pre-trained vision-language models (VLMs) on our synthetic dataset to generate guidance using only visual input (RGB images), task knowledge, and interaction history, without privileged simulator information. Beyond guidance, the models are also trained to emulate the reasoning patterns of the oracle assistant, which we found effective in improving guidance quality.See more details in the experiment section (§8.5).

Evaluation Setup. We create a held-out test set of 70 new scenarios (one per unique task condition). On this set, we evaluate different assistant models (oracle or trained models) paired with different users (simulated or real humans). For human evaluations in Stage 3, we developed an interactive web interface (see Appendix F.4.) allowing users to control the Arena avatar and communicate with the assistant through an integrated chat interface.

8.5 Experimental Results Analysis

We conduct extensive experiments to evaluate the effectiveness of our BASIS framework. Our experimental design addresses four key research questions:

- **RQ1: Oracle Effectiveness** – How effective is the oracle assistant in guiding human users? This serves as an upper bound for assistant performance.
- **RQ2: User Simulation Fidelity** – To what extent can we simulate realistic user behavior, and which simulation approach best approximates human interaction patterns?
- **RQ3: Assistant Modeling Performance** – How do different model architectures, reasoning strategies, and training conditions affect assistant modeling performance?
- **RQ4: Real-User Transferability** – How well do assistants trained on synthetic data transfer to real human users, and what are the performance and failure mode gaps between oracle and trained assistants?

8.5.1 Experiment Setups

To systematically investigate our research questions, we design a series of experiments to investigate specific aspects of the user-assistant interaction:

- **Exp1: Oracle Assistant + Human User** – We evaluate our GPT-4o oracle assistant, which has access to privileged information, with real human users. This establishes the performance upper bound achievable assuming perfect perception.
- **Exp2: Oracle Assistant + Simulated User Variants** – We pair the validated oracle assistant with various simulated user models (e.g., GPT-4o, o3-mini) under different perceptual conditions (with/without visual input, precise/fuzzy object names). This setup allows direct comparison among simulated user variants and against real human performance to assess simulation fidelity.

- **Exp3: Assistant Model Variants + Best Simulated User** – Using the highest-fidelity simulated user identified from Exp2, we evaluate multiple assistant models fine-tuned on our synthetic data. We use a pretrained Qwen2.5-VL backbone [19] and compare variants differing in size (3B vs. 7B), reasoning strategy (with/without Chain-of-Thought), knowledge integration (with/without task knowledge), and training data (1,381 synthetic interactions vs. 66 oracle-human interactions from Exp1). Note that real human interaction data is expensive to collect, whereas synthetic interactions is much easier to acquire. We also include a comparison against a strong zero-shot proprietary model (GPT-4o without privileged information) to assess the effectiveness of our assistant.

- **Exp4: Best Assistant Model + Human User** – Finally, we deploy the best-performing assistant model identified in Exp3 for evaluation with real human users. This model operates under realistic perceptual constraints (relying solely on visual input). This setup directly measures the sim-to-real transfer success of our framework and allows analysis of performance gaps and error patterns compared to the oracle baseline.

Evaluation Metrics. We employ complementary metrics to assess both effectiveness and efficiency. For effectiveness, we measure Success Rate (SR) and Subgoal Completion Rate (SGC) at both first attempt (@1) and after three attempts (@3). SR measures the percentage of tasks completed successfully, while SGC captures progress toward task completion even when the full task isn't achieved. For efficiency, we track the average number of steps (#Step) taken per task and average user dialogue turns (#U.Turn). Intuitively, a lower #Step indicates more efficient guidance, while a lower #U.Turn reflects better proactivity from the assistant. We also report the average assistant dialogue turns (#A.Turn) as an auxiliary metric to evaluate the assistant's communication efficiency.

8.5.2 Result Analysis

Oracle Effectiveness (RQ1). Our oracle assistant demonstrates strong effectiveness in guiding humans through complex tasks. As shown in Table 8.1, the GPT-4o-based oracle achieves an 88.6% success rate (SR@1) and a 93.0% subgoal completion rate (SGC@1) with human users. This high performance validates the oracle as both a reliable source for data generation and a concrete upper bound on what can be achieved with perfect perception.

User	Vision	Fuzzy Obj.	SR@1	SGC@1	SR@3	SGC@3	#Step	#U.Turn
Human	✓	✓	88.6%	93.0%	-	-	24.7	3.1
†GPT-4o	✓	✓	77.1%	83.2%	94.3%	94.6%	25.9	2.9
GPT-4o	✓	✗	82.9%	88.1%	100.0%	100.0%	22.1	2.2
GPT-4o	✗	✗	81.4%	89.2%	98.6%	98.9%	22.1	2.1
o3-mini	✗	✗	65.7%	69.7%	84.3%	87.0%	28.4	8.3
Gemini2F	✓	✗	68.6%	77.8%	94.3%	95.1%	21.1	1.8

Table 8.1: Performance comparison of different user variants paired with our oracle assistant, under different setups: with/without vision access (✓/✗) and precise/fuzzy naming (✓/✗) of uncommon objects. † indicates the simulated user selected for the following stages.

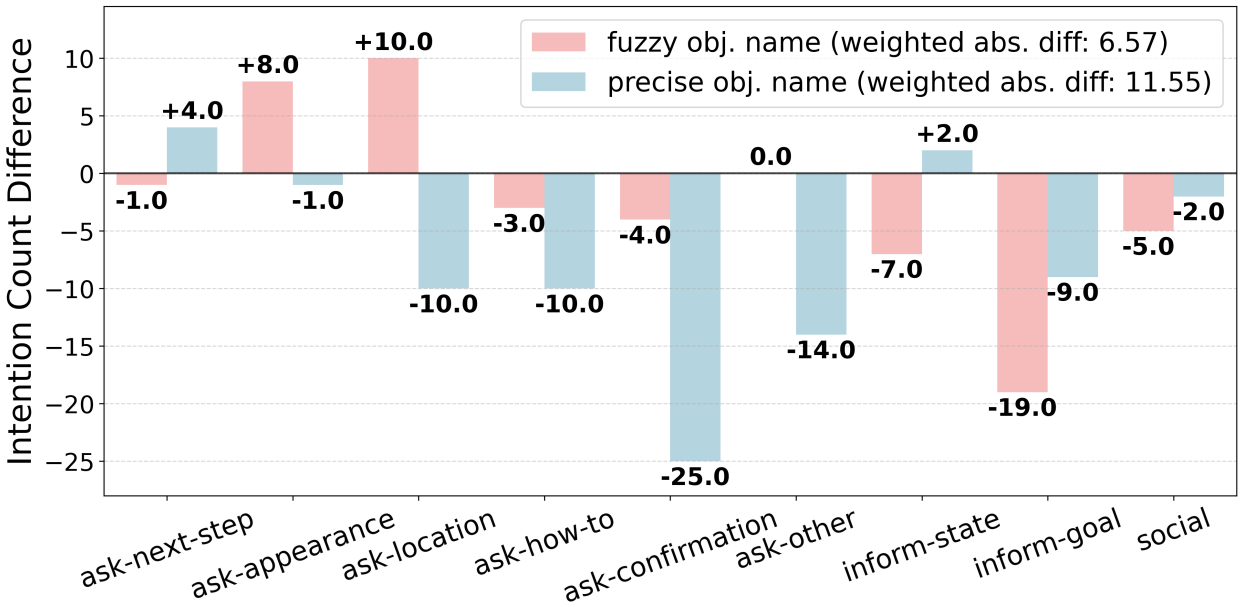


Figure 8.5: Intention differences between GPT-4o simulated users and real humans, comparing fuzzy vs. precise object naming. Positive/negative values indicate more/fewer instances than humans. Fuzzy naming better aligns with overall human behaviors (lower absolute differences) and better captures human uncertainty (more asking-related intentions).

User Simulation Fidelity (RQ2). To evaluate how perception impacts user simulation fidelity, we compare settings with and without vision inputs. We also compare precise vs. fuzzy object naming, where the former can improve object identification during task execution and the latter better reflects the uncertainty real users face when describing unfamiliar items. As shown in Table 8.1, GPT-4o consistently outperforms other models, achieving the highest success rate and fewest dialogue turns. In contrast, reasoning-focused models like o3-mini show significantly lower success (65.7%) and more user turns (8.3), while Gemini 2.0 Flash lags

Model	GT Obj.	SR@1	SGC@1	SR@3	SGC@3	#Step	#U.Turn	#A.Turn
Fine-Tuned								
3B	✗	60.0%	67.0%	75.7%	80.0%	29.8	4.4	16.0
3B CoT	✗	60.0%	67.6%	81.4%	85.9%	29.6	4.5	18.2
7B	✗	60.0%	68.1%	81.4%	83.2%	27.4	4.1	13.3
†7B CoT	✗	68.6%	73.5%	84.3%	85.9%	28.6	3.3	16.4
7B CoT (no knowledge)	✗	47.1%	58.4%	67.1%	69.2%	33.6	3.9	19.6
7B CoT (real data)	✗	40.0%	51.4%	67.1%	69.2%	31.1	5.2	17.8
7B CoT (w/ GT obj. info)	✓	70.0%	79.5%	92.9%	93.0%	27.8	3.5	17.0
Prompting								
GPT-4o CoT (img-only)	✗	50.0%	61.1%	68.6%	73.5%	35.2	4.2	20.6
GPT-4o CoT (oracle)	✓	77.1%	83.2%	94.3%	94.6%	25.9	2.9	14.7

Table 8.2: Performance comparison of different assistant models with our simulated user. All models are fine-tuned from Qwen2.5-VL 3B/7B. "GT Obj." denotes using ground-truth object information. † indicates the best assistant model selected for Stage 3.

across all metrics. Claude 3.5 failed to produce valid outputs and was excluded. Visual input further boosts GPT-4o's performance, confirming the importance of multimodal grounding.

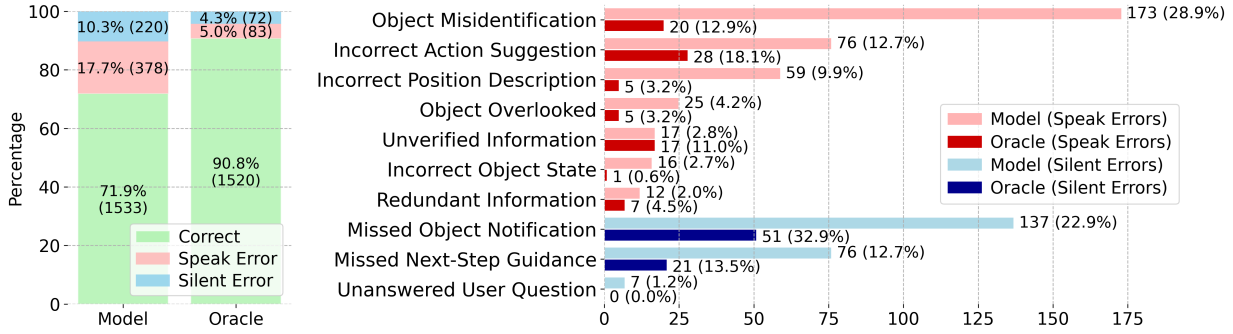
While accessing precise object names improves the success rate, it sacrifices the natural uncertainty that users exhibit when recognizing objects. As shown in Figure 8.5, using precise object names leads to less asking-relevant intentions in user dialogues, indicating a lack of human-like uncertainty. To balance realism and effectiveness, we adopt GPT-4o with vision access and fuzzy object naming as our default simulated user.

Assistant Modeling Performance (RQ3). Our analysis highlights key factors in visual assistant modeling (Table 8.2). Incorporating Chain-of-Thought (CoT) reasoning significantly boosts both success and efficiency across model sizes. Scaling from 3B to 7B improves SR@1 from 60.0% to 68.6% while reducing user dialogue turns. Removing task knowledge from model input leads to a major drop in performance (SR@1 from 68.6% to 47.1%), showing its importance even after fine-tuning. The same 7B CoT model trained on our synthetic data significantly outperforms its counterpart trained with real human-oracle interaction data, thanks to our scalable nature of our data collection framework. Our fine-tuned models also outperform zero-shot GPT-4o without ground-truth object information (68.6% vs. 50.0% SR@1), validating the effectiveness of our synthetic training. A variant with access to the same privileged information approaches oracle-level SR@3 and SGC@3, revealing the potential of small MLLMs under improved perception.

Real-User Transferability (RQ4) and More Error Analysis. As shown in Table 8.3, our best assistant model achieves a 72.9% success rate with human users, reaching

Assistant	Perception	SR@1	SGC@1	#Step	#U.Turn	#A.Turn
Oracle (GPT-4o)	GT Obj. Info	88.6%	93.0%	24.7	3.1	13.6
Model (Qwen2.5-VL 7B CoT)	RGB Images	72.9%	78.4%	28.3	4.3	15.0

Table 8.3: In Stage 3 human evaluations, our best-performing assistant, trained entirely on synthetic data and without access to privileged information, achieves 82.3% of the oracle assistant's performance in SR@1, demonstrating strong generalization to real users.



(a) Step-level error rate

(b) Fine-grained error type distribution

Figure 8.6: Stepwise error analysis comparing the assistant model versus the oracle assistant during interactions with real human users. Speak errors occur when the assistant provides guidance containing inaccuracies or mistakes, while silent errors indicate instances when the assistant fails to provide guidance despite it being needed.

82.3% of oracle-level performance (88.6%) despite relying only on RGB images, in contrast to the oracle's access to ground-truth object information. This result supports the core hypothesis of BASIS: high-fidelity user simulation enables training data that generalizes well to real human interactions. The perception gap is most evident in efficiency metrics, with our model requiring more interaction steps (28.3 vs. 24.7) and user dialogue turns (4.3 vs. 3.1), reflecting less optimal guidance. To understand this gap, we conducted a fine-grained analysis of error patterns by manually examining all interactions between both assistants and human users. As shown in Figure 8.6a, our assistant model makes errors in 28.1% of interactions—approximately three times higher than the oracle's 9.2% error rate. Figure 8.6b shows that object misidentification accounts for the most common speaking error (28.9% of assistant responses), highlighting challenges in grounding object references. These stem not only from novel object appearances in Arena simulator, but also from ambiguities in task descriptions, dialogue context, and the presence of multiple objects with similar types. For silent errors (where the assistant does not speak when guidance is needed), both missed object notifications and next-step guidance occur significantly more frequently than with

the oracle, suggesting that perceptual limitations lead to incorrect decisions about providing timely and proactive assistance. This detailed error analysis provides valuable insights for future research on improving the robustness of visual assistants in real-world scenarios.

8.6 Discussion of Limitations

While BASIS demonstrates promising results, several limitations remain. First, this work focuses on transferring from simulated to real *users*, with both operating in a simulated environment; we do not address sim-to-real *environment* transfer, which may involve domain gaps in visual input or interaction dynamics. Second, due to computational constraints, we limit our experiments to 3B and 7B vision-language models; larger models could offer improved reasoning and guidance capabilities. Third, our experiments are conducted within Alexa Arena, which, while rich and flexible, may not fully reflect the complexity and unpredictability of real-world daily tasks. Future work should explore more realistic task environments and physical deployment. Additionally, our simulated users, although designed to mimic human uncertainty and variability, may not fully capture the diversity and nuance of real human behavior, especially in terms of intent ambiguity, non-verbal cues, or long-term planning. Finally, while our assistant models exhibit strong performance, they currently lack mechanisms for real-time user adaptation or learning from interaction history across sessions, which is an important direction for building personalized, context-aware visual assistants.

8.7 Summary of Findings

This chapter introduced BASIS, a simulation-driven framework designed to overcome the data and evaluation bottlenecks in developing interactive visual assistants—addressing a central challenge in achieving robust communicative grounding. Instead of relying on costly human-collected data, BASIS leverages MLLM-powered agents within a simulated 3D environment (Alexa Arena) to generate rich, grounded interaction data, train visual assistants under realistic perceptual constraints, and evaluate their performance both in simulation and with real users.

Our key findings validate the BASIS approach:

- **Effective Sim-to-Real Transfer:** An assistant model (Qwen2.5-VL 7B CoT) trained entirely on synthetic data from BASIS achieved a 72.9% success rate with real human users, reaching 82.3% of the performance of an oracle assistant (88.6%) that used privileged information. This demonstrates that high-fidelity simulation can effectively bootstrap capable assistants.

- **User Simulation Fidelity:** Comparisons between different simulated user configurations and real human behavior revealed that GPT-4o simulating a user with visual input and fuzzy object naming best approximated human interaction patterns (dialogue turns) and uncertainty (asking intentions), validating our design choices for realistic simulation.
- **Scalable Development:** The framework enables rapid iteration and evaluation of different assistant model architectures using the simulated user. By comparing the performance of different model variants, we provide insight into the contribution of different design choices for the assistant model, such as model size, reasoning strategies (CoT), knowledge integration.
- **Bottleneck Identification:** Fine-grained error analysis during real-user validation pinpointed object misidentification (due to relying solely on visual input) as the primary performance gap between the trained model and the oracle, highlighting perception as a key area for future improvement.

In conclusion, BASIS provides a scalable and effective methodology for bootstrapping the development and evaluation of visual assistants capable of situated, interactive task guidance. By successfully bridging the gap between simulation and real-user performance in this context, this work offers a viable path forward for creating context-aware AI assistants that can effectively engage in communicative grounding to collaborate with humans in the physical world.

CHAPTER 9

Conclusion

This dissertation has explored the multifaceted challenge of situated language grounding, a cornerstone capability for developing multimodal AI assistants that can perceive, reason about, and interact effectively within the physical world alongside humans. The central goal has been to move beyond purely text-based understanding or simple object labeling towards systems that deeply connect language to rich, dynamic, multimodal contexts encompassing visual perception, embodied action, and interactive communication. While significant strides have been made in large language and multimodal models, achieving robust situated grounding remains a critical bottleneck, hindering the development of truly helpful and context-aware AI assistants. Addressing this, the research presented here investigated grounding across several key dimensions, developing novel approaches, datasets, and frameworks to advance the field.

Our journey began in Part I by tackling the foundational layer of symbol grounding—connecting language to precise visual referents under both standard and challenging perceptual conditions. Recognizing the limitations of prevalent bounding-box-based approaches for fine-grained understanding, Chapter 3 introduced GROUNDHOG, an MLLM designed for holistic visual grounding using segmentation masks. After being trained on the curated M3G2 dataset, GROUNDHOG demonstrated the ability to achieve superior grounding precision across diverse semantic granularities (objects, stuff, parts, text) and various tasks (GIC, RES, GVQA, RD) within a unified model, establishing the benefits of pixel-level grounding for reducing object hallucination and enhancing interpretability.

Complementing this focus on fine-grained accuracy, Chapter 4 addressed the crucial challenge of perceptual robustness by introducing GVIL, the first benchmark designed to systematically evaluate MLLM language grounding under visual illusions. By comparing model behavior against human perceptual norms on illusory stimuli, GVIL provided critical insights into the similarities and divergences between human and machine visual understanding, probing the limits of current models when perception deviates from physical reality and

providing a more robust perceptual foundation for subsequent grounding challenges.

Moving from static perception to dynamic actions, Part II focused on grounding language in embodied planning and execution within complex 3D environments. We explored two complementary neural-symbolic approaches. First, Chapter 5 introduced HiTUT, which tackled complex, multi-step tasks through hierarchical task learning. HiTUT mployed a unified Transformer architecture to decompose problems into sub-goals, navigation, and manipulation, learning task structure and action policies jointly. This hierarchical approach facilitated self-monitoring and backtracking, leading to improved generalization and success rates on the ALFRED benchmark, especially in unfamiliar settings.

Complementing this learned predictive policy, Chapter 6 presented DANLI, a deliberative agent emphasizing explicit world modeling and symbolic planning. DANLI constructs detailed semantic world representations, leverages LLMs to derive sub-goals from high-level instructions, and utilizes a classical planner for robust task execution and recovery. Evaluated on the demanding TEACh benchmark, DANLI demonstrated significant improvements in effectiveness and efficiency over reactive methods, highlighting the value of explicit reasoning and planning for complex, long-horizon tasks and providing interpretable decision processes. Together, HiTUT and DANLI showcase synergistic strategies for integrating language understanding with planning for effective embodied interaction.

Finally, Part III addressed the complexities of communicative grounding in dynamic, interactive scenarios, focusing on proactive assistance from a user's egocentric perspective. Recognizing the critical bottlenecks of data scarcity and evaluation for real-time interaction, this part presented two complementary contributions. Chapter 7 introduced ProAssist, a large-scale synthetic dialogue dataset generated from existing egocentric videos, along with novel evaluation metrics and an end-to-end streaming MLLM designed to learn when and what to communicate based on visually observed user activities. This work provided crucial resources and methods for studying proactive guidance.

From another perspective, Chapter 8 introduced the BASIS framework, a novel simulation-driven pipeline using MLLM-powered simulated users and oracle assistants within an interactive 3D environment. BASIS enables scalable generation of situated interaction data and automated evaluation, effectively bootstrapping the development of visual assistants. Experiments demonstrated successful sim-to-real transfer, with models trained purely on synthetic data achieving strong performance guiding real human users, further validating the approach and pinpointing perception as a key remaining challenge.

9.1 Broader Implications

Together, these contributions paint a comprehensive picture of situated language grounding, demonstrating concrete advancements across its perceptual, experiential, and communicative dimensions. The progression from grounding in vision (GROUNDHOG, GVIL), to grounding in embodied planning (HiTUT, DANLI), and finally to grounding in situated communication (PROASSIST, BASIS) illustrates a path towards increasingly sophisticated and capable AI assistants. This research offers several broader implications:

- **Enhanced AI Capabilities:** It provides methodologies for building AI assistant systems that can better understand multimodal context, follow complex instructions, and offer proactive, helpful guidance in physical tasks, benefiting applications in AR/VR, robotics, and personal assistance.
- **Scalable Development:** The introduction of large-scale synthetic data generation (PROASSIST) and simulation-based frameworks (BASIS) offers pathways to mitigate the data bottleneck and accelerate the development and evaluation cycle for next-generation perceptual, context-aware, and communicative AI assistants.
- **Improved Trust and Transparency:** The emphasis on interpretable grounding models (GROUNDHOG), understanding model perceptual limitations (GVIL), explicit reasoning (DANLI), and potentially hierarchical decomposition (HiTUT) contributes towards building more trustworthy AI systems whose decision-making processes can be better understood and diagnosed.

9.2 Limitations and Future Directions

Despite the progress made, this dissertation also highlights several limitations and opens avenues for future research:

- **Unified Grounding Architectures:** While we explored different facets of grounding, integrating symbol, experience, and communicative grounding seamlessly within a single, unified architecture remains an open challenge. Future work should focus on developing architectures that jointly optimize for precise perception, robust planning, and fluid interaction.
- **Real-World Deployment and Robustness:** Significant hurdles remain in transitioning these systems from controlled simulations and datasets to the dynamic and unpredictable real world. Bridging the sim-to-real gap is crucial for ensuring skills

learned in simulation transfer effectively to physical environments. Robustness must be enhanced to handle noisy perception, identify objects reliably beyond closed sets, track state under occlusion, and manage perceptual ambiguities like those highlighted by GVIL. Achieving real-time processing is essential for seamless interaction, particularly for computationally demanding models operating on continuous video streams, a key requirement for potential always-on visual assistants.

- **Open-World Adaptability:** Current systems often operate within relatively constrained domains. Future research must enable agents to learn and adapt dynamically to new objects, actions, environments, and task structures, potentially leveraging large models for open-world knowledge acquisition and continuous, lifelong learning.

- **Long-Term Interaction, Memory, and Safety:** The current work primarily examines task-specific interactions. Future directions must include developing robust long-term memory mechanisms, allowing assistants to retain context, user preferences, and environmental knowledge across extended periods for personalization and improved assistance. Evaluating long-term user adaptation, trust dynamics, and mitigating risks associated with user over-reliance are critical. Furthermore, ensuring the safety, privacy, and ethical behavior of proactive assistants, especially always-on systems that continuously process sensory input, requires careful consideration and robust safeguards.

- **Scalability and Fidelity of Synthesis/Simulation:** While promising, the quality, diversity, and realism of synthetic data generation and simulation environments, including agent behaviors, can be further advanced. Enhancements could involve incorporating more sophisticated user models, richer environmental interactions, or better leveraging web-scale multimodal data.

9.3 Concluding Remarks

Situated language grounding is fundamental to creating AI systems that can truly understand and collaborate with humans in the physical world. By developing novel models, datasets, benchmarks, and frameworks for grounding language in fine-grained visual details, under challenging perception, through embodied action planning, and via proactive communication, this dissertation contributes significantly to this goal. The findings highlight the importance of moving beyond reactive, text-centric paradigms towards deliberative, multimodal, and interactive approaches that account for both precise grounding and perceptual robustness. While challenges remain, the research presented here provides valuable insights and tools,

paving the way for the development of the next generation of helpful, context-aware multimodal AI assistants.

APPENDIX A

Appendix for Chapter 3

A.1 Mask2Former+ Implementation Details

Our enhancement of the original Mask2Former model focuses on broadening its segmentation capabilities beyond the 134 common object categories it currently handles, which include 80 things and 55 stuffs as defined in the COCO dataset. The primary goal is to enable the model to recognize an expanded range of object categories, as well as segmentation masks of various levels of granularities, such as semantic parts and visual text regions.

Training data. We have compiled a comprehensive dataset by combining multiple existing segmentation datasets. This ensemble encompasses a wide spectrum of entities (things and stuff), their semantic parts, and visual text, drawn from sources such as COCO [26], LVIS [93], Entity-v2 [208], Pascal [58], PACO [214], MHP-v2 [141], and TextOCR [238]. The resulting dataset comprises over 200K images and 4.5M masks, as summarized in Table A.1. Notably, the annotations from COCO, LVIS, and PACO are based on a shared set of COCO images. We merged these annotations to ensure comprehensive mask proposal coverage, thereby providing holistic instance coverage within each image, as can be illustrated in Figure A.1.

Model. Building on the foundation of the original Mask2Former [42], we developed Mask2-Former+, a panoptic segmentation model designed for multi-grained segmentation. We initialize our model from the Mask2Former checkpoint with the Swin-L backbone pre-trained on the COCO panoptic segmentation dataset [122]. Besides the 200 entity queries that are trained for thing and stuff proposals, we added 50 additional expert queries for the segmenting parts and the visual text regions, respectively. Given that not all images have annotations for every type of segmentation (for instance, the TextOCR dataset provides annotations only for visual text regions), our model computes the group-wise matching loss exclusively for the annotations available in each dataset. This approach ensures that the model benefits from

Dataset		Granularity			Dataset Size	
Name	Split	Entity	Part	Text	#Image	#Masks
LVIS [93] & PACO [214]	part	✓	✓		15,089	596,687
	no_part	✓			103,178	2,062,536
Entity-v2 [208]	cls	✓			31,913	579,076
Pascal [70]	train	✓	✓		4,998	93,322
	val	✓	✓		5,105	95,462
MHP [141]	train		✓		15,403	410,113
TextOCR [238]	train			✓	21,749	714,770
Total		✓	✓	✓	197,435	4,551,966

Table A.1: Summary of the training datasets for Mask2Former+. Entity includes both thing and stuff categories.

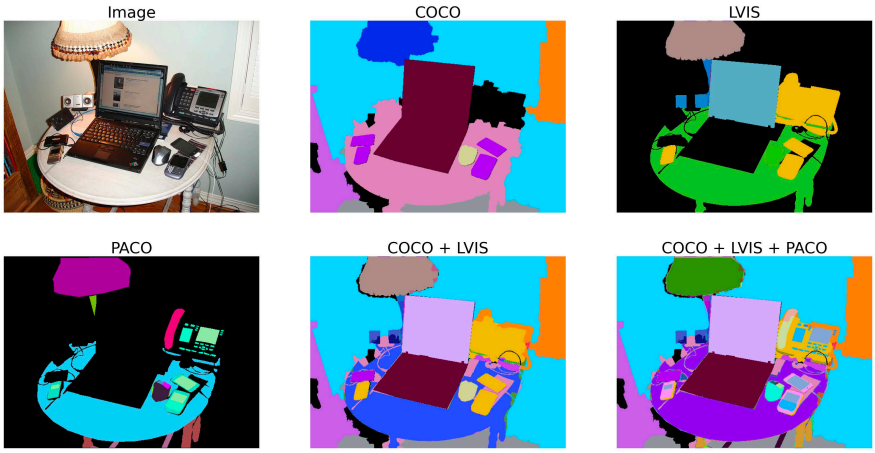


Figure A.1: Illustrations of the merged segmentation annotations from COCO Panoptic, LVIS, and PACO datasets.

partial annotations without compromising its ability to recognize other levels of granularity when certain annotations are unavailable. Although most samples in our dataset also have semantic annotations such as object categories, we do not use them but only train the model for class-agnostic mask proposals. We train the model for 20k iterations on our combined segmentation dataset with a batch size of 16 using the Detectron2 library [278].[1]

A.2 The M3G2 Dataset

In this section, we provide additional details for the M3G2 dataset. M3G2 is a comprehensive dataset consisting of 36 sub-problems, derived and augmented from 27 existing datasets with

[1] https://github.com/facebookresearch/detectron2

Metadata			Grounding Annotations			Semantic Granularity					Data Size	
Task Type	Dataset Name	Image Source	Mask	Box	Pointer	Thing	Stuff	Part	Multi.	Text	Train	Val / Test
Grd. Captioning (GCAP)	PNG	COCO	✓	✓		✓	✓		✓		132,045	8,435
	Flickr30K-Entity	Flickr30K		✓		✓	✓	✓	✓		148,915	1,000 / 1,000
Referential Expression Segmentation (RES)	RefCOCO	COCO	✓	✓		✓					113,311	-
	RefCOCO+	COCO	✓	✓		✓					112,441	-
	RefCOCOg	COCO	✓	✓		✓					80,322	-
	RefCLEF	ImageCLEF	✓	✓		✓					104,531	-
	gRefCOCO	COCO	✓	✓		✓					194,233	-
	PhraseCut	VG	✓	✓		✓	✓	✓	✓		84,688	-
	D-Cube	GRD	✓	✓		✓			✓		9,499	-
	ReasonSeg	OpenImages & ScanNetV2	✓	✓		✓	✓	✓	✓		1,315	344
	RIO	COCO	✓	✓		✓			✓		27,696	34,170
	SK-VG	VCR		✓		✓					23,404	-
Grounded Visual Question Answering (GVQA)	VizWiz-Grounding	VizWiz	✓	✓		✓	✓			✓	6,494	1,131 / 2,373
	TextVQA-X	OpenImages	✓	✓						✓	14,476	3,620
	GQA	VG		✓		✓	✓	✓	✓		301,623	-
	VQS	COCO		✓		✓					20,380	8,203
	Shikra-BinaryQA	Flickr30K		✓		✓	✓	✓	✓		4,044	1,159
	EntityCount	Entity-v2	✓	✓		✓	✓	✓	✓		11,088	453
	FoodSeg-QA	Recipe1M	✓	✓		✓			✓		7,114	-
	LVIS-QA	COCO	✓	✓		✓	✓		✓		94,860	3,611
Referential Dialog (RD)	RefCOCO-REG	COCO	✓	✓	✓	✓					17,395	-
	RefCOCO+-REG	COCO	✓	✓	✓	✓					17,383	-
	RefCOCOg-REG	COCO	✓	✓	✓	✓					22,057	-
	gRefCOCO-REG	COCO	✓	✓	✓	✓					20,282	-
	VG-SpotCap	VG		✓	✓	✓	✓	✓	✓		247,381	232,935
	V7W	COCO		✓	✓	✓					22,805	10,193 / 57,265
	PointQA-Local	VG			✓	✓					27,426	4,855 / 4,880
	PointQA-Twice	VG			✓	✓					36,762	14,668 / 5,710
	VCR-Open	VCR		✓	✓	✓					58,340	-
	VCR-Multichoice	VCR		✓	✓	✓					97,648	26,534 / 25,263
	ShikraRD	Flickr30K		✓	✓	✓	✓	✓	✓		1,878	-
	SVIT-RD	VG		✓	✓	✓	✓	✓	✓		32,571	-
	Guesswhat-Guesser	COCO	✓	✓	✓	✓					92,136	19,665
	Guesswhat-Oracle	COCO	✓	✓	✓	✓					101,256	21,643
	VG-RefMatch	VG		✓	✓	✓	✓	✓	✓		247,381	-
	HierText	OpenImages	✓	✓	✓					✓	6,058	3,885

Table A.2: The full list of datasets used in M3G2.

Instruction Templates for Brief Captioning	Instruction Templates for Detailed Captioning
Describe the image briefly.	Describe the image in detail.
Describe the image in a few words.	Describe the picture's every detail.
Describe the image in a short sentence.	Describe the given picture in very detail.
Describe the image in a clear and concise manner.	Make a fine description of the image.
Generate a short caption for the picture.	Generate a long caption for the given image.
Caption the image in a few words.	Give me a detailed caption of this image.

Table A.3: Instruction templates for the GIC task.

grounded vision-language annotations. The dataset is categorized into four main types: (1) Grounded Image Captioning (GIC), (2) Grounded Visual Question Answering (GVQA), (3) Referential Expression Segmentation (RES), and (4) Referential Dialog (RD). Details on the dataset sources, image origins, types of grounding annotations, semantic granularity, and data statistics are summarized in Table A.2. All datasets are formatted into the conversation format between a human user and a model assistant, where the user provides task objectives as instructions, and model responses are generated automatically based on the annotations.

Instruction Templates For RES	
Highlight "{}" in the image.	Segment "{}" in the image.
Segment: {}.	Help me segment out {}.
Localize "{}" in the image.	Help me localize {}.
Help me highlight the region of {}.	Demonstrate where "{}" is located in this image.
Show me where to find in this photo.	Identify and mark the region of {} for me.
Can you highlight "{}"?	Can you extract the segment: {} for me?
Can you localize "{}" in this image?	Could you please segment out {} in the image?

Table A.4: Templates used for the RES task.

A.3 GROUNDHOG Implementation Details

Data Balancing. In constructing the M3G2 dataset, we recognized the need to address the varying scales of the multiple constituent datasets to ensure a balanced data distribution during training. To achieve this, we have implemented dataset-specific sampling strategies, adjusting the volume of data from each source dataset through either up-sampling or down-sampling. The ratios we applied are as follows:

- PNG: up-sampled by a factor of 2.
- Flickr30k-Entities: up-sampled by 1.5 times.
- RefCOCO$^+$: up-sampled by 1.5 times.
- RefCOCOg: up-sampled by 1.5 times.
- SK-VG: up-sampled by a factor of 2.
- Dcube (multiturn): up-sampled by a factor of 10.
- ReasonSeg: up-sampled by a factor of 10.
- Shikra-Binary: up-sampled by a factor of 10.
- VCR-Open (multiturn): down-sampled by half.
- VCR-Multiturn: down-sampled to 10%.
- VizWiz: up-sampled by a factor of 3.
- LVIS-QA: down-sampled by half.
- TextVQAX: up-sampled by a factor of 2.

Instruction Templates For Short Response VQA.
{}Answer with a single word or a short phrase.
Given the image, answer the question "{}" with a single word or a short phrase.
Give a short answer to the question "{}" based on the image.
Instruction Templates For Chain-of-Thought Response VQA.
{}Let's think step by step.
{}Please include the reasoning process.
{}Before giving the answer, please explain your reasoning.
{}Explain your logic before giving the answer.
Please answer the following question "{}", and describe your thought process.
Instruction Templates For Grounding Answer to Masks.
Show where in the image you found your answer.
Mark the part of the image that supports your answer.
Please highlight your evidence in the image.
Point out the evidence from the image.
Indicate the area in the image that justifies your response.
Highlight the section of the image that backs up your answer.
Shade the section of the image that confirms your reply.
Emphasize the part of the image that relates to your answer.
Instruction Templates For Object Presence QA.
Is {} present in the image?
Is there any {} in this image?
Instruction Templates For Object Counting QA.
How many {} can you see in this image?
Count the number of {}.
Instruction Templates For Object Segmentation Request.
Segment {}.
Highlight all the {} in this image.
Show me all the {} presented in the picture.

Table A.5: Templates used for the GVQA task.

- EntityCount: up-sampled by a factor of 2.
- VG-SpotCap: down-sampled by half.
- Shikra-RD: up-sampled by a factor of 10.
- HierText: up-sampled by a factor of 5.
- GuessWhat-Oracle: down-sampled to 20%.

Instruction Templates For REG.
Provide a distinct description for that <PTR>
Describe the selected area in a unique way. <PTR>
Share a unique description of the region <PTR>
Offer a one-of-a-kind descriptor <PTR>
Describe the selected area <PTR> uniquely.
Point out <PTR> in the picture with a unique description.
Tell me how <PTR> stands out in the photo.
Use your words to highlight just <PTR> in the image.
Please describe <PTR> in the image in a way that it can be uniquely identified.
If you had to describe just <PTR> to someone, how would you do it?
What makes <PTR> different from everything else in the picture?
How can you describe <PTR> in the image in a way that it can be uniquely identified?
Can you provide a referring expression for <PTR> such that it sets it apart from others?
Let's play a game! Describe <PTR> in the photo so I can find it.
Instruction Templates For Region Captioning.
Describe it <PTR>.
Describe the region <PTR> in a few words.
Describe the region <PTR> in a short phrase.
Describe the selected area <PTR>.
Provide a brief description of this part <PTR>.
Give a short caption for this <PTR>.
Provide a brief description of the area marked <PTR>.
Tell me about the contents in the selected zone <PTR>.
Provide a concise description for this spot <PTR>.
Narrate what you see in the indicated area <PTR>.
What is in the region <PTR>? Describe in a phrase.
What can you see in this area <PTR>?
How would you describe the content at <PTR>?
How would you caption this particular region <PTR>?
What's depicted in the marked area <PTR>?

Table A.6: Templates used for the RD task.

- GuessWhat-Guesser: down-sampled to 20%.
- SVIT: up-sampled by a factor of 3.

The balanced sampled dataset contains 1.8 million samples in total.

Learning from Both Box and Mask Supervision. In the M3G2 dataset, not all sub-datasets include mask supervision, necessitating a hybrid loss approach to effectively benefit from grounded supervision from both mask and box annotations. We address this by employing different loss functions based on the type of annotation available. When

mask annotations are available, we apply the dice loss $\mathcal{L}_{\text{dice}}$ and binary cross-entropy loss $\mathcal{L}_{\text{bce}}$ between the predicted grounding masks and the ground truth masks of each phrase, following [42]. In cases where only box annotations are present, we apply the projection loss $\mathcal{L}_{\text{proj}}$ as introduced by [251], which selects the mask whose projection on the axis matches the best with the annotated box. Essentially, this can be seen as a 1D dice loss calculated between the projected masks and the edges of the ground truth boxes along both the x and y axes. Given that the primary objective of grounding is to accurately select the correct mask, we assign different weights to these loss components. The mask dice loss and box projection loss are both weighted at 1, while the mask bce loss is given a lower weight of 0.1. The final loss calculation is a summation of the language modeling loss $\mathcal{L}_{\text{lm}}$ and these mask-related losses.

LLM Configuration We adopt the Vicuna-7B model [44] as our base LLM, and use the OpenAI CLIP@336 [211] model and DINOv2-L/14-reg[54] pretrained checkpoints. We use the original conversation template from Vicuna, where all the interactions are formatted as `<system_message> <s> USER: <utterance> ASSISTANT: <utterance> </s>`.

Parameter-Efficient Training. We adopt the LLaMA2-7B model [255] as our base LLM, and initialized the weight from LLaVA-1.5 [153]. For the vision encoders, we use the OpenAI CLIP@336 [211] model and DINOv2-L/14-reg [54] pretrained checkpoints. We freeze all the parameters of the Mask2Former+, the CLIP, and the DINOv2 model during training. We use Low-Rank Adaptation (LoRA) [102] with $r = 16$ and $\alpha = 16$ to tune the LLM, including all the linear layers, input embeddings, and the LM head. We train all the new components introduced for connecting these models, including the MLP projection layer of CLIP and DINOv2, and the mask retrieval head. As a result, less than 2% of the total parameters are trainable in the whole model. We use the AdamW optimizer [163] with an initial learning rate of 2e-4 and a cosine annealing rate. We train our model on the balanced sampled M3G2 dataset for 2 epochs, which takes around 2 days using 8 40G A100 GPUs.

A.4 Additional Qualitative Results

We provide additional qualitative examples generated by GROUNDHOG. See Figure A.2 & A.3 for GIC, Figure A.4 for RES, Figure A.5 for GVQA, and Figure A.6 & A.7 for RD, respectively.

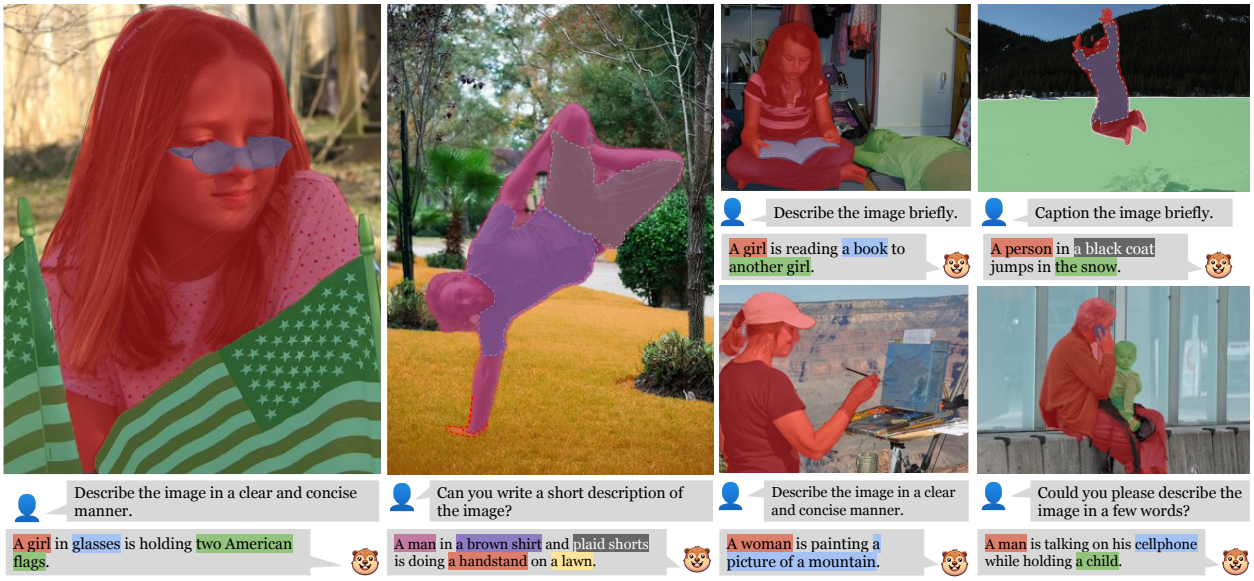


Figure A.2: Additional qualitative examples for short grounded captioning.



Figure A.3: Additional qualitative examples for detailed grounded captioning.



Figure A.4: Additional qualitative examples for referring expression segmentation.

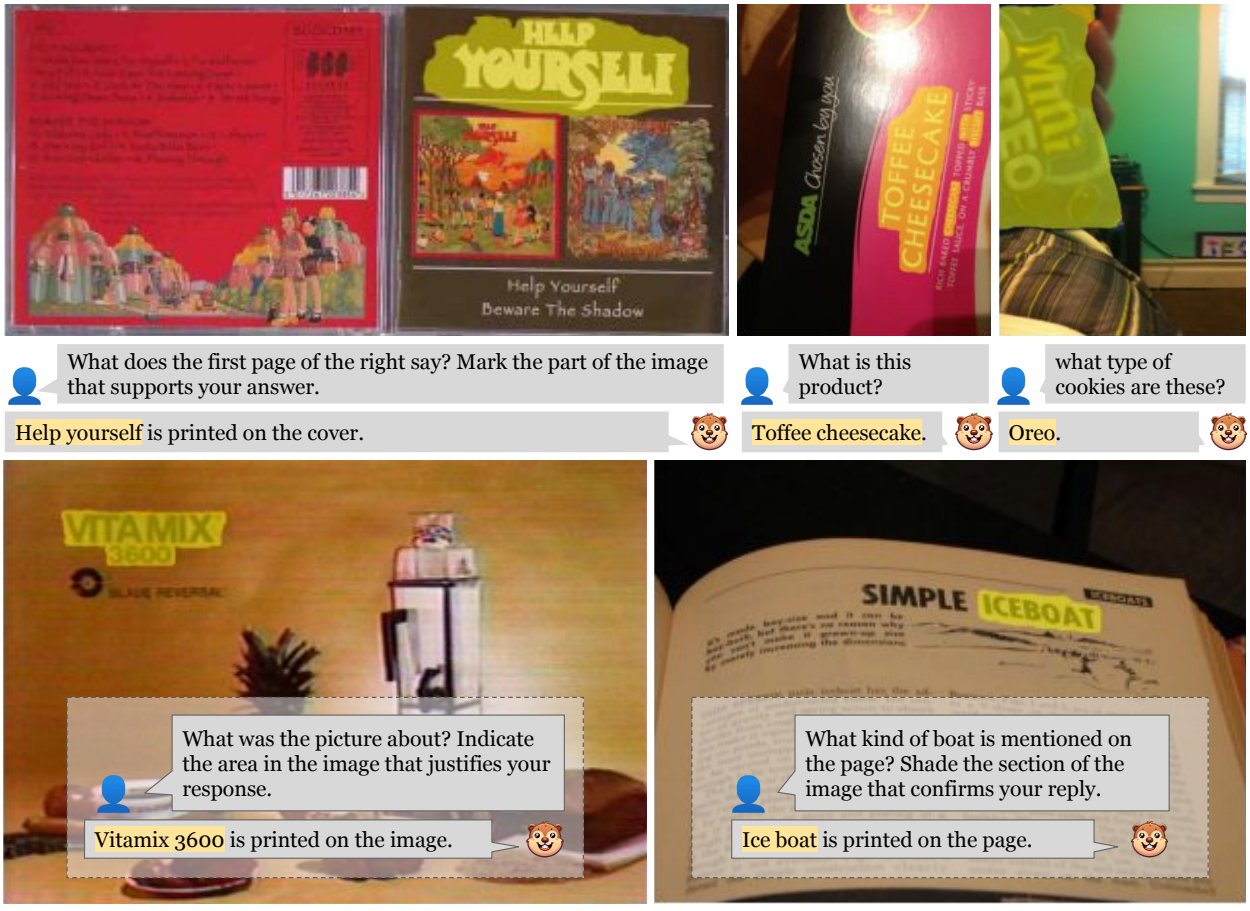


Figure A.5: Additional qualitative examples for grounded VQA of visual text.



Figure A.6: Additional qualitative examples for referential dialogue with pointer input.

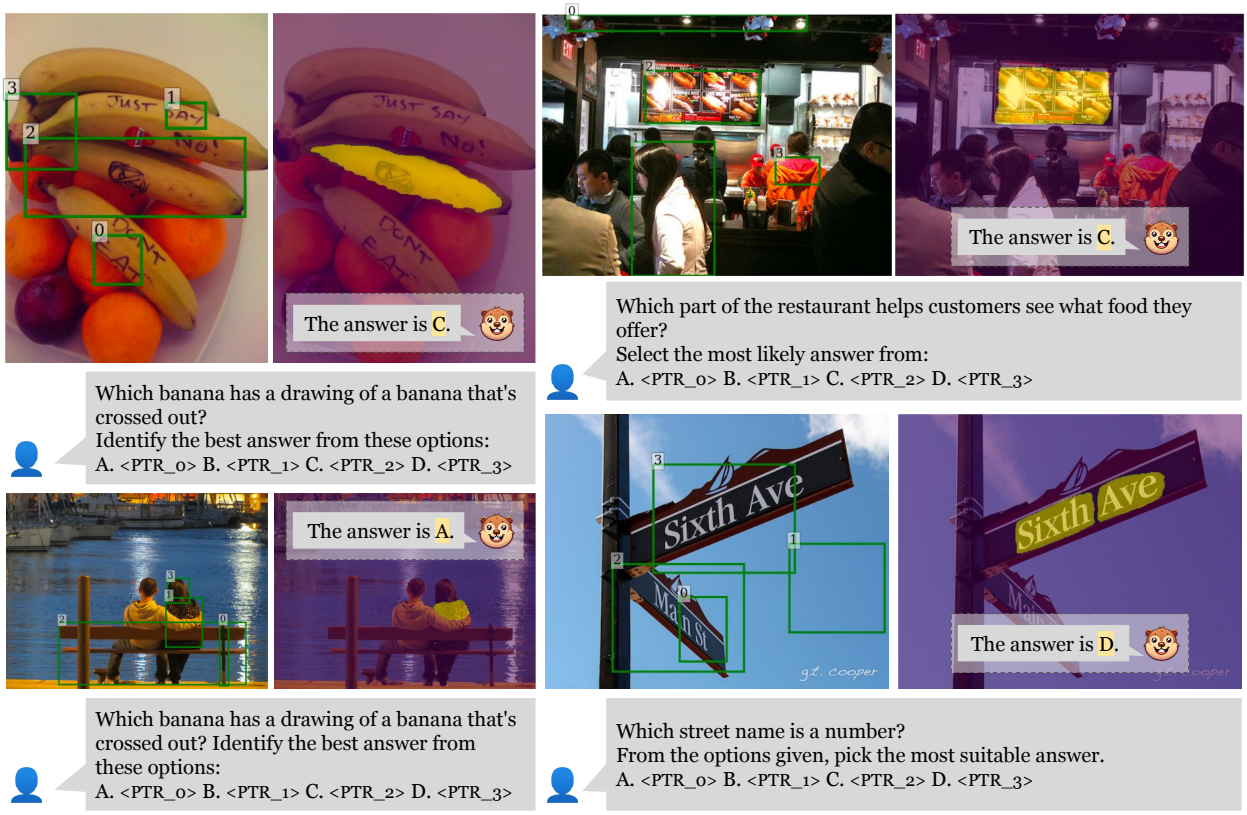


Figure A.7: Additional qualitative examples for referential dialogue with pointers as multiple choices input.

APPENDIX B

Appendix for Chapter 4

B.1 Statistical Analysis for Illusion Recognition

We conducted a statistical analysis to rigorously validate that our experimental setup is able to reveal a model's inclination towards either "humanlike" or "no-illusion", notwithstanding the high prevalence of N/A samples. Specifically, we applied a χ^2-test to the model predictions omitting the N/A samples. The null hypothesis posits that the humanlike and no-illusion samples are uniformly distributed, i.e., the model behaviors randomly. As shown in Table B.1, a significant proportion of the models reject the null hypothesis. Out of the 12 models tested, 8 models rejected the null hypothesis with a p-value < 0.001, and 9 models rejected the null hypothesis with a p-value < 0.01. These figures strongly suggest that most models perform better than what would be expected by chance alone, which is a piece of strong evidence that our dataset and experimental setup can support the evaluation of illusion recognition for the tested VLMs.

Model	#HL	#NI	χ^2	p-value
OFA-Tiny	17	172	129.45	<0.001***
OFA-Base	78	146	20.64	<0.001***
OFA-Large	58	73	1.72	0.190
OFA-Huge	60	136	29.47	<0.001***
UnifiedIO-Small	33	221	139.15	<0.001***
UnifiedIO-Base	33	62	9.38	0.002**
UnifiedIO-Large	115	103	0.66	0.416
UnifiedIO-XL	142	34	66.27	<0.001***
LLaVA-7B	13	202	12.89	<0.001***
LLaVA-13B	6	195	177.72	<0.001***
InstructBLIP-7B	91	223	55.49	<0.001***
InstructBLIP-13B	107	121	0.86	0.354

Table B.1: Chi-square test for the SameDiff Task (Figure 5). #HL and #NI denote the number of humanlike illusionary answers and no-illusion answers, respectively. Statistically significant results with $p < 0.001$ and $p < 0.05$ are marked with *** and **. respectively.

APPENDIX C

Appendix for Chapter 5

C.1 Additional Training Details

We use the RoBERTa [160] implementation from Huggingface [272]. The model is fine-tuned for 10 epochs with the Adam [121] optimizer on the ALFRED training set. The learning rate warms up over the first half of the first epoch to a peak value of 1e-5, and then linearly decayed. The model achieving the highest navigation action prediction accuracy on the validation seen set is selected for evaluation. All the models are trained on one NVIDIA V100 16GB GPU.

C.2 Additional Evaluation Details

We follow the evaluation setting in the ALFRED benchmark[1]. For each episode, the agent is given a task, which is composed of a goal directive G and several sub-goal instructions. The agent needs to sequentially perform actions to achieve the goal based on the visual observations of RGB image only. This progress ends if the agent predicts an *End* action (an *End* sub-goal for HiTUT), which is made after up to 10 failed interaction attempts or reaching the maximum step limitation. For HiTUT, there is also a maximum number of backtracking attempts, and the model will be forced to stop if the budget runs out. The maximum number of backtracking is 8 in all of our experiments without explicitly mentioning the backtracking number. We also leverage two techniques to reduce the interaction attempt failures. We use the obstruction detection trick proposed in [240] to avoid failures caused by repeated tries of moving toward obstructions. We propose a self-monitoring approach to check the validity of manipulation actions. If no mask is selected or a predicted action argument is not consistent with the class prediction from Mask R-CNN for the selected object, the manipulation action is decided as failed and the agent performs a backtrack without trying to execute the action.

[1] https://leaderboard.allenai.org/alfred/submissions/get-started

Note that in Table 5.4, we remove the interaction attempt constraint when comparing the effect of different allowed maximum backtracking numbers, thus the results of $\#BT = 8$ is slightly higher than those shown in Table 5.2.

C.3 Additional Results

A detailed per-task performance comparison of HiTUT and MOCA is shown in Table C.1. As the comparison might be unfair since HiTUT benefits from model pre-training, we also conduct an ablation study to show the effectiveness of pre-training. In Table C.2, we compare the fine-tuned RoBERTa model to a Transformer with the same size trained from scratch to show the role of the RoBERTa pretraining.We can see that RoBERTa consistently improves the performance over training from scratch both w/o and w/ backtracking with an absolute gain between 0.4% and 5% on task success rate. Notably, Scratch with 4 or 8 backtrackings still outperform MOCA by a large margin in terms of the unseen success rate.

Task-Type	MOCA		HiTUT	
	Seen	Unseen	Seen	Unseen
Pick & Place	29.5	5.0	**35.9**	**26.0**
Cool & Place	**26.1**	0.7	19.0	**4.6**
Stack & Place	5.2	1.8	**12.2**	**7.3**
Heat & Place	**15.8**	2.7	14.0	**11.9**
Clean & Place	22.3	2.4	**50.0**	**21.2**
Examine & Place	20.2	**13.2**	**26.6**	8.1
Pick Two & Place	11.2	1.1	**17.7**	**12.4**
Average	18.6	3.8	**25.2**	**12.4**

Table C.1: Success rates across 7 task types on the validation sets. Highest values per fold are **bold**.

Model	#Backtracking	Seen SR	Unseen SR
RoBERTa	no	10.5	5.2
Scratch	no	7.9	2.8
RoBERTa	4	23.1	12.9
Scratch	4	18.1	10.2
RoBERTa	8	27.2	16.2
Scratch	8	26.8	14.0
MOCA	-	19.15	3.78

Table C.2: The validation success rates for models pretrained and trained from scratch with different allowed maximum number of backtrackings.

APPENDIX D

Appendix for Chapter 6

D.1 DANLI System Implementation Details

A detailed overview of all system components and sub-components is provided in Figure D.1.

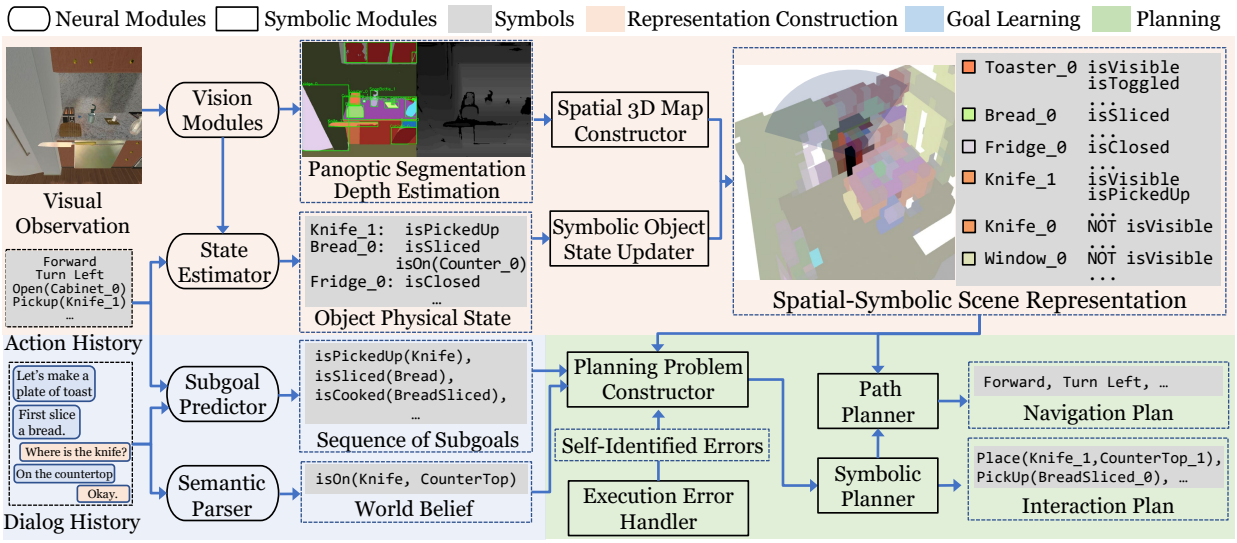


Figure D.1: Overview of our neuro-symbolic system, which learns a rich, visually-informed semantic representation of the surrounding environment, and captures hierarchical tasks in two phases. First, it uses a sequence-to-sequence model to predict the sequence of high-level subgoals, both completed and upcoming, and additionally extracts some spatial relations from the dialog to supplement the world representation. For each subgoal, we apply a specialized planner (either for navigation or manipulation) over the world representation to accurately execute the low-level actions and achieve the subgoal. Best viewed in color.

D.1.1 Perception System Details

Our perception system consists of a panoptic segmentation model, a depth estimation model and a physical state estimation model. At each time t, after receiving the RGB image I_t from

the egocentric vision, our perception system predicts egocentric depth estimation I_t^D, panoptic segmentation I_t^P, and corresponding state estimation I_t^S for each predicted objects in I_t^P. The output of this system serves as the foundation for building the symbolic representation for the agent,

We use main-stream pretrained models for the standard depth estimation and panoptic segmentation task and implement a custom Vision-Transformer [66] based model for physical state estimation. A dedicated dataset, TEACh-Perception, is collected for training these models.

D.1.1.1 Data Collection

We collect the visual perception data by replaying the TEACh dataset with additional filtering and data augmentation. Just like in the main experiment, we use the train split to build our training set, validation seen and validation unseen splits for two test sets.

We replay each of the games within the split and record egocentric images frame by frame, with their corresponding depth data, panoptic segmentation and object physical states ground truth extracted from the rendering engine. We find consecutive frames within the game can be very similar, degrading the dataset quality tremendously. Therefore, we propose to filter the frames on the fly. Namely, we use ImageHash [1] to calculate the similarity between the consecutive frames and only record the frame if its hash differs from the previous recorded frame by at least 3. To further improve dataset quality, we call the simulator's built-in lighting randomization and material randomization methods if the consecutive frames are similar by the aforementioned criteria. Following this method, we collect 179,144 annotated frames for the training set, 22,169 and 71,452 frames for the validation seen and unseen set respectively.

D.1.1.2 Model Design & Training

Depth Estimation We follow the customized U-Net [221] depth estimation model introduced in [24] and used its weight to initialize our model. To save computation, before passing I_t into the U-Net, we first downsample I_t's resolution from 900x900 to 300x300 and upsample the model's output back to 900x900 in the end. The model was finetuned with the same hyper-parameter for 30,000 steps on a single Nvidia A40 GPU on our TEACh-Perception dataset.

[1] https://github.com/JohannesBuchner/imagehash

Panoptic Segmentation Mask2Former [42] is used as the panoptic segmentation model. We use the Swin-B 21K model in MMDetection [37] for initialization, decrease the learning rate to 3×10^{-5} and finetune it for 80,000 steps on 8 Nvidia A40 GPUs.

Physical State Estimation As each object has different affordances,[2] we only predict the possible valid physical states for each object. For example, we will only predict a physical state value for the attribute `isFilledWithWater` for objects that are `Fillable`, e.g., bowls and cups.

We implement a custom model for state estimation tasks. The input consists of an object-centric image with its object's class id. After the image passes through an image encoder and class id through an embedding layer, two features are concatenated and go through two fully-connected layers to produce its state estimation. We use ViT-B/32 [66] with an additional projection layer as the image encoder and initialize its weights from CLiP [212]. The projection layer is a linear layer followed by a ReLU function, projecting the 512-dimension image feature down to 128 dimensions. The class id embedding has a dimension of 8. The following 2 fully-connected layers are with ReLU activation function, each with output dimensions 128 and 3. The model is then trained for 150,000 steps on our TEACh-Perception dataset with a learning rate of 1×10^{-5}.

Lastly, our system also requires predicting objects' spatial relationship of whether or not one object is on another object. We use a simple rule-based algorithm to predict these spatial relations, which is, given object A in class C_A with bounding box $B_A = (x_{A1}, y_{A1}, x_{A2}, y_{A2})$, object B in class C_B with bounding box $B_B = (x_{B1}, y_{B1}, x_{B2}, y_{B2})$, where bounding boxes are in absolute coordinates notation, the model predicts A is on B if and only if 1.) from the prior receptibility knowledge, it is possible for C_A to be on C_B, 2.) $(x_{B1} \leq (x_{A1} + x_{A2})/2 \leq x_{B2}) \wedge (y_{A2} \geq y_{B1}) \wedge ((y_{A1} + y_{A2})/2 \leq y_{B2})$.

During experimentation, we find three factors constraining state estimation accuracy and provide solutions respectively. First, the object-centric images are extracted from I_t by objects' bounding boxes, which include little information about its surrounding. For example, to predict the `isToggled` state of a cabinet, the spatial relationship between the cabinet and the desk must be included in the image. To solve this problem, we enlarge the bounding box by 30% on each side to include more spatial information. Secondly, we find when in distance, some objects will be too small to extract meaningful state information, hurting the accuracy of physical state estimation. Therefore, we filter out objects whose corresponding bounding boxes are less than 1000 pixels large in the egocentric image before passing into this model. Lastly, although whether or not a `StoveBurner` is turned on is mostly reflected on

[2]Object affordance is pre-configured in Thor [126], the virtual environment used to create TEACh.

the existence of fire flames above it, the corresponding `isToggled` state is stored in the `Knob` object controlling the `StoveBurner`. To solve this misalignment between the visual feature and the corresponding state, we manually move the `isToggled` state from the knob to the corresponding `StoveBurner`.

D.1.2 Map Construction and Navigation

We describe below the transformation procedure of converting egocentric visual observations to a 3D semantic map. Inspired by [24], we project the pixels from the panoptic segmentation model, with the help of a depth estimation model, to a 3D point cloud. The points in the point cloud are then binned into 0.25m^3 cubes, called *voxels*. Note that throughout this conversion process, object class information is preserved. The resulting map is $V_t \in [0,1]^{C\times X\times Y\times Z}$, where C is the number of object classes and X, Y, Z are the length, width and height of the map, respectively. Given this 3D semantic voxel map, we can plan navigation paths using a deterministic path planner like the Value Iteration Network (VIN) introduced in [24]. The path produced can be further converted to navigation action sequences, which can be directly followed by the agent through primitives. See Figure D.2 for an example of map construction and path planning results in a single episode. When exploring for missing objects, the agent randomly samples an object from the 5 farthest objects in its current world representation and navigates to it. There is no specific timeout for the search action, but each subgoal is limited to a maximum of 200 action steps.

D.1.3 Instance Match Algorithm

We use the following algorithm to match object instances from the current observation to instances already existing in the symbolic instance lookup table. If it is the first time that an instance is observed, we will register it into the lookup table. First, we group all the detections in the current observation based on its object type, and denote n_c^{2D} as the count of object of type c. Then for each object type, we get object instances of the same type from the lookup table that is visible (at least one voxel falls into the visibility mask) at the moment, and denote the count as n_c^{3D}. If $n_c^{2D} == n_c^{3D}$, we match each pair of them to minimize the pair-wise distance computed as the Euclidean distance between their centroids. If $n_c^{2D} > n_c^{3D}$, we match up to n_c^{3D} detections to the existing instances, and then register new instances into the lookup table. If $n_c^{2D} < n_c^{3D}$, we will match up to n_c^{2D} detections to the existing instances, and remove the remaining instances. When a detection is matched to a instance in the lookup table, we update its voxel mask by performing a logical-or operation of the projected voxel mask of the detection and the existing voxel mask. Thus different partial

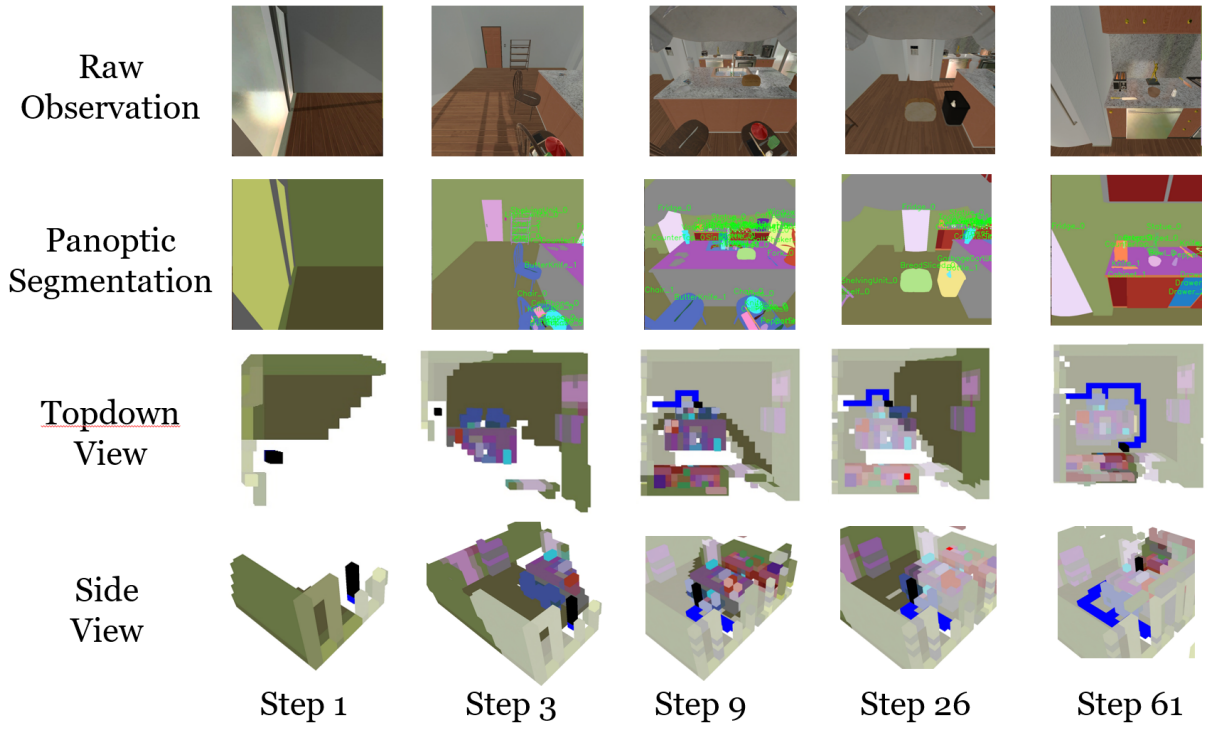


Figure D.2: Illustration of the 3D voxel map construction and interaction process of a making toast task. The agent is labeled in black, and the agent path is labeled in blue. Current navigation target in the agent's plan is labeled in red.

observations of the same instance at different positions can get merged.

D.1.4 Subgoal Prediction

D.1.4.1 Subgoal Labeling

To annotate the key subgoals for each task type in TEACh, we define a minimal set of object states that is essential for the completion of the task. For example, as previously shown in Figure 6.2, `(Knife,isPickedUp)` and `(Bread,isSliced)` are subgoals for making toast, while `(Knife,isPlacedTo,Counter)` is not. This is because while placing a knife on the counter may often happen when making toast in the TEACh data, it is not essential to achieve the goal conditions for making toast. We then use an automatic subgoal labeling algorithm to get these subgoal annotations for the entire dataset.

D.1.4.2 Subgoal Annotation

We propose a method to automatically annotate subgoals from the trajectories. Based on the metadata of task goal conditions and the oracle world state change of each step recorded

Commander	Follower			
Hello				
	Hi What is my task?			
Please make a slice of toast.				
	Goto CounterTop_0	FOR	Knife_0	Get Knives
	Pickup Knife_0	FROM	CounterTop_0	
	Slice Bread_0			Slice Bread
	Place Knife_0	TO	CounterTop_0	
	Pickup BreadSliced_0	FROM	CounterTop_0	Cook BreadSliced
	Place BreadSliced_0	TO	Toaster_0	
	Done			
Now put on the plate				
	Goto Toaster_0	FOR	BreadSliced_0	Place Toast To Plate
	Pickup BreadSliced_0	FROM	Toaster_0	
	Place BreadSliced_0	TO	Plate_0	
	Done			
Great job, task done.				

Figure D.3: An example of subgoal annotations.

in the dataset, we extract whether a goal condition is achieved at every step. If there is a goal condition completed, it is annotated as a subgoal being achieved in that step. We show an annotation example in Figure D.3.

D.1.5 Symbolic Planning Details

In this section we describe details of our symbolic planning module, including how PDDL planning works, the types of subgoal predicates we used in our experiments, and pseudocode to describe how our online planning pipeline works.

D.1.5.1 PDDL Planning

PDDL planners, such as FastDownward [98] and ENHSP [226], require as inputs *problems*, which symbolically specify the existing preconditions and desired goal, and *domains*, which describe the symbols that can be used and the possible actions that can be planned over. However, before forming PDDL problems, we first adjust the scene that the agent works with. This serves to speed up planning, as described in section 6.3.3.

The domain is formed via provided object and action symbols along with manually encoded interactions within the simulator. Primitive actions are qualified with *conditional effects*, which allows one to encode more complex, contextual interactions into the domain; e.g. if a plate is moved underneath a toggled faucet, then the planner should assume its state is changed to `isClean`. Within the domain one may also encode action costs, which specify

priorities for actions.

An example problem, which contains relevant objects, initial conditions, and goal conditions, looks like the following:

```
(define (problem knife_problem)
    (:domain knife_domain)
    (:objects
        Knife_0 - Knife
        DiningTable_0 - DiningTable
    )
    (:init
        (isVisible Knife_0)
        (isNear Knife_0)
        (isVisible DiningTable_0)
        (isNear DiningTable_0)
    )
    (:goal (isPickedUp Knife_0))
)
```

An example domain, which contains types of objects to plan with, predicates, and action schemas, may look like:

```
(define (domain knife_domain)
 (:requirements
     :adl :strips
     :typing :conditional-effects
 )
 (:types
     Knife - Pickupable
     DiningTable - Receptacle
     ... ; more types
 )
 (:predicates
     (isVisible ?x - Object)
     (isNear ?x - Object)
     (isPicked ?x - Pickupable)
     (parentReceptacles
         ?x - Object
         ?z - Receptacle
```

```
    )
    ... ; more predicates
)
(action Pickup
    :parameters (?x - Pickupable)
    :precondition (and
        (isNear ?x)
        (forall (?z - Pickupable)
            (not (isPickedUp ?z))
        )
    )
    :effect (and (isPickedUp ?x)
        ; remove from its parents
        (forall (?z - Receptacle)
            (not (parentReceptacles ?x ?z)))
        ; children should all be picked
        ; up if the parent is picked up
        (forall (?z - Object)
            (when
                (parentReceptacles ?z ?x)
                (isPickedUp ?z)
            )
        )
    )
)
... ; more actions
)
```

Subgoal predicates Below we describe the subgoal predicates used specifically in our TEACh experiments.

- `isCooked`: True if a food item has been cooked. Can be the result of being placed in a toaster, a microwave, or on a stove.
- `isClean`: True if an object is clean. Can be the result of being placed under running water.
- `isPickedUp`: True if an object is currently in the agent's hand.
- `isFilledWithLiquid`: True if an item is filled with some liquid, whether water or coffee. Can be the result of being placed under running water or in a coffee machine.

Algorithm 1: Hierarchical Planning

Input: PDDL domain D, Context C

1 **Function** `HierarchicalPlan`(D, C):
2 $S \leftarrow$ `InitializeRepr`(C)
3 $G \leftarrow$ `SubgoalPredictor`(C)
4 **foreach** $g \in G$ **do**
5 ▷ PDDL Planning
6 $P \leftarrow$ `FormPDDLProblem`(g, S, D)
7 $\Pi \leftarrow$ `PDDLPlanner`(P, D)
8 **foreach** $a \in \Pi$ **do**
9 $o, e \leftarrow$ `ExecuteAction`(a)
10 **if** $e \neq$ `None` **then**
11 ▷ Error handling
12 **if** $e \in$ *solvable* **then**
13 $g' \leftarrow$ `PlanRecovery`(o)
14 **goto** line 5 with $g := g'$
15 **else**
16 **return** with failure
17 **end**
18 **end**
19 ▷ Internal representation update
20 $S \leftarrow$ `UpdateRepr`(o, S)
21 **end**
22 **end**
23 **return**
24 **End Function**

- `isEmptied`: True if an item has nothing inside it.
- `isSliced`: True if an item exists as the result of a slicing operation, such as a bread slice or tomato slice.
- `simbotIsFilledWithCoffee`: True if an object is filled with coffee. Can be the result of an item being placed under a coffee machine.
- `isPlacedTo`: True if an object x currently lies in/on an object y.
- `isToggled`: True if an object has been toggled on, such as a light switch or a coffee machine.

D.1.5.2 Online Planning Algorithm

Our full planning algorithm, including the re-planning functionality to recover from failed actions, is listed in Algorithm 1. The subprocesses used in the algorithm are the following:

- `InitializeRepr`: given a context C, initialize the state representation S and return it
- `SubgoalPredictor`: given a context C, output a set of predicted subgoals G
- `FormPDDLProblem`: given a single goal g, a state representation S, and a PDDL domain D, output a primitive-level plan Π
- `ExecuteAction`: Given a single primitive action a, execute it and return an observation o and error state e.
- `UpdateRepr`: given an observation o and state representation S, update S with the new observation.

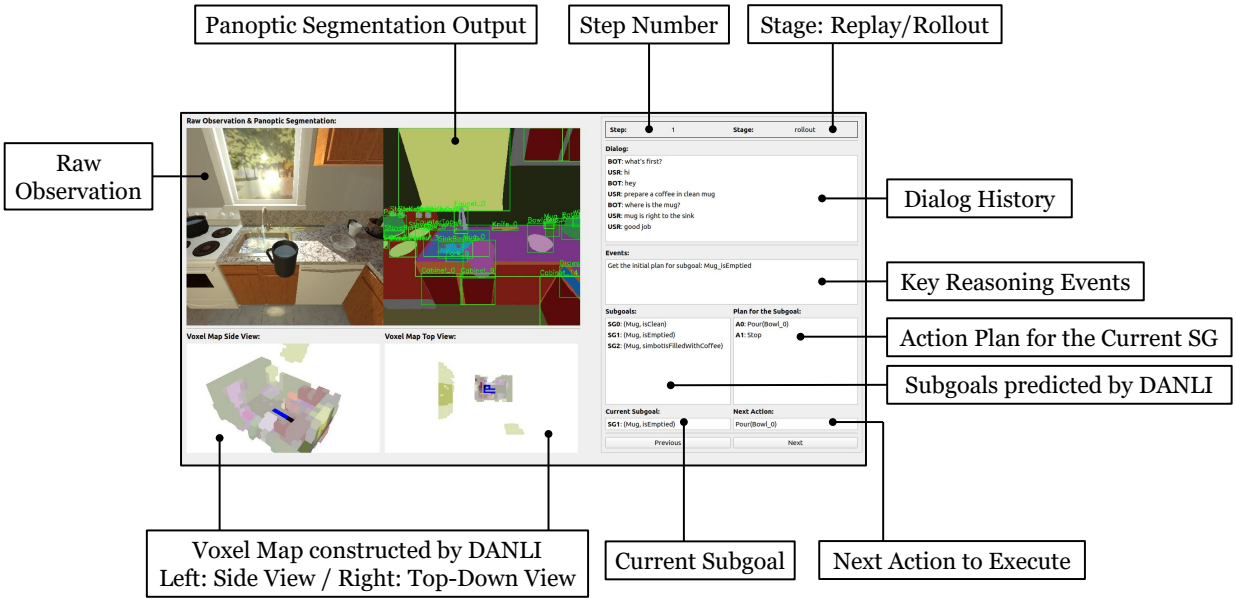


Figure D.4: Demo of the Graphical User Interface (GUI) created for visualizing DANLI's internal representations.

D.1.6 DANLI GUI Interface

Figure D.4 show a snapshot of the GUI interface we built for visualizing DANLI's intermediate representations.

D.2 Implementation Details of Baselines

The architecture of the HET model is shown in Figure D.5. We use two transformer models to implement the controller and the navigator respectively. For the HET-ON model, the

navigator is replaced with an oracle navigator which can get access to the ground truth environmental state for reliable navigation. We conduct an extensive hyperparameter search over the model architecture and optimization process. We find that using a 4-layer Transformer with a hidden size of 768 and 12 attention heads obtains the best results on the validation set (average of seen and unseen split). We find the navigator achieves the best performance when increasing the layer number from 4 to 6.

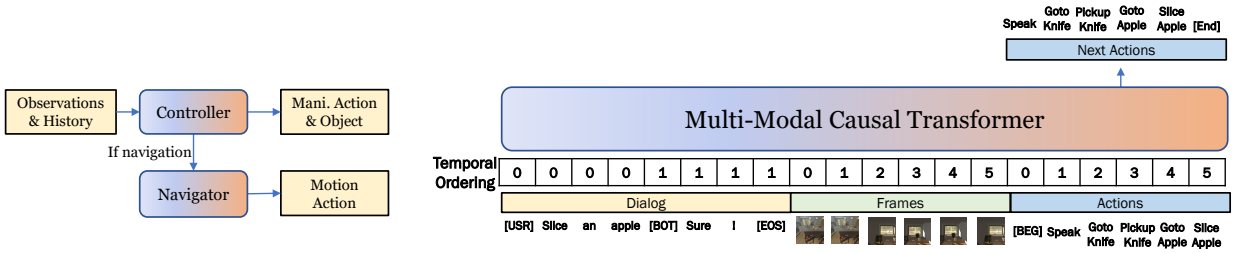


Figure D.5: Architecture of the hierarchical ET model.

D.2.1 Results on Original Validation Split

The original test splits of the TEACh dataset are used for the SimBot Challenge,[3] and are thus not publicly available at the time of writing this paper. To facilitate model evaluation, the authors of TEACh re-split the original validation set into new validation and test sets, which are used in this paper as the standard evaluation protocol. In Table D.1, we additionally provide the results on the EDH subtask of TEACh using the original validation splits. Future work evaluated on the original split can refer to these results when comparing with our model.

Model	Valid Seen		Valid Unseen	
	SR	GC	SR	GC
ET	8.55 (0.67)	9.10 (3.39)	7.86 (0.91)	6.20 (3.43)
HET	8.72 (1.00)	9.95 (5.29)	9.31 (1.74)	9.50 (5.35)
HET-ON	12.01 (3.43)	16.45 (9.93)	12.42 (3.43)	14.95 (9.41)
JARVIS	15.10 (3.30)	22.60 (8.70)	15.80 (2.60)	16.60 (8.20)
DANLI-PaP	15.62 (7.76)	20.39 (18.33)	14.19 (6.12)	19.32 (17.57)
DANLI (Ours)	**17.76 (9.28)**	**24.93 (22.20)**	**16.98 (7.24)**	**23.44 (19.95)**

Table D.1: EDH results on the original validation splits.

[3] https://eval.ai/web/challenges/challenge-page/1450/overview

APPENDIX E

Appendix for Chapter 7

E.1 Dataset

E.1.1 Synthetic Dialogue Data Generation Details

Dataset	**Domain**	**#Tasks**	**#Videos**	**Total/Avg Duration**	**Labels**
Ego4D-Goalstep [90, 241]	Cooking	86	851	368h/26m	C + F
EpicKitchen [50, 51]	Cooking	-	700	100h/9m	F
HoloAssist [267]	Manipulation; Assembly	20	2221	166h/5m	C + F + M + D
EgoExoLearn [104]	Cooking; Laboratory Tasks	8	432	96h/13m	C + F + R
Assembly101 [227]	Assembly	101*	356	42h/7m	C + F + M
WTaG [21]	Cooking	3	56	10h/11m	C + M + D + R

Table E.1: Summary of egocentric video datasets used in ProAssist. The statistics presented are as originally reported in the corresponding papers before filtering. The number of tasks indicates the types of tasks in each dataset, except for Assembly101, where it represents the unique number of toys. Label abbreviations are as follows: C (coarse action labels), F (fine-grained action labels), M (mistake and correction labels), D (human-collected assistant-user dialogues), and R (ground-truth recipes).

Videos in ProAssistare sourced from six extensively labeled datasets: Ego4D [90] with GoalStep annotations [241], EpicKitchen [50, 51], HoloAssist [267], Assembly101 [227], EgoExoLearn [104], and WTaG [21]. Detailed statistics and label types are summarized in Table E.1. When generating timestamped video descriptions for these videos, we leverage all available labels to provide a comprehensive understanding of the video content. In cases where both coarse- and fine-grained action labels are available, they are organized into hierarchical formats for clarity. An example of this unified timestamped video description, incorporating coarse and fine-grained actions, mistake corrections, and assistant-user dialogues, is shown in List E.1.

```
[11.0s-28.2s] The user grabs the GoPro.
```

```
 - [11.0s-12.2s] approach gopro
 - [13.7s] assistant: "Okay."
 - [15.1s] assistant: "You can pull the GoPro."
 - [16.9s] user: "GoPro."
 - [17.0s-22.2s] grab gopro
 - [22.2s-23.5s] flip bag
[29.9s-66.2s] The user changes the battery for the GoPro.
 - [30.2s] assistant: "Change the Battery."
 - [34.2s-41.0s] pull battery door
 - [41.0s-42.3s] open battery door
 - [43.5s-45.0s] grab battery
 - [45.1s-46.1s] withdraw battery
 - [45.6s] assistant: "Take out the battery."
 - [47.5s-49.6s] place battery
 - [48.8s] assistant: "Now, put it down"
 - [49.7s-50.9s] lift battery
 - [50.9s-51.7s] insert battery
 - [58.8s] assistant: "Close it."
 - [59.8s-61.0s] close battery door
 - [62.2s-63.5s] push battery door
 - [67.0s] assistant: "Now change the micro SD."
[68.4s-304.3s] The user opens the GoPro.
 - [68.6s-70.8s] grab battery door
 - [70.8s-72.7s] open battery door
 - [72.7s-73.9s] press battery (ERROR: The user presses the wrong place.)
 - [73.9s] assistant: "No that one."
 ...
```

Listing E.1: An example video description from HoloAssist.

Next we provide additional details for each step in our data curation pipeline.

Task Goal and Recipe Generation The objective is to generate a high-level task goal and recipe-style instructions that outline the key steps required to complete the task, derived from the video descriptions. For datasets where these elements are already provided in their labels (EgoExoLearn, WTaG), this step is skipped. During generation, we observe that the LLM can sometimes be distracted by irrelevant actions in the video descriptions, resulting in less accurate or unstable outputs across different sampling trials. To address this issue, we employ a two-step process. First, we generate 10 candidate recipes[1] using the prompt shown in List E.2. Next, we refine these recipes into a single cohesive and integrated version by calling the LLM one more time with the prompt in List E.3.

[1]We use vLLM[132] for efficient parallel sampling.

```
Here is a video description of an experienced user working on the task -
{goal_description}:
{video_descriptions}

Try to infer the **high-level** recipe from the descriptions. Note that the steps may
not belong to the same trial, so you have to infer the correct order of the steps based
on common sense, and re-order the steps if necessary. Do not hallucinate details that
are not mentioned in the descriptions. Also generate a more **informative** and
**descriptive** name for the task based on provided descriptions. The name should be a
description of the task, instead of the name of the recipe.

Give plain and concise text with numbered key steps in the following format:
[task name] 1. ...  2. ...
```

Listing E.2: Prompt for inferring task goal and recipe from video descriptions.

```
Here are {num_repeats} {knowledge_type}s:
{recipes}

Some may be incorrect or incomplete. Please give a single correct and complete
{knowledge_type} for the task, with numbered key steps. Pick the title that is
descriptive for the task, instead of a {knowledge_type} name.

Give plain, unformatted and concise text with numbered key steps in the following format:

[task name] 1. ... 2. ...

Do not include any other information or note.
```

Listing E.3: Prompt for task goal and recipe refinement.

Video Pre-Filtering The next step is to filter out videos that are unsuitable for proactive assistant-user dialogue modeling. First, we exclude videos with low label coverage, as their descriptions may lack sufficient detail to provide a clear understanding of the content. Then, using the video descriptions, task goals, and recipes generated in the previous step, along with the domain and recipe type derived from the dataset metadata, we prompt the LLM to classify each video into one of three categories:

- 0: The task does not belong to the target domain.
- 1: The camera wearer performs the target task following the recipe.

- 2: The camera wearer performs other tasks simultaneously while working on the target task.

We keep only the videos classified as category 1 for subsequent steps. To minimize noise, this classification process is repeated 10 times for each task, and the majority label is used as the final classification.

```
Here is a video description of an user working on the task - {goal_description}:
{video_descriptions}

Reference {knowledge_type}:
{knowledge}

Is this a {knowledge_type}? If so, was the user likely to:
1. perform the task roughly following the {knowledge_type} (**no** need to be strict), OR
2. perform other tasks (or another trial of the same task) simultaneously in a
multi-tasking manner?

Answer with your analysis, and end your response with "Final answer: 1, 2 or 0" (0
denotes that the activity is not related to {domain}).
```

Listing E.4: Prompt for video pre-filtering.

Multi-Round Dialog Generation To simulate realistic dialogues aligned with video content, we design a detailed instruction prompt, as shown in List E.5. We incorporate dataset-specific instructions to account for variations across data sources to improve generation quality (List E.6). To simulate diverse user behaviors, we define three type of user profiles that provide high-level guidelines for user interaction:

- `no_talk`: The user follows the assistant's instructions without speaking.
- `talk_some`: The user occasionally asks questions or seeks confirmation about instructions, accounting for approximately 20% of the steps.
- `talk_more`: The user is talkative, asking both task-related and unrelated questions, accounting for approximately 40% of the steps.

Since we can sample different user behaviors for the same video, our synthetic dataset can be easily expanded by generating multiple variations. Specifically, we create 10 dialogues per video, distributed across user types in a 2:4:4 ratio. In practice, we observe that very long videos can result in excessively lengthy prompts, which may lead to poor alignment between the video description and the generated dialogue due to performance degradation of LLMs

when processing long contexts [158]. To address this, we propose an iterative approach to generate dialogues within a limited time window chunk by chunk. For each chunk, we provide only the video description corresponding to that time window and up to the 10 most recent dialogue turns to ensure contextual consistency. This modification significantly improves alignment between the video description and the generated dialogue while stabilizing memory consumption due to the reduced prompt length. After generation, we conduct an additional refinement step to enhance dialogue quality. Specifically, we prompt the LLM to merge dialogue turns that occur close in time, improve naturalness and fluency by incorporating more coreference and pronouns, make assistant responses more concise, and avoid unfriendly behaviors. Refer to List E.7 for details.

```
Here is a video description of an user working on the task - {goal_description}:
{video_descriptions}

Your goal is to simulate a conversation between the user and an assistant, where the
user's actions are performed following the assistant's instructions. The user will first
mention the overall goal of the task. The assistant informs the user about the next step
at proper time. Importantly, the assistant is proactive and always provides the next
step even before the user asks for it. Before the task starts, the assistant may also
give a brief introduction about the task. {additional_requirement}

Requirements for the assistant:
- Time is crucial! Try to generate the dialog that strictly aligns with the video
timeline.
- Try to cover all the essential steps in the task. If the user asks a question at the
time the assistant should give the next step, the assistant turn should include both the
response to the question and instruction about the next step.
- Be helpful and friendly. If the user asks something that has been explained before,
the assistant should still provide the information with patience.
- Try to be encouraging when the user makes progress, but do not overdo it.
- Be concise! The dialog is verbal, so avoid long sentences.
- Do not say "can you do it for me" to the user.


Requirements for the user:
{user_requirement}


Generation format:
[time] User: ...
[time] Assistant: ...
[time] Assistant: ...
```

```
[time] User: ...
[time] Assistant: ...

Note that the minimal interval between each turn is 1 second, which means the user will
wait for at least 1 second after an assistant's turn, and two consecutive assistant's
turns should have at least 1 second interval. Combine close turns into a single turn if
necessary. One exception is that the assistant must respond **immediately** when the
user says something (i.e. give a response right after an user's turn at the same time).

{dialog_history}

In this round, please **only** generate the dialog for the video from time
[{start_time:.1f}s] to [{end_time:.1f}s]!
```

Listing E.5: Prompt for dialogue simulation.

```
HoloAssist: Note that the video description contains both the user's actions and the
user-assistant dialog. Anchor the simulated dialog to the existing dialog, and try to
rephrase the utterances to make them more coherent and human-like. You may add a few
more turns around the **essential steps** of the task, which are the underlying
intentions of the action instead of the actions themselves. Add a few turns to make the
dialog more fluent and helpful, but avoid being overwhelming.

EgoExoLearn: The simulated dialog should be centered around the **key steps** of the
task, not every single action of the user. Try to make the dialog more coherent and
helpful as what a human assistant will say.

Epickitchens: The simulated dialog should be centered around the **key steps** of the
task, not every single action of the user. Note that the user may make mistake or
perform suboptimal actions, the assistant should not give instructions on those actions,
but smartly select right time to give guidance. Try to make the dialog more coherent and
helpful as what a human assistant will say.

WTaG: Note that the video description contains both the step description and the
user-assistant dialog. Anchor the simulated dialog to the existing dialog, and try to
rephrase the utterances to make them more coherent and human-like. Add more details such
as assistant feedback or user question during long steps if necessary. Remember to
generate the response to user's question even if there isn't one in the original dialog
from the video description.

Assembly101: The mistakes made by the user are marked by (mistake: <mistake type>). If a
mistake happens, we want to simulate the dialog in the way that the assistant helps the
user correct the mistake. To be more specific, the assistant SHOULD NOT give
instructions if an action is 'wrong order', 'previous one is mistake' or 'shouldn't have
```

```
happened'. Instead, the assistant should give instruction of the CORRECT next step (i.e.
scan the future actions and select the nearest correct action). Afterwards, at the start
of actions marked as 'correction', the assistant should mention the previous mistake and
give insruction on how to correct it based on the corrective action. For 'wrong
position' mistakes, the assistant can give the instruction of that action, but need to
point out the mistake at the start time of corrective action for that mistake.
```

Listing E.6: Prompt specific for each dataset as additional requirements.

```
Here is a conversation between a user and an assistant:
{dialog_history}

For each assistant message, add labels regarding the assistant's initiativity and
intention:

Initiativity:
- initiative: The assistant says something proactively without the user asking for it.
- responsive: The assistant responds to the user's question or comment.

Intention:
- instruction: The assistant gives an instruction to the user.
- correction: The assistant corrects a mistake made by the user, either proactively or
responsively. Suggestions for alternative actions can also be included.
- info_sharing: The assistant shares some information with the user, such as explaining
something or giving a tip.
- feedback: The assistant gives feedback to the user, such as "good job" or "tips for
improvement".
- other: Other intentions that do not fall into the above categories.

Intention can be multiple, e.g., "instruction, info_sharing".

Generation format:
[time] User: ...
[time] Assistant: ... [initiativity|intentions]
[time] Assistant: ... [initiativity|intentions]
[time] User: ...
[time] Assistant: ... [initiativity|intentions]

When generating the dialog, you should also refine the dialogue following these
guidelines:
1. Merge turns that are close in time (less than 1 second apart) into a single turn,
when the content is similar or related.
2. Use more coreference and pronouns to make the dialog more coherent and human-like.
3. Decide the length of assistant messages smartly. Make them more clear and helpful
```

```
when necessary, but keep them concise and to the point in general.
4. Avoid repeating the same talking patterns or phrases. For example, do not say "make
sure ..." for every instruction.
5. Rephrase impolite or inappropriate language, such as "as I have mentioned this
earlier ...", to be more friendly and helpful. But keep concise and to the point.
6. Remove anything other than the dialog itself, such as the user's actions or
explanations of how the dialog is generated.
Do not just copy paste the original dialog!
```

Listing E.7: Prompt for dialogue refinement and intent labeling for assistant turns.

Dialogue Annotation To facilitate the analysis of our generated dialogues, we use LLM to annotate the initiativity (responsive or initiative) and intention type (instruction, correction, information sharing, feedback, and other) of each assistant turn. We find such annotation can be effectively generated within the dialogue refinement step, using a single LLM call with the prompt in List E.7. Additionally, we generate progress summaries at each assistant turn to support the iterative progress summarization approach (§7.4.3. These summaries include details such as the elapsed time, the task goal mentioned by the user, completed steps as progress, topics discussed by the user, and the current state or step of the task (List E.8).

```
Here is a conversation between a user and an assistant:
{dialog_history}

Summarize the task goal and progress so far, including:
1. The task goal mentioned by the user.
2. What has been done.
3. Other topics mentioned by the user in the conversation, if any.
4. The current state/step of the task.
Be faithful and try to include all the relevant information.

Give your response in plain text of a single line in the following format:
SUMMARY: <progress summary>
```

Listing E.8: Prompt for progress summary generation

Automatic Quality Evaluation To evaluate the quality of the generated dialogues, we assess their alignment and step coverage with the corresponding video descriptions. We first extract all time steps from the video descriptions, denoted as T_v, and all time steps from the generated dialogues, denoted as T_d. For each time step in T_d, we identify its closest time step in T_v and compute the average time difference across all pairs, normalized by the number of dialogue turns. Similarly, for each time step in T_v, we find its closest match in T_d and

calculate the average time difference, normalized by the number of video description steps. These values approximate the precision of dialogue turns relative to the video (p) and the recall of task steps in the video descriptions (r). Additionally, to ensure the assistant remains responsive, we count the number of user turns without an immediate assistant response as a penalty term nr. The final quality score is computed as $score = 10 - p - r - nr$, the higher the better.

Post-Filtering and Data Splitting We derive our training set from the training splits provided by each original dataset, while our validation and testing sets are based on the respective validation splits. For the training set, we filter out dialogues with a score below 3. For the validation sets, we retain only the highest-scoring dialogue for each user type. If any dialogue for a video scores below 5, the video is removed. From the remaining videos, where each has three dialogues, we evenly split them into validation and test sets. This process removes approximately 25% of the dialogues and 41 hours of video.

E.1.2 Implementation Details

We utilize LLaMA-3.1-70B-Instruct [67] as the LLM for all the steps described above. The model is hosted locally using vLLM [132], running a FP8-quantized version[2] on four H100 GPUs.

E.1.3 Data Statistics and Distributions

Figure E.1 provides a comprehensive overview of the ProAssistdata statistics. Specifically, Figure E.1a presents the distribution of task domains by video durations, showing that the dataset predominantly contains cooking tasks (58.3%), followed by object manipulation (25.4%), assembly (12.0%), and laboratory tasks (4.3%). Figure E.1b illustrates the dialogue length distribution, measured by the number of turns per dialogue, highlighting a significant variability with some dialogues exceeding 200 turns. Figure E.1c shows the task duration distribution where the majority of tasks last under 20 minutes, while some extend up to an hour. Figure E.1d visualizes the length distribution of user and assistant utterances, as well as the generated progress summaries. Assistant utterances are generally longer (mean = 16.1 words) compared to user utterances (mean = 6.5 words). For the generated summaries, the average length is 91.3 words, with the largest summary less than 200 words, showing that the information can be successfully compressed into such summarizes without growing linearly with the dialogue. Figure E.1e highlights assistant initiativity and intention distributions

[2] https://huggingface.co/neuralmagic/Meta-Llama-3.1-70B-Instruct-FP8-dynamic

across three user types. As users speak more frequently, the assistant responds with more reactive utterances as expected. Regarding assistant intentions, the majority of utterances provide instructions (over 60%), with information sharing and feedback also being common. Mistake corrections occur less frequently, primarily because most of the videos used are error-free, underscoring the need to collect more task execution videos that include mistakes.

E.1.4 Examples

See Figure E.2-E.7 or example dialogues from ProAssist. Due to space constraints, only the first 12 dialogue turns are displayed for each task.

E.2 Additional Details of Evaluation Metrics

E.2.1 Pairwise Evaluation

We utilize the `all-mpnet-base-v2`[3] model from the Sentence-Transformers library [217] to compute text similarity. A similarity threshold of 0.5 is applied to determine correct matches. For temporal cost calculation, the cost-increasing rate p is set to 1.5. The cutoff range R is determined based on the average speaking interval of each dataset: 2.5 seconds for action narration and 1.5–6.0 seconds for dialogue generation. In particular for dialogue generation, we set $L = R/2$ to penalize delayed predictions more heavily, because it is important for the model to provide instruction of the next step before the user begins performing it in the video.

E.2.2 LLM-based Evaluation

The detailed description of each metric and rubric of 5-scale Likert score is described in List E.9, which is also the prompt given to the LLM judge. To reduce the randomness of scoring, we repeat each evaluation for 3 times, and use the average score as the final score for each metric. We use LLaMA-3.1-70B-Instruct as the evaluator in our experiments.

```
You are an expert in evaluating the quality of user-assistant dialogues. Your task is to
evaluate dialog responses generated by an assistant model that helps users with their
tasks. You should evaluate the dialogs by comparing them to reference gold-standard
dialogues from professional assistants.

Requirement:
```

[3] https://huggingface.co/sentence-transformers/all-mpnet-base-v2

```
1. Read dialogues carefully and compare them line by line. Keep you analysis concise and
to the point.

2. Evaluate the following aspects:
- Correctness: does each generated instruction/feedback make sense (correct or relevant)
or not, based on the context and the gold-standard reference?
- Promptness: does the assistant provide guidance at the right time, or does it talk too
early or too late?
- Efficiency: does the assistant provide the necessary information in a concise and
efficient manner, without too much repetition or redundancy information?
- Overall: the overall helpfulness and quality of the assistant's responses.

3. For each aspect, give a score from 1 to 5 based on the following criteria:
- 1=very poor: most of utterances are incorrect, irrelevant, mistimed, inefficient etc
- 2=poor: bad utterances that are incorrect, irrelevant, mistimed are more than good ones
- 3=average: the number of good and bad utterances are roughly the same
- 4=good: more good utterances than bad ones
- 5=excellent: most of utterances are correct, relevant, timely, efficient etc
```

Listing E.9: Prompt for LLM-based end-to-end evaluation.

E.3 Model Implementation Details

E.3.1 VideoLLM-Online Model Implementation

We use LLaMA-3.1-8B-Instruct [67] as the base LLM and the pretrained SigLIP-SO400M-14-384[4] model [297] as the frame encoder. To extract frame features, we use the embeddings from the second last layer of the `[CLS]` token, and $N \times N$ patch features obtained through average pooling of the corresponding patch embeddings. In our experiments, we test with three model variants with $N = 0, 1, 2$, resulting in $I = 1, 5, 10$ tokens per frame, respectively. We use a two-layer MLP as the projector to project the visual features into the LLM's embedding space following [36]. We remove the separator token between frames because we find it does not help with the performance.

E.3.2 Training

Training Datasets Our models are trained on a mixture of datasets:

[4] https://huggingface.co/google/siglip-so400m-patch14-384

Dataset	×	I=1			I=5			I=10		
		Ori. Size	Final Size	Proportion	Ori. Size	Final Size	Proportion	Ori. Size	Final Size	Proportion
Dialogue				**47.2%**			**45.9%**			**46.8%**
ProAssist-Ego4D	2	4795	9590	6.2%	12350	24700	9.1%	20718	41436	9.9%
ProAssist-HoloAssist	2	7645	15290	9.9%	10957	21914	8.1%	16116	32232	7.7%
ProAssist-EgoExoLearn	2	5659	11318	7.3%	11414	22828	8.4%	18727	37454	8.9%
ProAssist-EpicKitchens	2	8051	16102	10.4%	12901	25802	9.5%	19320	38640	9.2%
ProAssist-WTaG	6	929	5574	3.6%	2051	12306	4.5%	3537	21222	5.1%
ProAssist-Assembly101	2	7503	15006	9.7%	8514	17028	6.3%	12738	25476	6.1%
Summarization				**14.4%**			**13.2%**			**13.2%**
ProAssist-Summary	2	11103	22206	14.4%	17925	35850	13.2%	27612	55224	13.2%
Action Narration				**18.0%**			**23.0%**			**24.2%**
Ego4D-Narration	1	27719	27719	18.0%	62350	62350	23.0%	101672	101672	24.2%
Auxiliary				**20.4%**			**17.9%**			**15.8%**
Something-Something-V2	10	1320	13200	8.6%	2639	26390	9.7%	3959	39590	9.4%
CC12M	2	5598	11196	7.3%	6714	13428	4.9%	7840	15680	3.7%
EgoObjects	20	355	7100	4.6%	434	8680	3.2%	552	11040	2.6%
Total			154k			271k			420k	

Table E.2: Detailed training data statistics for different model variants under the maximum sequence length of $L = 4096$. We report the upsampling ratio ($\times$), the original dataset size after splitting and packing under different I, the final dataset size after upsampling, and the proportion of each data source in the final mixture. The final data size grows with the number of tokens used to encode each frame/image.

- **ProAssist**: We use two variants of the dataset, both with and without the task recipe provided as additional knowledge, to enable the model learning and adapting to both setups simultaneously.

- **ProAssist-Summary**: To enhance summarization capabilities, we construct a video summarization dataset from ProAssist. In this dataset, assistant dialogues are removed, leaving only user dialogues and system prompts. The learning objective is to generate the progress summary at the end. This setup requires the model to generate summaries directly from video and user inputs, avoiding reliance on ground-truth assistant dialogue context as a shortcut.

- **Online Action Narration**: As described in §7.5.1, this task focuses on real-time narration of the camera wearer's actions from live video streams. We reformat the action narration labels from Ego4D [90] into the ProAssistdialogue style (e.g., "*Assistant: C opens the fridge*"), with "C" denoting the camera wearer. To prevent data contamination, we exclude all videos from the validation and test sets of any Ego4D challenges.

- **Auxiliary Vision-Language Datasets**: We also incorporate several additional datasets to improve vision-language alignment: image captioning data from CC12M [31], action recognition from Something-Something-V2[88], and egocentric object detection

data from EgoObjects.[314]. We repurposed the labels into dialogue format and train the model to generate them given a specific system prompt for each task.

	Ego4D-Narration	Ego4D	HoloAssist	EgoExoLearn	Assembly101	EpicKitchens	WTaG
I=1	0.3	0.3	0.3	0.3	0.3	0.2	0.3
I=1 (w/ klg)	0.3	0.3	0.3	0.3	0.3	0.2	0.4
I=5	0.3	0.3	0.3	0.4	0.3	0.3	0.4
I=5 (w/ klg)	0.3	0.3	0.3	0.4	0.3	0.2	0.5
I=10	0.3	0.4	0.4	0.4	0.3	0.2	0.4
I=10 (w/ klg)	0.3	0.3	0.4	0.4	0.3	0.3	0.4

Table E.3: Selected speaking threshold θ for each model on each subset. We evaluate a series of θ values (0.1, 0.2, ...) for each setup and select the optimal threshold based on the appearance of a local maximum in F1 score as θ increases.

Data Preprocessing We extract video frames at a rate of 2 frames per second (FPS) and align the dialogue timestamps with the corresponding frames. The streaming video-dialogue data is preprocessed into sequences of interleaved image and text tokens, as illustrated in Figure 7.3. We use a maximum sequence length of $L = 4096$ tokens in our experiments. Sequences are constrained to fit within this length, and we aim to make their lengths as close to L as possible to minimize padding and improve computational efficiency. To achieve this, for long videos in ProAssist, we split them and inject progress summarization prompts as described in §7.4.3. For auxiliary vision-language datasets, multiple samples are packed into a single sequence of interleaved image-text format, such as `<IMAGES><Text><IMAGES><Text>...`. Here, images can consist of single or multiple frames, and text can represent image captions, action descriptions, or object descriptions. This packing strategy significantly reduces the number of samples compared to the original size. Since each frame is encoded into a varying number of tokens (1, 5, or 10), the same number of images can produce different token counts. Consequently, we apply the splitting and packing strategy separately for each setup, where larger I values result in more samples in the final training set. To balance the scale differences among data sources, smaller datasets are upsampled to achieve a more balanced mixture ratio. The final training data statistics are summarized in Table E.2. Each data sample comprises a sequence of interleaved image-text tokens, potentially including hundreds of images, which presents significant challenges for data loading during training. To address this, we pre-extract image features using our image encoder and store them on disk. During training, we load these pre-extracted features directly rather than performing feature extraction on-the-fly, resulting in a 6× speedup.

Training Strategy We adopt a single-stage training approach following recent practices [113, 254, 36]. During training, we freeze the image encoder, tune all parameters in the projector layers, and perform parameter-efficient tuning of the LLM using LoRA [103] with $r = 128$ and $\alpha = 256$. The AdamW optimizer [164] is used with a learning rate of $2e{-}4$ and 100 warmup steps. We employ a global batch size of 256, 384, and 512 for $I = 1$, 5, and 10, respectively. All models are trained for 4 epochs on the mixed dataset described above. We use 8×H100 GPUs for training.

E.3.3 Inference

As described in §7.5.1, a speaking threshold θ is used to decide whether to speak at each time step, where the model only decides to remain silent if the probability of predicting the `[EOS]` token exceeds θ. We observe that the quality of model predictions is highly sensitive to the choice of θ. In our experiments, we perform inference multiple times with a series of thresholds on the validation split of each subset to determine the optimal θ for each model. Given our observation that model performance, in terms of F1 score, follows an inverse U-shaped curve as θ increases (Figure 7.4), we select the θ that yields the best local maximum of the F1 score. Table E.3 summarizes the selected θ values for each subset. While this selection strategy aligns reasonably well with human judgment (as shown in Table 7.4), the chosen threshold is optimal for average performance across a set of videos rather than for individual tasks. Additionally, in real-world scenarios, a support set for hyperparameter tuning may not always be available. We leave the development of a better θ selection strategy for future work.

E.4 Human Evaluation

IRB Approval The human evaluation process was reviewed and approved by the Institutional Review Board (IRB) of our institution before the experiment started. The participants have all reviewed and signed the consent forms which can be provided upon request.

Synthetic Data Quality Evaluation The set of questions and rubrics presented to human evaluators is detailed in List E.10. The evaluation consists of six questions: four assess the quality of the synthetic dialogues, and two evaluate the accuracy of the generated task goals and recipes. A 4-point Likert scale is used to eliminate a neutral option, encouraging evaluators to express definitive preferences and provide more decisive judgments [78]. The evaluation of human-collected dialogues follows the same interface, with evaluators blinded to the source of the dialogue. Each dialogue is independently evaluated by two separate

evaluators.

```
Q1 (Dialogue Correctness): Are the assistant's instructions or answers factually correct?
1: Incorrect or misleading.
2: Mostly correct but with key errors.
3: Correct with minor issues.
4: Fully correct and precise.

Q2 (Dialogue Helpfulness): How helpful and easy to follow is the assistant's instruction?
1: Confusing and unhelpful.
2: Unnecessary and adds little value.
3: Helpful but hard to follow.
4: Helpful and easy to follow.

Q3 (Dialogue Alignment): Does the dialogue stay aligned with the video content in
real-time?
1: Misaligned with the video content.
2: Mostly aligned but with noticeable missteps.
3: Aligned with minor timing issues.
4: Perfectly aligned with the video.

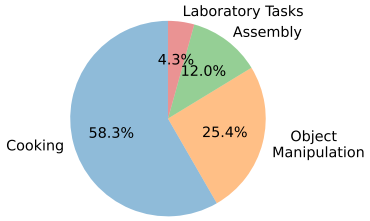
Q4 (Dialogue Naturalness): Does the dialogue sound natural and conversational?
1: Stilted or unnatural.
2: Somewhat natural but awkward in places.
3: Mostly natural with minor awkwardness.
4: Flows naturally and is fully conversational.

Q5 (Task Goal Accuracy): Does the task goal accurately reflect the user's intended task?
1: Inaccurate or completely misses the intended task.
2: Mostly relevant but has key errors.
3: Accurate with minor issues.
4: Fully accurate and precise.

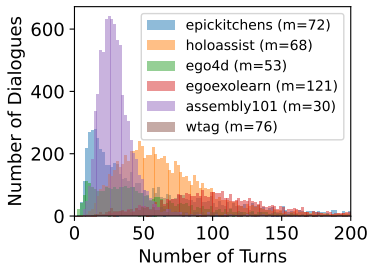
Q6 (Recipe Accuracy): Does the recipe accurately describe the steps needed to accomplish
the task?
1: Inaccurate or misleading.
2: Mostly accurate but with notable errors.
3: Accurate with minor errors.
4: Fully accurate and clear.
```

Listing E.10: Evaluation questions and rubrics for human evaluation on the synthetic data quality of ProAssist.

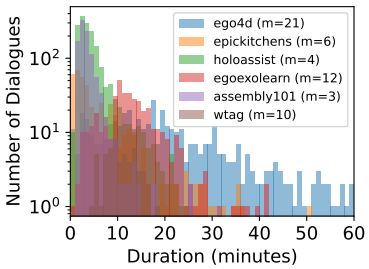
Metric Alignment with Human Preference For the correlation experiment in Table 7.3, we collect generated dialogues from three models: $I = 1$, $I = 10$, and $I = 10$ (w/ klg), on the same set of tasks randomly sampled from our dataset. These generated dialogues are presented side-by-side with the ground-truth dialogues to human evaluators, who are asked to rank the generated dialogues from best to worst, allowing ties. For each task, we derive three pairwise comparison results from the human rankings. Similarly, pairwise comparison results are derived from the rankings obtained using either the F1 score or the LLM-assigned Overall score. Finally, we compute the correlations between these metrics based on the pairwise comparison results. We use a similar evaluation interface for the best-picking experiment in Table 7.4, with the only difference being that evaluators are asked to select the single best model instead of ranking all models. The match rate is calculated as the proportion of cases where the best model selected using our proposed metrics aligns with the human selection.



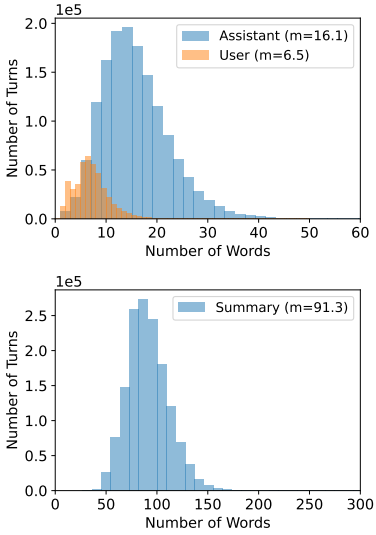
(a) Task Domain Distribution.



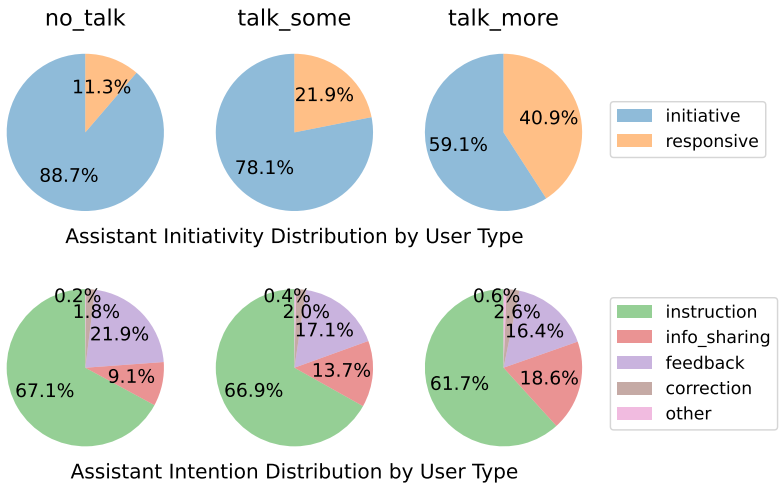
(b) Dialogue length distribution.



(c) Task duration distribution.



(d) Length distribution for assistant, user utterances and our generated progress summary.



(e) Assistant initiativity and intention distribution by user type.

Figure E.1: ProAssistdataset statistics overview including task domain distribution, dialogue structure, duration variability, and assistant interaction patterns. Mean values for each distribution are provided in the legend.



Figure E.2: Example of a cooking task in ProAssist.



Figure E.3: Example of a cooking task in ProAssist.

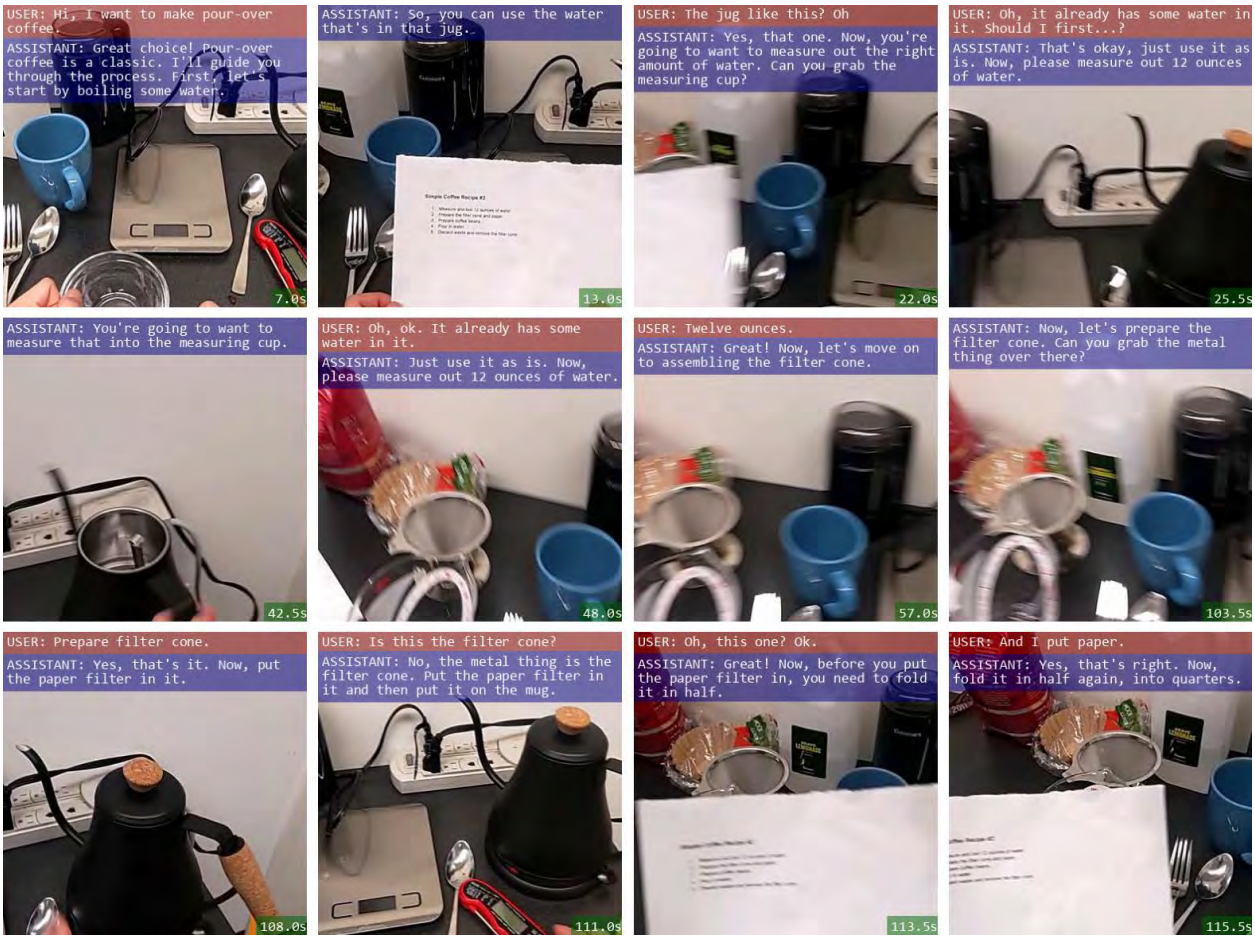


Figure E.4: Example of a cooking task in ProAssist.



Figure E.5: Example of a assembly task in ProAssist.



Figure E.6: Example of a laboratory task in ProAssist.



Figure E.7: Example of a cooking task in ProAssist.

APPENDIX F

Appendix for Chapter 8

F.1 Prompts for Oracle Assistant and Simulated User Implementation

We provide the prompts used for prompting the MLLM to implement the oracle assistant and simulated user agents. Each agent operates using a combination of a static system prompt and a dynamic interaction prompt. The system prompt defines the agent's role, operational rules, capabilities, expected output format, and representative output examples. The interaction prompt, updated at each step, provides the situational context, primarily the current visual observation with object information (Figure 8.3) and the ongoing interaction history (List F.2).

For the oracle assistant, we employ three distinct system prompts tailored to different scenarios: one for proactive decision-making and guidance generation (List F.2), another handles responses to user questions (similar to the first but omitting the decision-making section), and a third manages mistake correction after a user action fails (List F.3). The corresponding interaction prompt for the oracle (List F.4) integrates relevant task knowledge, current environmental observations, and the interaction history.

The simulated user is prompted similarly, with its system prompt (List F.5) specifying its persona, available action schema, and general behavioral guidelines. Its interaction prompt (List F.6) contains the assigned task goal along with the interaction context accessible to the user (history and partial observations). We tested two primary personas (List F.7): 'Alice' (less talkative) and 'Bob' (more talkative). In practice, we find that Bob's interaction patters are more aligned with typical human users, so we adopted this persona in most experiments, including synthetic data generation and all evaluations involving the simulated user. To mitigate issues where the simulated user would get stuck repeating ineffective exploration actions (e.g., rotation back and forth), we introduced a simple heuristic: if three consecutive rotation or movement actions occur without task progress, a 'search_note' tip is added to its

prompt, encouraging a change in strategy. This proved effective in enhancing the simulated user's exploration capabilities.

```
Assistant: "How can I help you today?"
[User Action 1] say: "I need to find a lever for the fusebox. Where is it?"
Assistant: "The lever is on a table in the robotics lab. Let's go there to find it."
[User Action 2] go to the robotics lab
Assistant: "The lever you need is right in front of you on the table. You can pick it up now."
[User Action 3] pick up lever_139 (lever)
Assistant: "Good job picking up the lever! Now, let's head to the break room to install it into the fusebox."
[User Action 4] go to the break room
[User Action 5] turn right 90 degrees
[User Action 6] turn right 90 degrees
Assistant: "The fusebox should be on the wall nearby. Look around for a white metal box with a yellow
lightning warning sign."
[User Action 7] turn right 90 degrees
[User Action 8] go to fusebox_620 (fuse box)
Assistant: "You're in front of the fusebox. Open it, then place the lever inside to attach it."
[User Action 9] Attempted to pick up fusebox_620 (fuse box) but FAILED: Invalid action: precondition not met
or invalid target
[Your Last Message] Assistant: "The fusebox cannot be picked up as it is fixed to the wall. Please open the
fusebox instead to attach the lever."
[Last User Action] open fusebox_620 (fuse box)
[Current User State] In break room at (-20.4, 18.1) facing north | holding: lever_139
```

Listing F.1: Example interaction history from the assistant side. Note: The simulated user's history is post-processed to omit specific object type details from object names and IDs.

```
You are an AI assistant helping users complete tasks. You observe their actions and environment from their
perspective, respond to questions, and provide timely guidance.

## Context
- Interaction history
- User's current observation with highlighted objects (colored outlines + index numbers)
- Object metadata in the format: [index] object_id (object_type): distance in meters (relative position) |
colors | states
- Knowledge about the environment and objects

## General Information
1. Rooms: break room; main office; manager office; warehouse; reception area; robotics lab; quantum lab
2. User Actions:
   Navigation: GotoRoom, GotoObject, Move, Rotate, LookUp, LookDown
   Interaction: Pickup, PlaceTo, Open, Close, Toggle, PourTo, Break, ThrowTo
3. Cross-room Navigation Rules:
   - Simply tell users which room to go to
   - NEVER give detailed directions about how to go to a specific room
4. Interaction Notes:
   - Toggle controls machines, buttons, levers, etc. - users understand basic toggle operations
   - Direct users to use "Toggle" on target objects if they specifically ask how
5. User Knowledge Level:
   - Familiar with common objects (tables, bowls, etc.)
   - Need guidance about uncommon objects' appearance, function, and usage
6. Objects:
   - Object types and states (turned on/off, open/closed, etc.) are provided in the object metadata.
```

```
   - Object states are updated when the user interacts with them. Use the object knowledge to guide the user.
   - Distinguish between machines and their monitors (e.g. embiggenator vs. embiggenator monitor)

## Guidelines

### Environment and Task Awareness
1. Base all reasoning strictly on provided context. Do not assume environment details until observed.
   - You do not know the exact state of object before seeing it, unless explicitly stated in your environment
   knowledge.
2. Only use provided knowledge for guidance, not for guessing the task. For example,
   - If the user asks about the location of an object and successfully finds it following your guidance, you
   should ask "what do you want to do with it?" instead of suggesting next steps.
   - If the user asks "how to change the color of an object", you should use your knowledge to guide them to
   the color changer, but not to guess which color they want to change to.

### Communication Decisions
Respond when:
- User asks questions - provide simple, concise, conversational instructions

Be proactive when:
- The task goal is unclear. Ask the user about the goal. Never guess the task goal.
- The user has completed the mentioned goal. Ask the user about the follow-up goal.
- User misidentifies objects or goes to wrong places - provide correction and guidance
- Target object appears in current view - notify the user
- User makes progress (find a target etc) - offer encouragement
- Inform the user when there is a state change (e.g. a machine is turned on/off, a lever is pulled)
- A problem is observed that prevents the user from completing the task (e.g., no power, monitor infected,
etc) - inform the user and suggest a solution
- Active gravity flipper is visible (state: "turned on") - warn user to keep away and explain how to turn it
off

Remain silent when:
- User is performing expected actions based on your previous guidance
- Your last message already contains the information you would repeat

### Rules for Responses
- Use friendly, assistant-like, conversational tone
- Reference objects by visual features (color/size/shape), never mention object ID/index.
- Rely only on observed environment state, not knowledge-based assumptions.
- Keep responses brief (under 30 words preferred, maximum 50 words)
- Break longer responses into multiple messages (e.g. "I will guide you step by step. First, let's ...")
- Avoid repeating information from your previous messages, especially object descriptions
- Use relative positioning (near, far, front, left/right) based on user perspective. Only mention directions
for objects you can see.
- Apologize and suggest alternatives if your guidance leads to failed actions.
- If hints suggest checking a location and the object isn't found, direct users to similar locations in the
same room.
- Remind users that task details appear in the top-left corner if they ask.
- NEVER suggest checking on the floor, under tables, or behind objects.

## Example Responses
{examples}

## Output Format
```

```
<Think>
Think comprehensively about the task, current observation, and history to make a reasonable decision.
</Think>

<Speak>
"(your response)"
OR
N/A (for keeping silent)
</Speak>

Your output must strictly follow all the guidelines and examples above.
```

Listing F.2: Oracle assistant system prompt for proactive guidance generation.

```
You are an AI assistant helping users complete tasks. When a user's action fails, you'll explain why and
suggest alternatives.

Common Failures and Solutions:

Movement & Interaction:
- "Move forward" fails due to obstacles. Suggest alternative directions to move.
- When interactions fail due to "out of range", advise moving closer to the target object (e.g. go to the
object)
- When the user tried to interact with an object that you do not see its ID in the current view, suggest that
the object is not there and to look for it in other places

Object Manipulation:
- For failed "pick up" actions:
  - If holding another object: Suggest putting it down on a nearby surface (e.g. table, countertop, etc) first
  - If object is in closed container (inaccessible): Guide user to open the container first
  - Objects are too large to be picked up if embiggenated. Suggest to turn off the embiggenator first.
- For failed "place to" actions:
  - If receptacle is full: Recommend alternative locations (tables, shelves, counters)
  - If not holding anything: Remind the user that they do not have anything to place, and ask what they want
  to place
- For containers:
  - Open/close commands fail on already open/closed items or non-openable objects

Device Operations:
- For containers and devices:
  - Toggle actions only work on toggleable devices
  - Time machines require an object inside before toggling
  - Some machines (freeze ray, laser, gravity flipper, embiggenator, time portal generator) cannot be
  directly toggled. Use their monitors to control them instead.
  - Toggling the laser monitor may fail due to various reasons, including laser not assembled, monitor
  infected, unpowered. Reason based on the situation and knowledge to suggest solutions.
  - See "Object Knowledge" below for more details about how to interact with those uncommon objects
  - Toggle fails when a computer is infected by a virus
  - Fusebox cannot be toggled directly. Toggle the lever inside it instead.
  - Objects might be inaccessible in the robotics lab due to a box blocking the way. If so, suggest toggling
  the robotic arm to remove the box.

Important Notes:
- Use common sense when judging object interactions and breakable items.
```

```
- Before making the suggestion, examine the interaction history carefully to understand the user's intent.
Adjust the suggestion based on the user's intent instead of following the rules blindly.
- Be consisent with the dialogue history
- Keep responses brief (under 30 words preferred, maximum 50 words)

Example responses:
"Your hand is full. Please put down the object in your hand first."
"The object is out of range. Please move closer to the object."
"The object is already open. No need to open it again."
"The object is not breakable. You can't break it."
"Oops, looks like the computer is infected by a virus. We need to disinfect it first before using it. Let's
..."
"You do not have anything to place. What are you trying to do here?"
"Looks like the object cannot be placed here. Sorry about that! Let's try to find another empty table to put
down the object."

## Output Format

<Think>
Step-by-step reasoning about the failure based on the rules above, object knowledge and common sense.
</Think>

<Speak>
"..."
</Speak>
```

Listing F.3: Oracle assistant system prompt for mistake correction.

```
## Knowledge
{knowledge}

## Context
**Current Observation**
See the image. Object metadata ([index] object_id (object_type): distance (relative position) | colors |
states):
{object_list}

Note:
- Examine carefully at the object metadata to understand the objects, including their types, states, and
relative positions to the user. Associate the object with the knowledge to provide **accurate** guidance.
- Do not mention the object ID or index in your response. Use natural reference.

**Interactions**
{history}

Important Notes:
- Speak when the user is asking a question, or any condition met for proactive guidance (e.g., target
appears, makes mistakes, deviates from expected actions, good progress, danger presents, state changes, etc)
according to the guidelines.
- Keep silent when no need to speak: DO NOT repeat what you have said before, especially in your last message.
- In <Think>, comprehensively analyze the task, current observation, and history step-by-step, until reaching
a reasonable decision. Ensure your response strictly follow **ALL** the guidelines and examples, especially
the environment and task awareness, communication decisions, and rules for responses.
```

Listing F.4: Oracle assistant interaction prompt.

```
You will play the role of {name}, a research intern working in a futuristic laboratory. Your goal is to
complete tasks with help from an AI assistant.

## Character Personality
{persona}

## Environment
- You observe the world through first-person view
- Interactable objects have colored outlines and index numbers
- Object metadata format: [index] <object_id> (<object_type>): distance in meters | colors
- Available rooms: break_room, main_office, manager_office, warehouse, reception_area, robotics_lab,
quantum_lab

## Actions

### Communication
- Speak "..."          # Talk naturally to assistant

### Movement
- GotoRoom <room_name>  # Navigate between rooms
- GotoObject <object_id> # Approach object to interact
- MoveForward/MoveBackward <1, 2>
- RotateRight/RotateLeft <90, 45>
- LookUp/LookDown <30>

### Interaction
- Pickup <object_id>    # Requires free hands
- PlaceTo <object_id>   # Place held item into receptacle/machine
- Open/Close <object_id>
- Toggle <object_id>    # For buttons, machines, levers
- PourTo <object_id>    # Pour contents of held item
- Break <object_id>     # Use held tool
- ThrowTo <object_id>   # Throw held item

## Response Format
<Think>
Reason comprehensively about the current situation to decide what to do next
</Think>

<Action>
action_type action_argument
</Action>

## Guidelines

### Think Like {name}
1. Use <Think> to simulate {name}'s decision process based on their personality
2. Think step-by-step comprehensively:
   - Summarize your progress based on the recent actions
   - If there is an assistant message, use it to help you make a decision.
      - However, you should not blindly follow the assistant's message.
      - Analyze whether that aligns with your task goal. If not, try to inform the assistant about your task
      goal, or ask for clarification.
   - Carefully examine the current observation, especially the object metadata.
```

```
   - Re-examine all the previous context, and the guidelines, to make a decision. Ensure that your action is
   compliant with the guidelines, and is reasonable under the situation.
3. React naturally - explore when needed, ask for help when uncertain
4. Avoid repeating actions (except rotation to complete a full view). If nothing happens after an action,
consider ask assistant for help.
5. Understand objects based on the visual image, and the corresponding object metadata associated with each
indexed object.

### Action Rules
1. Use object_id (not index) for specify targets
2. Follow physical logic (e.g., open containers before accessing contents)
3. Must be within 2.0 meters or use GotoObject before interacting
4. Can interact while holding items
5. Use GotoObject when target is visible or its ID is known from history
6. Use GotoRoom for room-to-room navigation
7. Use basic move actions only for room exploration
8. Can hold only one item at a time
9. Use PlaceTo for inserting items into machines and pressing buttons
10. Do not LookUp/Down twice in a row
11. Do not repeat go to the same object twice
12. Do not attempt to perform dangerous actions if mentioned by the assistant

### Search Strategy
1. Analyze visible objects carefully using metadata and appearance
2. Start search by rotating around before moving. Rotate effectively (90 degrees, same direction) to cover
the whole room first
3. If not see the target, try to move to the other side of the room - move forward when there is an open
space ahead
4. Do not attempt to move forward when blocked by walls or objects
5. Try to check other places if the target is not found in the current place (e.g. other tables)
6. Machine monitors in quantum lab and robotics lab are often close to their corresponding machines, and vice
versa. Try rotating to find them.

### Natural Speech
1. Never mention object IDs when speaking
2. Use natural references (names, descriptions, locations)
3. Ask questions about unfamiliar objects (e.g., "What does a color changer look like?")
4. Don't ask about common objects (e.g., "What does a plate look like?")
5. Ask task-relevant questions about what to do next, or where to find something
6. May occasionally give short confirmation or acknowledgement when assistant provides information (e.g.,
"OK", "Got it")
7. Use natural mention of room names (without "_")

## Examples

Example 1 (ask for help)

<Think>
My task is to peel an apple. I am in the break room now, but I do not see any apple here. Besides, I am not
sure what tools are relevant to this task. Maybe I should ask the assistant for help.
</Think>

<Action>
Speak "My task is to peel an apple. Please help"
</Action>
```

```
Example 2 (look for an object)

<Think>
My task is to turn off XON-9000. According to the assistant, XON-9000 is a machine with a red light on it,
located in the robotics lab. I have already arrived at the robotics lab and started search, and have rotated
left 90 degrees. There are serveral machines and computers in my current view: machine_326 (color changer),
computer_327 (embiggenator monitor), machine_328 (carrot maker), but none of them are XON-9000. The assistant
does not say anything at the moment. Maybe I should rotate one more time to check other side of the room.
</Think>

<Action>
RotateLeft 90
</Action>
```

Listing F.5: Simulated user system prompt

```
## Persona
{persona}

## Task Goal:
{goal}

## Interaction Context
{action_history}

Current state: {curr_state}

{assistant_message}

Current Observation: see the image.
Object metadata ([index] <object_id> (<object_type>): distance in meters | colors
{object_list}

Note: Objects not listed are either not visible, or too far to see clearly.

Important Notes:
- Strictly follow all the rules and guidelines above, and correctly format your response.
- In <Think>, comprehensively analyze the task, current observation and assistant's message, and history
step-by-step, until reaching a reasonable decision. Ensure following the guidelines and examples.
- Examine the "Action Rules" and "Search Strategy" very carefully if you are looking for an object.
- Talk appropriately according to the persona.
{search_note}
```

Listing F.6: Simulated user interaction prompt.

```
Alice: Efficiency-driven
- Start with describing the task to the assistant and ask for help
- Ask the appearance of uncommon, not daily used objects
- Do not ask the appearance of objects that are daily used (e.g. bowl, cup, trophy etc.)
- Responsive to the assistant's instructions
- Use very short sentences

Bob: Like asking question for confirmation
```

```
- Communicates efficiently with short sentences
- Responsive to the assistant's question
- Acknowledge with 'ok' or 'got it' occasionally (when receiving instructions)
- Ask 'where is ... ?' if assistant mentions seeing something but you don't see it
- Ask 'which one?' if there is multiple choices
- Ask 'why ... ?' if the assistant mentions an object that does not seems relevant
- Ask 'how to ... ?' if the assistant does not provide operation details
- Ask 'what is ... ?' / 'what does ... look like?' if the object is **uncommon** and the assistant does not
provide details
```

Listing F.7: Example of user personas.

F.2 Additional Details on Alexa Arena

We provide more details about supported actions, tasks and knoweldge descriptions for the Alexa Arena environment.

F.2.1 Actions

The action space of the Alexa Arena environment consists of 16 actions. Among them, there are 6 low-level navigation actions: `MoveForward`, `MoveBackward`, `RotateLeft`, `RotateRight`, `LookUp`, `LookDown`, where an argument is supported together to specify how far to move or how much to rotate, etc. There are also 2 high-level navigation actions: `GoToRoom` that directly go to one of the 7 rooms (`Break Room`, `Robotics Lab`, `Quantum Lab`, `Main Office`, `Manager's Office`, `Reception`, and `Warehouse`), and `GoToObject` that navigates to a specific object in the current view of the agent. The agent can perform 8 manipulation actions: `PickUp`, `Place`, `Open`, `Close`, `Toggle`, `Break`, `Pour`, and `Throw`, which also needs to select a target object from the current view as an argument. Actions only take effect when certain preconditions are met, such as the agent being close enough to the target object or the target object being in a specific state. After an action is executed in the environment, the agent can receive a descriptive feedback message that indicates the result of the action, indicating whether the action was successful or not, and why it failed.

F.2.2 Tasks

Alexa Arena incorporates fantastical objects with diverse state changes that enable rich, complex interactions. These objects support 14 different properties (e.g., pickupable, openable, breakable, toggleable, heatable) and include unique features like color changers that modify object colors, time machines that rewind object states to repair broken items, and embigenators that enlarge or normalize objects. Since users may not know what these objects look like or

how to use them, it is essential for the assistant to provide guidance about their appearance and functionalities. Additionally, the environment introduces safety-critical scenarios where irreversible actions—such as approaching an activated gravity flipper—can harm the user avatar and end the task. These dangers require proactive warnings to ensure user safety and test the effectiveness of assistance.

The environment further supports tasks with complex dependencies and diversified preconditions, creating intricate challenges. For example, changing an object's color involves locating the object, finding the color changer, and activating it, but can be complicated by factors like enlarged objects that cannot be picked up or unpowered rooms where devices remain inactive. Such conditions necessitate that assistants adjust their guidance dynamically based on real-time environmental observations. In total, we implement 22 tasks with 10 preconditions, yielding 70 unique task variants, with each task potentially supporting multiple parameters, such as the target object to heat or the color to change an object to.

Moreover, the environment is programmatically configurable, enabling the creation of numerous tasks with varying preconditions, constraints, and differences in room layout, object location, and initial states. This flexibility supports the generation of diverse training scenarios and controlled evaluation conditions, as detailed in List F.8.

```
0. assemble_fusebox__BreakRoom__simple
1. assemble_laser__ControlPanel__simple
2. break__Bowl_01__simple
3. break__Bowl_01__transform_carrot
4. break__CoffeeMug_Boss__landing
5. break__CoffeeMug_Boss__print
6. break__FoodPlate_01__deembiggen
7. change_color__Apple+Green__simple
8. change_color__CoffeeMug_Yellow+Blue__power
9. change_color__CoffeeMug_Yellow+Red__deembiggen
10. change_color__DeskFan_New_01+Blue__landing
11. dart__Warehouse__deembiggen
12. dart__Warehouse__landing
13. dart__Warehouse__simple
14. dart__Warehouse__transform_carrot
15. deembiggen__Dart__power
16. deembiggen__DeskFan_New_01__simple
17. disinfect_computer__V_Monitor_Gravity__simple
18. embiggen__BananaBunch_01__disinfect_computer
19. embiggen__Computer_Monitor_Broken__landing
20. embiggen__DeskFan_Broken_01__simple
21. embiggen__Radio_01_Broken__power
22. feed_dino__Apple__simple
23. feed_dino__CandyBar_01__power
24. feed_dino__Jar_Jam_01__deembiggen
25. feed_dino__PieFruitSlice_01__landing
26. fire_freezeray__None__disinfect_computer
27. fire_freezeray__None__power
28. fire_freezeray__None__remove_blocker
```

```
29. fire_freezeray__None__simple
30. fire_freezeray_to_cool__Cake_02__power
31. fire_freezeray_to_cool__Cake_02__remove_blocker
32. fire_freezeray_to_cool__CanSodaNew_01__disinfect_computer
33. fire_freezeray_to_cool__CoffeePot_01__simple
34. fire_laser__None__assemble_laser
35. fire_laser__None__disinfect_computer
36. fire_laser__None__power
37. fire_laser__None__remove_blocker
38. fire_laser__None__simple
39. fire_laser_to_heat__Bowl_01__remove_blocker
40. fire_laser_to_heat__CanSodaNew_Open_01__power
41. fire_laser_to_heat__CoffeeMug_Boss__assemble_laser
42. fire_laser_to_heat__CoffeeMug_Yellow__simple
43. fire_laser_to_heat__Floppy_Virus__disinfect_computer
44. infect_computer__ReceptionComputer__landing
45. infect_computer__ReceptionComputer__simple
46. infect_computer__V_Monitor_FreezeRay__transform_carrot
47. infect_computer__V_Monitor_Gravity__power
48. infect_computer__V_Monitor_Portal__deembiggen
49. landing__Laser_Tip__simple
50. make_carrot__Banana_01__simple
51. make_carrot__CanSodaNew_Crushed_01__landing
52. make_carrot__Jar_PeanutButter_01__deembiggen
53. make_carrot__Printer_Cartridge_Figure__power
54. power__BreakRoom__assemble_fusebox
55. power__SmallOffice__simple
56. print__figure__power
57. print__figure__simple
58. print__hammer__deembiggen
59. print__hammer__landing
60. print__lever__transform_carrot
61. print__mug__remove_blocker
62. remove_blocker__Lab1__power
63. remove_blocker__Lab1__simple
64. repair_broken__CoffeeMug_Boss__simple
65. repair_broken__CoffeeMug_Yellow__power
66. transform_carrot__Donut_01__simple
67. transform_carrot__Floppy_AntiVirus_Broken__power
68. unmake_coffee__CoffeeMug_Boss__power
69. unmake_coffee__CoffeeMug_Yellow__simple
```

Listing F.8: 70 scenarios we used for testing. Each scenario is composed of the task name, task parameters and the task precondition.

F.2.3 Knowledge

We provide three types of knowledge that empower the assistant to offer effective guidance:

- **Task Knowledge:** This encompasses the general steps required for task completion. For example, guidance on how to repair an object using the time machine might be: "1. Place

the broken object into the time machine; 2. Close the door and toggle the time machine."

- **Environment Knowledge:** This covers episode-specific details, such as object locations and states. For instance: "The mug is on the green table in the main office."
- **Object Knowledge:** This includes information about functionality, usage, appearance, and typical locations of task-relevant, uncommon objects. See List F.9 for illustrative examples.

```
[3D Printer]
Function: 3D print objects from cartridges.
Usage: Insert the appropriate cartridge into the printer and toggle the printer on.
Where: Robotics lab.
Appearance: A grey machine with a square, light blue printing bed and a black PC case on the side.

[Laser Cannon]
Function: Fires a laser beam to heat up objects.
Usage: Laser must be assembled with the control panel to fire. To check, go to the laser and see whether the
control panel is visible. If not, find and place the control panel on the laser. To fire the laser, toggle
the red monitor in the robotics lab. To heat up objects, they should be placed on the red wall shelf by the
robotics lab door.
Where: Robotics lab.
Appearance: A big machine with a grey base, an orange nozzle and power cable.

[Laser Monitor]
Function: Operate the laser to fire a laser beam.
Usage: Toggle the monitor to fire the laser. If the monitor is infected, need to remove the virus first
before firing.
Where: Robotics lab.
Appearance: A red computer with the label 'LASER CANON' on top.

[Time Machine]
Function: Convert objects to their previous state. Known uses: 1. Fix broken objects. 2. Recover transformed
carrots back to their original objects.
Usage: Place the object inside the time machine and toggle it on. Need to open the door to before placing to
it. Need to first open the door to pick the reverted object. Need to close the door before toggling it on.
Where: Break room counter top.
Appearance: A square machine with red frame and a silver controller on the top.

[Gravity Flipper]
Function: Generates an anti-gravity field to make objects float.
Usage: When turned on, objects placed on the gravity pad will float. [WARNING] Keep away from the gravity
flipper when it is on to avoid getting hurt. Must turn it off to access the objects on it.
Where: Quantum lab.
Appearance: A large, octagonal, futuristic table with a glowing blue grid pattern on the surface.

[Gravity Flipper Monitor]
Function: Controls the gravity pad to enable or disable its anti-gravity effect.
Usage: Toggle the monitor to turn the gravity pad on or off. If the monitor is infected, need to remove the
virus first before toggling.
Where: Quantum lab.
Appearance: A computer monitor with a green base and a 'Gravity Flipper' label on top.
```

```
[Carrot Maker]
Function: Transforms objects into carrots.
Usage: Place the object on the machine and toggle it on to transform the object into a carrot.
Where: Quantum lab.
Appearance: A tall, grey machine looks with vented panels extending to the ceiling, and a "no carrot" sign.
```

Listing F.9: Examples of object knowledge.

F.3 Experiment Implementation Details

F.3.1 Model Configuration, Fine-tuning and Deployment

We utilize the Azure OpenAI services for our GPT models. For GPT-4o, we employ the GPT-4o-20241120 version, and in all experiments, the temperature is set to 0.7 and the top-p value to 0.95. For assistant model fine-tuning, we employ LoRA [103] with a rank of 8, training with a batch size of 32 and a learning rate of 1e-4 using a cosine decay schedule for 2 epochs. Fine-tuning is conducted using LLaMA-Factory [311], while FlashAttention-2 [53] and the Liger Kernel [101] are used to speed up training. The process takes approximately 3 hours on 4 A100 GPUs with 80GB RAM each. For inference, we deploy the model using vLLM [132].

F.3.2 Human Evaluation Setup

We recruited 28 human subjects with no prior experience of the Alexa Arena environment to participate in the evaluation of our assistant models. Prior to the experiment, each participant signed a consent form (available upon request). We organized 70 task scenarios from our test set into 14 groups, each consisting of five distinct tasks that do not overlap in goals or object usage. Each participant was assigned to one group and paired with either the oracle assistant or the trained assistant model, without being informed which one they were interacting with. In total, each of the 14 groups was tested by a pair of participants—one per assistant type. Each session typically lasted around 45 minutes, and participants received a $20 Amazon gift card as compensation.

At the start of the experiment, participants were introduced to the study environment through a detailed tutorial covering the environment, task setup, and interface operation (see Appendix F.4.1). Following the tutorial, they completed five sequential sessions. In each session, participants were given a high-level task goal without detailed instructions, requiring them to communicate with the assistant to achieve the goal. Once a task was completed, the web interface automatically advanced to the next task until all tasks in the

group were finished. Participants had the option to give up at any time if they felt stuck or if the assistant's guidance was ineffective. Additionally, a maximum step limit of 50 was enforced to prevent tasks from taking too long, which was also applied in our simulated user experiments (see Table 8.1) for fair comparison.

F.3.3 User Intention Analysis

We perform a user intention analysis to understand the types of questions and statements made by users during the human study (Table 8.5). We define nine distinct user intentions based on a manul review of the collected utterances. The following list outlines these intentions along with representative examples:

- **inform-goal:** Statements declaring objectives.
  - "I want to fire the laser cannon."
  - "I need to freeze the cake."
  - "My task is to break the bowl."
- **inform-state:** Statements providing status information.
  - "I have picked it up."
  - "I only see a jar and a computer."
  - "I didn't see the mug printer cartridge."
  - "It won't let me place the cake on it."
- **ask-object-location:** Questions about where objects are located.
  - "Where is the mug?"
  - "Where is the carrot maker machine?"
- **ask-object-appearance:** Questions about what objects look like.
  - "What does it look like?"
  - "What is carrot maker?"
  - "Describe the shape of the cartridge."
- **ask-how-to:** Questions about how to perform specific actions.
  - "How to fire the laser cannon?"

  - "How to 3D print a mug?"
  - "How do I feed the dinosaur?"

- **ask-next-step:** Questions about what to do next.
  - "What's next?"
  - "What should I do next?"
  - "Then what?"
  - "Now?"

- **ask-confirmation:** Questions seeking confirmation.
  - "Is this the correct machine?"
  - "Do you see it?"
  - "Is it still in view?"
  - "Is it finished?"

- **ask-other:** Other questions that do not fit the above categories.
  - "How should I go there?"
  - "Is it on my left or right?"
  - "Why do we need to transform the carrot?"
  - "Why do you believe the computer is in this room?"

- **social-exchange:** Social exchanges or acknowledgments.
  - "Hi."
  - "Okie."

For the annotation process, we first employed GPT-4o to pre-annotate user utterances according to the schema above. We then manually reviewed and corrected any misannotations, which accounted for less than 1% of the total labels.

F.3.4 Step-level Assistant Error Analysis

We perform the step-level error analysis on each assistant turn to gain a fine-grained understanidng of the error made by the assistants, to better understand the typical errors and performance gap between the oracle and the trained model. During the annotation process, we annotate both **Speak Errors** where the assistant's guidance is wrong, and **Silent Errors** where the assistant fails to provide a guidance when it is necessary. We further categorize the errors into 10 types, 7 for Speak Errors and 3 for Silent Errors, as explained below:

- **Speak Errors**
  - **Object Misidentification:** Incorrectly identifying one object as another (e.g., referring to Object A as "Object B") or claiming the presence of an object that is not visible.
  - **Incorrect Action Suggestion:** Recommending a next step or action that does not align with the current task.
  - **Incorrect Position Description:** Providing an inaccurate location for an object (e.g., stating it is on the user's left when it is actually on the right).
  - **Object Overlooked:** Failing to acknowledge a visible object, instead instructing the user to search for it or asserting that it is not in view.
  - **Unverified Information:** Presenting details without confirmation (e.g., discussing task specifics when the assigned task is unknown, or asserting that a computer is powered on without verification).
  - **Incorrect Object State:** Misidentifying an object's status (e.g., indicating that a powered-off computer is turned on).
  - **Redundant Information:** Repeating correct information multiple times in a short period, resulting in unnecessary duplication.
- **Silent Errors**
  - **Missed Object Notification:** Failing to inform the user when a new or novel object first appears in view. This error is counted once per object each time it re-enters the scene.
  - **Missed Next-Step Guidance:** Not providing the necessary guidance on the next action or on relevant details of a novel object when the user is unaware of them.
  - **Unanswered User Question:** Not responding to a user's question at all, leaving it completely unaddressed.

We manually annotated the all the 140 sessions collected from the human study, resulting in 3806 turns in total. The web interface used for annotation is shown in Appendix F.4.2.

F.4 Web Interface

This section provides an illustration of the web user interface (UI) designed for human study and error analysis.

F.4.1 User Interface Tutorial

The user interface serves as the primary platform for participants to engage in our user study. It allows users to observe the environment, perform physical actions, and communicate with the assistant through an integreated chatbox. We show the tutorial that is provided to the users below for an overview of the interface functionalities.

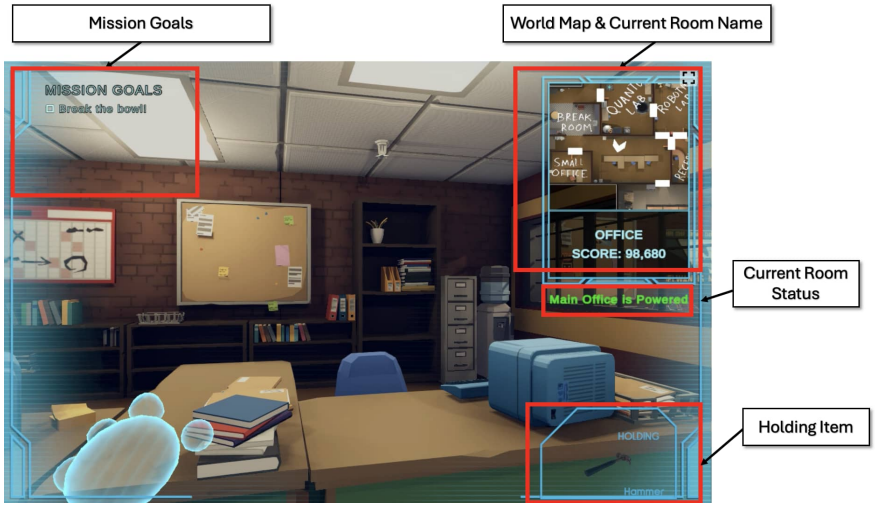


Figure F.1: **Game Video Introduction (Tutorial 1/6)**: Your goal is to complete the task assigned to you with the help of an AI assistant. You can move around, interact with objects, and talk to the assistant. The assistant shares the same view as you do, and will provide you with instructions and guidance throughout the task. **Mission Goals:** Displays the goal of the current task. This is only visible to you but not the assistant. **Map & Current Room:** Shows your position in the world and the current room you are in. Score can be ignored for now. **Current Room Status:** Indicates the operational status of the room, such as whether it is powered or not. **Holding Item:** Displays the object you are holding. This area will be empty if you are not holding an item. **Note**: You can only hold one item at a time, so you need to place the current item down before picking up a new one.

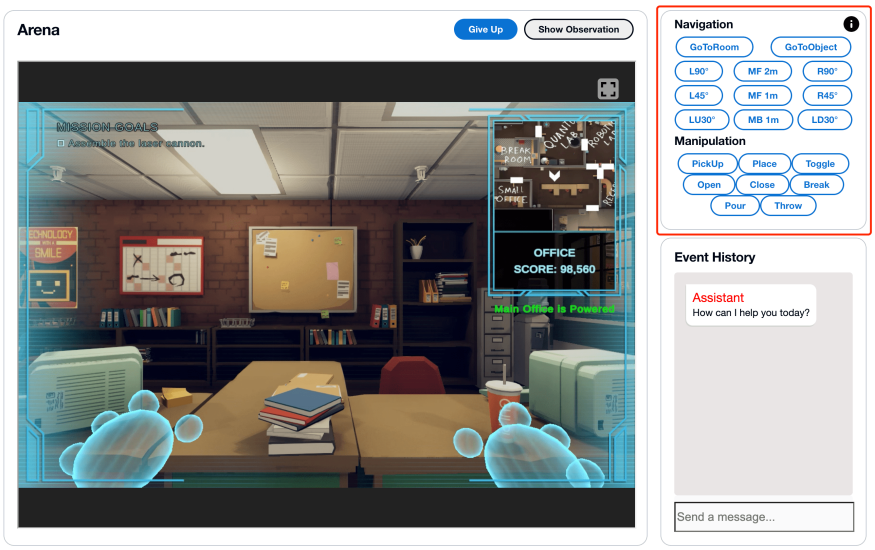


Figure F.2: **Action Control Panel (Tutorial 2/6)**: The **Action Control** panel lets you move around and interact with objects. Use **Movement Actions** for navigation, and **Object Manipulation** for actions like picking up or placing items, toggling machines and computers, and opening/closing doors. Hover over the question mark icon for details of each action. **Go to Object:** Quickly navigate to specific objects in view. **Go to Room:** Directly move to a specific room. **Note**: Go to Object/Room is much more efficient than moving around manually, so it is recommended to use them whenever possible.

F.4.2 Assistant Interface

The Assistant Interface (Figure F.7) is a monitoring page displaying the interaction from the assistant's viewpoint. It includes: **Task Selection** for choosing tasks within a study group; **Screen Recording** to capture user activity; **Interaction History** showing the dialogue between user and assistant; and **Knowledge Display** providing access to the task, environment, and object knowledge used by the assistant to provide effective guidance.

F.4.3 Annotation Interface

We develop an annotation interface for step-level error analysis of assistant performance. Annotators can choose from a list of pre-defined labels to annotate each action made by the assistant, including both "speak errors" that there is something wrong with the guidance, and "silent errors" where the assistant keeps silent when there is a need of speaking. If an action has no error, then a "correct" label should be selected. The knowledge and the full interaction history with the generated thought from the assistant is displayed to help the annotator understand what's going on.

We built an Annotation Interface for step-level error analysis of the assistant. Annotators

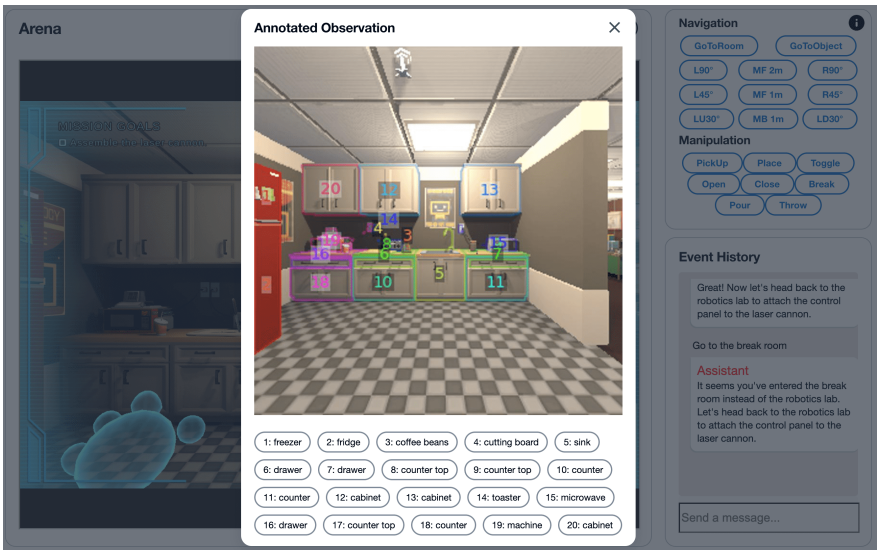


Figure F.3: **Show Observation (Tutorial 3/6)**: A pop-up window displaying all the interactable objects in the current view will show up after clicking the **Show Observation** button. It can help you better understand the objects and plan your actions. Objects that are not common in daily life are displayed with their base category (e.g., "machine") instead of their specific type (e.g., what type of machine it is). The assistant may help you identify them if they are relevant to the current task.

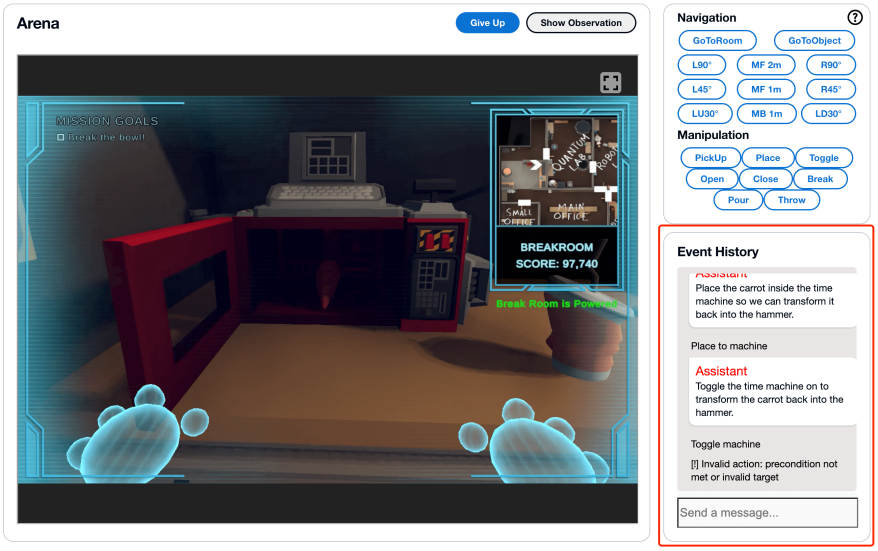


Figure F.4: **Event History Intro (Tutorial 4/6)**: The **Event History** panel shows: (1) Chat history between you and the assistant; (2) Actions you have taken; (3) Action execution feedback when an action has failed (starting with [!]); (4) System messages (in grey) indicating the start and end of a task. You can use the input field at the bottom to send a message to the assistant.

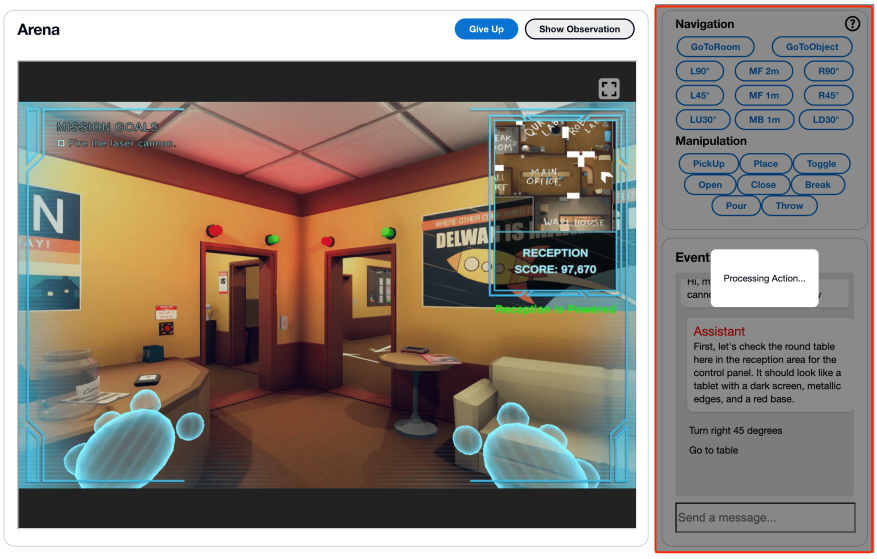


Figure F.5: **Blocking Page Intro (Tutorial 5/6)**: After you select an action or send a message, a blocking overlay will be shown while the system processes your request, which normally takes a few seconds. You can only continue after the overlay disappears.

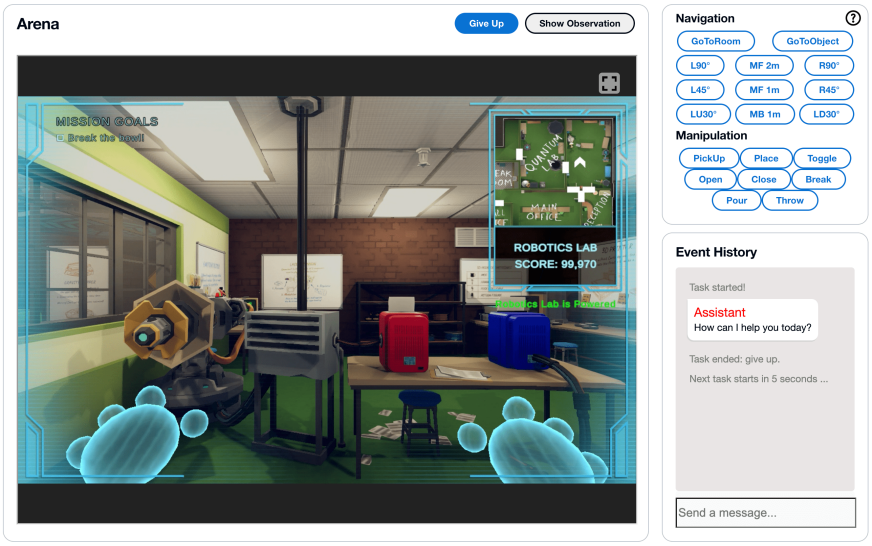


Figure F.6: **Give Up or Task Completion (Tutorial 6/6)**: The task will end if (1) you successfully complete the task; (2) you click the **Give Up** button; (3) max steps (50 actions) is reached. **Note**: the assistant is not perfect, so it is normal that the assistant's instructions are not always correct. You may choose to give up on a task at any time, when you feel getting stuck and the assistant is not helpful. After a task is finished, the system will automatically advance you to the next task. You will work on a total of 5 tasks in this session.

label each assistant action using pre-defined categories, identifying "speak errors" (incorrect guidance), "silent errors" (missed necessary interventions), or marking actions as "correct".

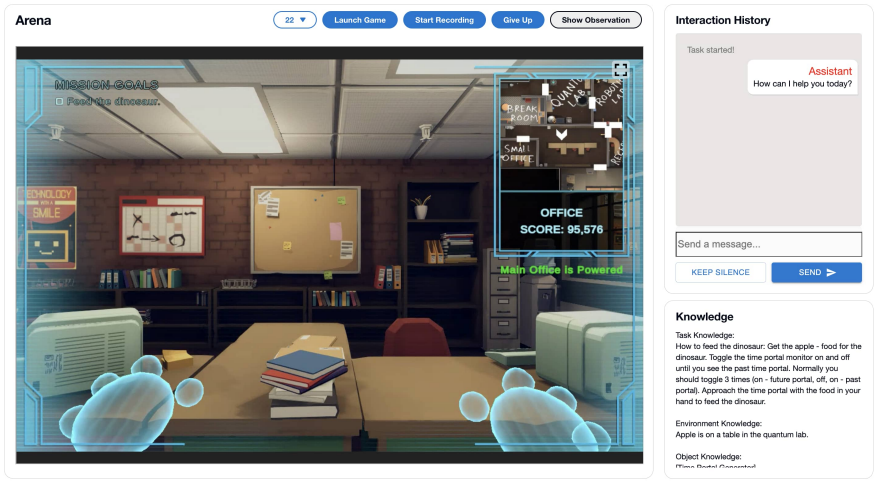


Figure F.7: Assistant interface for task monitoring and debugging.

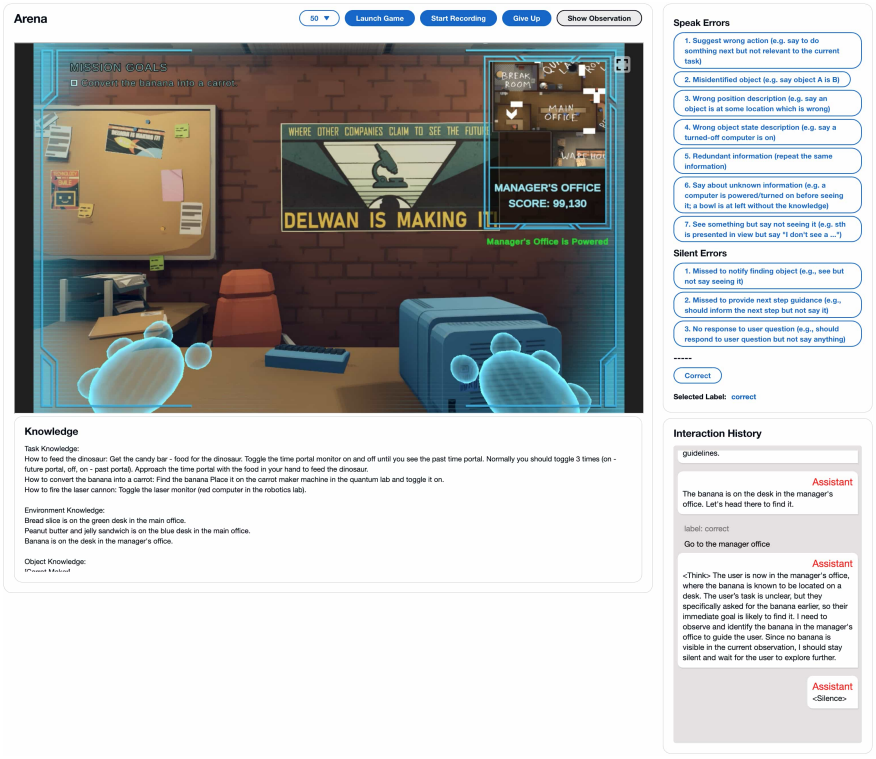


Figure F.8: Annotation interface for assistant error anntation.

The interface shows the full interaction history including actions and the assistant's thought process, and relevant knowledge to support annotation decisions.

F.4.4 Qualitative Examples

In this section, we present two examples of assistant–user interactions, highlighting key events that demonstrate the assistant's decision-making.

F.4.4.1 Example Interaction Between Oracle Assistant and Human User

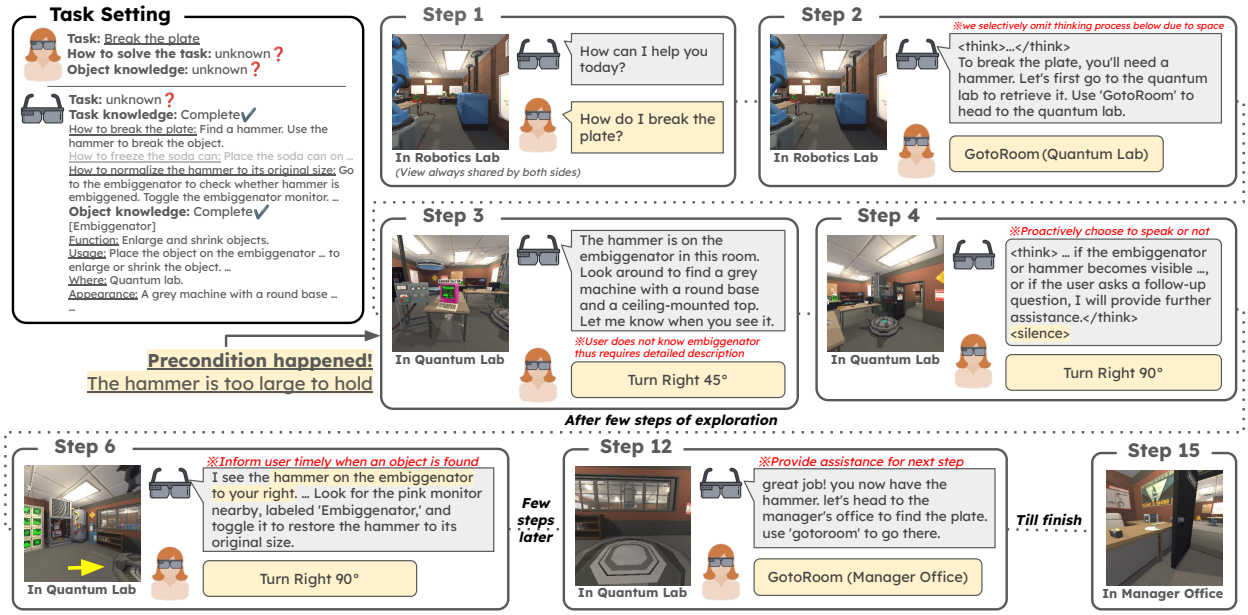


Figure F.9: Illustration of the asymmetric task setting and key events during the interaction, where the oracle assistant provides assistance for the user to normalize a hammer using the embiggenator and use it to break a plate.

Figure F.9 shows an interaction where the user is tasked with breaking a plate, but is initially unaware of the necessary steps or object locations. The oracle assistant possesses complete task and object knowledge and provides step-by-step guidance:

- **Step 1:** The assistant initiates by asking, "How can I help you today?" When the user responds, "How do I break the plate?" the assistant confirms the task and begins reasoning.
- **Step 2:** Concluding that a hammer is required, the assistant directs the user to the Quantum Lab to retrieve it.
- **Step 3:** In the Quantum Lab, the assistant describes the embbiggenator—a tool unfamiliar to the user—by noting its appearance (a grey machine with a round base and a ceiling-mounted top), helping the user identify it.

- **Step 4:** The assistant then intentionally remains silent to avoid redundant guidance, monitoring the user's progress.
- **Step 6:** Observing the user near the embbiggenator and hammer, the assistant notices that the hammar has been embiggenated and thus too large for the user to pick up. The assistant then instructs the user to toggle the embbiggenator to resize the hammer before picking it up.
- **Step 12 - 15:** After several additional steps, the user retrieves the hammer, is praised by the assistant, and is subsequently guided to the manager's office to break the plate, successfully completing the task.

Overall, the example illustrates how the oracle assistant effectively bridges the user's knowledge gap through adaptive, contextual, and timely guidance.

F.4.5 Example Interaction Between Trained Assistant Model and Human User

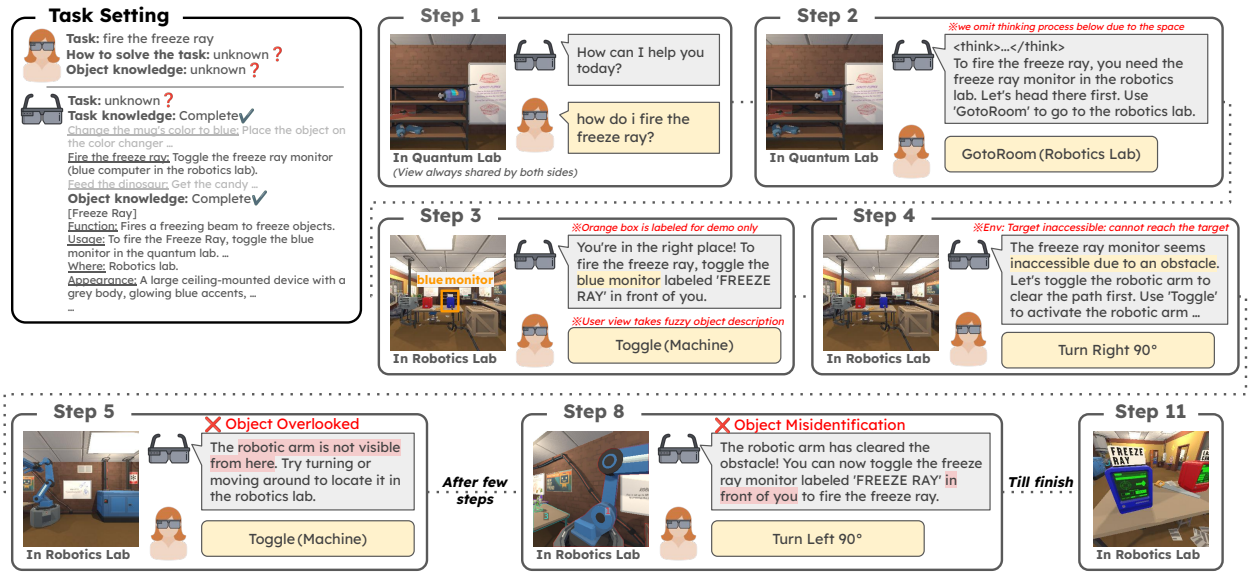


Figure F.10: Illustration of interactions between the **trained model assistant** and a human user where the task is to fire the freeze ray. Two assistant errors are highlighted in Steps 5 and 8.

Figure F.10 illustrates an interaction with our trained model assistant where we show two common errors observed in our experiments (also seen with oracle assistants):

- **Step 5:** The user seeks a robotic arm to remove an obstacle. Although the robotic arm is clearly visible on the left, the assistant overlooks it and advises the user to search the area. This typical perception error leads to inefficient navigation and additional steps.
- **Step 8:** After the obstacle is cleared, the assistant misidentifies the object as the freeze ray monitor. Despite this error, the user relies on previously learned visual cues to correctly locate the actual freeze ray monitor. This demonstrates how human memory and situational reasoning capability can compensate for the assistant's perceptual limitations.

BIBLIOGRAPHY

[1] Yelaman Abdullin, Diego Molla-Aliod, Bahadorreza Ofoghi, John Yearwood, and Qingyang Li. Synthetic dialogue dataset generation using llm agents. arXiv preprint arXiv:2401.17461, 2024.

[2] Josh Achiam, Steven Adler, Sandhini Agarwal, Lama Ahmad, Ilge Akkaya, Florencia Leoni Aleman, Diogo Almeida, Janko Altenschmidt, Sam Altman, Shyamal Anadkat, et al. Gpt-4 technical report. arXiv preprint arXiv:2303.08774, 2023.

[3] Edward H Adelson. Checkershadow illusion, 1995.

[4] Constructions Aeronautiques, Adele Howe, Craig Knoblock, ISI Drew McDermott, Ashwin Ram, Manuela Veloso, Daniel Weld, David Wilkins SRI, Anthony Barrett, Dave Christianson, et al. Pddl| the planning domain definition language. Technical Report, Tech. Rep., 1998.

[5] Mahmoud Afifi and Michael S Brown. What else can fool deep learning? addressing color constancy errors on deep neural network performance. In Proceedings of the IEEE/CVF International Conference on Computer Vision, pages 243–252, 2019.

[6] Estelle Aflalo, Meng Du, Shao-Yen Tseng, Yongfei Liu, Chenfei Wu, Nan Duan, and Vasudev Lal. Vl-interpret: An interactive visualization tool for interpreting vision-language transformers. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pages 21406–21415, 2022.

[7] Christopher Agia, Krishna Murthy Jatavallabhula, Mohamed Khodeir, Ondrej Miksik, Vibhav Vineet, Mustafa Mukadam, Liam Paull, and Florian Shkurti. Taskography: Evaluating robot task planning over large 3d scene graphs. In Aleksandra Faust, David Hsu, and Gerhard Neumann, editors, Proceedings of the 5th Conference on Robot Learning, volume 164 of Proceedings of Machine Learning Research, pages 46–58. PMLR, 08–11 Nov 2022.

[8] 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.

[9] Jean-Baptiste Alayrac, Jeff Donahue, Pauline Luc, Antoine Miech, Iain Barr, Yana Hasson, Karel Lenc, Arthur Mensch, Katherine Millican, Malcolm Reynolds, et al. Flamingo:

a visual language model for few-shot learning. Advances in Neural Information Processing Systems, 35:23716–23736, 2022.

[10] Peter Anderson, Basura Fernando, Mark Johnson, and Stephen Gould. Spice: Semantic propositional image caption evaluation. In Proceedings of the 14th European Conference on Computer Vision, pages 382–398, 2016.

[11] Peter Anderson, Qi Wu, Damien Teney, Jake Bruce, Mark Johnson, Niko Sünderhauf, Ian Reid, Stephen Gould, and Anton van den Hengel. Vision-and-Language Navigation: Interpreting visually-grounded navigation instructions in real environments. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2018.

[12] Peter Anderson, Qi Wu, Damien Teney, Jake Bruce, Mark Johnson, Niko Sünderhauf, Ian D. Reid, Stephen Gould, and Anton van den Hengel. Vision-and-language navigation: Interpreting visually-grounded navigation instructions in real environments. In 2018 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2018, Salt Lake City, UT, USA, June 18-22, 2018, pages 3674–3683. IEEE Computer Society, 2018.

[13] Anthropic. The claude 3 model family: Opus, sonnet, haiku. In Claude 3 Technical Report, 2024.

[14] Stanislaw Antol, Aishwarya Agrawal, Jiasen Lu, Margaret Mitchell, Dhruv Batra, C. Lawrence Zitnick, and Devi Parikh. VQA: Visual Question Answering. In International Conference on Computer Vision (ICCV), 2015.

[15] Iro Armeni, Zhi-Yang He, Amir Zamir, Junyoung Gwak, Jitendra Malik, Martin Fischer, and Silvio Savarese. 3d scene graph: A structure for unified semantics, 3d space, and camera. 2019 IEEE/CVF International Conference on Computer Vision (ICCV), Oct 2019.

[16] Pierre-Luc Bacon, Jean Harb, and Doina Precup. The option-critic architecture. Proceedings of the AAAI Conference on Artificial Intelligence, 31(1), Feb. 2017.

[17] Jinze Bai, Shuai Bai, Shusheng Yang, Shijie Wang, Sinan Tan, Peng Wang, Junyang Lin, Chang Zhou, and Jingren Zhou. Qwen-vl: A frontier large vision-language model with versatile abilities. arXiv preprint arXiv:2308.12966, 2023.

[18] Jinze Bai, Shuai Bai, Shusheng Yang, Shijie Wang, Sinan Tan, Peng Wang, Junyang Lin, Chang Zhou, and Jingren Zhou. Qwen-vl: A versatile vision-language model for understanding, localization, text reading, and beyond. arXiv preprint arXiv:2308.12966, 2023.

[19] Shuai Bai, Keqin Chen, Xuejing Liu, Jialin Wang, Wenbin Ge, Sibo Song, Kai Dang, Peng Wang, Shijie Wang, Jun Tang, et al. Qwen2. 5-vl technical report. arXiv preprint arXiv:2502.13923, 2025.

[20] Satanjeev Banerjee and Alon Lavie. Meteor: An automatic metric for mt evaluation with improved correlation with human judgments. In Proceedings of the acl workshop on intrinsic and extrinsic evaluation measures for machine translation and/or summarization, pages 65–72, 2005.

[21] Yuwei Bao, Keunwoo Peter Yu, Yichi Zhang, Shane Storks, Itamar Bar-Yossef, Alexander De La Iglesia, Megan Su, Xiao Lin Zheng, and Joyce Chai. Can foundation models watch, talk and guide you step by step to make a cake? arXiv preprint arXiv:2311.00738, 2023.

[22] A Benjamin, Cheng Qiu, Ling-Qi Zhang, K Kording, and A Stocker. Shared visual illusions between humans and artificial neural networks. In 2019 Conference on Cognitive Computational Neuroscience, volume 10, pages 2019–1299, 2019.

[23] Yonatan Bisk, Ari Holtzman, Jesse Thomason, Jacob Andreas, Yoshua Bengio, Joyce Chai, Mirella Lapata, Angeliki Lazaridou, Jonathan May, Aleksandr Nisnevich, et al. Experience grounds language. In Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP), pages 8718–8735, 2020.

[24] Valts Blukis, Chris Paxton, Dieter Fox, Animesh Garg, and Yoav Artzi. A persistent spatial semantic representation for high-level natural language instruction execution. In Conference on Robot Learning, pages 706–717. PMLR, 2022.

[25] Dan Bohus and Eric Horvitz. On the challenges and opportunities of physically situated dialog. In AAAI Fall Symposium: Dialog with Robots, 2010.

[26] Holger Caesar, Jasper Uijlings, and Vittorio Ferrari. Coco-stuff: Thing and stuff classes in context. In Proceedings of the IEEE conference on computer vision and pattern recognition, pages 1209–1218, 2018.

[27] Claus-Christian Carbon. Understanding human perception by human-made illusions. Frontiers in human neuroscience, 8:566, 2014.

[28] David P Carey. Do action systems resist visual illusions? Trends in cognitive sciences, 5(3):109–113, 2001.

[29] J. Y. Chai, L. She, R. Fang, S. Ottarson, C. Littley, C. Liu, and K. Hanson. Collaborative effort towards common ground in situated human robot dialogue. In The 9th ACM/IEEE International Conference on Human-Robot Interaction, Bielefeld, Germany, 2014.

[30] Joyce Chai, Qiaozi Gao, Lanbo She, Shaohua Yang, Sari Saba-Sadiya, and Guangyue Xu. Language to action: Towards interactive task learning with physical agents. In IJCAI, pages 2–9, 2018.

[31] Soravit Changpinyo, Piyush Sharma, Nan Ding, and Radu Soricut. Conceptual 12m: Pushing web-scale image-text pre-training to recognize long-tail visual concepts. In Proceedings of the IEEE/CVF conference on computer vision and pattern recognition, pages 3558–3568, 2021.

[32] Devendra Singh Chaplot, Dhiraj Gandhi, Abhinav Gupta, and Ruslan Salakhutdinov. Object goal navigation using goal-oriented semantic exploration. In In Neural Information Processing Systems, 2020.

[33] Chi Chen, Ruoyu Qin, Fuwen Luo, Xiaoyue Mi, Peng Li, Maosong Sun, and Yang Liu. Position-enhanced visual instruction tuning for multimodal large language models. arXiv preprint arXiv:2308.13437, 2023.

[34] Chongyan Chen, Samreen Anjum, and Danna Gurari. Grounding answers for visual questions asked by visually impaired people. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pages 19098–19107, 2022.

[35] Howard Chen, Alane Suhr, Dipendra Misra, Noah Snavely, and Yoav Artzi. TOUCHDOWN: natural language navigation and spatial reasoning in visual street environments. In IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2019, Long Beach, CA, USA, June 16-20, 2019, pages 12538–12547. Computer Vision Foundation / IEEE, 2019.

[36] Joya Chen, Zhaoyang Lv, Shiwei Wu, Kevin Qinghong Lin, Chenan Song, Difei Gao, Jia-Wei Liu, Ziteng Gao, Dongxing Mao, and Mike Zheng Shou. Videollm-online: Online video large language model for streaming video. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pages 18407–18418, 2024.

[37] Kai Chen, Jiaqi Wang, Jiangmiao Pang, Yuhang Cao, Yu Xiong, Xiaoxiao Li, Shuyang Sun, Wansen Feng, Ziwei Liu, Jiarui Xu, Zheng Zhang, Dazhi Cheng, Chenchen Zhu, Tianheng Cheng, Qijie Zhao, Buyu Li, Xin Lu, Rui Zhu, Yue Wu, Jifeng Dai, Jingdong Wang, Jianping Shi, Wanli Ouyang, Chen Change Loy, and Dahua Lin. MMDetection: Open mmlab detection toolbox and benchmark. arXiv preprint arXiv:1906.07155, 2019.

[38] Keqin Chen, Zhao Zhang, Weili Zeng, Richong Zhang, Feng Zhu, and Rui Zhao. Shikra: Unleashing multimodal llm's referential dialogue magic. arXiv preprint arXiv:2306.15195, 2023.

[39] Maximillian Chen, Alexandros Papangelis, Chenyang Tao, Seokhwan Kim, Andy Rosenbaum, Yang Liu, Zhou Yu, and Dilek Hakkani-Tur. Places: Prompting language models for social conversation synthesis. In Findings of the Association for Computational Linguistics: EACL 2023, pages 844–868, 2023.

[40] Ting Chen, Saurabh Saxena, Lala Li, David J Fleet, and Geoffrey Hinton. Pix2seq: A language modeling framework for object detection. In International Conference on Learning Representations, 2021.

[41] Yi Chen, Yuying Ge, Yixiao Ge, Mingyu Ding, Bohao Li, Rui Wang, Ruifeng Xu, Ying Shan, and Xihui Liu. Egoplan-bench: Benchmarking multimodal large language models for human-level planning. arXiv preprint arXiv:2312.06722, 2023.

[42] Bowen Cheng, Ishan Misra, Alexander G Schwing, Alexander Kirillov, and Rohit Girdhar. Masked-attention mask transformer for universal image segmentation. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pages 1290–1299, 2022.

[43] Alexis Chevalier, Alexander Wettig, Anirudh Ajith, and Danqi Chen. Adapting language models to compress contexts. In Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing, pages 3829–3846, 2023.

[44] Wei-Lin Chiang, Zhuohan Li, Zi Lin, Ying Sheng, Zhanghao Wu, Hao Zhang, Lianmin Zheng, Siyuan Zhuang, Yonghao Zhuang, Joseph E Gonzalez, et al. Vicuna: An open-source chatbot impressing gpt-4 with 90%* chatgpt quality. https://vicuna.lmsys.org, 2023.

[45] Sanjiban Choudhury, Mohak Bhardwaj, Sankalp Arora, Ashish Kapoor, Gireeja Ranade, Sebastian Scherer, and Debadeepta Dey. Data-driven planning via imitation learning. The International Journal of Robotics Research, 37(13-14):1632–1672, 2018.

[46] Herbert H Clark. Using language. Cambridge university press, 1996.

[47] Jifeng Dai, Kaiming He, and Jian Sun. Convolutional feature masking for joint object and stuff segmentation. In Proceedings of the IEEE conference on computer vision and pattern recognition, pages 3992–4000, 2015.

[48] Wenliang Dai, Junnan Li, Dongxu Li, Anthony Meng Huat Tiong, Junqi Zhao, Weisheng Wang, Boyang Li, Pascale Fung, and Steven Hoi. Instructblip: Towards general-purpose vision-language models with instruction tuning. arXiv preprint arXiv:2305.06500, 2023.

[49] Wenliang Dai, Zihan Liu, Ziwei Ji, Dan Su, and Pascale Fung. Plausible may not be faithful: Probing object hallucination in vision-language pre-training. In Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics, pages 2128–2140, 2023.

[50] Dima Damen, Hazel Doughty, Giovanni Maria Farinella, Sanja Fidler, Antonino Furnari, Evangelos Kazakos, Davide Moltisanti, Jonathan Munro, Toby Perrett, Will Price, et al. The epic-kitchens dataset: Collection, challenges and baselines. IEEE Transactions on Pattern Analysis and Machine Intelligence, 43(11):4125–4141, 2020.

[51] Dima Damen, Hazel Doughty, Giovanni Maria Farinella, Antonino Furnari, Jian Ma, Evangelos Kazakos, Davide Moltisanti, Jonathan Munro, Toby Perrett, Will Price, and Michael Wray. Rescaling egocentric vision: Collection, pipeline and challenges for epic-kitchens-100. International Journal of Computer Vision (IJCV), 130:33–55, 2022.

[52] Christian Daniel, Gerhard Neumann, Oliver Kroemer, and Jan Peters. Hierarchical relative entropy policy search. Journal of Machine Learning Research, 17(93):1–50, 2016.

[53] Tri Dao. FlashAttention-2: Faster attention with better parallelism and work partitioning. In International Conference on Learning Representations (ICLR), 2024.

[54] Timothée Darcet, Maxime Oquab, Julien Mairal, and Piotr Bojanowski. Vision transformers need registers. arXiv preprint arXiv:2309.16588, 2023.

[55] Abhishek Das, Samyak Datta, Georgia Gkioxari, Stefan Lee, Devi Parikh, and Dhruv Batra. Embodied question answering. In 2018 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2018, Salt Lake City, UT, USA, June 18-22, 2018, pages 1–10. IEEE Computer Society, 2018.

[56] Abhishek Das, Georgia Gkioxari, Stefan Lee, Devi Parikh, and Dhruv Batra. Neural modular control for embodied question answering. In Conference on Robot Learning, pages 53–62. PMLR, 2018.

[57] RH Day. The nature of, perceptual illusions. Interdisciplinary Science Reviews, 9(1):47–58, 1984.

[58] Daan de Geus, Panagiotis Meletis, Chenyang Lu, Xiaoxiao Wen, and Gijs Dubbelman. Part-aware panoptic segmentation. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pages 5485–5494, 2021.

[59] Harm De Vries, Florian Strub, Sarath Chandar, Olivier Pietquin, Hugo Larochelle, and Aaron Courville. Guesswhat?! visual object discovery through multi-modal dialogue. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pages 5503–5512, 2017.

[60] Matt Deitke, Winson Han, Alvaro Herrasti, Aniruddha Kembhavi, Eric Kolve, Roozbeh Mottaghi, Jordi Salvador, Dustin Schwenk, Eli VanderBilt, Matthew Wallingford, et al. Robothor: An open simulation-to-real embodied ai platform. In Proceedings of the IEEE/CVF conference on computer vision and pattern recognition, pages 3164–3174, 2020.

[61] Matt Deitke, Eli VanderBilt, Alvaro Herrasti, Luca Weihs, Kiana Ehsani, Jordi Salvador, Winson Han, Eric Kolve, Aniruddha Kembhavi, and Roozbeh Mottaghi. Procthor: Large-scale embodied ai using procedural generation. Advances in Neural Information Processing Systems, 35:5982–5994, 2022.

[62] Franz Joseph Delboeuf. Note sur certaines illusions d'optique: Essai d'une théorie psychophysique de la maniere dont l'oeil apprécie les distances et les angles. Bulletins de l'Académie Royale des Sciences, 19:195–216, 1865.

[63] Jan Deriu, Alvaro Rodrigo, Arantxa Otegi, Guillermo Echegoyen, Sophie Rosset, Eneko Agirre, and Mark Cieliebak. Survey on evaluation methods for dialogue systems. Artificial Intelligence Review, 54:755–810, 2021.

[64] Thomas G. Dietterich. Hierarchical reinforcement learning with the maxq value function decomposition. J. Artif. Int. Res., 13(1):227–303, nov 2000.

[65] Carlos Diuk, Andre Cohen, and Michael L Littman. An object-oriented representation for efficient reinforcement learning. In Proceedings of the 25th international conference on Machine learning, pages 240–247, 2008.

[66] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, Jakob Uszkoreit, and Neil Houlsby. An image is worth 16x16 words: Transformers for image recognition at scale. ICLR, 2021.

[67] 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.

[68] Kiana Ehsani, Tanmay Gupta, Rose Hendrix, Jordi Salvador, Luca Weihs, Kuo-Hao Zeng, Kunal Pratap Singh, Yejin Kim, Winson Han, Alvaro Herrasti, et al. Spoc: Imitating shortest paths in simulation enables effective navigation and manipulation in the real world. arXiv preprint arXiv:2312.02976, 2023.

[69] Kutluhan Erol, James Hendler, and Dana S Nau. Htn planning: Complexity and expressivity. In AAAI, volume 94, pages 1123–1128, 1994.

[70] Mark Everingham, Luc Van Gool, Christopher KI Williams, John Winn, and Andrew Zisserman. The pascal visual object classes (voc) challenge. International journal of computer vision, 88:303–338, 2010.

[71] C. Galindo, J.-A. Fernandez-Madrigal, and J. Gonzalez. Improving efficiency in mobile robot task planning through world abstraction. IEEE Transactions on Robotics, 20(4):677–690, 2004.

[72] Cipriano Galindo, Juan-Antonio Fernández-Madrigal, Javier González, and Alessandro Saffiotti. Robot task planning using semantic maps. Robotics and Autonomous Systems, 56(11):955–966, 2008. Semantic Knowledge in Robotics.

[73] Chuang Gan, Yandong Li, Haoxiang Li, Chen Sun, and Boqing Gong. Vqs: Linking segmentations to questions and answers for supervised attention in vqa and question-focused semantic segmentation. In Proceedings of the IEEE international conference on computer vision, pages 1811–1820, 2017.

[74] Chuang Gan, Jeremy Schwartz, Seth Alter, Damian Mrowca, Martin Schrimpf, James Traer, Julian De Freitas, Jonas Kubilius, Abhishek Bhandwaldar, Nick Haber, et al. Threedworld: A platform for interactive multi-modal physical simulation. arXiv preprint arXiv:2007.04954, 2020.

[75] Qiaozi Gao, Malcolm Doering, Shaohua Yang, and Joyce Chai. Physical causality of action verbs in grounded language understanding. In Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), pages 1814–1824, Berlin, Germany, 2016. Association for Computational Linguistics.

[76] Qiaozi Gao, Govind Thattai, Suhaila Shakiah, Xiaofeng Gao, Shreyas Pansare, Vasu Sharma, Gaurav Sukhatme, Hangjie Shi, Bofei Yang, Desheng Zhang, et al. Alexa arena: A user-centric interactive platform for embodied ai. Advances in Neural Information Processing Systems, 36:19170–19194, 2023.

[77] Xiaofeng Gao, Qiaozi Gao, Ran Gong, Kaixiang Lin, Govind Thattai, and Gaurav S Sukhatme. Dialfred: Dialogue-enabled agents for embodied instruction following. IEEE Robotics and Automation Letters, 7(4):10049–10056, 2022.

[78] Ron Garland. The mid-point on a rating scale: Is it desirable. Marketing bulletin, 2(1):66–70, 1991.

[79] Maurizio Gentilucci, Sergio Chieffi, Elena Daprati, M Cristina Saetti, and Ivan Toni. Visual illusion and action. Neuropsychologia, 34(5):369–376, 1996.

[80] Alex Gomez-Villa, Adrián Martín, Javier Vazquez-Corral, Marcelo Bertalmío, and Jesús Malo. On the synthesis of visual illusions using deep generative models. Journal of Vision, 22(8):2–2, 2022.

[81] Alexander Gomez-Villa, Adrian Martin, Javier Vazquez-Corral, and Marcelo Bertalmío. Convolutional neural networks can be deceived by visual illusions. In Proceedings of the IEEE/CVF conference on computer vision and pattern recognition, pages 12309–12317, 2019.

[82] Alexander Gomez-Villa, Adrian Martín, Javier Vazquez-Corral, Marcelo Bertalmío, and Jesús Malo. Color illusions also deceive cnns for low-level vision tasks: Analysis and implications. Vision Research, 176:156–174, 2020.

[83] Tao Gong, Chengqi Lyu, Shilong Zhang, Yudong Wang, Miao Zheng, Qian Zhao, Kuikun Liu, Wenwei Zhang, Ping Luo, and Kai Chen. Multimodal-gpt: A vision and language model for dialogue with humans. arXiv preprint arXiv:2305.04790, 2023.

[84] Cristina González, Nicolás Ayobi, Isabela Hernández, José Hernández, Jordi Pont-Tuset, and Pablo Arbeláez. Panoptic narrative grounding. In Proceedings of the IEEE/CVF International Conference on Computer Vision, pages 1364–1373, 2021.

[85] Cristina González, Nicolás Ayobi, Isabela Hernández, Jordi Pont-Tuset, and Pablo Arbeláez. Piglet: Pixel-level grounding of language expressions with transformers. IEEE Transactions on Pattern Analysis and Machine Intelligence, 2023.

[86] Abel Gonzalez-Garcia, Davide Modolo, and Vittorio Ferrari. Objects as context for detecting their semantic parts. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pages 6907–6916, 2018.

[87] Daniel Gordon, Aniruddha Kembhavi, Mohammad Rastegari, Joseph Redmon, Dieter Fox, and Ali Farhadi. IQA: visual question answering in interactive environments. In 2018 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2018, Salt Lake City, UT, USA, June 18-22, 2018, pages 4089–4098. IEEE Computer Society, 2018.

[88] Raghav Goyal, Samira Ebrahimi Kahou, Vincent Michalski, Joanna Materzynska, Susanne Westphal, Heuna Kim, Valentin Haenel, Ingo Fruend, Peter Yianilos, Moritz Mueller-Freitag, et al. The" something something" video database for learning and

evaluating visual common sense. In Proceedings of the IEEE international conference on computer vision, pages 5842–5850, 2017.

[89] Yash Goyal, Tejas Khot, Douglas Summers-Stay, Dhruv Batra, and Devi Parikh. Making the v in vqa matter: Elevating the role of image understanding in visual question answering. In Proceedings of the IEEE conference on computer vision and pattern recognition, pages 6904–6913, 2017.

[90] Kristen Grauman, Andrew Westbury, Eugene Byrne, Zachary Chavis, Antonino Furnari, Rohit Girdhar, Jackson Hamburger, Hao Jiang, Miao Liu, Xingyu Liu, et al. Ego4d: Around the world in 3,000 hours of egocentric video. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pages 18995–19012, 2022.

[91] Kristen Grauman, Andrew Westbury, Lorenzo Torresani, Kris Kitani, Jitendra Malik, Triantafyllos Afouras, Kumar Ashutosh, Vijay Baiyya, Siddhant Bansal, Bikram Boote, et al. Ego-exo4d: Understanding skilled human activity from first-and third-person perspectives. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pages 19383–19400, 2024.

[92] Richard L Gregory. Visual illusions classified. Trends in cognitive sciences, 1(5):190–194, 1997.

[93] Agrim Gupta, Piotr Dollar, and Ross Girshick. Lvis: A dataset for large vocabulary instance segmentation. In Proceedings of the IEEE/CVF conference on computer vision and pattern recognition, pages 5356–5364, 2019.

[94] Kai Han, Yunhe Wang, Jianyuan Guo, Yehui Tang, and Enhua Wu. Vision gnn: An image is worth graph of nodes. Advances in Neural Information Processing Systems, 35:8291–8303, 2022.

[95] Stevan Harnad. The symbol grounding problem. Physica D: Nonlinear Phenomena, 42(1-3):335–346, 1990.

[96] Bo He, Hengduo Li, Young Kyun Jang, Menglin Jia, Xuefei Cao, Ashish Shah, Abhinav Shrivastava, and Ser-Nam Lim. Ma-lmm: Memory-augmented large multimodal model for long-term video understanding. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pages 13504–13514, 2024.

[97] Kaiming He, Georgia Gkioxari, Piotr Dollár, and Ross B. Girshick. Mask R-CNN. In IEEE International Conference on Computer Vision, ICCV 2017, Venice, Italy, October 22-29, 2017, pages 2980–2988. IEEE Computer Society, 2017.

[98] M. Helmert. The fast downward planning system. Journal of Artificial Intelligence Research, 26:191–246, jul 2006.

[99] Karl Moritz Hermann, Felix Hill, Simon Green, Fumin Wang, Ryan Faulkner, Hubert Soyer, David Szepesvari, Wojciech Marian Czarnecki, Max Jaderberg, Denis Teplyashin, et al. Grounded language learning in a simulated 3d world. arXiv preprint arXiv:1706.06551, 2017.

[100] Elad Hirsch and Ayellet Tal. Color visual illusions: A statistics-based computational model. Advances in Neural Information Processing Systems, 33:9447–9458, 2020.

[101] Pin-Lun Hsu, Yun Dai, Vignesh Kothapalli, Qingquan Song, Shao Tang, Siyu Zhu, Steven Shimizu, Shivam Sahni, Haowen Ning, and Yanning Chen. Liger kernel: Efficient triton kernels for llm training. arXiv preprint arXiv:2410.10989, 2024.

[102] Edward J Hu, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, Weizhu Chen, et al. Lora: Low-rank adaptation of large language models. In International Conference on Learning Representations, 2021.

[103] Edward J Hu, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, Weizhu Chen, et al. Lora: Low-rank adaptation of large language models. In International Conference on Learning Representations, 2022.

[104] Yifei Huang, Guo Chen, Jilan Xu, Mingfang Zhang, Lijin Yang, Baoqi Pei, Hongjie Zhang, Lu Dong, Yali Wang, Limin Wang, et al. Egoexolearn: A dataset for bridging asynchronous ego-and exo-centric view of procedural activities in real world. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pages 22072–22086, 2024.

[105] Drew A Hudson and Christopher D Manning. Gqa: A new dataset for real-world visual reasoning and compositional question answering. In Proceedings of the IEEE/CVF conference on computer vision and pattern recognition, pages 6700–6709, 2019.

[106] Ahmed Hussein, Mohamed Medhat Gaber, Eyad Elyan, and Chrisina Jayne. Imitation learning: A survey of learning methods. ACM Computing Surveys (CSUR), 50(2):1–35, 2017.

[107] Muhammet Ilaslan, Chenan Song, Joya Chen, Difei Gao, Weixian Lei, Qianli Xu, Joo Lim, and Mike Shou. Gazevqa: A video question answering dataset for multiview eye-gaze task-oriented collaborations. In Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing, pages 10462–10479, 2023.

[108] Peter Jansen. Visually-grounded planning without vision: Language models infer detailed plans from high-level instructions. In Findings of the Association for Computational Linguistics: EMNLP 2020, pages 4412–4417, Online, 2020. Association for Computational Linguistics.

[109] Joseph Jastrow. Studies from the laboratory of experimental psychology of the university of wisconsin. ii. The American Journal of Psychology, 4(3):381–428, 1892.

[110] Roy Jonker and Ton Volgenant. A shortest augmenting path algorithm for dense and sparse linear assignment problems. In DGOR/NSOR: Papers of the 16th Annual Meeting of DGOR in Cooperation with NSOR/Vorträge der 16. Jahrestagung der DGOR zusammen mit der NSOR, pages 622–622. Springer, 1988.

[111] Aishwarya Kamath, Mannat Singh, Yann LeCun, Gabriel Synnaeve, Ishan Misra, and Nicolas Carion. Mdetr-modulated detection for end-to-end multi-modal understanding. In Proceedings of the IEEE/CVF International Conference on Computer Vision, pages 1780–1790, 2021.

[112] Amita Kamath, Jack Hessel, and Kai-Wei Chang. What's "up" with vision-language models? investigating their struggle with spatial reasoning. In Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing, pages 9161–9175, 2023.

[113] Siddharth Karamcheti, Suraj Nair, Ashwin Balakrishna, Percy Liang, Thomas Kollar, and Dorsa Sadigh. Prismatic vlms: Investigating the design space of visually-conditioned language models. In Forty-first International Conference on Machine Learning, 2024.

[114] L.E. Kavraki, P. Svestka, J.-C. Latombe, and M.H. Overmars. Probabilistic roadmaps for path planning in high-dimensional configuration spaces. IEEE Transactions on Robotics and Automation, 12(4):566–580, 1996.

[115] Osman Semih Kayhan, Bart Vredebregt, and Jan C Van Gemert. Hallucination in object detection—a study in visual part verification. In 2021 IEEE International Conference on Image Processing (ICIP), pages 2234–2238. IEEE, 2021.

[116] Sahar Kazemzadeh, Vicente Ordonez, Mark Matten, and Tamara Berg. Referitgame: Referring to objects in photographs of natural scenes. In Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP), pages 787–798, 2014.

[117] Liyiming Ke, Xiujun Li, Yonatan Bisk, Ari Holtzman, Zhe Gan, Jingjing Liu, Jianfeng Gao, Yejin Choi, and Siddhartha Srinivasa. Tactical rewind: Self-correction via backtracking in vision-and-language navigation. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pages 6741–6749, 2019.

[118] Apoorv Khandelwal, Luca Weihs, Roozbeh Mottaghi, and Aniruddha Kembhavi. Simple but effective: Clip embeddings for embodied ai. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pages 14829–14838, 2022.

[119] Seyedalireza Khoshsirat and Chandra Kambhamettu. Sentence attention blocks for answer grounding. In Proceedings of the IEEE/CVF International Conference on Computer Vision, pages 6080–6090, 2023.

[120] Ue-Hwan Kim, Jin-Man Park, Taek-jin Song, and Jong-Hwan Kim. 3-d scene graph: A sparse and semantic representation of physical environments for intelligent agents. IEEE Transactions on Cybernetics, 50(12):4921–4933, Dec 2020.

[121] Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization. In Yoshua Bengio and Yann LeCun, editors, 3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings, 2015.

[122] Alexander Kirillov, Kaiming He, Ross Girshick, Carsten Rother, and Piotr Dollár. Panoptic segmentation. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pages 9404–9413, 2019.

[123] Alexander Kirillov, Eric Mintun, Nikhila Ravi, Hanzi Mao, Chloe Rolland, Laura Gustafson, Tete Xiao, Spencer Whitehead, Alexander C. Berg, Wan-Yen Lo, Piotr Dollar, and Ross Girshick. Segment anything. In Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV), pages 4015–4026, October 2023.

[124] Akiyoshi Kitaoka. A brief classification of colour illusions. Colour: Design & Creativity, 5(3):1–9, 2010.

[125] Eric Kolve, Roozbeh Mottaghi, Winson Han, Eli VanderBilt, Luca Weihs, Alvaro Herrasti, Matt Deitke, Kiana Ehsani, Daniel Gordon, Yuke Zhu, et al. Ai2-thor: An interactive 3d environment for visual ai. arXiv preprint arXiv:1712.05474, 2017.

[126] Eric Kolve, Roozbeh Mottaghi, Winson Han, Eli VanderBilt, Luca Weihs, Alvaro Herrasti, Daniel Gordon, Yuke Zhu, Abhinav Gupta, and Ali Farhadi. Ai2-thor: An interactive 3d environment for visual ai, 2017.

[127] George Konidaris, Scott Kuindersma, Roderic Grupen, and Andrew Barto. Robot learning from demonstration by constructing skill trees. International Journal of Robotic Research - IJRR, 31:360–375, 05 2012.

[128] Satwik Kottur, Seungwhan Moon, Alborz Geramifard, and Babak Damavandi. Simmc 2.0: A task-oriented dialog dataset for immersive multimodal conversations. In Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing, pages 4903–4912, 2021.

[129] Ranjay Krishna, Yuke Zhu, Oliver Groth, Justin Johnson, Kenji Hata, Joshua Kravitz, Stephanie Chen, Yannis Kalantidis, Li-Jia Li, David A Shamma, et al. Visual genome: Connecting language and vision using crowdsourced dense image annotations. International journal of computer vision, 123:32–73, 2017.

[130] Girish Kulkarni, Visruth Premraj, Vicente Ordonez, Sagnik Dhar, Siming Li, Yejin Choi, Alexander C Berg, and Tamara L Berg. Babytalk: Understanding and generating simple image descriptions. IEEE transactions on pattern analysis and machine intelligence, 35(12):2891–2903, 2013.

[131] Tejas D Kulkarni, Karthik Narasimhan, Ardavan Saeedi, and Josh Tenenbaum. Hierarchical deep reinforcement learning: Integrating temporal abstraction and intrinsic motivation. In D. Lee, M. Sugiyama, U. Luxburg, I. Guyon, and R. Garnett, editors, Advances in Neural Information Processing Systems, volume 29. Curran Associates, Inc., 2016.

[132] Woosuk Kwon, Zhuohan Li, Siyuan Zhuang, Ying Sheng, Lianmin Zheng, Cody Hao Yu, Joseph E. Gonzalez, Hao Zhang, and Ion Stoica. Efficient memory management for large language model serving with pagedattention. In Proceedings of the ACM SIGOPS 29th Symposium on Operating Systems Principles, 2023.

[133] Xin Lai, Zhuotao Tian, Yukang Chen, Yanwei Li, Yuhui Yuan, Shu Liu, and Jiaya Jia. Lisa: Reasoning segmentation via large language model. arXiv preprint arXiv:2308.00692, 2023.

[134] Leonardo Lamanna, Luciano Serafini, Alessandro Saetti, Alfonso Gerevini, and Paolo Traverso. Online grounding of pddl domains by acting and sensing in unknown environments. arXiv preprint arXiv:2112.10007, 2021.

[135] Shih-Po Lee, Zijia Lu, Zekun Zhang, Minh Hoai, and Ehsan Elhamifar. Error detection in egocentric procedural task videos. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pages 18655–18666, 2024.

[136] Teesid Leelasawassuk, Dima Damen, and Walterio Mayol-Cuevas. Automated capture and delivery of assistive task guidance with an eyewear computer: the glaciar system. In Proceedings of the 8th Augmented Human International Conference, pages 1–9, 2017.

[137] Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Veselin Stoyanov, and Luke Zettlemoyer. BART: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. In Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, pages 7871–7880, Online, July 2020. Association for Computational Linguistics.

[138] Bo Li, Yuanhan Zhang, Liangyu Chen, Jinghao Wang, Jingkang Yang, and Ziwei Liu. Otter: A multi-modal model with in-context instruction tuning. arXiv preprint arXiv:2305.03726, 2023.

[139] Chengshu Li, Fei Xia, Roberto Martín-Martín, Michael Lingelbach, Sanjana Srivastava, Bokui Shen, Kent Vainio, Cem Gokmen, Gokul Dharan, Tanish Jain, et al. igibson 2.0: Object-centric simulation for robot learning of everyday household tasks. arXiv preprint arXiv:2108.03272, 2021.

[140] Chunyuan Li, Zhe Gan, Zhengyuan Yang, Jianwei Yang, Linjie Li, Lijuan Wang, and Jianfeng Gao. Multimodal foundation models: From specialists to general-purpose assistants. arXiv preprint arXiv:2309.10020, 2023.

[141] Jianshu Li, Jian Zhao, Yunchao Wei, Congyan Lang, Yidong Li, Terence Sim, Shuicheng Yan, and Jiashi Feng. Multiple-human parsing in the wild. arXiv preprint arXiv:1705.07206, 2017.

[142] Junnan Li, Dongxu Li, Silvio Savarese, and Steven Hoi. Blip-2: Bootstrapping language-image pre-training with frozen image encoders and large language models. In International conference on machine learning, pages 19730–19742. PMLR, 2023.

[143] KunChang Li, Yinan He, Yi Wang, Yizhuo Li, Wenhai Wang, Ping Luo, Yali Wang, Limin Wang, and Yu Qiao. Videochat: Chat-centric video understanding. arXiv preprint arXiv:2305.06355, 2023.

[144] Liunian Harold Li, Pengchuan Zhang, Haotian Zhang, Jianwei Yang, Chunyuan Li, Yiwu Zhong, Lijuan Wang, Lu Yuan, Lei Zhang, Jenq-Neng Hwang, et al. Grounded language-image pre-training. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pages 10965–10975, 2022.

[145] Xinghang Li, Di Guo, Huaping Liu, and Fuchun Sun. Embodied semantic scene graph generation. In Aleksandra Faust, David Hsu, and Gerhard Neumann, editors, Proceedings of the 5th Conference on Robot Learning, volume 164 of Proceedings of Machine Learning Research, pages 1585–1594. PMLR, 08–11 Nov 2022.

[146] Yanwei Li, Chengyao Wang, and Jiaya Jia. Llama-vid: An image is worth 2 tokens in large language models. In European Conference on Computer Vision, pages 323–340. Springer, 2025.

[147] Yifan Li, Yifan Du, Kun Zhou, Jinpeng Wang, Wayne Xin Zhao, and Ji-Rong Wen. Evaluating object hallucination in large vision-language models. In Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing, 2023.

[148] Zekun Li, Wenhu Chen, Shiyang Li, Hong Wang, Jing Qian, and Xifeng Yan. Controllable dialogue simulation with in-context learning. In Findings of the Association for Computational Linguistics: EMNLP 2022, pages 4330–4347, 2022.

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

[150] Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C Lawrence Zitnick. Microsoft coco: Common objects in context. In Computer Vision–ECCV 2014: 13th European Conference, Zurich, Switzerland, September 6-12, 2014, Proceedings, Part V 13, pages 740–755. Springer, 2014.

[151] Chang Liu, Henghui Ding, and Xudong Jiang. Gres: Generalized referring expression segmentation. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pages 23592–23601, 2023.

[152] Changsong Liu, Rui Fang, Lanbo She, and Joyce Chai. Modeling collaborative referring for situated referential grounding. In Proceedings of the SIGDIAL 2013 Conference, pages 78–86, Metz, France, 2013. Association for Computational Linguistics.

[153] Haotian Liu, Chunyuan Li, Yuheng Li, and Yong Jae Lee. Improved baselines with visual instruction tuning. arXiv preprint arXiv:2310.03744, 2023.

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

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

[156] Haotian Liu, Chunyuan Li, Qingyang Wu, and Yong Jae Lee. Visual instruction tuning. Advances in neural information processing systems, 36, 2024.

[157] Jiang Liu, Hui Ding, Zhaowei Cai, Yuting Zhang, Ravi Kumar Satzoda, Vijay Mahadevan, and R Manmatha. Polyformer: Referring image segmentation as sequential polygon generation. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pages 18653–18663, 2023.

[158] Nelson F Liu, Kevin Lin, John Hewitt, Ashwin Paranjape, Michele Bevilacqua, Fabio Petroni, and Percy Liang. Lost in the middle: How language models use long contexts. Transactions of the Association for Computational Linguistics, 12:157–173, 2024.

[159] Xiaodong Liu, Pengcheng He, Weizhu Chen, and Jianfeng Gao. Multi-task deep neural networks for natural language understanding. In Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics, pages 4487–4496, Florence, Italy, 2019. Association for Computational Linguistics.

[160] Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. Roberta: A robustly optimized bert pretraining approach. arXiv preprint arXiv:1907.11692, 2019.

[161] Shangbang Long, Siyang Qin, Dmitry Panteleev, Alessandro Bissacco, Yasuhisa Fujii, and Michalis Raptis. Towards end-to-end unified scene text detection and layout analysis. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pages 1049–1059, 2022.

[162] Ben Lonnqvist, Alban Bornet, Adrien Doerig, and Michael H Herzog. A comparative biology approach to dnn modeling of vision: A focus on differences, not similarities. Journal of Vision, 21(10):17–17, 2021.

[163] Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. arXiv preprint arXiv:1711.05101, 2017.

[164] Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. In International Conference on Learning Representations, 2017.

[165] Jiasen Lu, Christopher Clark, Sangho Lee, Zichen Zhang, Savya Khosla, Ryan Marten, Derek Hoiem, and Aniruddha Kembhavi. Unified-io 2: Scaling autoregressive multimodal models with vision language audio and action. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pages 26439–26455, 2024.

[166] Jiasen Lu, Christopher Clark, Rowan Zellers, Roozbeh Mottaghi, and Aniruddha Kembhavi. Unified-io: A unified model for vision, language, and multi-modal tasks. In The Eleventh International Conference on Learning Representations, 2022.

[167] Jiasen Lu, Jianwei Yang, Dhruv Batra, and Devi Parikh. Neural baby talk. In Proceedings of the IEEE conference on computer vision and pattern recognition, pages 7219–7228, 2018.

[168] Yao Lu and Walterio Mayol-Cuevas. Higs: Hand interaction guidance system. In 2019 IEEE international symposium on mixed and augmented reality adjunct (ISMAR-Adjunct), pages 376–381. IEEE, 2019.

[169] Ruipu Luo, Ziwang Zhao, Min Yang, Junwei Dong, Da Li, Pengcheng Lu, Tao Wang, Linmei Hu, Minghui Qiu, and Zhongyu Wei. Valley: Video assistant with large language model enhanced ability. arXiv preprint arXiv:2306.07207, 2023.

[170] Chih-Yao Ma, Jiasen Lu, Zuxuan Wu, Ghassan AlRegib, Zsolt Kira, Richard Socher, and Caiming Xiong. Self-monitoring navigation agent via auxiliary progress estimation. In 7th International Conference on Learning Representations, ICLR 2019, New Orleans, LA, USA, May 6-9, 2019. OpenReview.net, 2019.

[171] Ziqiao Ma, Jiayi Pan, and Joyce Chai. World-to-words: Grounded open vocabulary acquisition through fast mapping in vision-language models. In Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), pages 524–544, 2023.

[172] Ziqiao Ma, Benjamin VanDerPloeg, Cristian-Paul Bara, Yidong Huang, Eui-In Kim, Felix Gervits, Matthew Marge, and Joyce Chai. Dorothie: Spoken dialogue for handling unexpected situations in interactive autonomous driving agents. In Findings of the Association for Computational Linguistics: EMNLP 2022, pages 4800–4822, 2022.

[173] Muhammad Maaz, Hanoona Rasheed, Salman Khan, and Fahad Shahbaz Khan. Video-chatgpt: Towards detailed video understanding via large vision and language models. arXiv preprint arXiv:2306.05424, 2023.

[174] Bruce MacEvoy. Color vision, 2008.

[175] James MacGlashan, Monica Babes-Vroman, Marie desJardins, Michael L. Littman, Smaranda Muresan, S. Squire, Stefanie Tellex, Dilip Arumugam, and Lei Yang. Grounding english commands to reward functions. In Robotics: Science and Systems, 2015.

[176] Arjun Mani, Nobline Yoo, Will Hinthorn, and Olga Russakovsky. Point and ask: Incorporating pointing into visual question answering. arXiv preprint arXiv:2011.13681, 2020.

[177] Ramesh Manuvinakurike, Sovan Biswas, Giuseppe Raffa, Richard Beckwith, Anthony Rhodes, Meng Shi, Gesem Gudino Mejia, Saurav Sahay, and Lama Nachman. Human in the loop approaches in multi-modal conversational task guidance system development. arXiv preprint arXiv:2211.01824, 2022.

[178] Junhua Mao, Jonathan Huang, Alexander Toshev, Oana Camburu, Alan L Yuille, and Kevin Murphy. Generation and comprehension of unambiguous object descriptions. In Proceedings of the IEEE conference on computer vision and pattern recognition, pages 11–20, 2016.

[179] Junhua Mao, Jonathan Huang, Alexander Toshev, Oana Camburu, Alan L Yuille, and Kevin Murphy. Generation and comprehension of unambiguous object descriptions. In Proceedings of the IEEE conference on computer vision and pattern recognition, pages 11–20, 2016.

[180] Amy McGovern and Andrew G. Barto. Automatic discovery of subgoals in reinforcement learning using diverse density. In Proceedings of the Eighteenth International Conference on Machine Learning, ICML '01, page 361–368, San Francisco, CA, USA, 2001. Morgan Kaufmann Publishers Inc.

[181] Christian Meurisch, Cristina A Mihale-Wilson, Adrian Hawlitschek, Florian Giger, Florian Müller, Oliver Hinz, and Max Mühlhäuser. Exploring user expectations of proactive ai systems. Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies, 4(4):1–22, 2020.

[182] So Yeon Min, Devendra Singh Chaplot, Pradeep Kumar Ravikumar, Yonatan Bisk, and Ruslan Salakhutdinov. FILM: Following instructions in language with modular methods. In International Conference on Learning Representations, 2022.

[183] Dipendra Misra, John Langford, and Yoav Artzi. Mapping instructions and visual observations to actions with reinforcement learning. In Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing, pages 1004–1015, Copenhagen, Denmark, 2017. Association for Computational Linguistics.

[184] Seungwhan Moon, Satwik Kottur, Paul A Crook, Ankita De, Shivani Poddar, Theodore Levin, David Whitney, Daniel Difranco, Ahmad Beirami, Eunjoon Cho, et al. Situated and interactive multimodal conversations. In Proceedings of the 28th International Conference on Computational Linguistics, pages 1103–1121, 2020.

[185] Seungwhan Moon, Andrea Madotto, Zhaojiang Lin, Tushar Nagarajan, Matt Smith, Shashank Jain, Chun-Fu Yeh, Prakash Murugesan, Peyman Heidari, Yue Liu, et al. Anymal: An efficient and scalable any-modality augmented language model. In Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing: Industry Track, pages 1314–1332, 2024.

[186] I Munasinghe, J Asensio-Cubero, S Reddy Bethi, ST Huang, S Zylfo, T Nica, A Mitrocsak, S Mezza, et al. Building proactive voice assistants: When and how (not) to interact. arXiv preprint arXiv:2005.01322, 2020.

[187] FC Müller-Lyer. Optische urteilstäuschungen. Archiv für Physiologie Suppl, pages 263–270, 1889.

[188] Simra Nazim, Saba Firdous, Swaroop Ramaswamy Pillai, and Vinod Kumar Shukla. Smart glasses: a visual assistant for the blind. In 2022 International Mobile and Embedded Technology Conference (MECON), pages 621–626. IEEE, 2022.

[189] Van-Quang Nguyen and Takayuki Okatani. A hierarchical attention model for action learning from realistic environments and directives. ECCV EVAL Workshop, 2020.

[190] Jennifer Ockerman and Amy Pritchett. A review and reappraisal of task guidance: Aiding workers in procedure following. International Journal of Cognitive Ergonomics, 4(3):191–212, 2000.

[191] Jennifer J Ockerman and Amy R Pritchett. Preliminary investigation of wearable computers for task guidance in aircraft inspection. In Digest of Papers. Second International Symposium on Wearable Computers (Cat. No. 98EX215), pages 33–40. IEEE, 1998.

[192] Maxime Oquab, Timothée Darcet, Théo Moutakanni, Huy Vo, Marc Szafraniec, Vasil Khalidov, Pierre Fernandez, Daniel Haziza, Francisco Massa, Alaaeldin El-Nouby, et al. Dinov2: Learning robust visual features without supervision. arXiv preprint arXiv:2304.07193, 2023.

[193] Aishwarya Padmakumar, Jesse Thomason, Ayush Shrivastava, Patrick Lange, Anjali Narayan-Chen, Spandana Gella, Robinson Piramuthu, Gokhan Tur, and Dilek Hakkani-Tur. Teach: Task-driven embodied agents that chat. In Proceedings of the AAAI Conference on Artificial Intelligence, volume 36, pages 2017–2025, 2022.

[194] Georgios Pantazopoulos, Malvina Nikandrou, Amit Parekh, Bhathiya Hemanthage, Arash Eshghi, Ioannis Konstas, Verena Rieser, Oliver Lemon, and Alessandro Suglia. Multitask multimodal prompted training for interactive embodied task completion. In Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing, pages 768–789, 2023.

[195] Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. Bleu: a method for automatic evaluation of machine translation. In Proceedings of the 40th annual meeting of the Association for Computational Linguistics, pages 311–318, 2002.

[196] Joon Sung Park, Joseph O'Brien, Carrie Jun Cai, Meredith Ringel Morris, Percy Liang, and Michael S Bernstein. Generative agents: Interactive simulacra of human behavior. In Proceedings of the 36th annual acm symposium on user interface software and technology, pages 1–22, 2023.

[197] Alexander Pashevich, Cordelia Schmid, and Chen Sun. Episodic transformer for vision-and-language navigation. 2021 IEEE/CVF International Conference on Computer Vision (ICCV), pages 15922–15932, 2021.

[198] Rohith Peddi, Shivvrat Arya, Bharath Challa, Likhitha Pallapothula, Akshay Vyas, Jikai Wang, Qifan Zhang, Vasundhara Komaragiri, Eric Ragan, Nicholas Ruozzi, et al. Captaincook4d: A dataset for understanding errors in procedural activities. arXiv preprint arXiv:2312.14556, 2023.

[199] Zhiliang Peng, Wenhui Wang, Li Dong, Yaru Hao, Shaohan Huang, Shuming Ma, and Furu Wei. Kosmos-2: Grounding multimodal large language models to the world. arXiv preprint arXiv:2306.14824, 2023.

[200] Daniel Philipov, Vardhan Dongre, Gokhan Tur, and Dilek Hakkani-Tür. Simulating user agents for embodied conversational-ai. arXiv preprint arXiv:2410.23535, 2024.

[201] Renjie Pi, Jiahui Gao, Shizhe Diao, Rui Pan, Hanze Dong, Jipeng Zhang, Lewei Yao, Jianhua Han, Hang Xu, and Lingpeng Kong Tong Zhang. Detgpt: Detect what you need via reasoning. arXiv preprint arXiv:2305.14167, 2023.

[202] Chiara Plizzari, Gabriele Goletto, Antonino Furnari, Siddhant Bansal, Francesco Ragusa, Giovanni Maria Farinella, Dima Damen, and Tatiana Tommasi. An outlook into the future of egocentric vision. International Journal of Computer Vision, 132(11):4880–4936, 2024.

[203] Bryan A Plummer, Liwei Wang, Chris M Cervantes, Juan C Caicedo, Julia Hockenmaier, and Svetlana Lazebnik. Flickr30k entities: Collecting region-to-phrase correspondences for richer image-to-sentence models. In Proceedings of the IEEE international conference on computer vision, pages 2641–2649, 2015.

[204] Jordi Pont-Tuset, Jasper Uijlings, Soravit Changpinyo, Radu Soricut, and Vittorio Ferrari. Connecting vision and language with localized narratives. In Proceedings of the 16th European Conference on Computer Vision, pages 647–664, 2020.

[205] Mario Ponzo. Intorno ad alcune illusioni nel campo delle sensazioni tattili sull. Archives Italiennes de Biologie, 1911.

[206] Xavier Puig, Kevin Ra, Marko Boben, Jiaman Li, Tingwu Wang, Sanja Fidler, and Antonio Torralba. Virtualhome: Simulating household activities via programs. In Proceedings of the IEEE conference on computer vision and pattern recognition, pages 8494–8502, 2018.

[207] Xavier Puig, Eric Undersander, Andrew Szot, Mikael Dallaire Cote, Tsung-Yen Yang, Ruslan Partsey, Ruta Desai, Alexander Clegg, Michal Hlavac, So Yeon Min, et al. Habitat 3.0: A co-habitat for humans, avatars, and robots. In The Twelfth International Conference on Learning Representations, 2024.

[208] Lu Qi, Jason Kuen, Tiancheng Shen, Jiuxiang Gu, Wenbo Li, Weidong Guo, Jiaya Jia, Zhe Lin, and Ming-Hsuan Yang. High quality entity segmentation. In Proceedings of the IEEE/CVF International Conference on Computer Vision, pages 4047–4056, 2023.

[209] Yuankai Qi, Qi Wu, Peter Anderson, Xin Wang, William Yang Wang, Chunhua Shen, and Anton van den Hengel. Reverie: Remote embodied visual referring expression in real indoor environments. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2020.

[210] Mengxue Qu, Yu Wu, Wu Liu, Xiaodan Liang, Jingkuan Song, Yao Zhao, and Yunchao Wei. RIO: A benchmark for reasoning intention-oriented objects in open environments. In Thirty-seventh Conference on Neural Information Processing Systems Datasets and Benchmarks Track, 2023.

[211] Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, et al. Learning transferable visual models from natural language supervision. In International conference on machine learning, pages 8748–8763. PMLR, 2021.

[212] Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, et al. Learning transferable visual models from natural language supervision. In International conference on machine learning, pages 8748–8763. PMLR, 2021.

[213] Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, Gretchen Krueger, and Ilya Sutskever. Learning transferable visual models from natural language supervision. In ICML, 2021.

[214] Vignesh Ramanathan, Anmol Kalia, Vladan Petrovic, Yi Wen, Baixue Zheng, Baishan Guo, Rui Wang, Aaron Marquez, Rama Kovvuri, Abhishek Kadian, et al. Paco: Parts and attributes of common objects. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pages 7141–7151, 2023.

[215] Varun Nagaraj Rao, Xingjian Zhen, Karen Hovsepian, and Mingwei Shen. A first look: Towards explainable textvqa models via visual and textual explanations. arXiv preprint arXiv:2105.02626, 2021.

[216] Abhinav Rastogi, Xiaoxue Zang, Srinivas Sunkara, Raghav Gupta, and Pranav Khaitan. Towards scalable multi-domain conversational agents: The schema-guided dialogue dataset. In Proceedings of the AAAI conference on artificial intelligence, volume 34, pages 8689–8696, 2020.

[217] Nils Reimers and Iryna Gurevych. Making monolingual sentence embeddings multilingual using knowledge distillation. In Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing. Association for Computational Linguistics, 11 2020.

[218] Arvin Christopher C Reyes, Neil Patrick A Del Gallego, and Jordan Aiko P Deja. Mixed reality guidance system for motherboard assembly using tangible augmented reality. In Proceedings of the 2020 4th International Conference on Virtual and Augmented Reality Simulations, pages 1–6, 2020.

[219] James Outram Robinson. The psychology of visual illusion. Courier Corporation, 2013.

[220] Anna Rohrbach, Marcus Rohrbach, Ronghang Hu, Trevor Darrell, and Bernt Schiele. Grounding of textual phrases in images by reconstruction. In Proceedings of the 14th European Conference on Computer Vision, pages 817–834. Springer, 2016.

[221] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-net: Convolutional networks for biomedical image segmentation. In International Conference on Medical image computing and computer-assisted intervention, pages 234–241. Springer, 2015.

[222] Daniel Rossos, Alex Mihailidis, and Brokoslaw Laschowski. Ai-powered smart glasses for sensing and recognition of human-robot walking environments. In 2024 10th IEEE RAS/EMBS International Conference for Biomedical Robotics and Biomechatronics (BioRob), pages 62–67. IEEE, 2024.

[223] Homagni Saha, Fateme Fotouhif, Qisai Liu, and Soumik Sarkar. A modular vision language navigation and manipulation framework for long horizon compositional tasks in indoor environment, 2021.

[224] Ayaka Sato, Keita Watanabe, and Jun Rekimoto. Mimicook: a cooking assistant system with situated guidance. In Proceedings of the 8th international conference on tangible, embedded and embodied interaction, pages 121–124, 2014.

[225] Manolis Savva, Abhishek Kadian, Oleksandr Maksymets, Yili Zhao, Erik Wijmans, Bhavana Jain, Julian Straub, Jia Liu, Vladlen Koltun, Jitendra Malik, et al. Habitat: A platform for embodied ai research. In Proceedings of the IEEE/CVF international conference on computer vision, pages 9339–9347, 2019.

[226] Enrico Scala. Enhsp. https://sites.google.com/view/enhsp/, 2020. Accessed: 2022-06-20.

[227] Fadime Sener, Dibyadip Chatterjee, Daniel Shelepov, Kun He, Dipika Singhania, Robert Wang, and Angela Yao. Assembly101: A large-scale multi-view video dataset for understanding procedural activities. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pages 21096–21106, 2022.

[228] Pararth Shah, Dilek Hakkani-Tür, Gokhan Tür, Abhinav Rastogi, Ankur Bapna, Neha Nayak, and Larry Heck. Building a conversational agent overnight with dialogue self-play. arXiv preprint arXiv:1801.04871, 2018.

[229] Murray Shanahan, Kyle McDonell, and Laria Reynolds. Role play with large language models. Nature, 623(7987):493–498, 2023.

[230] Pratyusha Sharma, Antonio Torralba, and Jacob Andreas. Skill induction and planning with latent language. In Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), pages 1713–1726, Dublin, Ireland, May 2022. Association for Computational Linguistics.

[231] Lanbo She and Joyce Chai. Interactive learning of grounded verb semantics towards human-robot communication. In Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), pages 1634–1644, 2017.

[232] Lanbo She, Shaohua Yang, Yu Cheng, Yunyi Jia, Joyce Chai, and Ning Xi. Back to the blocks world: Learning new actions through situated human-robot dialogue. In Proceedings of the SIGDIAL 2014 Conference, Philadelphia, US, June 2014.

[233] Bokui Shen, Fei Xia, Chengshu Li, Roberto Martín-Martín, Linxi Fan, Guanzhi Wang, Claudia Pérez-D'Arpino, Shyamal Buch, Sanjana Srivastava, Lyne Tchapmi, et al. igibson 1.0: A simulation environment for interactive tasks in large realistic scenes. In 2021 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), pages 7520–7527. IEEE, 2021.

[234] Mohit Shridhar, Jesse Thomason, Daniel Gordon, Yonatan Bisk, Winson Han, Roozbeh Mottaghi, Luke Zettlemoyer, and Dieter Fox. Alfred: A benchmark for interpreting grounded instructions for everyday tasks. In Proceedings of the IEEE/CVF conference on computer vision and pattern recognition, pages 10740–10749, 2020.

[235] Mohit Shridhar, Xingdi Yuan, Marc-Alexandre Côté, Yonatan Bisk, Adam Trischler, and Matthew J. Hausknecht. Alfworld: Aligning text and embodied environments for interactive learning. arXiv preprint arXiv:2010.03768, 2020.

[236] Tom Silver, Rohan Chitnis, Joshua Tenenbaum, Leslie Pack Kaelbling, and Tomás Lozano-Pérez. Learning symbolic operators for task and motion planning. In 2021 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), pages 3182–3189. IEEE, 2021.

[237] Amanpreet Singh, Vivek Natarjan, Meet Shah, Yu Jiang, Xinlei Chen, Devi Parikh, and Marcus Rohrbach. Towards vqa models that can read. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pages 8317–8326, 2019.

[238] Amanpreet Singh, Guan Pang, Mandy Toh, Jing Huang, Wojciech Galuba, and Tal Hassner. Textocr: Towards large-scale end-to-end reasoning for arbitrary-shaped scene text. In Proceedings of the IEEE/CVF conference on computer vision and pattern recognition, pages 8802–8812, 2021.

[239] Bharat Singh, Hengduo Li, Abhishek Sharma, and Larry S Davis. R-fcn-3000 at 30fps: Decoupling detection and classification. In Proceedings of the IEEE conference on computer vision and pattern recognition, pages 1081–1090, 2018.

[240] Kunal Pratap Singh, Suvaansh Bhambri, Byeonghwi Kim, Roozbeh Mottaghi, and Jonghyun Choi. MOCA: A modular object-centric approach for interactive instruction following. arXiv preprint arXiv:2012.03208, 2020.

[241] Yale Song, Eugene Byrne, Tushar Nagarajan, Huiyu Wang, Miguel Martin, and Lorenzo Torresani. Ego4d goal-step: Toward hierarchical understanding of procedural activities. Advances in Neural Information Processing Systems, 36, 2024.

[242] Sanjana Srivastava, Chengshu Li, Michael Lingelbach, Roberto Martín-Martín, Fei Xia, Kent Elliott Vainio, Zheng Lian, Cem Gokmen, Shyamal Buch, Karen Liu, et al. Behavior: Benchmark for everyday household activities in virtual, interactive, and ecological environments. In Conference on Robot Learning, pages 477–490, 2021.

[243] Shane Storks, Qiaozi Gao, Govind Thattai, and Gokhan Tur. Are we there yet? learning to localize in embodied instruction following. arXiv preprint arXiv:2101.03431, 2021.

[244] Masanori Suganuma, Takayuki Okatani, et al. Look wide and interpret twice: Improving performance on interactive instruction-following tasks. In 30th International Joint Conference on Artificial Intelligence, IJCAI 2021, pages 923–930. International Joint Conferences on Artificial Intelligence, 2021.

[245] Alessandro Suglia, Qiaozi Gao, Jesse Thomason, Govind Thattai, and Gaurav S. Sukhatme. Embodied bert: A transformer model for embodied, language-guided visual task completion. ArXiv, abs/2108.04927, 2021.

[246] Eric D Sun and Ron Dekel. Imagenet-trained deep neural networks exhibit illusion-like response to the scintillating grid. Journal of Vision, 21(11):15–15, 2021.

[247] 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):181–211, 1999.

[248] Andrew Szot, Alex Clegg, Eric Undersander, Erik Wijmans, Yili Zhao, John Turner, Noah Maestre, Mustafa Mukadam, Devendra Chaplot, Oleksandr Maksymets, Aaron Gokaslan, Vladimir Vondrus, Sameer Dharur, Franziska Meier, Wojciech Galuba, Angel Chang, Zsolt Kira, Vladlen Koltun, Jitendra Malik, Manolis Savva, and Dhruv Batra. Habitat 2.0: Training home assistants to rearrange their habitat. In Advances in Neural Information Processing Systems (NeurIPS), 2021.

[249] Gemini Team, Petko Georgiev, Ving Ian Lei, Ryan Burnell, Libin Bai, Anmol Gulati, Garrett Tanzer, Damien Vincent, Zhufeng Pan, Shibo Wang, et al. Gemini 1.5: Unlocking multimodal understanding across millions of tokens of context. arXiv preprint arXiv:2403.05530, 2024.

[250] Jesse Thomason, Michael Murray, Maya Cakmak, and Luke Zettlemoyer. Vision-and-dialog navigation. In 2019 Conference on Robot Learning (CoRL 2019), 2019.

[251] Zhi Tian, Chunhua Shen, Xinlong Wang, and Hao Chen. Boxinst: High-performance instance segmentation with box annotations. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pages 5443–5452, 2021.

[252] Josh Tobin, Rachel Fong, Alex Ray, Jonas Schneider, Wojciech Zaremba, and Pieter Abbeel. Domain randomization for transferring deep neural networks from simulation to the real world. In 2017 IEEE/RSJ international conference on intelligent robots and systems (IROS), pages 23–30. IEEE, 2017.

[253] Dejan Todorović. What are visual illusions? Perception, 49(11):1128–1199, 2020.

[254] Shengbang Tong, Ellis Brown, Penghao Wu, Sanghyun Woo, Manoj Middepogu, Sai Charitha Akula, Jihan Yang, Shusheng Yang, Adithya Iyer, Xichen Pan, et al. Cambrian-1: A fully open, vision-centric exploration of multimodal llms. arXiv preprint arXiv:2406.16860, 2024.

[255] Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, et al. Llama 2: Open foundation and fine-tuned chat models. arXiv preprint arXiv:2307.09288, 2023.

[256] Maria Tsimpoukelli, Jacob L Menick, Serkan Cabi, SM Eslami, Oriol Vinyals, and Felix Hill. Multimodal few-shot learning with frozen language models. Advances in Neural Information Processing Systems, 34:200–212, 2021.

[257] Karthik Valmeekam, Alberto Olmo, Sarath Sreedharan, and Subbarao Kambhampati. Large language models still can't plan (a benchmark for llms on planning and reasoning about change). In NeurIPS 2022 Foundation Models for Decision Making Workshop, 2022.

[258] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention is all you need. In Isabelle Guyon, Ulrike von Luxburg, Samy Bengio, Hanna M. Wallach, Rob Fergus, S. V. N. Vishwanathan, and Roman Garnett, editors, Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, December 4-9, 2017, Long Beach, CA, USA, pages 5998–6008, 2017.

[259] Ramakrishna Vedantam, C Lawrence Zitnick, and Devi Parikh. Cider: Consensus-based image description evaluation. In Proceedings of the IEEE conference on computer vision and pattern recognition, pages 4566–4575, 2015.

[260] Ethan Waisberg, Joshua Ong, Mouayad Masalkhi, Nasif Zaman, Prithul Sarker, Andrew G Lee, and Alireza Tavakkoli. Meta smart glasses—large language models and the future for assistive glasses for individuals with vision impairments. Eye, 38(6):1036–1038, 2024.

[261] Chen Wang, Danfei Xu, and Li Fei-Fei. Generalizable task planning through representation pretraining. arXiv preprint arXiv:2205.07993, 2022.

[262] Noah Wang, Zy Peng, Haoran Que, Jiaheng Liu, Wangchunshu Zhou, Yuhan Wu, Hongcheng Guo, Ruitong Gan, Zehao Ni, Jian Yang, et al. Rolellm: Benchmarking, eliciting, and enhancing role-playing abilities of large language models. In Findings of the Association for Computational Linguistics ACL 2024, pages 14743–14777, 2024.

[263] Peng Wang, Shuai Bai, Sinan Tan, Shijie Wang, Zhihao Fan, Jinze Bai, Keqin Chen, Xuejing Liu, Jialin Wang, Wenbin Ge, et al. Qwen2-vl: Enhancing vision-language model's perception of the world at any resolution. arXiv preprint arXiv:2409.12191, 2024.

[264] Peng Wang, An Yang, Rui Men, Junyang Lin, Shuai Bai, Zhikang Li, Jianxin Ma, Chang Zhou, Jingren Zhou, and Hongxia Yang. Ofa: Unifying architectures, tasks, and modalities through a simple sequence-to-sequence learning framework. In International Conference on Machine Learning, pages 23318–23340. PMLR, 2022.

[265] Qingyue Wang, Liang Ding, Yanan Cao, Zhiliang Tian, Shi Wang, Dacheng Tao, and Li Guo. Recursively summarizing enables long-term dialogue memory in large language models. arXiv preprint arXiv:2308.15022, 2023.

[266] Wenhai Wang, Zhe Chen, Xiaokang Chen, Jiannan Wu, Xizhou Zhu, Gang Zeng, Ping Luo, Tong Lu, Jie Zhou, Yu Qiao, et al. Visionllm: Large language model is also an open-ended decoder for vision-centric tasks. arXiv preprint arXiv:2305.11175, 2023.

[267] Xin Wang, Taein Kwon, Mahdi Rad, Bowen Pan, Ishani Chakraborty, Sean Andrist, Dan Bohus, Ashley Feniello, Bugra Tekin, Felipe Vieira Frujeri, et al. Holoassist: an egocentric human interaction dataset for interactive ai assistants in the real world. In Proceedings of the IEEE/CVF International Conference on Computer Vision, pages 20270–20281, 2023.

[268] Xuan Wang, SK Ong, and Andrew Yeh-Ching Nee. Multi-modal augmented-reality assembly guidance based on bare-hand interface. Advanced Engineering Informatics, 30(3):406–421, 2016.

[269] Zhanyu Wang, Longyue Wang, Zhen Zhao, Minghao Wu, Chenyang Lyu, Huayang Li, Deng Cai, Luping Zhou, Shuming Shi, and Zhaopeng Tu. Gpt4video: A unified multimodal large language model for lnstruction-followed understanding and safety-aware generation. In Proceedings of the 32nd ACM International Conference on Multimedia, pages 3907–3916, 2024.

[270] Zhaoqing Wang, Yu Lu, Qiang Li, Xunqiang Tao, Yandong Guo, Mingming Gong, and Tongliang Liu. Cris: Clip-driven referring image segmentation. In Proceedings of the IEEE/CVF conference on computer vision and pattern recognition, pages 11686–11695, 2022.

[271] Yuetian Weng, Mingfei Han, Haoyu He, Xiaojun Chang, and Bohan Zhuang. Longvlm: Efficient long video understanding via large language models. In European Conference on Computer Vision, pages 453–470. Springer, 2025.

[272] Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, et al. Huggingface's transformers: State-of-the-art natural language processing. arXiv preprint arXiv:1910.03771, 2019.

[273] Benita Wong, Joya Chen, You Wu, Stan Weixian Lei, Dongxing Mao, Difei Gao, and Mike Zheng Shou. Assistq: Affordance-centric question-driven task completion for egocentric assistant. In European Conference on Computer Vision, pages 485–501. Springer, 2022.

[274] M Wooldridge. Conceptualising and developing agents. Proceedings of the UNICOM Seminar on Agent Software, page 42, 1995.

[275] Chenyun Wu, Zhe Lin, Scott Cohen, Trung Bui, and Subhransu Maji. Phrasecut: Language-based image segmentation in the wild. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pages 10216–10225, 2020.

[276] Te-Lin Wu, Satwik Kottur, Andrea Madotto, Mahmoud Azab, Pedro Rodriguez, Babak Damavandi, Nanyun Peng, and Seungwhan Moon. SIMMC-VR: A task-oriented multimodal dialog dataset with situated and immersive VR streams. In Anna Rogers, Jordan Boyd-Graber, and Naoaki Okazaki, editors, Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), pages 6273–6291, Toronto, Canada, July 2023. Association for Computational Linguistics.

[277] Yixuan Wu, Zhao Zhang, Chi Xie, Feng Zhu, and Rui Zhao. Advancing referring expression segmentation beyond single image. In Proceedings of the IEEE/CVF International Conference on Computer Vision, pages 2628–2638, 2023.

[278] Yuxin Wu, Alexander Kirillov, Francisco Massa, Wan-Yen Lo, and Ross Girshick. Detectron2. https://github.com/facebookresearch/detectron2, 2019.

[279] Zhenyu Wu, Ziwei Wang, Xiuwei Xu, Jiwen Lu, and Haibin Yan. Embodied task planning with large language models. arXiv preprint arXiv:2307.01848, 2023.

[280] Guangxuan Xiao, Yuandong Tian, Beidi Chen, Song Han, and Mike Lewis. Efficient streaming language models with attention sinks. In The Twelfth International Conference on Learning Representations, 2024.

[281] Chi Xie, Zhao Zhang, Yixuan Wu, Feng Zhu, Rui Zhao, and Shuang Liang. Described object detection: Liberating object detection with flexible expressions. In Thirty-seventh Conference on Neural Information Processing Systems, 2023.

[282] Ruinian Xu, Hongyi Chen, Yunzhi Lin, and Patricio A Vela. Sgl: Symbolic goal learning in a hybrid, modular framework for human instruction following. arXiv preprint arXiv:2202.12912, 2022.

[283] Bin Yan, Yi Jiang, Jiannan Wu, Dong Wang, Ping Luo, Zehuan Yuan, and Huchuan Lu. Universal instance perception as object discovery and retrieval. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), pages 15325–15336, June 2023.

[284] Jianwei Yang, Hao Zhang, Feng Li, Xueyan Zou, Chunyuan Li, and Jianfeng Gao. Set-of-mark prompting unleashes extraordinary visual grounding in gpt-4v. arXiv preprint arXiv:2310.11441, 2023.

[285] Jihan Yang, Shusheng Yang, Anjali W Gupta, Rilyn Han, Li Fei-Fei, and Saining Xie. Thinking in space: How multimodal large language models see, remember, and recall spaces. arXiv preprint arXiv:2412.14171, 2024.

[286] Yue Yang, Fan-Yun Sun, Luca Weihs, Eli VanderBilt, Alvaro Herrasti, Winson Han, Jiajun Wu, Nick Haber, Ranjay Krishna, Lingjie Liu, et al. Holodeck: Language guided generation of 3d embodied ai environments. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pages 16227–16237, 2024.

[287] Zhao Yang, Jiaqi Wang, Yansong Tang, Kai Chen, Hengshuang Zhao, and Philip HS Torr. Lavt: Language-aware vision transformer for referring image segmentation. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pages 18155–18165, 2022.

[288] Zhengyuan Yang, Zhe Gan, Jianfeng Wang, Xiaowei Hu, Faisal Ahmed, Zicheng Liu, Yumao Lu, and Lijuan Wang. Unitab: Unifying text and box outputs for grounded vision-language modeling. In European Conference on Computer Vision, pages 521–539, 2022.

[289] Qinghao Ye, Haiyang Xu, Guohai Xu, Jiabo Ye, Ming Yan, Yiyang Zhou, Junyang Wang, Anwen Hu, Pengcheng Shi, Yaya Shi, et al. mplug-owl: Modularization empowers large language models with multimodality. arXiv preprint arXiv:2304.14178, 2023.

[290] Yi-Ting Yeh, Maxine Eskenazi, and Shikib Mehri. A comprehensive assessment of dialog evaluation metrics. In The First Workshop on Evaluations and Assessments of Neural Conversation Systems, pages 15–33, 2021.

[291] Shukang Yin, Chaoyou Fu, Sirui Zhao, Ke Li, Xing Sun, Tong Xu, and Enhong Chen. A survey on multimodal large language models. arXiv preprint arXiv:2306.13549, 2023.

[292] Haoxuan You, Haotian Zhang, Zhe Gan, Xianzhi Du, Bowen Zhang, Zirui Wang, Liangliang Cao, Shih-Fu Chang, and Yinfei Yang. Ferret: Refer and ground anything anywhere at any granularity. arXiv preprint arXiv:2310.07704, 2023.

[293] Licheng Yu, Patrick Poirson, Shan Yang, Alexander C Berg, and Tamara L Berg. Modeling context in referring expressions. In Computer Vision–ECCV 2016: 14th European Conference, Amsterdam, The Netherlands, October 11-14, 2016, Proceedings, Part II 14, pages 69–85. Springer, 2016.

[294] Yuhang Zang, Wei Li, Jun Han, Kaiyang Zhou, and Chen Change Loy. Contextual object detection with multimodal large language models. arXiv preprint arXiv:2305.18279, 2023.

[295] Rowan Zellers, Yonatan Bisk, Ali Farhadi, and Yejin Choi. From recognition to cognition: Visual commonsense reasoning. In Proceedings of the IEEE/CVF conference on computer vision and pattern recognition, pages 6720–6731, 2019.

[296] Rowan Zellers, Ari Holtzman, Matthew E Peters, Roozbeh Mottaghi, Aniruddha Kembhavi, Ali Farhadi, and Yejin Choi. Piglet: Language grounding through neuro-symbolic interaction in a 3d world. In Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers), pages 2040–2050, 2021.

[297] Xiaohua Zhai, Basil Mustafa, Alexander Kolesnikov, and Lucas Beyer. Sigmoid loss for language image pre-training. In Proceedings of the IEEE/CVF International Conference on Computer Vision, pages 11975–11986, 2023.

[298] Haolan Zhan, Sameen Maruf, Ingrid Zukerman, and Gholamreza Haffari. Going beyond imagination! enhancing multi-modal dialogue agents with synthetic visual descriptions. In Proceedings of the 25th Annual Meeting of the Special Interest Group on Discourse and Dialogue, pages 420–427, 2024.

[299] Chen Zhang, Yiming Chen, Luis Fernando D'Haro, Yan Zhang, Thomas Friedrichs, Grandee Lee, and Haizhou Li. Dynaeval: Unifying turn and dialogue level evaluation. In Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers), pages 5676–5689, 2021.

[300] Hang Zhang, Xin Li, and Lidong Bing. Video-llama: An instruction-tuned audio-visual language model for video understanding. arXiv preprint arXiv:2306.02858, 2023.

[301] Shilong Zhang, Peize Sun, Shoufa Chen, Min Xiao, Wenqi Shao, Wenwei Zhang, Kai Chen, and Ping Luo. Gpt4roi: Instruction tuning large language model on region-of-interest. arXiv preprint arXiv:2307.03601, 2023.

[302] Yichi Zhang and Joyce Chai. Hierarchical task learning from language instructions with unified transformers and self-monitoring. In Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021, pages 4202–4213, 2021.

[303] Yichi Zhang, Ziqiao Ma, Xiaofeng Gao, Suhaila Shakiah, Qiaozi Gao, and Joyce Chai. Groundhog: Grounding large language models to holistic segmentation. In Proceedings of the IEEE/CVF conference on computer vision and pattern recognition, pages 14227–14238, 2024.

[304] Yichi Zhang, Jiayi Pan, Yuchen Zhou, Rui Pan, and Joyce Chai. Grounding visual illusions in language: Do vision-language models perceive illusions like humans? In Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing, pages 5718–5728, 2023.

[305] Yichi Zhang, Jianing Yang, Jiayi Pan, Shane Storks, Nikhil Devraj, Ziqiao Ma, Keunwoo Peter Yu, Yuwei Bao, and Joyce Chai. Danli: Deliberative agent for following natural language instructions. arXiv preprint arXiv:2210.12485, 2022.

[306] Zheyuan Zhang, Fengyuan Hu, Jayjun Lee, Freda Shi, Parisa Kordjamshidi, Joyce Chai, and Ziqiao Ma. Do vision-language models represent space and how? evaluating spatial frame of refer. ICLR, 2025.

[307] Bo Zhao, Boya Wu, and Tiejun Huang. Svit: Scaling up visual instruction tuning. arXiv preprint arXiv:2307.04087, 2023.

[308] Yang Zhao, Zhijie Lin, Daquan Zhou, Zilong Huang, Jiashi Feng, and Bingyi Kang. Bubogpt: Enabling visual grounding in multi-modal llms. arXiv preprint arXiv:2307.08581, 2023.

[309] Kaizhi Zheng, Kaiwen Zhou, Jing Gu, Yue Fan, Jialu Wang, Zonglin Li, Xuehai He, and Xin Eric Wang. Jarvis: A neuro-symbolic commonsense reasoning framework for conversational embodied agents. arXiv preprint arXiv:2208.13266, 2022.

[310] Lianmin Zheng, Wei-Lin Chiang, Ying Sheng, Siyuan Zhuang, Zhanghao Wu, Yonghao Zhuang, Zi Lin, Zhuohan Li, Dacheng Li, Eric Xing, et al. Judging llm-as-a-judge with mt-bench and chatbot arena. Advances in Neural Information Processing Systems, 36:46595–46623, 2023.

[311] Yaowei Zheng, Richong Zhang, Junhao Zhang, YeYanhan YeYanhan, and Zheyan Luo. Llamafactory: Unified efficient fine-tuning of 100+ language models. In Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 3: System Demonstrations), pages 400–410, 2024.

[312] Yiwu Zhong, Jianwei Yang, Pengchuan Zhang, Chunyuan Li, Noel Codella, Liunian Harold Li, Luowei Zhou, Xiyang Dai, Lu Yuan, Yin Li, et al. Regionclip: Region-based language-image pretraining. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pages 16793–16803, 2022.

[313] Shuyan Zhou, Pengcheng Yin, and Graham Neubig. Hierarchical control of situated agents through natural language. arXiv preprint arXiv:2109.08214, 2021.

[314] Chenchen Zhu, Fanyi Xiao, Andrés Alvarado, Yasmine Babaei, Jiabo Hu, Hichem El-Mohri, Sean Culatana, Roshan Sumbaly, and Zhicheng Yan. Egoobjects: A large-scale egocentric dataset for fine-grained object understanding. In Proceedings of the IEEE/CVF International Conference on Computer Vision, pages 20110–20120, 2023.

[315] Deyao Zhu, Jun Chen, Xiaoqian Shen, Xiang Li, and Mohamed Elhoseiny. Minigpt-4: Enhancing vision-language understanding with advanced large language models. arXiv preprint arXiv:2304.10592, 2023.

[316] Yuke Zhu, Daniel Gordon, Eric Kolve, Dieter Fox, Li Fei-Fei, Abhinav Gupta, Roozbeh Mottaghi, and Ali Farhadi. Visual semantic planning using deep successor representations. In IEEE International Conference on Computer Vision, ICCV 2017, Venice, Italy, October 22-29, 2017, pages 483–492. IEEE Computer Society, 2017.

[317] Yuke Zhu, Oliver Groth, Michael Bernstein, and Li Fei-Fei. Visual7w: Grounded question answering in images. In Proceedings of the IEEE conference on computer vision and pattern recognition, pages 4995–5004, 2016.