# Language-guided navigation and manipulation in robotics using transformers.

Pierre-Louis Guhur

PSL
UNIVERSITÉ PARIS

THÈSE DE DOCTORAT
DE L'UNIVERSITÉ PSL

Préparée à l'École Normale Supérieure

# Navigation et manipulation guidées par le langage en robotique grâce aux transformers.

## Language-guided navigation and manipulation in robotics using transformers.


Soutenue par
**Pierre-Louis Guhur**
Le 1 février 2023


École doctorale n°8548
**DIENS**

Spécialité
**Informatique**

Préparée à
Inria, centre de Paris

Composition du jury :

| | |
|---|---|
| M. Matthieu Cord<br>Sorbonne Université | *Président du jury* |
| M. Wolfram Burgard<br>Technische Universität Nürnberg | *Rapporteur* |
| M. Christian Wolf<br>Naver Labs | *Rapporteur* |
| Mme. Cordelia Schmid<br>ENS | *Co-encadrante* |
| M. Ivan Laptev<br>ENS | *Co-encadrant* |

ENS | PSL

# Résumé

Les progrès réalisés dans le domaine de l'apprentissage automatique ont permis des percées importantes, notamment en vision par ordinateur, en traitement du langage naturel et en robotique. Peut-on aller plus loin et combiner ces domaines de recherche ? Cela développera de nouvelles applications, comme la robotique guidée par le langage, où un robot doit suivre les instructions fournies par un opérateur.

Alors que les humains apprennent à suivre des instructions dès leur enfance, la même tâche est difficile pour des robots, et cela pour plusieurs raisons: (i) le manque de données d'entraînements, (ii) les raisonnements faits sur multiples niveaux d'abstraction, et (iii) l'espace d'actions ayant une haute dimension.

L'objectif de cette thèse est d'améliorer la robotique guidée par le langage en relevant ces défis. Nous décomposons la difficulté de la robotique guidée par le langage en considérant deux types de tâches : (i) un robot mobile doit se rendre à un endroit cible décrit par des instructions ; (ii) les instructions décrivent une séquence d'actions qu'un bras robotique doit opérer sur des objets placés sur une table.

Nos contributions sont les suivantes : (i) pour résoudre le manque de données d'entraînement, nous avons développé une procédure efficace de pré-entraînement basé sur le nouveau jeu de données BnB, (ii) nous avons construit de nouvelles architectures neuronales basées sur une approche hiérarchique pour encoder plusieurs niveaux d'abstractions, et (iii) nous avons proposé une nouvelle méthode pour prédire des actions continus et en sur plusieurs dimensions pour résoudre un grand nombre de tâches.

---

**Mots clés :** Robotique, vision par ordinateur, traitement du langage naturel, transformer

# Abstract

Recent progress in machine learning has enabled groundbreaking improvements notably in computer vision, natural language processing, and robotics. Can we go one step further and combine these research fields? This would allow new applications, such as language-guided robotics, where a robot must follow instructions provided by an operator.

While people learn to follow natural language instructions from their childhood, the same task is difficult for robots. Current challenges include (i) the limited amount of training data, (ii) the multiple levels of reasoning, and (iii) the multi-dimensional continuous action space.

The goal of this thesis is to improve language-guided robotics by addressing these challenges. We break down the difficulty of language-guided robotics by considering two types of tasks: (i) vision-and-language navigation, where a mobile robot must go to a target location, and (ii) vision-and-language manipulation, where a robotic arm should manipulate objects on a tabletop.

Our contributions are the following: (i) we address the scarcity of training data and develop an efficient pre-training procedure based on the new BnB dataset, (ii) we propose a hierarchical approach based on the Transformer architecture to encode several layers of abstractions, and (iii) we propose a new method predicting continuous and multi-dimensional actions for solving a large number of robotics tasks on a tabletop. Methods developed in this thesis have been tested in photo-realistic simulators and on a real-world robot. They have outperformed the state-of-the-art performance on a dozen of benchmarks.

---

**Keywords :** Robotics, computer vision, natural language processing, transformer

# Acknowledgments

I would like to express my deepest gratitude to all those who have supported and contributed to the completion of my doctoral thesis. Their unwavering encouragement, guidance, and assistance have been invaluable throughout my academic journey.

First and foremost, I would like to extend my heartfelt appreciation to my partner, Claire, for her unlimited patience. Not only did you endure the challenges of my thesis, but also supported me through various other projects. Your presence and belief in me have been constant sources of inspiration. This achievement would not have been possible without you by my side.

I am immensely grateful to my family for their unconditional love. To my mother, my brother, his wife, my step-dad, and his children, thank you for always being there for me. I would also like to extend special thoughts to the memory of my late father. The values he instilled in me continue to motivate and inspire my journey.

I would like to express my gratitude to my dearest friends, including Philippe, Farah, Paul, Florian, Matthis, Charles, Agathe, Thomas, and Chloé, for the countless moments of joy and laughter we have shared. Your friendship has been paramount to keep me going. I am also deeply thankful to my friends at Better Vote and at Climate Fresk, particularly Chloé, Paloma, David, Rida, and Antoine, whose determination and dedication inspire me.

My profound appreciation goes to my primary academic advisors, Cordelia Schmid and Ivan Laptev, for their exceptional guidance, expertise, and continuous availability. Their weekly meetings and demanding expectations pushed me to excel and provided the necessary direction for my research. I am also thankful to Makarand Tapaswi and Shizhe Chen for their valuable contributions and inspiring ideas that enriched my work. Our brainstorming sessions will forever be cherished. Additionally, I would like to extend my gratitude to the members of my thesis jury, Matthieu Cord (president), Wolfram Burgard, and Christian Wolf (examiners), for their insightful feedback and constructive evaluations.

I am indebted to the research teams Willow and Sierra at Inria for providing a conducive environment for collaborative research. I would like to express my sincere appreciation to all the individuals who provided technical and intellectual support during my doctoral journey. Yann, Yana, Charlotte, the climate crew (Bertille, Yann, Benjamin), Ricardo (with whom I had the pleasure of co-authoring a paper), Vivien, Armand, Robin, and the colleagues with whom I shared my room (Thomas, Sarah, Wilson), have played instrumental roles in shaping my ideas and providing valuable insights. To all my other colleagues whom I regret not having had more time to spend with, your presence and

camaraderie have been greatly appreciated.

To all those whose names I may have unintentionally omitted but have played a significant role in my academic journey, please accept my sincerest appreciation.

Thank you all for being an integral part of this incredible chapter of my life.

# Contents

# Chapter 1

# Introduction

This thesis addresses problems at the intersection of computer vision, natural language understanding, and robotics. Our main contributions concern language-guided visual navigation and manipulation. In this chapter, we first outline the goal and motivation of our work in Sections 1.1-1.2 and then summarize the challenges and contributions of our work in Sections 1.3-1.4. Section 1.5 provides a list of publications, datasets, and awards obtained in the course of this thesis.

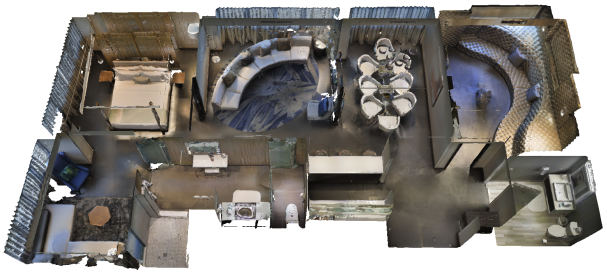
## 1.1 Goal: language-guided robots

Artificial intelligence has a great potential to help people with a variety of tasks including healthcare, transportation, medicine, and education. Over the last decades, machine learning-based algorithms have surpassed people in multiple tasks which have been associated exclusively with human cognitive abilities. Examples of such tasks include playing the game of go [258], the detection of tumoural cells [76], reading comprehension [301], or protein folding prediction [221].

Despite its success in solving particular problems, artificial intelligence (AI) is still far from human intelligence in general. The success of AI also builds on the huge amount of data and computing power that may not scale well to the large variety of existing problems. Thanks to large computational resources, models can be trained on larger datasets than humans could study over their lifetime. Scaling AI to a variety of problems with limited data, hence, requires new methods with a higher degree of generalization.

Animals and humans exhibit learning abilities and understandings of the world that are far beyond the capabilities of current AI [176]. For instance, if a child learns how to speak a language with limited exposure, a teenager can similarly learn to drive a car in about 20 hours of practice. People and other animals can understand how their environment works and interact with it while using a limited amount of observations and trials. Moreover, they demonstrate a stronger level of robustness than current AI systems [61].

Current AI systems mostly have indirect knowledge about the physical world and



**(a) Navigation:** a mobile robot follows instructions to find a target location in an unseen photo-realistic environment, on Matterport [43]

**(b) Manipulation:** a robotic arm follows instructions to execute household tasks in RLBench [143]

**Figure 1.1:** An overview of the two robotics tasks considered in this thesis.

arguably lack embodied intelligence. To reduce the gap between artificial and human intelligence, AI systems may need to acquire knowledge through direct interaction with physical environments or their simulations. Towards this goal, AI systems can be equipped with bodies that (a) obey physical laws such as gravity and friction, (b) are able to perceive the environment through visual, touch, and other sensors, and (c) can change the state of the environment through actuators. In the rest of this thesis, we refer to such systems as embodied agents or robots. In particular, we consider scenarios where robots must follow instructions either for navigating the environment or for manipulating objects as illustrated in Figure 1.1. While a 5-year-old child would be able to solve these tasks, we will see that they remain challenging for robots.

Solving navigation and manipulation tasks requires understanding visual sensor data, and natural language while acting in an unknown environment. While much progress has been done on these tasks separately, language-guided visual navigation and manipulation require a joint approach. The goal of this thesis is to present such a joint approach that can leverage limited annotated resources using weakly-supervised learning, reinforcement learning, and new architectures.

The first part of this thesis is focused on navigation tasks, where a mobile robot is supposed to navigate to a target location, inside an indoor environment that it has never seen before. As described in Figure 1.2, instructions provided to the robot consist either of step-by-step indications, such as in "walk straight then turn right and walk down the hallway until you get near the large painting of a man", or of a description of the target location, as in "Go to the dining room and pet the dog". In particular, we investigate how the robot can efficiently cope with unseen objects and unseen referring expressions. We approach the problem from two directions by (a) exploring additional readily-available data in combination with auxiliary losses and (b) new neural architectures based on recent Transformer models.

In the second part of the thesis, we turn to a related and less addressed problem of language-guided manipulation. Compared to navigation, object manipulation is more challenging as it requires appropriate contact between the object and the robot end-effector. As illustrated in Figure 1.3, we consider here object manipulation tasks

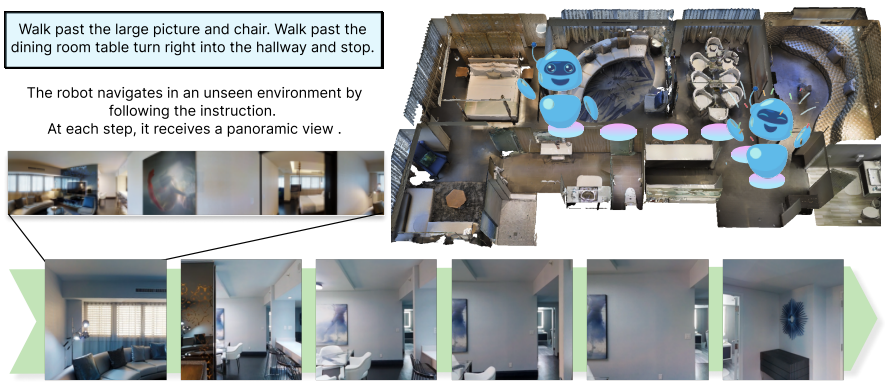



**Figure 1.2:** An example of a language-guided visual navigation task. The mobile robot receives instructions as well as a panoramic view (obtained from a 360 degrees rotation) at its current location. It must predict the sequence of locations to reach a target location.

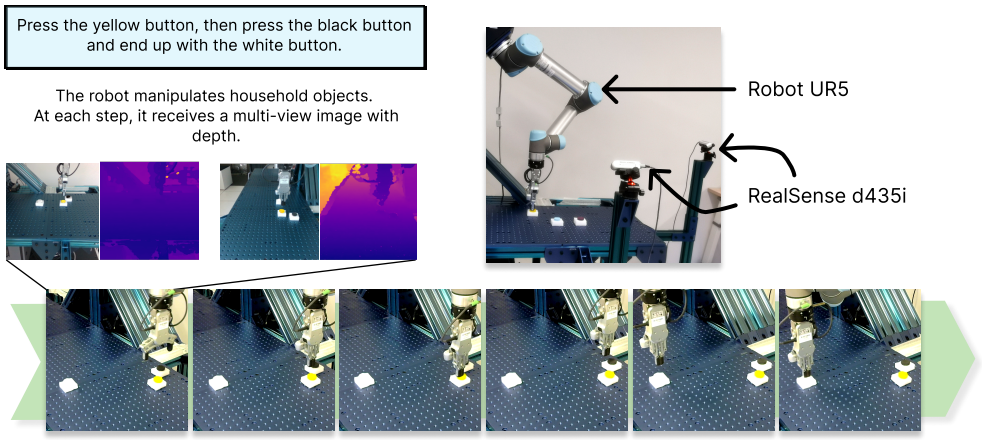



**Figure 1.3:** An example of a language-guided manipulation task. The robotics arm receives instructions as well as multiple views of the countertop.

using a robotics arm. We show that with a single architecture, we are able to learn several dozens of robotics tasks. We also demonstrate promising generalizations to human-written natural language instructions and to unseen variations of a given task.

## 1.2 Motivation

Robots can reduce our workload, improve our productivity and safety, and extend capabilities beyond what the human body and mind can do on their own. Therefore, robots that can follow instructions in unknown environments are beneficial for a myriad of tasks and can open up new possibilities for applications in an increasing number of domains, such as healthcare, education, search and rescue, and logistics.

**Autonomous navigation.** Search and rescue are key applications for robots that can navigate unknown environments. Autonomous robots could be deployed in hazardous locations, such as collapsed buildings, landmines, or contaminated areas. The benefits could be huge when it comes to saving lives in these scenarios, and the number of potential applications is growing. For example, fire departments are already using robots to find survivors in dangerous buildings [35], and robots able to communicate with those survivors would help them by offering emotional and psychological support, or warn them about dangers, such as gas leaks or unstable structures.

In addition to search and rescue, these robots could also be applied to urban exploration. Autonomous robots can be used to gather data in challenging and seemingly inaccessible areas, such as underground subway systems and sewers. Robots are already being used in many cities to inspect pipes, sewer lines, and other vital infrastructure [343]. This type of data gathering combined with social skills can provide invaluable insights into urban development, allowing city planners to make better decisions about citywide infrastructure.

Another exciting place where robots can be used is in the area of logistics. Autonomous robots are already used to more efficiently move goods around a warehouse [137], but they require challenging and expensive programming, slowing down their deployment. Robots able to follow instructions could more easily be deployed in existing factories [325]. This would drastically reduce the costs associated with manual labor and increase efficiency in various sectors.

**Healthcare.** In healthcare, robots have been proven to be able to fulfill therapeutic roles, like helping children with learning disabilities [324], motivating teens with diabetes to exercise, and keeping elderly Alzheimer's patients company [274].

**Education.** Robots able to communicate with people have also the potential to be useful in education because they can provide personalized and interactive learning experiences for students. For example, robots could be used to provide one-on-one tutoring sessions, where the robot can answer questions, provide feedback, and reinforce learning objectives [227]. Additionally, social robots can serve as a bridge between physical and digital learning environments, allowing students to access resources and content in a more engaging and efficient manner. As such, social robots can be used to supplement traditional teaching methods, providing students with an enhanced learning experience that is both effective and enjoyable. Robots have already been tested and

Belpaeme *et al*. [29] shows that on restricted tasks, robots used as tutors or peer learners can achieve a similar impact that human tutoring.

**Understanding of our brain.** This thesis builds on the Transformer neural network architecture. Exploration of neural network architectures may give us hints towards an understanding of the human brain [326]. This understanding can help us to better model language [42], which can lead to improved natural language processing applications, such as voice assistants and machine translation. Additionally, a better understanding of how our brains process information can help us to develop more effective artificial intelligence algorithms [91]. Indeed, the neocortex has also highly uniform architecture, although it is involved in numerous tasks, such as sensory perception, language, and generation of motor commands.

**Enhancing further research.** First of all, studies at the interface between computer vision, natural language processing, and robotics could strengthen research on each field independently, notably when consolidating them into a unique formulation. Consolidation is a major milestone for any scientific field, a valuable recognition of its maturity. Secondly, a ubiquitous architecture also helps to learn from the enormous volume of data available on the Internet (more than 500 hours of videos are uploaded to YouTube every minute [342] and almost 800 billion webpages have been recorded in the Wayback Machine [17]). Several research papers [220, 337] showed that leveraging such datasets can boost performance for downstream tasks. Similarly, in Chapter 3, we will see how we can leverage samples from a rental marketplace to pre-train a language-guided robot to navigate in unseen environments.

## 1.3 Challenges and contributions

By encompassing research topics related to natural language processing, robotics, and computer vision, the idea of controlling robots using instructions is cumulating their difficulties. In this Section, we go through some of the most challenging ones.

### 1.3.1 Diversity of the real world

Modeling our reality is a fundamental challenge due to its complexity and diversity. In this perspective, the Matterport3D simulator, used in Chapters 3 and 4, provides reconstructions of environments using RGB-D photos. But, collecting these photos is costly, and it can hardly be done at scale, whereas our algorithms are trained on a portion of them, and then tested on the others. For example, an unseen environment contains a Christmas tree, but how can we expect the model to "stop in front of the Christmas tree" when such an object does not appear in any of the training environments?

**Human-written instructions.** The large diversity does not only encompass objects, but also our language. Indeed, not only vocabularies are extensive, with each word having a large number of synonyms, and our syntax is flexible, but also our utterances must be considered in a specific context. For example, if we ask a robot to bring us water while lying on a sofa, we are implicitly asking the robot to use a glass of water and not a bottle. Since we have access only to a limited amount of instructions during training,

our models again suffer from a gap between training and testing. In this thesis, we will consider several strategies: in Chapter 3, we study how we can create new instructions using other datasets; in Chapter 5, our model is trained on synthetic instructions and then tested on human-written instructions thanks to a powerful CLIP's embeddings [254].

**Robots must learn with limited demonstrations.** Having a limited number of instructions also means that we have a limited number of demonstrations, and this is particularly challenging for supervised learning algorithms, such as behavioral cloning, which are known to easily overfit due to their exposure bias [262, 308]. As a solution, we will study how self-supervised learning and reinforcement learning can help us in Chapter 4. The scarcity of demonstrations is all the more penalizing that the action space is large. Indeed the larger the action space, the smaller it has been explored during training with a limited set of demonstrations. In Chapter 3, we will consider training our models on a large-scale dataset collected in a rental marketplace. In Chapters 3 and 4, we partially circumvent this problem by considering environments discretized into a graph smaller than one hundred nodes. In Chapter 5, the problem must be directly addressed, because our action space is made of 7 dimensions. We will see that our offered model can solve this issue.

### 1.3.2 Partially-observable environments

Throughout this thesis, we are interested in tasks, that require solving a sequence of steps in a specific order. For example, when a robot is asked to "clean the sink of the bathroom located on the second floor", the robot must first find the stairways, walk up the stairs, find the bathroom, and finally clean the sink. Sequential tasks raise a series of challenges: to avoid climbing several floors, the robot must remember that it has already climbed one floor, and hence its previous actions. Moreover, the robot might not find the bathroom on its first attempt, but it could enter several rooms first: how can it remember what has been seen before?

**Long-term planning** is hence required, but it is also challenging because of the exponentially increasing number of possible combinations of actions that can be taken as the time horizon increases. This results in large search spaces and increases the complexity of the problem, making it more difficult to find optimal solutions. Additionally, there may be hidden patterns or dependencies between different points in the sequence that a model may not be able to detect. Long sequences also usually require data from previous timesteps to make decisions at later times, which makes handling temporal data a challenge. In Chapters 4 and 5, we tackle this issue by including previous steps when inferring the next action.

**High dimensional observations.** Including the previous steps is not an easy task, since inputs have a high dimension. High dimensional data are known to be challenging to deal with due to the curse of dimensionality [162]. In the context of this thesis, we will use the transformer architecture, which was demonstrated to support high dimension data [136]. However, this performance comes at a price, since the transformer suffers from a $O(N^2)$ algorithmic complexity, where $N$ is the number of tokens provided to the transformer. A token is a small piece of data, such as a block of pixels or a segment of a word. Through this thesis, $N$ is particularly high: its value is between a few hundred

to a thousand dimensions. Indeed, our transformer-based architectures are fed with instructions and with multi-views (or even panoramic views). Moreover, we show the positive impact of reasoning not only from the instant when the robot generates the next command but also from all previous ones.

**Multiple levels of abstraction.** A solution to face this problem, as well as the scarcity of resources is to limit what needs to be learned by decomposing our problem into several layers of abstraction: (i) visual and textual representations extract meaning from raw data, (ii) motion planning computes a sequence of valid configurations to move a robot, (iii) and task planning breaks down a task into a sequence of steps executable by the motion planner. It is noticeable that this layout of layers is naturally conducted by humans. For example, when we are driving, our eyes observe the scene at a high frame rate (we are able to detect any change in the environment very quickly). But, reacting to a pedestrian appearing suddenly takes roughly a second, because we need to update our short-term plan. Moreover, in case we miss an exit on the road, we would need several seconds to come up with a new long-term plan. This thesis focuses on learning a task planner while using pre-trained models for extracting visual and textual representations and out-of-the-self inverse kinematics as a motion planner.

### 1.3.3 Real-world experiments

**Hardware limitations.** While RGB cameras provide higher-resolution images than human eyes, most of the other robotic sensors remain noisy and inaccurate. In particular, depth sensors, used in Chapter 5, suffer from many artifacts such as ghost points and important Gaussian noise. Robotic hardware is also limited by grippers. While precise grippers exist [4], their costs and their fragility are prohibitive for public research laboratories. Grippers have generally between two to three fingers with only a single degree of freedom, meaning that a robot can only close or open its pawn. Moreover, grippers have rarely haptic sensors able to provide feedback signals. Moreover, robots are slow: even pressing a button takes roughly 5 seconds. Running several millions of *episodes* (a complete trial of the robot) as we did during training is impossible to do. Another limitation to running experiments in the real world is the fragility of the hardware. For example, during this thesis, a bug in the motion planner severely damaged the gripper. As a consequence, interactions of a robot with its environment remain limited. While a human baby can spend several years playing with toys to discover object affordances, and learn manipulation skills, learning robotics policies from years-long real-interaction data is impractical. Instead, we preferred to focus on simplified versions of robots running on a simulator [247]. Simulated embodied agents follow physical rules, such as gravity or collisions. However, they benefit from defectless sensors and accurate actioners.

**Simulation-to-reality gap.** Simulated embodied agents have several advantages: (i) obviously, any damage caused on a simulator has no impact; (ii) simulators provide a high-frequency rendering, allowing to run episodes faster than in the real world ; (iii) simulators can be run in parallel, providing a fleet of robots at a considerably lower cost than an equivalent real one. However, teaching skills to an embodied agent does not mean we are able to execute such skills in the real world. First, because they

are visually unrealistic [234] (with the exception of photo-realistic renderings [43]), providing a large domain gap between reality and simulation. Second, contacts are typically hard to simulate: it happens frequently that a gripper is going through an object. As a consequence, grasping is dramatically simplified: if the embodied agent closes its gripper while an object is located close to the gripper, the object is "magically" grasped, even if it is physically impossible. In Chapter 5, we address this challenge, by finetuning a model pre-trained on a simulator with only one hundred demonstrations collected on a real-world robot.

## 1.4 Outline

This thesis consists of six chapters including this introduction. Our main technical contributions are separated into two parts: the first one is focused on language-guided navigation tasks, whereas the second addresses language-guided manipulation.

**Literature survey.** Chapter 2 reviews the related works in the literature on robotics controlled by instructions with a particular focus on (i) tasks based on navigation, dubbed vision-and-language navigation (VLN), and (ii) tasks based on manipulation, called similarly VLM for vision-and-language manipulation.

**Self-supervised pre-training for sequential tasks.** To improve generalization to unseen environments, we study strategies to pre-train a transformer architecture in Chapter 3. The generic formulation of the transformer deprived this architecture of prior knowledge, which allows this architecture to work well with different types of modalities. Although previous methods tried to pre-train a transformer on generic image-caption datasets, their approaches provided limited improvements in results. We introduce BnB, a large-scale and diverse in-domain VLN dataset collected from hundreds of thousands of listings from an online rental marketplace. As such, BnB is closer to the VLN domain than generic image-caption datasets. We mitigate even further the domain gap between image-caption pairs and the VLN task by rephrasing descriptions into navigation-like instructions and by adding contextual views to a single image. Hence, BnB contains (i) more than a million image-caption pairs, and (ii) automatically generated millions of VLN path-instruction pairs. Furthermore, a shuffling loss is proposed to enhance temporal order reasoning inside path-instructions pairs.

First, we use BnB to pre-train our Airbert model in a *discriminative setting*. In this setting, Airbert must decide which trajectory fits the most the instructions given a pool of several candidate trajectories. Airbert needs hence an internal representation of the environment, as visiting each candidate trajectory in the real world is not realistic. To circumvent this limitation, Airbert is also extended to a *generative setting*, where it is predicting its next step given previously visited locations. Airbert is here trained to fit a ground truth demonstration with supervised learning. It induces an exposure bias as the training set is not explored enough. We fix this bias using reinforcement learning which increases the exploration of training environments. Overall, we show that Airbert outperforms state-of-the-art for R2R and REVERIE benchmarks. Moreover, our in-domain pretraining significantly increases performance on a challenging few-shot VLN evaluation, where we train the model only on VLN instructions from a few houses.

**A hierarchical approach for the transformer.** In the generative setting previously presented, the previously visited locations are encoded through a state vector. Similarly to recurrent neural networks, the state vector is propagated from step to step. It is suboptimal because the state vector encodes only a summary of the history. We would prefer to infer the next actions of a robot given all images from all previous steps. However, the robot observes its environment through multiple views, providing a input that are too large to be processed directly with a Transformer architecture. To mitigate this constraint, we employ a hierarchical approach in Chapter 4: each panoramic view is encoded into a single token, whereas, for the current time step, each image of the panoramic grid is encoded into a token. which first encodes individual images, then models spatial relation between images in a panoramic observation, and finally takes into account temporal relation between panoramas in history. Our obtained model dubbed a History Aware Multimodal Transformer (HAMT), is hence combining text, history, and current observation to predict the next action.

**Manipulation-based tasks in the real world.** So far, we consider scenarios in which robots must find a target location given some human-written instructions. In Chapter 5, We consider another scenario: a robotic arm with a gripper lying on a table has to accomplish various manipulation-based tasks given simple natural language instructions. Yet, some of these tasks can be highly challenging as it requires fine-grained motor control (*e.g.* playing Jenga), and long-term memory (*e.g.* when rearranging groceries in a cupboard) well as a generalization to previously unseen tasks and environments. To address these challenges, we propose a unified transformer-based approach that takes into account multiple inputs. In particular, our transformer architecture integrates (i) natural language instructions and (ii) multi-view scene observations while (iii) keeping track of the full history of observations and actions. Such an approach enables learning dependencies between history and instructions and improves manipulation precision using multiple views. We evaluate our method on the challenging RLBench benchmark and on a real-world robot. Notably, our approach scales to 74 diverse RLBench tasks and outperforms the state-of-the-art. We also address instruction-conditioned tasks and demonstrate excellent generalization to previously unseen variations.

**Conclusion.** We conclude in Chapter 6 with a summary of contributions, a discussion of open problems, and future work.

## 1.5 Publications and Awards

In the following, we list the publications, awards, as well as software and dataset releases that were obtained during this thesis.

### 1.5.1 List of Publications

Contributions of this thesis have led to the following publications:

1. Pierre-Louis Guhur, Makarand Tapaswi, Shizhe Chen, Ivan Laptev, and Cordelia Schmid. “Airbert: In-domain pretraining for vision-and-language navigation.”

In Proceedings of the IEEE/CVF International Conference on Computer Vision. 2021.

2. Shizhe Chen, Pierre-Louis Guhur, Cordelia Schmid, and Ivan Laptev. "History aware multimodal transformer for vision-and-language navigation." Advances in Neural Information Processing Systems. 2021.

3. Pierre-Louis Guhur, Shizhe Chen, Ricardo Pinel, Makarand Tapaswi, Ivan Laptev, and Cordelia Schmid. "Instruction-driven history-aware policies for robotic manipulations". In Proceedings of the Conference on Robotics Learning (Oral). 2022.

In addition, three other papers were published during this Ph.D., extending the works presented in this thesis on language-guided navigation tasks, and in language referring tasks in 3D environments.

1. Shizhe Chen, Pierre-Louis Guhur, Makarand Tapaswi, Cordelia Schmid, and Ivan Laptev. "Think Global, Act Local: Dual-scale Graph Transformer for Vision-and-Language Navigation." In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (Oral). 2022.

2. Shizhe Chen, Pierre-Louis Guhur, Makarand Tapaswi, Ivan Laptev, and Cordelia Schmid. "Learning from Unlabeled 3D Environments for Vision-and-Language Navigation" In Proceedings of the IEEE/CVF European Conference on Computer Vision. 2021.

3. Shizhe Chen, Pierre-Louis Guhur, Makarand Tapaswi, Cordelia Schmid, and Ivan Laptev. "Language Conditioned Spatial Relation Reasoning for 3D Object Grounding" Advances in Neural Information Processing Systems. 2021.

### 1.5.2 List of Awards

Our method [58] was ranked first at the REVERIE and SOON Challenges organized in conjunction with the ICCV 2021 Workshop on Human Interaction for Robotic Navigation. Our follow-up submission has obtained the second place at the REVERIE Challenge @ CSIG 2022 on two tracks.

### 1.5.3 List of Datasets

**BnB.** We collect 1.4 million images (half of them have a caption) over 144 thousand listings in a rental marketplace. This is several orders of magnitude higher than the number of images in Matterport3DSimulator [10]. Airbnb has a total of 6 million listings, so using the provided code source might provide 50 times more images.

**R2R-Back.** In Chapter 4, we assess the efficiency of our history module. That is why we offer a new variant of R2R [9] in which the robot must reach the target location and be able to go back to its initial location.

**R2R-Last.** Simularly, to judge the spatial reasoning of our approach, we create another variant, where only the last sentence of the instructions is provided.

**Auto-VLN.** In Vil3DRel [56], we offer a new dataset made of pseudo-labels on 900 3D environments. This dataset has been successfully employed to pretrain a model and obtains new state-of-the-art results.

**RLBench testing instructions.** In Chapter 5, we collect 162 human-written instructions to test our approach on two tasks: one where the robot presses up to three colored buttons, and another where the robot stacks colored blocks. In both cases, the sequence of colors is provided by the instructions.

# Chapter 2

# Literature review

This Chapter surveys the literature on vision, language, and robotics. We first review recent progress in computer vision and robotics in Sections 2.1 and 2.2 respectively. Related work on the intersection of natural language understanding, computer vision and robotics is then presented in Section 2.3 with a particular focus on language-guided manipulation in Section 2.3.3 and language-guided navigation in Section 2.3.4.

## 2.1 Computer vision

In recent years, the field of computer vision has seen enormous progress due to the emergence of deep learning algorithms. Major factors in this progress have been the emergence of new large-scale annotated datasets [26] and the significant increase in computing power, which together enabled successful performance of convolutional neural networks (CNNs) [177] for a large range of problems such as image and video classification [298], object detection [199] and image segmentation [260].

**CNNs** are composed of layers of neurons, where each neuron is connected to a certain number of neighboring neurons in the previous layer, as well as a certain number of weights associated with each of those connections. CNNs have been used extensively in the field of image recognition and classification. Notably, LeCun *et al*. [177] used CNNs to classify handwritten digits from the MNIST dataset in 1998, achieving a classification accuracy of 98%. Since then, a variety of other CNN architectures have been developed for image recognition and classification, such as AlexNet [168], VGGNet [280], GoogLeNet [293], and ResNet [112]. These architectures have been used to achieve state-of-the-art results on a variety of tasks, such as image classification [66, 177], object detection, and semantic segmentation [195, 90, 78]. The success of CNNs on this task led to more challenging applications, such as optical flow estimation [306], human pose estimation [310], or shape reconstruction [109]. The predominance of CNNs in computer vision applications can be explained by their efficient convolutional structure, but this structure may also impose disadvantages. For example, CNNs have a limited receptive field, which can limit the amount of context they can use in the decision-making process [293].

**Transformer** [312] is an architecture based on a self-attention mechanism, which allows the network to learn long-term dependencies between input and output. Transformers have been originally developed for natural language processing, but they were recently found to outperform CNNs [72], notably for image classification [18], image segmentation [287], and image generation [45]. This success can be impeded by the fact they are better suited to capturing long-range correlations between input features [360], and they have less bias in their formulation [336]. Furthermore, Transformers enable convenient joint modeling of multiple modalities such as text, images, video, and sound [136, 294, 204]. However, their adaptation to computer vision remains an open challenge, partially because they require additional resources such as memory and computing to be able to extract features from the data [174]. Moreover, their lack of inductive biases makes them hard to generalize to small-scale datasets [18].

In this work, we aim to address the above limitations of the Transformer architecture, by collecting additional datasets (Ch. 3), reformulating this architecture with a hierarchical approach (Ch. 4), or by combining it with CNNs (Ch. 5), allowing us to capture long-range correlations using a very limited corpus of demonstrations.

## 2.2 Robotics

**Control** is an essential component of robotics, as it allows robots to interact with the environment and carry out their intended tasks. Real-time control of autonomous robots requires an efficient and robust control architecture, which can rapidly respond to environmental changes. This includes the ability to detect and avoid obstacles, as well as accurately track and follow a given path. To improve the accuracy and reliability of robotic motion control, model predictive control [38] uses a model of the system to predict the future state of the system and then optimizes a cost function to determine a control action. Furthermore, control architectures suitable for autonomous robots need to be robust enough to handle dynamic environments, and for this purpose, adaptive control [22] uses feedback to adjust the parameters of a controller in response to changes in the environment or system. In the context of this thesis, we assume the environment is controlled and fixed, allowing to use of more basic controllers, such as inverse kinematics [171], a technique that solves the problem of finding the joint angles necessary to achieve a desired end effector position.

**Motion planning.** Robotics research has made great strides in motion planning [338], to generate collision-free paths that can be followed by the robot. Much of the existing work assumes a known structure of the environment. Notably, visual servoing [47] is a process by which a robot is able to navigate through a given environment by relying on visual feedback from its sensors, and it has been widely used in applications such as robotic navigation and robotic manipulation. Furthermore, rapidly exploring random trees (RRT) [151] have been applied to motion planning. RRT generates a tree-like structure to efficiently search for the optimal path from a given start to a given goal. RRT is often used for visual servoing applications since RRT is able to generate paths that are close to the desired goal and can be easily followed by the robot. However, having an access to a known structure of the environment is not realistic for unknown

and complex environments [289].

**Simultaneous Localization and Mapping** [304] (SLAM) has been introduced to overcome this limitation. It enables robots to create a map of their environment and localize themselves within the map. SLAM algorithms use a combination of computer vision, odometry, and sensor fusion to create accurate maps and enable robots to autonomously navigate in unknown environments. Techniques for SLAM include Monte-Carlo localization [28], extended Kalman filter [307], Rao–Blackwellized particle filter [98], visual odometry [341] and optic flow [352].

**Policy learning** is an alternative line of work that directly learns sensorimotor policies from raw sensor inputs. For example, imitation learning methods [133] allow robots to learn from demonstrations of desired behaviors, while reinforcement learning (RL) [292] methods enable robots to learn from a reward signal. These methods have been successful for a variety of robotic tasks, such as robot navigation, object manipulation, and locomotion. Nevertheless, learning-based methods are often data-hungry and require extensive training in simulation before being able to transfer to the real world. This is known as the sim-to-real problem [356] and is an area of active research. Recent work has attempted to address this issue by combining imitation and RL [288], using domain randomization to make models more robust to domain shifts [173], and using meta-learning to enable an agent to rapidly adapt to new environments [82]. In our work, we present a method for learning visuomotor controllers for unknown and complex environments. Our method is based on a combination of imitation learning and RL and is capable of generalizing to unseen environments. We also tested the method developed in Chapter 5 on a real-world robot.

## 2.3 Language-guided agents

Does language influence the way we think? Linguists [263] have been passionate about this question for a century. Whorf [327] suggests the linguistic relativity hypothesis saying that language determines thought and hence cognition abilities. Later, the role of language has experimentally been studied in the acquisition of counter-factual analysis [33, 84], memory [286], navigation [182], and color [108, 206, 264].

Does it also hold for AI systems? In other words, could AI systems benefit from embodiment and language acquisition? According to Bisk *et al.* [32], "in addition to learning basic physical properties of the world from interaction, [embodiment] also allows the agent to construct rich pre-linguistic representations from which to generalize."

### 2.3.1 Connecting language to embodied agents

**Translating natural language to formal language.** Winograd [329] offered seminal works in this direction: using exhaustive symbolic rules to translate natural language into a set of executable instructions, his program called SHRDLU is able to assemble geometric shapes given an order provided by an operator. Further works improved this idea by translating natural language instructions into a set of pre-defined actions [300],

lambda calculus expressions [349, 351], or reward functions [282]. Those works have been using probabilistic graphical models [158, 124], weakly supervised learning [19], or sequence-to-sequence recurrent networks [95, 20].

Using formal language as an intermediary connection between language and sensorimotor processing has a significant consequence: the environment does not help to learn the language. In other words, the learning process is unilateral, contrary to previous examples from embodied language processing. To address this limitation, can we bring closer language and sensorimotor processing?

**Connecting language to vision.** A fundamental task is semantic segmentation, which consists in assigning each pixel of an image to a semantic label. First neural architectures [177, 168] emerged for this purpose. Further improvements were suggested, such as regional CNNs [92, 93], feature pyramids [194], recurrent CNNs [248], or encoder-decoder [6]. Those solutions have thereafter been employed for visual question answering [16, 132] and video question answering [298, 179], offering innovative approaches based on LSTMs [146], and hierarchical learning [357].

**Connecting language to spatial localization.** Going one step further, several vision and language tasks assess the agent's abilities to ground spatial referring expressions [252]. In ReferIt [153], the agent receives a referring expression describing the localization of an item in an image, and the agent must localize such objects. The tasks were popularized with the 2D ReferIt Game [153] and it was developed in further datasets [71, 160, 166]. A similar task [203, 167] consists in detecting visual relationships in images.

**Connecting language to temporal localization** is another attempt to bring closer language understanding with an agent's cognition processing. The temporal activity localization via language query [87, 52, 15] formulated as a clip retrieval task based on the description of an activity. Methods are based on auxiliary losses [335], contrastive learning [214], and transformers [208].

These aforementioned tasks and methods are restricted to 2D representations of an environment, failing to capture the true 3D nature of our world. Chen *et al*. [50] mentioned notably that the "first giraffe" is particularly challenging to recognize on a 2D image of two interlaced giraffes. In the general case, an image alone might not provide precise position cues, which are essential for robotic tasks such as grasping.

### 2.3.2 Referring expressions

We consider now tasks that were inspired by a 2D setup and were extended to a 3D environment.

**Datasets.** Similarly to the 2D case, a referential expression is identified common objects in a 3D environment, and the embodied agent must retrieve the target object. The SUN-Spot [217] is the first dataset doing so, but with a relatively little amount of annotations. It was later extended in ReferIt3D [1] and ScanRefer [50]. These datasets provide instructions to localize annotated objects in the ScanNet dataset [63]. SUNRefer [198] and OCID-Ref [319] offer a more challenging setup on which the embodied agent must deal with occlusions.

**Methods.** Approaches for this task usually consist in two steps: first objects are detected,

and then they are match to instructions [1, 50, 259], notably by using a graphical representation of the scene [348, 129] to infer spatial relations. Latest methods [340, 110, 259, 130, 209, 355] have adopted the transformer architecture [313], allowing to detect and match simultaneously [138, 209]. ReCLIP [291] is using precomputed embeddings on the instructions.

### 2.3.3 Language-guided manipulation

Interest for robotics in research communities was boosted with international competitions, notably RoboCup [21], and Amazon Robotics Challenge [75], but also easily-reproducible benchmarks such as ARCV [180], OpenGrasp [181], and VisGrab [161]. Although the robot remains fixed, manipulation tasks are particularly challenging as they require dealing with a continuous action space across several dimensions.

**Simulators and datasets.** Simulations of physically-realistic robots require the development of a specific range of simulators. For example, ManipulaTHOR [74] introduced realistic interaction with objects using a mobile manipulator with seven degrees of freedom.

Based on the simulator CoppeliaSim [261] (formerly V-REP), RLBench [143] offers more than 100 tasks with automatically generated demonstrations. Moreover, RLBench is composed of synthetic-generated instructions. Built as an extension of RLBench, VLMBench [358] is focused on language-guided embodied agents. It is composed of eight tasks but thousands of variations of these tasks.

CALVIN [219] is another simulated benchmark to learn long-horizon tasks specified with human language. Compared to VLMBench, CALVIN tasks are more complex in terms of sequence length and language. Moreover, it can evaluate embodied agents in unseen environments. Drawing inspirations from Learning from Play [210, 211], CALVIN provides annotations with 20k language directives of 24 hours of teleoperated unstructured play data.

**Imitation learning.** Assuming the existence of demonstrations, an embodied agent can learn to reproduce ground truth trajectories conditioned on instructions. Behavioral cloning-based approaches [308, 288] rely on supervised learning to learn the trajectories.

Imitation learning has intensively been employed in robotics manipulation, notably to train recurrent neural networks [210], U-Net structures [200], or with active learning [262]. Particularly impressive results were achieved for generalization to new skills [83, 145, 65].

Imitation learning is the main paradigm for training neural networks for solving language-guided manipulation tasks, as the continuous action space and the limited amount of data are particularly challenging for reinforcement learning-based algorithms. The transporter network was adapted to this class of tasks in CLIPort [275] and was extended to a seven-dimensional action space in VLMBench [358]. Hristov *et al*. [126], Lynch *et al*. [211] and CALVIN's baseline [219] are based on a sequence- to-sequence conditional variational auto-encoder network.

**Applications to real-world.** Tellex *et al*. [300] and Misra *et al*. [224] succeeded in applying algorithms on a real-world robot. However, using a translation of the instructions into a formal language, the set of executable actions remained limited in

their work. Among the aforementioned methods that predict low-level commands, only a few have been tested on a real-world robot [275, 125]. However, CLIPort [275] is using a simplified action space, whereas Hristov *et al*. [125] is using simplified instructions. In Chapter 5, we will see that our methods can be applied in a real-world robot with seven degrees of freedom and human-written instructions.

### 2.3.4 Language-guided navigation

While the aforementioned tasks have popularized language-guided embodied agents, the latter have visual access to most of the environment though some parts are visually occluded. This is not the case for mobile agents that must navigate to a different room of their environment to fulfill their tasks.

**Simulators.** To handle embodied agents, more simulators have been offered to the research community. They differ by their scale [147, 267, 266], their degree of customization [27, 283], their latency [154], their interactivity [37, 331, 159, 272, 334], or their visual realism [43].

A new generation of simulators is appearing, combining most of these advantages, notably with Gibson 2.0 [184], BEHAVIOR-1K [183], ThreeDWorld [86]. The emergence of these simulators led to new challenges as described below.

**Embodied Question Answering.** In embodied question answering [64], a question is asked to an embodied agent. But contrary to visual question answering [16], the agent must navigate in its environment to find the answer. For example, when asked "what is the color of the car in the garage", the agent must first navigate to the garage. Das *et al*. [64]'s seminal work was developed in the House3D environment [332] was been extended to multi-agents [297], multi-targets [344], and photo-realistic environments [328]. VideoNavQA [39] extended the dataset to 28 types of questions to improve the diagnostics of an agent's reasoning abilities.

**Vision-and-language navigation.** Because of the versatile nature of such embodied agents, different tasks have been offered. Touchdown [51], R2R [13, 9], R4R [140], and RxR [169], Landmark-RxR [113], Talk2Nav [311], StreetLearn [222] provide fine-grained navigation instructions to the agent that must find a target location. In REVERIE [251] and SOON [361], the task is similar, but the agent receives only a coarse-grained instruction, describing the target location and not the path towards this location. Other datasets have encompassed the presence of dialogs [302, 231, 232]. Efforts were also put forward to increase interactions, embodied agents are also asked to interact with objects in CHAI [223] and ALFRED [276] or fine-grained action space with VLN-CE [164] and RoboVLN [135]. Finally, dialog and interaction have been combined in DialFRED [88] and TEACh [237].

**Evaluation of vision-and-language navigation.** Metrics are generally of the success of the goal by measuring the agent's proximity to the goal. For example, in datasets based on Matterport3D [44], an agent is successfully completing a task when it stops on the target location with a tolerance of 3 meters. The success rate [13, 51] then how frequently the agent has completed tasks on the testing set. However, this metric does not take into account the path taken by the agent, whereas taking the shortest path is reasonably harder than exploring first the environment. For this reason, the success

rate is often weighted by the path length [251]. Some other metrics are penalizing long trajectories, such as the path fidelity score [140] or the normalized dynamic time warping [134]. As the numbers of datasets grow, the methods for addressing them have also increased. Here, we summarize the main directions in the literature.

**Pre-training.** The multimodality nature of those tasks allowed methods to study different types of pre-training: on text [188], on semantic segmentation [204], on other vision and language datasets [106]. Several articles [273, 155, 185] show the advantages of using CLIP embeddings [254], achieving notably remarkable performance in zero-shot learning [269]. In Chapter 3, we study how we can benefit from an in-domain pretraining dataset for pretraining a transformer-based architecture.

**Data-Augmentation.** Similarly, several works suggest data-augmentation methods, notably by generating novel instructions using a speaker model [85, 295, 73], editing environments [186, 295], or mixing several existing environments [197]. Several methods are also generating subgoal instructions [192, 120, 269].

**Memory.**. Since the task is a sequential process, encoding a memory mechanism is crucial to remember previous steps achieved by the agent. First algorithms employed recurrent neural networks [13, 85, 212, 188, 106, 122], but it remains difficult to remember long sequences of steps. Another solution consists in building a separate memory model dedicated to the summarization of previous steps [364, 193, 233]. Recent works [241, 226] are encoding the full history through a multi-modal transformer architecture. This solution is further explored in Chapter 4.

**Auxiliary tasks** have been shown to be beneficial to agents [362], either acting as regularizers [212], or improving the alignment between several modalities [106]. In Chapter 3, we provide an auxiliary task enhancing the agent's capacity to reason over time, and we combine this task as an additional loss term. In Chapter 4, four different auxiliary tasks are shown to significantly improve our architecture performance.

**Map representation.** When optimizing the SPL metrics, we can not assume that the robot has access to a previously scan of the environment, contrary to Chapter 3. In this case, we need some mechanisms to track previously seen locations [213], and notably a map incorporating structured information from observations and eventually from instructions. An example of map representation in language-guided embodied agents can be found in Blukis *et al*. [34] using a 2D representation of the trajectory of a quadcopter. In the context of the aforementioned datasets, Anderson *et al*. [12] employs metrics maps based on SLAM [304, 46] tasks. But, to alleviate the difficulties of constructing such maps and then encoding them, a topological graph structure is proposed in several works using pre-exploration [53] or back-tracking [213], at the risk of increasing the path length. Topological maps have been encoded in separate modules without language grounding and using recurrent neural networks [135, 316, 67]. We have provided a solution addressing these issues in DUET [58].

**Reinforcement learning** enables embodied agents to learn how to act from a reward provided by the environment [292]. This framework formulates a sequential decision as an optimization process, maximizing the expected discounted cumulative return, which is the sum of rewards received during an episode. Reinforcement learning has been used to solve tasks conditioned on language [207] and it has been shown efficient to

improve the environment exploration [321, 317, 295]. However, it suffers from the credit assignment problem [292], since the success signal is provided only at the end of the task, making it difficult to know which steps to penalize.

# Chapter 3

# In-domain Pretraining for Vision and Language Navigation

This Chapter presents our first contribution to vision-and-language navigation (VLN), which focuses on pre-training a transformer on a large-scale and diverse dataset collected on the web. This work is addressing the scarcity of domain-specific training data and the high diversity of image and language inputs, making the generalization of VLN agents to unseen environments particularly challenging.

Recent methods explore pretraining to improve generalization, however, the use of generic image-caption datasets or existing small-scale VLN environments is suboptimal and results in limited improvements. In this chapter, we introduce BnB[1], a large-scale and diverse in-domain VLN dataset. We first collect image-caption (IC) pairs from hundreds of thousands of listings from online rental marketplaces. Using IC pairs we next propose automatic strategies to generate millions of VLN path-instruction (PI) pairs. We further propose a shuffling loss that improves the learning of temporal order inside PI pairs. We use BnB to pretrain our Airbert[2] model. For two years, our method using a discriminative setting has been ranked first on the leaderboard of the Room-to-Room (R2R) navigation benchmark [77], which is a rare feat in a competitive research field.

This model is assuming that the action space has been discretized: to navigate from the initial location to the target location, the agent jumps from node to node, assuming that a lower-level motion planner can effectively control the robot between the two nodes, notably by avoiding collisions. At first, we also assume that the model has access to an overview scan of the environmnent; but, we will relax this hypothesis afterwards.

## 3.1 Introduction

In vision-and-language navigation (VLN), an agent is asked to navigate in home environments following natural language instructions [9, 13]. This task is attractive to many

[1] Bed and Breakfast

[2] *Airbert* is an Old Irish word meaning *practice*, here referring to model pretraining on pretext tasks similar to VLN.

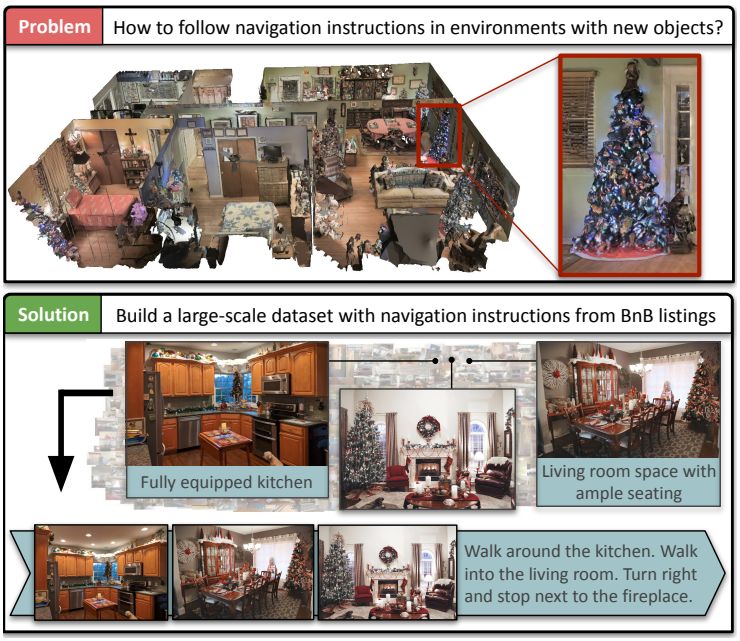



**Figure 3.1:** VLN tasks are evaluated on unseen environments at test time. *Top*: None of the training houses contain a Christmas theme making this test environment particularly challenging. *Bottom*: We build a large-scale, visually diverse, and in-domain dataset by creating path-instruction pairs close to a VLN-like setup and show the benefits of self-supervised pretraining.

real-world applications such as domestic robotics and personal assistants. However, given the high diversity of VLN data across environments and the difficulty of the manual collection and annotation of VLN training data at scale, the performance of current methods remains limited, especially for previously unseen environments [353].

This chapter is motivated by significant improvements in vision and language pretraining [5, 59, 190, 204, 205, 290], where deep transformer models [312] are trained via self-supervised proxy tasks [69] using large-scale, automatically harvested image-text datasets [236, 271]. Such pretraining enables learning transferable multi-modal representations achieving state-of-the-art performance in various vision and language tasks. Similarly, with the goal of learning an embodied agent that generalizes, recent works [106, 128, 188, 215] have explored different pretraining approaches for VLN tasks.

In [106, 128], annotated path-instruction pairs are augmented with a *speaker* model that generates instructions for random unseen paths. However, as these paths originate from a small set of 61 houses used during training, they are limited in visual diversity. The limited pretraining environments do not equip agents with visual understanding abilities that enable generalization to unseen houses, see Fig. 3.1. To address this problem, VLN-BERT [215] proposes to pretrain the agent on generic image-caption datasets that are abundant and cover diverse visio-linguistic knowledge. However, these image-caption pairs are quite different from the dynamic visual stream (path)

and navigable instructions observed by a VLN agent. Such out-of-domain pretraining, although promising, only brings limited gains to the navigation performance. Besides the above limitations, existing pretraining methods do not place much emphasis on temporal reasoning abilities in their proxy tasks such as one-step action prediction [106] and path-instruction pairing [215], while such reasoning is important to a sequential decision making task like VLN. As a result, even if performance in downstream tasks is improved, the pretrained models may still be brittle. For example, a simple corruption of instructions by swapping noun phrases within the instruction, or replacing them with other nouns, leads to significant confusion as models are unable to pick the correct original pair.

In this chapter, we explore a different data source and proxy tasks to address the above limitations in pretraining a generic VLN agent. Though navigation instructions are rarely found on the Internet, image-caption pairs from home environments are abundant in online marketplaces (*e.g. Airbnb*), which include images and descriptions of rental listings. We collect BnB, a new large-scale dataset with 1.4M indoor images and 0.7M captions. First, we show that in-domain image-caption pairs bring additional benefits for downstream VLN tasks when applied with generic web data [215]. In order to further reduce the domain gap between the BnB pretraining and the VLN task, we present an approach to transform static image-caption pairs into visual paths and navigation-like instructions (Fig. 3.1 bottom), leading to large additional performance gains. We also propose a shuffling loss that improves the model's temporal reasoning abilities by learning a temporal alignment between a path and the corresponding instruction.

Our pretrained model, Airbert, is a generic transformer backbone that can be readily integrated in both discriminative VLN tasks such as path-instruction compatibility prediction [215] and generative VLN tasks [121] in R2R navigation [13] and REVERIE remote referring expression [251]. We achieve state-of-the-art performance on these VLN tasks with our pretrained model. Beyond the standard evaluation, our in-domain pretraining opens an exciting new direction of *one/few-shot VLN* where the agent is trained on examples only from one/few environment(s) and expected to generalize to other unseen environments.

In summary, the contributions of this chapter are three-fold. (1) We collect a new large-scale in-domain dataset, BnB, to promote pretraining for vision-and-language navigation tasks. (2) We curate the dataset in different ways to reduce the distribution shift between pretraining and VLN and also propose the shuffling loss to improve temporal reasoning abilities. (3) Our pretrained Airbert can be plugged into generative or discriminative architectures and achieves state-of-the-art performance on R2R and REVERIE datasets. Moreover, our model generalizes well under a challenging one/few-shot VLN evaluation, truly highlighting the capabilities of our learning paradigm.

## 3.2 Related work

**Vision-and-language navigation.** VLN [13] has received significant attention with a large number of followup tasks introduced in recent years [9, 51, 165, 170, 231, 232, 251, 276, 302]. Early days of VLN saw the use of sequence-to-sequence LSTMs to predict low-level actions [13] or high-level directions in a panoramic action space [85].

Different attention mechanisms [212, 250] are proposed to improve cross-modal alignment. Various reinforcement learning based training algorithms [295, 318, 321, 322] and searching algorithms in inference [85, 212, 213] have also been explored to improve the VLN performance.

To improve an agent's generalization to unseen environments, data augmentation is performed by using a *speaker* model [85] that generates instructions for random paths in seen environments, and environment dropout [295] is used to mimic new environments. While pretraining LSTMs for transferable representations is adopted by [128], recently, there has been a shift towards transformer models [106] to learn generic multimodal representations. This is further extended to a recurrent model that significantly improves sequential action prediction [121]. However, the limited environments in pretraining [106, 128] constrain the generalization ability to unseen scenarios. The most related work, VLN-BERT [215] transfers knowledge from abundant, but out-of-domain image-text data to improve path-instruction matching. In this chapter, we not only create a large-scale, *in-domain* BnB dataset, but also propose effective pretraining strategies to mitigate the domain-shift between webly crawled image-text pairs and VLN data.

**Large-scale visio-linguistic pretraining.** Thanks to large-scale image-caption pairs automatically collected from the web [220, 236, 255, 271], visio-linguistic pretraining (VLP) has made great breakthroughs in recent years. Several VLP models [59, 190, 204, 294] have been proposed based on the transformer architecture [312]. These models are often pretrained with self-supervised objectives akin to those in BERT [69]: masked language modeling, masked region modeling and vision-text pairing. Fine-tuning them on downstream datasets achieves state-of-the-art performance on various VL tasks [16, 153, 320, 315]. While such pretraining focuses on learning correlations between vision and text, it is not designed for sequential decision making as required in embodied VLN. The goal of this chapter is not to improve VLP architectures but to present in-domain training strategies that lead to performance improvements for VLN tasks.

## 3.3 BnB Dataset

Hosts that rent places on online marketplaces often upload attractive and unique photos along with descriptions. One such marketplace, *Airbnb*, has 5.6M listings from over 100K cities all around the world [3]. We propose to use this abundant and curated data for large-scale in-domain VLN pretraining. In this section, we first describe how we collect image-caption pairs from *Airbnb*. Then, we propose methods to transform images and captions into VLN-like path-instruction pairs to reduce the domain gap between webly crawled image-caption pairs and VLN tasks (see Fig. 3.3).

### 3.3.1 Collecting BnB Image-Caption Pairs

**Collection process.** We restrict our dataset to listings from the US (about 10% of *Airbnb*) to ensure high quality English captions and visual similarity with Matterport environments [44]. The data collection proceeds as follows: (1) obtain a list of locations

**(a)** Gorgeous ocean views from bedroom.


**(b)** Main stage

**(c)** Spring excitement

**Figure 3.2:** Examples of outdoor images with their corresponding captions.

from Wikipedia; (2) find listings in these locations by querying the *Airbnb* search engine; (3) download listings and their metadata; (4) remove *outdoors* images[3] as classified by a ResNet model pretrained on Places365 [359]; and (5) remove invalid image captions such as emails, URLs and duplicates.

**Statistics.** We downloaded almost 150k listings and their metadata (1/4 of the listings in the US) in step 3, leading to over 3M images and 1M captions. After data cleaning with steps 4 and 5, we obtain 713K image-caption pairs and 676K images without captions. Table 3.1 compares our BnB dataset to other datasets used in previous works for VLN (pre-)training. It is larger than R2R [13], REVERIE [251] and includes a large diversity of rooms and objects, which is not the case for Conceptual Captions [271]. We posit that such in-domain data is crucial to deal with the data scarcity challenge in VLN environments as illustrated in Fig. 3.1. We use 95% of our BnB dataset for training and the remaining 5% for validation.

Apart from images and captions, our collected listings contain structured data including a list of amenities, a general description, reviews, location, and rental price, which may offer additional applications in the future.

### 3.3.2 Filtering image-caption pairs: Outdoor images

Images of outdoor scenes are almost never seen in the environments used in downstream VLN tasks. In fact, not only are the images out-of-domain (such images are rarely seen in the VLN environments), their captions are often irrelevant to a VLN task. In order to alleviate the impact of such noisy images and captions, we discard outdoor images from the pretraining process. Figure 3.2 illustrates several examples of misleading outdoor image-caption pairs. Captions as written by the host are presented in the label below the image. The caption for the image in Figure 3.2a refers to a "*bedroom*", however, the image does not show a bedroom. Similarly, the image-caption pair in the Figure 3.2b talks about activities or festivals that take place in the neighborhood of the listing, however, they are not relevant for solving indoor navigation tasks. Finally, Figure 3.2c shows an outdoor scene with several birds along with a noisy caption that is not directly related to the image content, but the emotion that the image may evoke.

[3]While outdoor images may contain interesting features (*e.g.* a patio), we observe that removing them increases performance.

| Dataset | Source | #Envs | #Imgs | #Texts |
|---|---|---|---|---|
| R2R [13] | Matterport | 90 | 10.8K | 21.7K |
| REVERIE [251] | Matterport | 86 | 10.6K | 10.6K |
| Speaker [295] | Matterport | 60 | 7.8K | 0.2M |
| ConCaps [271] | Web images | - | 3.3M | 3.3M |
| **BnB** (ours) | Airbnb | 140K | 1.4M | 0.7M |

**Table 3.1:** Comparing BnB to other existing VLN datasets. The #images from Matterport environments [44] refers to the #panoramas. The speaker model [295] generates instructions for randomly selected trajectories, but is limited to panoramas from 60 training environments. Note that the data from Conceptual Captions (ConCaps) may feature some houses, but it is not the main category.

### 3.3.3 Creating BnB Path-Instruction Pairs

BnB image-caption (IC) pairs are complementary to Conceptual Captions (ConCaps) as they capture diverse VLN environments. However, they still have large differences from path-instruction (PI) pairs in VLN tasks. For example, during navigation, an agent observes a sequence of panoramic views rather than a single image, and the instruction may contain multiple sentences. To mitigate this domain gap, we propose strategies to automatically craft path-instruction pairs starting from BnB-IC pairs.

#### Concatenating Images and Texts in a BnB Listing

Images in a BnB listing usually depict different locations in a house, mimicking the sequential visual observations an agent makes while navigating in the house. To create a VLN-like path-instruction pair, we randomly select and concatenate $K$[4] image-caption pairs from the listing. In between each caption, we randomly add a word from "*and*", "*then*", "." or nothing to make the concatenated instruction more fluent and diverse.

#### Augmenting *Paths* with Visual Contexts

In the above concatenated path, each location only contains one BnB image, and perhaps with a limited view angle as hosts may focus on objects or amenities they wish to highlight. Therefore, it lacks the panoramic visual context at each location that the agent receives in real navigation paths. Moreover, each location in the concatenated instruction is described by a unique sentence, while adjacent locations are often expressed together in one sentence in VLN instructions [119]. To address the above issues with concatenation, we propose two approaches to compose paths that have more visual context and can also leverage the abundant images without captions (denoted as *captionless images*).

**1. Image merging** extends the panoramic context of a location by grouping images from similar room categories (see Fig. 3.3). For example, if the image depicts a kitchen sink, it is natural to expect images of other objects such as forks and knives nearby.

[4] typically 4 - 7 to match the number of steps in the R2R dataset

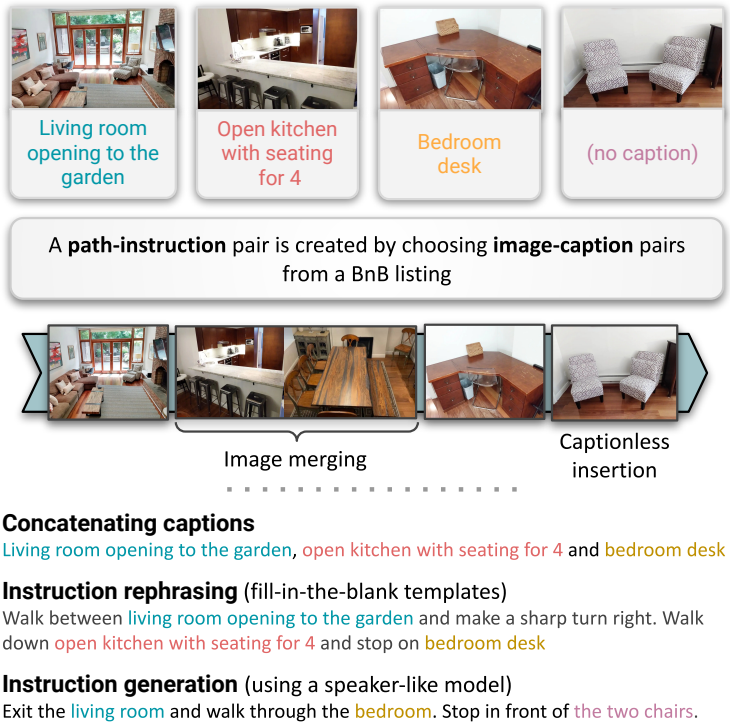



**Figure 3.3:** We explore several strategies to automatically create navigation-like instructions from image-caption pairs.

Specifically, we first cluster images of similar categories (*e.g.* *kitchen*) using room labels predicted by a pretrained Places365 model [359]. Then, we extract multiple regions from this *merged* set of images, and use them as an approximation to the panoramic visual representation.

**2. Captionless image insertion.** Table 1 shows that half of the BnB images are captionless. Using them allows to increase the size of the dataset. When creating a path-instruction pair from the concatenation approach, a captionless image is inserted as if its caption was an empty string. The BnB PI pairs hence better approximate the distribution of the R2R path-instructions: (1) some images in the path are not described and (2) instructions have similar number of noun phrases.

### Crafting *Instructions* with Fluent Transitions

The concatenated captions mainly describe rooms or objects at different locations, but do not contain any of the actionable verbs as in navigation instructions, *e.g.* "*turn left at the door*" or "*walk straight down the corridor*". We suggest two strategies to create fake instructions that have fluent transitions between sentences.

**1. Instruction rephrasing.** We use a fill-in-the-blanks approach to replace noun-phrases

in human annotated navigation instructions [13] by those in BnB captions (see Fig. 3.3). Concretely, we create more than 10K instruction templates containing 2-7 blanks, and fill the blanks with noun-phrases extracted from BnB captions. The noun-phrases matched to object categories from the Visual Genome [167] dataset are preferred during selection. This allows us to create VLN-like instructions with actionable verbs interspersed with room and object references for visual cues that are part of the BnB path (see Fig. 3.3).

**2. Instruction generation** is a video captioning like model that takes in a sequence of images and generates an instruction corresponding to an agent's path through an environment. To train this model, we adopt ViLBERT and train it to generate captions for single BnB image-caption pairs. Further, this model is fine-tuned on trajectories of the R2R dataset to generate instructions. Finally, we use this model to generate BnB PI pairs by producing an instruction for a concatenated image sequence from BnB (the path).

### 3.3.4 Dataset details and Statistics

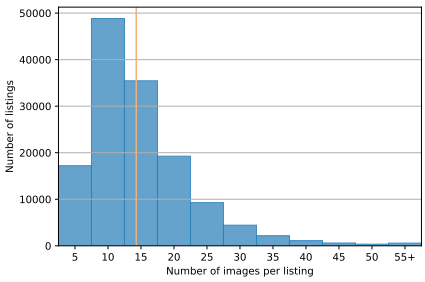
**BnB image-caption pairs.** We collect BnB IC pairs from 150K listings on *Airbnb* resulting in 713K image-caption pairs and 676K images without captions. In Figure 3.4, we present some key statistics about this data. Figure 3.4a presents a histogram of the number of images found in each listing. While most listings have less than 20 images, this is still a sufficiently large and diverse in-domain distribution. In Figure 3.4b, we summarize the rooms depicted in the images through predicted category labels obtained using a CNN trained on the *Places365* dataset [359]. These category labels are used as part of our proposed extensions such as *image merging*.

**Creating BnB path-instruction pretraining samples.** We create the BnB PI pairs on-the-fly during training to mimic the agent's visual trajectory and a corresponding instruction through an environment. Each sample in a batch is created by randomly sampling a listing without replacement during an epoch (one epoch consists of one PI pair from each listing). Then, the number of IC pairs $K$ that form the PI pair are chosen (as an integer) from a uniform distribution, $K \sim U[4,7]$. We sample $N \sim U[2,K]$ IC pairs that have a non-empty caption and the remainder $K-N$ images are chosen from the set of captionless images. Any image in the path may include additional visual context (from the same room) via the *image merging* strategy. Similarly, the *instruction rephrasing* strategy may be employed by using existing R2R instruction templates and filling them with noun phrases extracted from the image captions.

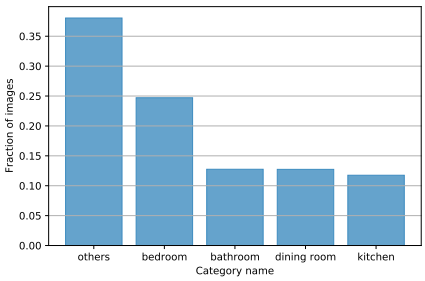
The above procedure results in creating one correctly aligned (positive) PI pair, $X^{+}$. To employ the shuffling loss for each sample, we create 9 additional negatives, $X_n^{-}$, by shuffling either the sequence of images or captions, ensuring that the post-shuffling order does not align with the positive pair.

**Statistics for BnB PI pairs.** Due to the large number of possible combinations, we can (theoretically) create 200 billion path-instruction pairs, using the simple concatenation strategy. This number grows to over 300 quadrillion when considering additional visual context augmentations and fluent instructions.

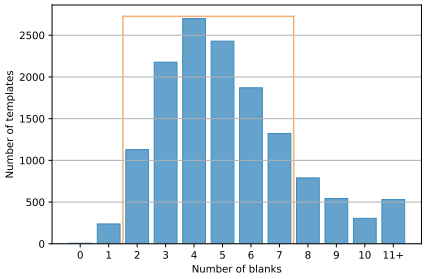
For *instruction rephrasing*, we create 11,626 fill-in-the-blank templates from the R2R training set. Figure 3.4c shows the distribution of the number of blanks in the



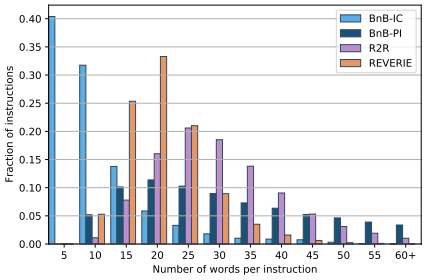

**(a)** Distribution of the number of images per listing.




**(b)** Distribution of predicted scene categories on BnB images.




**(c)** Fill-in-the-blanks templates built using the R2R training set.




**(d)** Distribution of the instruction lengths.

**Figure 3.4:** Statistics of BnB Dataset.

templates – most instruction templates have 2-7 blanks into which we insert noun phrases from the BnB captions.

While we are unable to generate the entire BnB PI dataset for computing statistics, we generate 50K PI pairs as a representative sample. Figure 3.4d presents the distribution of instruction lengths (number of words) for different datasets. We see that the captions in BnB IC pairs are much shorter than typical instructions in R2R and REVERIE, while our automatically created instructions in BnB PI pairs exhibit a high level of similarity in terms of their length.

### 3.3.5 Examples of BnB PI Pairs

Figure 3.5 presents generated BnB PI pairs using various strategies proposed in this chapter, including naive concatenation, instruction rephrasing, instruction generation, image merging and captionless image insertion.

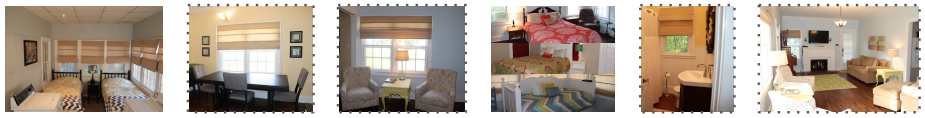
Among the methods to create an instruction, simple concatenation lacks action verbs between sentences for fluent transition leading to a domain shift from real instructions. Instruction rephrasing selects noun phrases from BnB image descriptions and inserts them into real instruction templates, providing a natural feel to the created instruction. Finally, while the learning approach of instruction generation (recall, this is learned on downstream VLN dataset) produces fluent sentences, it is unable to leverage the diverse

**Concatenation:** extra guest room with comfy full bed on top floor of house and top floor shared bathroom for both guest rooms then adjoining modern private bath with stall shower bath and beach towels provided then granny's treasures add a homey touch
**Instruction rephrasing:** exit extra guest room and turn left. pass top floor shared bathroom then turn right. walk toward a homey touch and wait there.
**Instruction generation:** walk to the other side of the bathroom and stop next to the last corner on the wall with the candles.

**(a)** Example 1

**Concatenation:** full bath and open floor plan living opens to deck, kitchen / dining area
**Instruction rephrasing:** go around full bath, then open floor plan living down to kitchen / dining area.
**Instruction generation:** walk into the bathroom and turn right. walk to the end of the landing and turn left. walk into the sitting area and turn right. walk past the chair and stop.

**(b)** Example 2



**Concatenation:** bedroom 3 ( picture 2 of 2 ) - 2 twin beds w / full size washer then bedroom 2 - queen bed - 1st floor, bathroom 1st floor. w / tub and shower.
**Instruction rephrasing:** exit the bedroom 3 and go right into bedroom 2 next to tub and shower.
**Instruction generation:** walk straight through the doorway and turn right. walk straight through the doorway and turn left. walk through the doorway and stop.

**(c)** Example 3

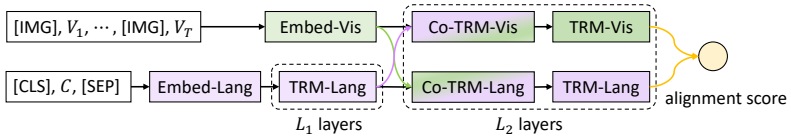
**Figure 3.5:** Examples of path-instruction pairs created by different strategies. The images with dotted borders are images chosen from the *captionless image insertion* strategy, and the clustered images are from the *image merging* strategy.

captions of BnB images due to the limited vocabulary stemming from the downstream VLN dataset. For example, the generated instruction in Figure 3.5c does not contain noun phrases related to images in the path. Better caption generation models such as Pointer network [314] may help avoid such problems, however are left for future work.

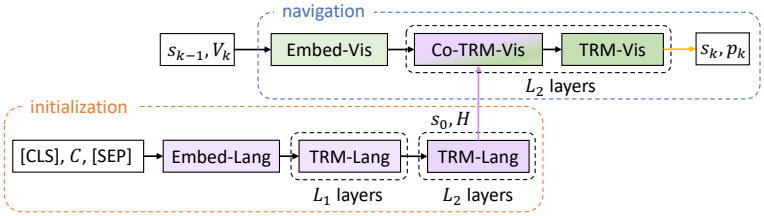
Among augmentations for path generation, we can see that *image merging* helps to expand relevant visual context from single images to semi-panoramic views, see the bedroom in Figure 3.5a or the kitchen in Figure 3.5b. *Captionless image insertion* also improves the path diversity by mimicking unmentioned viewpoints in the instruction (indicated by images with a dotted border).

## 3.4 Airbert: A Pretrained VLN Model

In this section, we present Airbert, our multi-modal transformer pretrained on the BnB dataset with masking and shuffling losses. We first introduce the architecture of Airbert, and then describe datasets and pretext tasks in pretraining. Finally, we show




**(a)** Adapting Airbert to a discriminative setting to predict path-instruction alignment score, similar to [215].




**(b)** Adapting Airbert to a generative setting based on the Recurrent VLN-BERT [121].

**Figure 3.6:** The adapted Airbert model in both discriminative and generative settings for downstream VLN tasks.

how Airbert can be adapted to downstream VLN tasks.

### 3.4.1 ViLBERT-like Architecture

ViLBERT [204] is a multi-modal transformer extended from BERT [69] to learn joint visio-linguistic representations from image-caption pairs, as illustrated in Fig. 3.7.

Given an image-caption pair $(V, C)$, the model encodes the image as region features $[v_1, \ldots, v_{\mathscr{V}}]$ via a pretrained Faster R-CNN [11], and embeds the text as a series of tokens: $[\texttt{[CLS]}, w_1, \ldots, w_T, \texttt{[SEP]}]$, where `[CLS]` and `[SEP]` are special tokens added to the text. ViLBERT contains two separate transformers that encode $V$ and $C$ and it learns cross-modal interactions via co-attention [204].

We follow a similar strategy to encode path-instruction pairs (created in Sec. 3.3.3) that contain multiple images and captions $\{(V_k, C_k)\}_{k=1}^{K}$. Here, each $V_k$ is represented as visual regions $v_i^k$ and $C_k$ as word tokens $w_t^k$. Respectively, the visual and text inputs to Airbert are:

$$X_V = [\texttt{[IMG]}, v_1^1, \ldots, v_{\mathscr{V}_1}^1, \ldots, \texttt{[IMG]}, v_1^K, \ldots, v_{\mathscr{V}_K}^K], \tag{3.1}$$

$$X_C = [\texttt{[CLS]}, w_1^1, \ldots, w_{T_1}^1, \ldots, w_1^K, \ldots, w_{T_K}^K, \texttt{[SEP]}], \tag{3.2}$$

where the `[IMG]` token is used to separate image region features taken at different locations.

Note that while our approach is not limited to a ViLBERT-like architecture, we choose ViLBERT for a fair comparison with previous work [215].

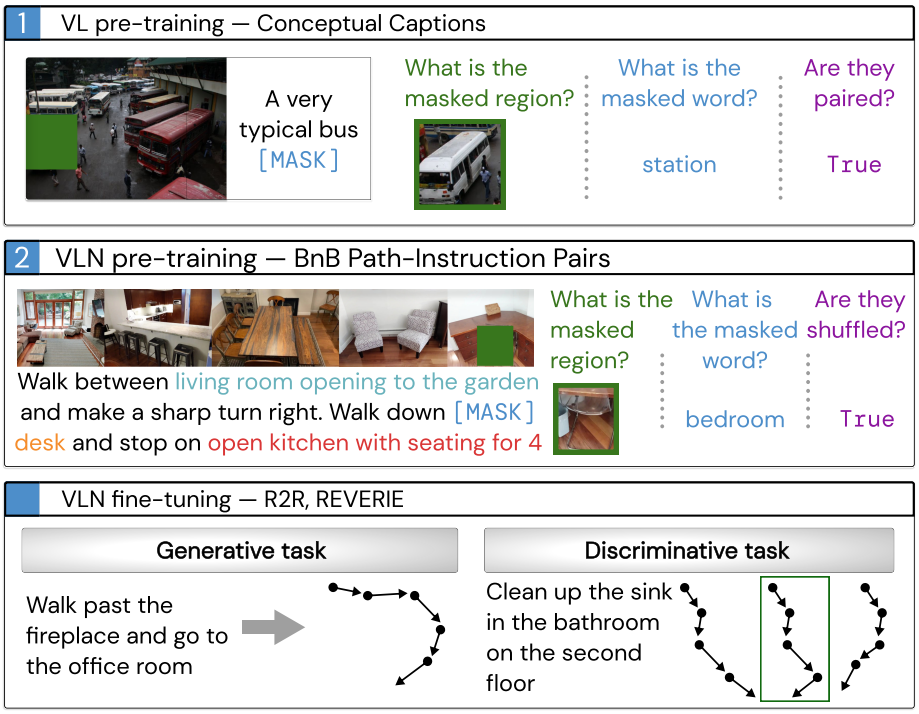



**Figure 3.7:** Overview of our pretraining approach. Instead of the usual VL pretraining (panel 1), we adopt in-domain data and use the path-instruction pairs to train Airbert with the masking and shuffling losses (panel 2). We fine-tune Airbert on downstream VLN tasks using both discriminative or generative models (panel 3).

### 3.4.2 Datasets and Pretext Tasks for Pretraining

We use Conceptual Captions (ConCaps) [271] and BnB-PI in subsequent pretraining steps (see Fig. 3.7) to reduce the domain gap for downstream VLN tasks.

Previous multi-modal pretraining efforts [204, 215, 128] commonly use two self-supervised losses given image-caption (IC) pairs or path-instruction (PI) pairs: (1) *Masking* loss: An input image region or word is randomly replaced by a `[MASK]` token. The output feature of this masked token is trained to predict the region label or the word given its multi-modal context. (2) *Pairing* loss: Given the output features of `[IMG]` and `[CLS]` tokens, a binary classifier is trained to predict whether the image (path) and caption (instruction) are paired.

The above two pretext tasks mainly focus on learning object-word associations instead of reasoning about the temporal order of paths and instructions. For example, if an image $V_i$ appears before $V_j$, then words from its caption $C_i$ should appear before $C_j$. In order to promote such a temporal reasoning ability, we propose an additional *shuffling* loss to enforce alignment between PI pairs.

Given an aligned PI pair $X^+ = \{(V_k, C_k)\}_{k=1}^K$, we generate $\mathcal{N}$ negative pairs $X_n^- = \{(V_k, C_l)\}, k \neq l$, by shuffling the composed images or the captions. We train our model to choose the aligned PI pair as compared to the shuffled negatives by minimizing the cross-entropy loss:

$$L = -\log \frac{\exp(f(X^+))}{\exp(f(X^+)) + \sum_n \exp(f(X_n^-))}, \tag{3.3}$$

where $f(X)$ denotes the similarity score (logit) computed via Airbert for the PI pair $X$.

### 3.4.3 Adaptations for Downstream VLN tasks

We consider two VLN tasks: goal-oriented navigation (R2R [13]) and object-oriented navigation (REVERIE [251]). Airbert can be readily integrated in discriminative and generative models for the above VLN tasks.

**Discriminative Model: Navigation as Path-Selection [215].** The navigation problem on the R2R dataset is formulated as a path selection task in [215]. Several candidate paths are generated via beam search from a navigation agent such as [295], and a discriminative model is trained to choose the best path among them. We fine-tune Airbert on the R2R dataset for path selection. A two-stage fine-tuning process is adopted: in the first phase, we use *masking* and *shuffling* losses on the PI pairs of the target VLN dataset in a manner similar to BnB PI pairs; in the second phase, we choose a positive candidate path as one that arrives within 3m of the goal, and contrast it against 3 negative candidate paths. We also compare multiple strategies to mine additional negative pairs (other than the 3 negative candidates), and in fact, empirically show that negatives created using shuffling outperform other options.

**Generative Model: Recurrent VLN-BERT [121].** The Recurrent VLN-BERT model adds recurrence to a state in the transformer to sequentially predict actions, achieving state-of-the-art performance on R2R and REVERIE tasks. We use our Airbert architecture as its backbone and apply it to the two tasks as follows. First, the language

transformer encodes the instruction via self-attention. Then, the embedded `[CLS]` token in the instruction is used to track history and concatenated with visual tokens (observable navigable views or objects) in each action step. Self-attention and cross-attention on embedded instructions are employed to update the state and visual tokens and the attention score from the state token to visual tokens is used to decide the action at each step. We fine-tune the Recurrent VLN-BERT model with Airbert as the backbone in the same way as [121].

## 3.5 Experimental Results

We first perform ablation studies evaluating alternative ways to pretrain Airbert in Sec. 3.5.4. Then, we compare Airbert with state-of-the-art methods on R2R and REVERIE tasks in Sec. 3.5.5. Finally, in Sec. 3.5.8, we evaluate models in a more challenging setup: VLN few-shot learning where an agent is trained on examples taken from one/few houses.

### 3.5.1 Implementation details

We present the implementation details for learning Airbert via pretraining using BnB, and subsequent fine-tuning in both discriminative or generative settings.

#### Airbert Pretraining

Airbert's architecture is the same as VLN-BERT (see Figure 3.6a where the number of layers $L_1 = L_2 = 6$). The feature vector $v_i^k$ (corresponding to $i$th image region of the $k$th image) is composed of three terms: the first term is the visual feature extracted by the Bottom-Up Top-Down attention model [11]; the second term encodes the location of the region in the image as $\text{MLP}(l_i^k)$, where $l_i^k$ is the 5-dim location vector of the given image region defined as the top corner $(x, y)$, the width, height and area; and the last term $\text{Emb}(k)$ encodes the position, where Emb is an embedding layer for the image order.

We use 8 V100 SXM2 GPUs (32 GB each) for pretraining Airbert. The model is trained for 15 epochs with a batch size of 64 and learning rate of $4 \times 10^{-5}$. Each epoch consists of one randomly sampled PI pair from 95% of the listings, while the remaining 5% are used for validation and preventing overfitting.

### 3.5.2 Fine-tuning in Discriminative Setting

In the discriminative setting, R2R navigation is formulated as a path selection problem given the instruction. The pretrained Airbert model can be directly fine-tuned without any modifications to the architecture to predict the path-instruction alignment (or compatibility) score as shown in Figure 3.6a.

We follow the same fine-tuning setup as VLN-BERT [215] to allow for a fair comparison. We use the Adam optimizer with a learning rate of $4 \times 10^{-5}$. The optimizer is controlled by a learning rate scheduler with a linear warmup and cooldown. We

fine-tune Airbert for 30 epochs with a batch size of 64. Samples from the R2R training set are used for fine-tuning and the model checkpoint with the highest success rate on the unseen validation set (val unseen) is selected for the test set and leaderboard submission.

#### Fine-tuning in Generative Setting

In the generative setting, an agent is required to predict navigable actions step by step. We adopt the state-of-the-art generative model Recurrent VLN-BERT [121] for R2R and REVERIE tasks. The model uses a pretrained multimodal transformer as a backbone and adds recurrence to a state token to keep track of history for sequential action prediction. Although the original Recurrent VLN-BERT model only implements an LXMERT-like [294] architecture PREVALENT [106], and one-stream BERT-like architecture OSCAR [190], it is easy to plug our two-stream ViLBERT architecture as the backbone.

The adapted model is shown in Figure 3.6b. For initialization, the language stream is used to encode the instruction $C$ into an instruction representation $H$. As no visual inputs are used during the initialization, the co-attention modules in the original language stream of ViLBERT are removed, and the output feature of the `[CLS]` token is used as the agent's initial state $s_0$. For navigation at each step $k$, the visual stream takes the previous state $s_{k-1}$, visual observations $V_k$ at step $k$ and the encoded language features $H$ to generate a new state $s_k$ and action decision $p_k$.

When fine-tuning on the R2R dataset, we use scene features with a ResNet-152 pretrained on Places365 [359] and augment the training data with generated path-instruction pairs from [106]. We train the model via imitation learning and A2C reinforcement learning for 300,000 iterations with a batch size of 16 and learning rate of $10^{-5}$. When fine-tuning on the REVERIE dataset, object features encoded by a Bottom-Up Top-Down attention model [11] are used along with the scene features. The model is trained for 200,000 iterations with a batch size of 8. All the experimental setups for fine-tuning are the same as [121] for a fair comparison.

### 3.5.3 Datasets and metrics

**R2R Setup.** Most of our experiments are conducted on the R2R dataset [13], where we adopt standard splits and metrics defined by the task. We focus on success rate (SR), which is the ratio of predicted paths that stop within 3m of the goal. Please refer to [13, 215] for a more detailed explanation of the metrics. In particular, as the discriminative model uses path selection for R2R, we follow the pre-explored environment setting adopted by VLN-BERT [215].

**REVERIE Setup.** We also adopt standard splits and metrics on the REVERIE task [251]. Here, the success rate (SR) is the ratio of paths for which the agent stops at a viewpoint where the target object is visible. Remote Grounding Success Rate (RGS) measures accuracy of localizing the target object in the stopped viewpoint, and RGS per path length (RGSPL) is a path length weighted version.

| | Cat | Instruction | | Path | | SR on Val | |
|---|---|---|---|---|---|---|---|
| | | Rep | Gen | Merge | Insert | Seen | Unseen |
| 1 | - | - | - | - | - | 71.21 | 62.45 |
| 2 | ✓ | - | - | - | - | 73.84 | 62.71 |
| 3 | - | ✓ | - | - | - | 72.67 | 63.35 |
| 4 | - | - | ✓ | - | - | 71.19 | 63.11 |
| 5 | - | - | - | ✓ | - | 70.51 | 64.07 |
| 6 | - | - | - | - | ✓ | 74.43 | 66.05 |
| 7 | - | ✓ | - | ✓ | ✓ | 73.57 | **66.52** |

**Table 3.2:** Comparison between various BnB PI pair creation strategies for pretraining. The first row denotes the use of image-caption pairs. All methods from the second row use masking and shuffling during pretraining. Cat: naive concatenation; Rep: instruction rephrasing; Gen: instruction generation; Merge: image merging; and Insert: captionless image insertion.

| | BnB | | Speaker | | R2R | | SR on Val | |
|---|---|---|---|---|---|---|---|---|
| | Mask | Shuf. | Rank | Shuf. | Rank | Shuf. | Seen | Unseen |
| 1 | - | - | - | - | ✓ | - | 70.20 | 59.26 |
| 2 | - | - | ✓ | ✓ | ✓ | ✓ | 73.12 | 65.50 |
| 3 | ✓ | - | - | - | ✓ | - | 73.24 | 64.21 |
| 4 | ✓ | ✓ | - | - | ✓ | - | 73.57 | 66.52 |
| 5 | ✓ | ✓ | - | - | ✓ | ✓ | 74.69 | 66.90 |
| 6 | ✓ | - | ✓ | - | ✓ | - | 70.21 | 65.52 |
| 7 | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | 73.83 | **68.67** |

**Table 3.3:** Impact of shuffling during pretraining and fine-tuning. While additional data helps, we see that using the shuffling loss (abbreviated as Shuf.) consistently improves model performance. Row 1 corresponds to VLN-BERT [215].

### 3.5.4 Pretraining with BnB

We perform ablation studies on the impact of various methods for creating path-instruction pairs. We also present ablation studies that highlight the impact of using the shuffling loss during Airbert's pretraining as well as fine-tuning stages. Throughout this section, our primary focus is on the SR on the unseen validation set and we compare our results against VLN-BERT [215], which achieves a SR of 59.26%.

**1. Impact of creating path-instruction pairs.** Table 3.2 presents the performance of multiple ways of using the BnB dataset after ConCaps pretraining as illustrated in Fig. 3.7. In row 1, we show that directly using BnB IC pairs without any strategies to reduce domain gap improves performance over VLN-BERT by 3.2%. Even if we skip ConCaps pretraining, we achieve 60.54% outperforming 59.26% of VLN-BERT. It proves that our BnB dataset is more beneficial to VLN than the generic ConCaps dataset.

| | Fine-tuning Strategies | Additional Negatives | SR on Val Seen | SR on Val Unseen |
|---|---|---|---|---|
| 1 | VLN-BERT [215] | 0 | 70.20 | 59.26 |
| 2 | (1) + Wrong trajectories | 2 | 70.11 | 59.11 |
| 3 | (1) + Highlight keywords | 0 | 71.89 | 61.37 |
| 4 | (1) + Hard negatives | 2 | 71.89 | 61.63 |
| 5 | (1) + Shuffling (Ours) | 2 | 72.46 | **61.98** |

**Table 3.4:** Comparison between different strategies for fine-tuning a ViLBERT model on the R2R task. VLN-BERT [215] fine-tunes ViLBERT with a masking and ranking loss. Each row (described in the text) is an independent data augmentation and can be compared directly against the baseline (row 1).

| Model | Replace-Nouns | | Swap-Nouns | | Directions | |
|---|---|---|---|---|---|---|
| | Seen | Unseen | Seen | Unseen | Seen | Unseen |
| VLN-BERT | 60.3 | 58.7 | 53.4 | 52.3 | 46.2 | 45.3 |
| Airbert | 68.3 | 66.6 | 66.6 | 61.1 | 47.3 | 49.8 |

**Table 3.5:** Accuracy of models attempting to pick the correct PI pair from a pool of correct + 10 negatives created by simple corruptions such as replacing or swapping noun phrases and switching directions (left with right). Random performance is $\frac{1}{11}$ or 9.1%.

Naive concatenation (row 2) does only slightly better than using the IC pairs (row 1) as there are still domain shifts with respect to fluency of transitions and lack of visual context. Rows 3-6 show that each method mitigates domain-shift to some extent. Instruction rephrasing (row 3) performs better at improving instructions than instruction generation (row 4), possibly since the generator is unable to use the diverse vocabulary of the BnB captions. Inserting captionless images at random locations (row 6) reduces the domain-shift significantly and achieves the highest individual performance. Finally, a combination of instruction rephrasing, image merging and captionless insertion provides an overall 3.8% improvement over concatenation, and a large 7.2% improvement over VLN-BERT.

**2. Shuffling loss applied during pretraining.** Table 3.3 demonstrates that shuffling is an effective strategy to train the model to reason about temporal order, and enforce alignment between PI pairs. Rows 3-5 show that shuffling is beneficial both during pretraining with BnB-PI data, or during fine-tuning with R2R data, and results in 2.3% and 0.4% improvements respectively. In combination with the *Speaker* dataset (paths from seen houses with generated instruction yielding 178K additional PI pairs [295]), we see that the shuffling loss provides 3.1% overall improvement (row 6 vs. 7). The BnB-PI data brings more improvements than the Speaker dataset (row 2 vs. 5). Putting together the BnB-PI data, Speaker dataset and shuffling, we achieve 68.67% SR on the R2R dataset with a single model.

**3. Shuffling loss applied during fine-tuning.** The final stage of model training on

| | Airbert | VLN-BERT [215] | Speaker [295] | Follower [295] | Val Seen | | | | | Val Unseen | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | | | PL | NE | SPL | OSR | SR | PL | NE | SPL | OSR | SR |
| 1 | - | ✓ | - | - | 10.28 | 3.73 | 0.66 | 76.47 | 70.20 | 9.60 | 4.10 | 0.55 | 69.22 | 59.26 |
| 2 | ✓ | - | - | - | 10.59 | 3.21 | 0.69 | 80.71 | **73.85** | 10.03 | 3.24 | 0.63 | 78.45 | **68.67** |
| 3 | - | - | ✓ | ✓ | 10.69 | 2.72 | 0.70 | 82.94 | 74.22 | 10.10 | 3.32 | 0.63 | 76.63 | 67.90 |
| 4 | - | ✓ | ✓ | ✓ | 10.61 | 2.35 | 0.78 | 86.57 | **81.86** | 10.00 | 2.76 | 0.68 | 81.91 | 73.61 |
| 5 | ✓ | - | ✓ | ✓ | 10.63 | 2.13 | 0.77 | 87.17 | **81.40** | 9.99 | 2.69 | 0.70 | 82.89 | **75.01** |

**Table 3.6:** Performance of single models and the impact of ensembling VLN-BERT or Airbert with the speaker and follower.

| Methods | Validation Seen | | | | | | Validation Unseen | | | | | | Test Unseen | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | Navigation | | | | RGS | RGSPL | Navigation | | | | RGS | RGSPL | Navigation | | | | RGS | RGSPL |
| | SR | OSR | SPL | TL | | | SR | OSR | SPL | TL | | | SR | OSR | SPL | TL | | |
| Seq2Seq-SF [13] | 29.59 | 35.70 | 24.01 | 12.88 | 18.97 | 14.96 | 4.20 | 8.07 | 2.84 | 11.07 | 2.16 | 1.63 | 3.99 | 6.88 | 3.09 | 10.89 | 2.00 | 1.58 |
| RCM [321] | 23.33 | 29.44 | 21.82 | 10.70 | 16.23 | 15.36 | 9.29 | 14.23 | 6.97 | 11.98 | 4.89 | 3.89 | 7.84 | 11.68 | 6.67 | 10.60 | 3.67 | 3.14 |
| SMNA [212] | 41.25 | 43.29 | 39.61 | 7.54 | 30.07 | 28.98 | 8.15 | 11.28 | 6.44 | 9.07 | 4.54 | 3.61 | 5.80 | 8.39 | 4.53 | 9.23 | 3.10 | 2.39 |
| FAST-MATTN [251] | **50.53** | **55.17** | **45.50** | **16.35** | 31.97 | 29.66 | 14.40 | 28.20 | 7.19 | 45.28 | 7.84 | 4.67 | 19.88 | 30.63 | 11.61 | 39.05 | 11.28 | 6.08 |
| Rec (OSCAR) [121] | 39.85 | 41.32 | 35.86 | 12.85 | 24.46 | 22.28 | 25.53 | 27.66 | 21.06 | 14.35 | 14.20 | 12.00 | 24.62 | 26.67 | 19.48 | 14.88 | 12.65 | 10.00 |
| Rec (ViLBERT) | 43.64 | 45.61 | 37.86 | 15.75 | 31.69 | 27.58 | 24.57 | 29.91 | 19.81 | 17.83 | 15.14 | 12.15 | 22.17 | 25.51 | 17.28 | 18.22 | 12.87 | 10.00 |
| Rec (VLN-BERT) | 41.11 | 42.87 | 35.55 | 15.62 | 28.39 | 24.99 | 25.53 | 29.42 | 20.51 | 16.94 | 16.42 | 13.29 | 23.57 | 26.83 | 18.73 | 17.63 | 14.24 | 11.63 |
| Rec (Airbert) | 47.01 | 48.98 | 42.34 | 15.16 | **32.75** | **30.01** | **27.89** | **34.51** | **21.88** | **18.71** | **18.23** | **14.18** | **30.28** | **34.20** | **23.61** | **17.91** | **16.83** | **13.28** |

**Table 3.7:** Navigation and object localization performance on the REVERIE dataset, including results on the unseen test set (leaderboard).

R2R involves fine-tuning to rank multiple candidate paths that form the path selection task. We compare various approaches to improve this fine-tuning procedure (results in Table 3.4). (1) In row 2, we explore the impact of using additional negative paths. Unsurprisingly, this does not improve performance. (2) Inspired by [102], we highlight keywords in the instruction using a part-of-speech tagger [148], and include an extra loss term that encourages the model to pay attention to their similarity scores (row 3). (3) Another alternative suggested by [102] involves masking keywords in the instruction and using VLP models to suggest replacements, resulting in hard negatives (row 4).

Hard negatives and highlighting keywords improve performance by 2.1-2.3%, but at the cost of extra parsers or VLP models. In contrast, shuffling visual paths to create two additional negatives results in highest improvement (row 5, +2.7% on val unseen) and appears to be a strong strategy to enforce temporal order reasoning, that neither requires external parsers nor additional VLP models.

**4. Error analysis.** We study the areas in which Airbert brings major improvements by analyzing scores for aligned PI pairs and simple corruptions that involve replacing noun phrases (*e.g.* *bedroom* by *sofa*), swapping noun phrases appearing within the instruction, or switching left and right directions (*e.g.* *turn left/right* or *leftmost/rightmost chair*). In particular, for every ground-truth aligned PI pair, we create 10 additional negatives by corrupting the instruction, and measure the accuracy of the model selecting the correct pair. Table 3.5 shows that Airbert with in-domain training and the shuffling loss achieves large improvements ($> 8\%$) for corruptions involving replacement or swapping of noun phrases. On the other hand, distinguishing directions continues to be a challenging problem; but here as well we see Airbert outperform VLN-BERT by 4.5%.

| Model | Test Unseen | | | | |
|---|---|---|---|---|---|
| | PL | NE | SPL | OSR | SR |
| Speaker-Follower [85] | 1,257 | 4.87 | 0.01 | 96 | 53 |
| PreSS [188] | 10.5 | 24.5 | 0.63 | 57 | 53 |
| PREVALENT [106] | 10.21 | 4.52 | 0.56 | 64 | 59 |
| Self-Monitoring [212] | 373 | 4.48 | 0.02 | 97 | 61 |
| Reinforced CM [321] | 358 | 4.03 | 0.02 | 96 | 63 |
| EnvDrop [13] | 687 | 3.26 | 0.01 | 99 | 69 |
| AuxRN [362] | 41 | 3.24 | 0.21 | 81 | 71 |
| VLN-BERT [215] | 687 | 3.09 | 0.01 | 99 | 73 |
| Airbert (ours) | 687 | 2.69 | 0.01 | 99 | 77 |

**Table 3.8:** Navigation performance on the R2R unseen test set as indicated on the benchmark leaderboard.

### 3.5.5 Comparison against state-of-the-art

**R2R.** We first evaluate the discriminative model for the R2R task. Similar to VLN-BERT, we evaluate Airbert as an ensemble model created by a linear combination (chosen through grid search) of multiple model outputs (see Table 3.6). First, we see that Airbert alone (row 2) outperforms VLN-BERT (row 1) by 9.4% on the unseen environments and a strong ensemble of speaker and follower models [295] (row 3) by 0.7%. Ensembling Airbert results in a gain of 1.4% over the VLN-BERT ensemble (row 4 vs. 5).

We also obtain results on the test set by submitting our best method to the R2R leaderboard[5]. As seen from Table 3.8, our method of ensembling Airbert, speaker, and follower (similar to VLN-BERT with speaker and follower [69]) achieves the highest success rate at 77% and is ranked first as of the submission deadline. Both VLN-BERT and Airbert use 30 candidate trajectories sampled by beam search with EnvDrop [295], inducing the same path length (PL) for the three methods. As the SPL metric on the leaderboard takes into account the total path length over the 30 trajectories, the SPL is very low and similar across the approaches. Airbert also benefits generative models for the R2R task.

**REVERIE.** Table 3.7 presents results for the REVERIE dataset. The last four rows in the table use Recurrent VLN-BERT [121] with different backbones or parameter initialization. The OSCAR and ViLBERT backbones are pretrained on out-of-domain image-caption pairs. As compared to OSCAR, we observe slight improvements using the ViLBERT backbone for the REVERIE task. VLN-BERT shares the same architecture as ViLBERT, but is pretrained on the R2R dataset, resulting in performance improvement on the unseen environments. Our pretrained Airbert achieves significantly better performance than VLN-BERT, with over 2.4% gain on navigation SR and 1.8% gain on RGS in unseen environments (val unseen). Without any special adaptation, we

[5] https://eval.ai/web/challenges/challenge-page/97/overview also shows performance for ensembling Airbert, VLN-BERT, speaker and follower at a unseen test set SR of 78%.

see that Airbert brings benefits from pretraining on the BnB dataset. We also achieve the state-of-the-art performance on the REVERIE test set by the time of submission, surpassing previous works by a large margin.

### 3.5.6 Results on R2R with Generative Models

Table 3.10 shows the performance of different generative models on the R2R dataset. The OSCAR and ViLBERT backbones for Recurrent VLN-BERT [121] (Rec) are all pretrained on large-scale out-of-domain image-caption pairs with object features and similar self-supervised tasks. On the other hand, the PREVALENT [106] backbone is pretrained on in-domain R2R dataset with scene features and fine-tuned with an additional action prediction task. We suspect that this is the reason for PREVALENT's higher performance as compared to using OSCAR or VLN-BERT as backbones. Note that our Airbert backbone is not fine-tuned further on downstream tasks after pretraining.

Replacing OSCAR's single BERT-like architecture with the ViLBERT architecture slightly improves the performance (similar to our results on the REVERIE dataset presented in the Table 3.7). The VLN-BERT model further fine-tunes ViLBERT on the R2R dataset (with the masking loss). This is beneficial to the navigation performance on the unseen environments validation set[6]. Our Airbert initialization achieves substantial performance improvement as compared to the OSCAR and VLN-BERT backbones on unseen environments, while achieving comparable performance with the PREVALENT initialization.

### 3.5.7 Qualitative results

We visualize the predicted paths from VLN-BERT and Airbert models. In the following figures, ● is the starting viewpoint of the agent, ■ denotes viewpoints in the ground-truth path, ■ for VLN-BERT and ■ for Airbert. Arrows indicate the navigation direction.

**New houses.** In Figure 3.8, we compare predicted paths from VLN-BERT and Airbert in new houses beyond the training environments. Benefiting from BnB dataset that provides diverse visual environments in pretraining, our Airbert model generalizes better to recognize different room types in new houses (see Figure 3.8a-3.8d), and performs better on significantly different environments such as a church (Figure 3.8e) or castle (Figure 3.8f).

**New objects.** Airbert also improves the understanding of new objects in home environments, *e.g.* through noun phrases related to household objects. As shown in Figure 3.9, it is successful at following instructions containing noun phrases that rarely occur or are even unseen on the training set, while the VLN-BERT model that is trained on a large image-caption corpus not pertaining to houses fails.

**Similar environments and instructions.** Figure 3.10 displays examples where the environments and the instructions are similar to those on the training set, with the aim to show that the shuffling loss in pretraining also benefits learning. For example, in Figure 3.10a, the VLN-BERT agent ■ focuses on the stairs (in the last step) and

[6]The performance of VLN-BERT on the seen validation set is lower because the model checkpoint is selected to maximize performance on validation unseen set which happens to be at an earlier iteration.

| Methods | Validation Seen | | | | Validation Unseen | | | |
|---|---|---|---|---|---|---|---|---|
| | TL | NE | SR | SPL | TL | NE | SR | SPL |
| Seq2Seq-SF [13] | 11.33 | 6.01 | 39 | - | 8.39 | 7.81 | 22 | - |
| Speaker-Follower [85] | - | 3.36 | 66 | - | - | 6.62 | 35 | - |
| PRESS [188] | 10.57 | 4.39 | 58 | 55 | 10.36 | 5.28 | 49 | 45 |
| EnvDrop [295] | 11.00 | 3.99 | 62 | 59 | 10.70 | 5.22 | 52 | 48 |
| PREVALENT [106] | 10.32 | 3.67 | 69 | 65 | 10.19 | 4.71 | 58 | 53 |
| Rec (no init. OSCAR) [121] | 9.78 | 3.92 | 62 | 59 | 10.31 | 5.10 | 50 | 46 |
| Rec (OSCAR) [121] | 10.79 | 3.11 | 71 | 67 | 11.86 | 4.29 | 59 | 53 |
| Rec (PREVALENT) [121] | 11.13 | 2.90 | 72 | 68 | 12.01 | 3.93 | 63 | 57 |
| Rec (ViLBERT) | 11.16 | 2.54 | 75 | 71 | 12.44 | 4.20 | 60 | 54 |
| Rec (VLN-BERT) | 10.95 | 3.37 | 68 | 64 | 11.33 | 4.19 | 60 | 55 |
| Rec (Airbert) | 11.09 | 2.68 | 75 | 70 | 11.78 | 4.01 | 62 | 56 |

**Table 3.9:** Navigation performance of different generative models on the R2R dataset for the validation sets.

| Methods | Test Unseen | | | |
|---|---|---|---|---|
| | TL | NE | SR | SPL |
| Seq2Seq-SF [13] | 8.13 | 7.85 | 20 | 18 |
| Speaker-Follower [85] | 14.82 | 6.62 | 35 | 28 |
| PRESS [188] | 10.77 | 5.49 | 49 | 45 |
| EnvDrop [295] | 11.66 | 5.23 | 51 | 47 |
| PREVALENT [106] | 10.51 | 5.30 | 54 | 51 |
| Rec (no init. OSCAR) [121] | 11.15 | 5.45 | 51 | 47 |
| Rec (OSCAR) [121] | 12.34 | 4.59 | 57 | 53 |
| Rec (PREVALENT) [121] | 12.35 | 4.09 | 63 | 57 |
| Rec (Airbert) | 12.41 | 4.13 | 62 | 57 |

**Table 3.10:** Navigation performance of different generative models on the R2R dataset for the testing set.

| # Env. | Traj. | Val Seen SR | | | | | Val Unseen SR | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|
| | | PL | NE | SPL | OSR | SR | PL | NE | SPL | OSR | SR |
| 1 | Rand | 10.97 | 5.36 | 0.44 | 63.74 | 47.87 ±0.03 | 10.84 | 4.86 | 0.51 | 68.46 | 54.48 ±0.04 |
| 6 | Rand | 9.84 | 5.49 | 0.47 | 65.93 | 50.00 ±0.02 | 9.55 | 4.55 | 0.55 | 70.89 | 57.97 ±0.01 |
| 61 | Rand | 10.91 | 4.87 | 0.60 | 76.23 | 64.24 | 9.50 | 3.70 | 0.62 | 76.24 | 65.60 |
| 61 | [295] | 10.59 | 3.21 | 0.69 | 80.71 | 73.85 | 10.03 | 3.24 | 0.63 | 78.45 | 68.67 |

**Table 3.11:** Performance of Airbert on R2R few-shot evaluation. During training, only a subset of the Matterport [44] environments are accessible.

goes upstairs incorrectly, whereas Airbert learns to consider intermediate steps such as "*lounge chairs*" and "*cabinet*" besides the last step by learning from the shuffling task. Similarly, in Figure 3.10c, we see that the VLN-BERT agent stops at the wrong stairs, while Airbert considers intermediate steps such as "*hallway*" and "*wooden doors*", and ends within the acceptable range of 3m from the goal.

**Failure cases.** Figure 3.11 presents some failure cases for both VLN-BERT and Airbert. It reveals that current models still struggle to deal with relationships such as "*between*" (Figure 3.11a), or directional instructions such as "*on the left*" (Figure 3.11b). Similar failures are also highlighted by Table 3.5 where we show that models fail to choose the correct instruction when a direction keyword (left/right) is switched.

### 3.5.8 Training a navigation agent on few houses

We hypothesize that in-domain pretraining, especially one that leverages proposed PI pair generation methods, can achieve superior performance while requiring less training data. To evaluate this, we propose a novel few shot evaluation paradigm for VLN: models are allowed to fine-tune on samples (PI pairs) from one (or few) environments. Few-shot learning for VLN is particularly interesting as visual appearance of houses may differ vastly across geographies, and while training data is hard to obtain, pretraining data like BnB may be readily available.

**One/few shot tasks.** We considered two types of setups: (1) learning from a single environment, which we refer as one-shot learning; and (2) learning from 6 environments (representing 10% of the total training size). For both cases, we randomly sample 5 sets of environments, and report average results. As the number of paths in an environment may have a large impact on performance, we exclude 17 of 61 environments with less than 80 paths.

**Results.** We adopt VLN-BERT, pretrained on ConCaps, as a baseline for few-shot tasks. Recall that fine-tuning VLN-BERT and Airbert on R2R relies on candidate paths drawn from an existing model (EnvDrop [295]). However, as this would lead to unfair comparisons (EnvDrop is trained on the full dataset), candidate paths are sampled as the shortest path between two random positions.

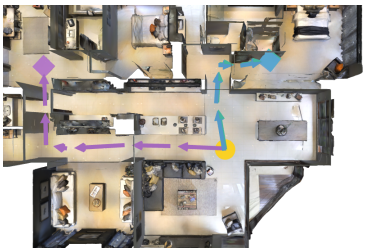
Table 3.11 shows that Airbert largely outperforms VLN-BERT on the unseen validation set: 27.6% with 1 house and 22% with 6 houses. Airbert fine-tuned on 6 houses is almost as good as VLN-BERT on the entire training set. The last two rows of the

table shows that using random paths does not lead to a large performance drop for both models and is a testament to the power of pretrained networks.

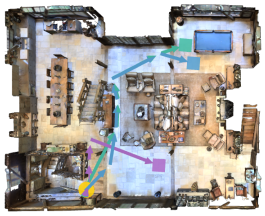
## 3.6 Conclusion

We introduced BnB, a large-scale, in-domain, image-text dataset from houses listed on online rental marketplaces and showed how domain gaps between BnB image-caption pairs and VLN tasks can be mitigated through the creation of path-instruction pairs. We also proposed shuffling, as a means to improve an agent's reasoning about temporal order. Our pretrained model Airbert, achieved state-of-the-art on R2R through the discriminative path-selection setting, and REVERIE through a generative setting. We also demonstrated large performance improvements when applying our model to a challenging one/few-shot VLN setup, highlighting the impact of good pretraining in VLN tasks.

**(a) R2R ✔:** Walk over the kitchen counter, turn left, walk ahead till wall, turn right, walk to the closet room, wait at front.

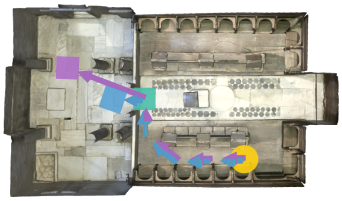


**(b) REVERIE ✔:** Walk past the kitchen and enter the hallway. Turn right at the artwork and wait by the closet.

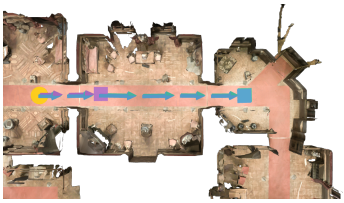


**(c) R2R ✔:** Walk forward to the sitting area to the right of the stairs. Walk to the wall of windows and take a right into the recreation room and stop before you reach the pool table.

**(d) R2R ✔:** Go between the counters, turn left, turn right, and stop before the display and dining room.

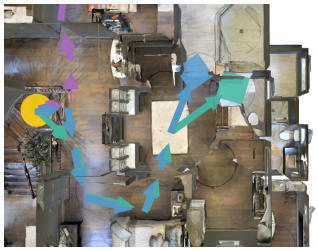


**(e) R2R ✔:** Turn right and head towards the end. Once you reach the end make a right and stop.



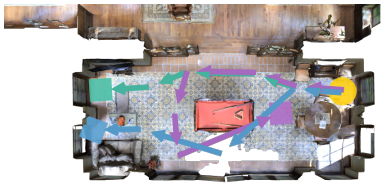
**(f) R2R ✔:** Walk straight out the door in front of you and follow the red carpet. Keep going through the room with the ropes and stop when you enter the next room with ropes.

**Figure 3.8:** When navigating in new houses, our Airbert model not only successfully recognizes the closet room in (a) and (b), pool table (c), living room (d), but also generalizes better to challenging environments, such as the church (e) and castle (f).

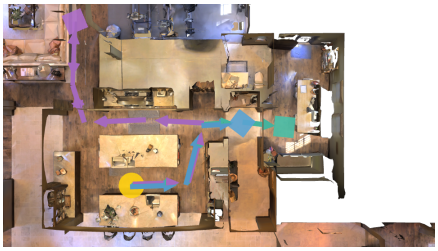


**(a) R2R ✔:** Walk up the stairs and take a right. Walk into the bedroom and take a left . Take another left at the night stand and walk out of the bedroom. Wait by the toilet in the second door on the right.

**(b) R2R ✔:** Go straight past the table and chairs then turn left and continue to go past the table and chairs. Wait near the white antique furniture with the two chairs on on each side.



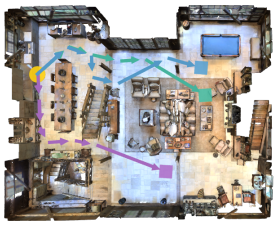
**(c) REVERIE ✔:** Walk past the pool table and towards the TV on the far side of the room and grab the coffee table that is located in front of the couch



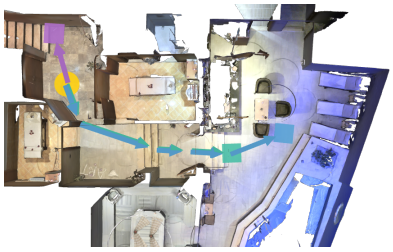
**(d) REVERIE ✔:** Please go to the pantry room with the two large freezers and kitchen appliances on the large table and reset the flipped breaker in the breaker panel box to the right of the freezers

**Figure 3.9:** The Airbert model outperforms VLN-BERT to recognize rare or even unseen objects in training set. (a) Rare object "*night stand*"; (b) unseen object "*antique furniture*"; (c) rare object "*pool table*"; and (d) unseen object "*freezer*".

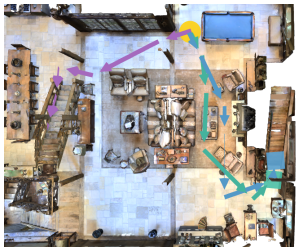
**(a) R2R ✔:** Walk from dining room to living room turning slightly right before lounge chairs, walk straight following cabinet. Turn slight right and stop at stairs.



**(b) R2R ✔:** Walk on into the kitchen and turn to the right. Walk past the staircase, behind the chairs. Walk to the right of the pillar. Stop and wait by the footstool.



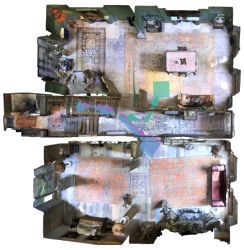
**(c) R2R ✔:** Walk out of the hallway and turn left. Walk down the steps and through the wooden doors. Walk down the steps and stop.



**(d) R2R ✔:** Go straight passed the coffee table turn left and go through the left door to the stairs. Stop in front of the stairs.

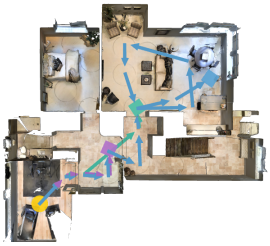
**Figure 3.10:** Examples in similar environments and instructions to the training set. The improvements of Airbert model can be contributed to the shuffling loss in pretraining.

**(a) R2R ✘:** Walk between the two kitchen islands and then turn right. Pass through the stone archway and stop just after you pass through it. Wait there.



**(b) R2R ✘:** Exit the bathroom and go down the stairs. Enter the last doorway on the left and stop just before stepping on the rug.

**(c) REVERIE ✘:** go to level 3 bathroom in the first bedroom left of the stairs and grab the mirror on the wall



**(d) REVERIE ✘:** Go to the lounge on this level and polish the black leather armchair in the corner

**Figure 3.11:** Failure cases for both VLN-BERT and Airbert models.

# Chapter 4

# History Aware Multimodal Transformer for Vision-and-Language Navigation

Vision-and-language navigation aims to build autonomous visual agents that follow instructions and navigate in real scenes. To remember previously visited locations and actions taken, most approaches to VLN implement memory using recurrent states, as in Chapter 3.

In this Chapter, we introduce a History Aware Multimodal Transformer (HAMT) to incorporate a long-horizon history into multimodal decision making. HAMT efficiently encodes all the past panoramic observations via a hierarchical vision transformer (ViT), which first encodes individual images with ViT, then models spatial relation between images in a panoramic observation and finally takes into account temporal relation between panoramas in the history. It, then, jointly combines text, history and current observation to predict the next action. We first train HAMT end-to-end using several proxy tasks including single step action prediction and spatial relation prediction, and then use reinforcement learning to further improve the navigation policy.

HAMT achieves new state of the art on a broad range of VLN tasks, including VLN with *fine-grained instructions* (R2R, RxR), *high-level instructions* (R2R-Last, REVERIE), *dialogs* (CVDN) as well as *long-horizon VLN* (R4R, R2R-Back). We demonstrate HAMT to be particularly effective for navigation tasks with longer trajectories.

## 4.1 Introduction

Vision-and-language navigation (VLN) has recently received growing attention [9, 51, 139, 354, 122]. VLN requires an agent to understand natural language instructions, perceive the visual world, and perform navigation actions to arrive at a target location. A number of datasets have been proposed to support various VLN tasks such as indoor

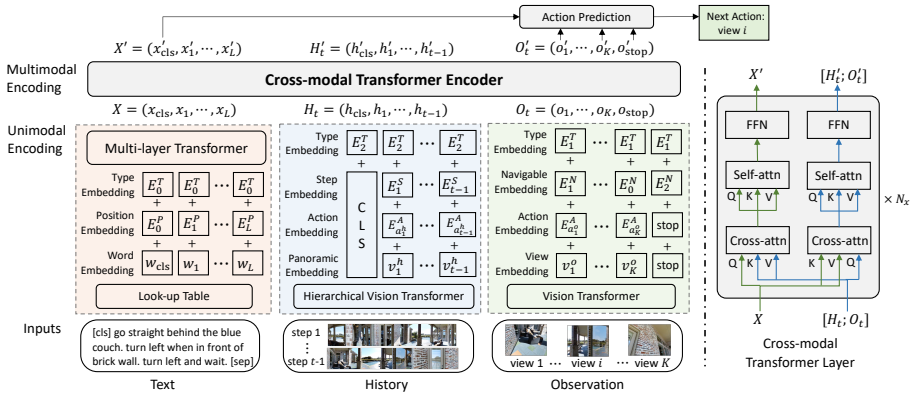



**Figure 4.1:** The architecture of History Aware Multimodal Tranformer (HAMT). HAMT jointly encodes textual instruction, full history of previous observations and actions, and current observation to predict the next action.

and outdoor navigation with fine-grained instructions [51, 14, 169], language-driven remote object finding [251] and navigation in dialogs [303].

VLN agents are faced with several challenges. First, as opposed to static vision-text grounding [345], the agent continuously receives new visual observations and should align them with instructions. Most of existing works adopt recurrent neural networks (RNNs) [14, 85, 296, 212, 321, 118, 196] to encode historical observations and actions within a fixed-size state vector to predict the next action. Such condensed states might be sub-optimal for capturing essential information in extended trajectories [79]. For instance, *"bring the spoon to me"* requires the agent to remember its start location after navigating to the *"spoon"*, while early memories are prone to fade in the recurrent state. Few endeavors [67, 316] construct external map-like memories for received observations. Nevertheless, these approaches still rely on RNNs to track the navigation state. As the history plays an important role in environment understanding and instruction grounding, we propose to explicitly encode the history as a sequence of previous actions and observations instead of using recurrent states.

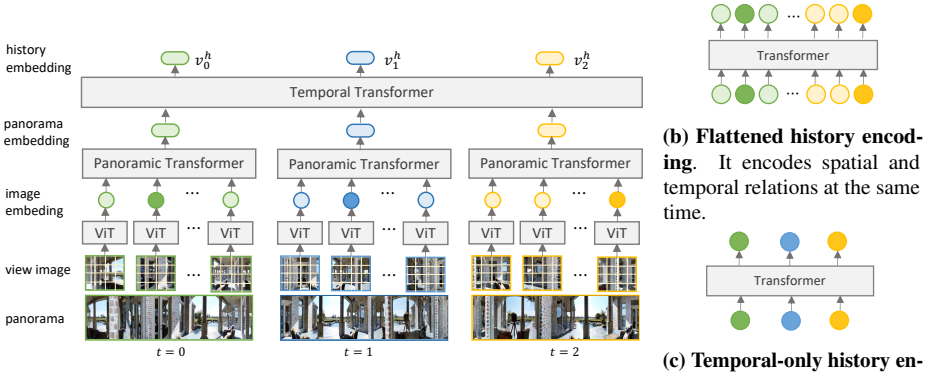
Another VLN challenge concerns the generalizations of agents to new environments that have not been observed during training [354]. One direction is to learn more generic text-image representations. The PRESS model [189] improves language representation with a pretrained BERT encoder [70], and PREVALENT [107] uses pairs of instruction and single-step observations to pretrain a multimodal transformer. Though achieved promising results, these works do not optimize visual representation for the target navigation task. Moreover, lack of history in training [107] makes it hard to learn cross-modal alignment and increases the risk of overfitting to training environments. Another direction towards better generalization is to overcome exposure bias [256] due to discrepancy between training and inference. Different methods have been adopted for VLN including DAgger [14, 262] and scheduled sampling [189, 30]. Reinforcement Learning (RL) [296, 292] is one of the most effective approach among them, but it is considered unstable to directly train large-scale transformers via RL [239].

To address the above challenges, we propose the History Aware Multimodal Transformer (HAMT), a fully transformer-based architecture for multimodal decision making in VLN tasks. As illustrated in Figure 4.1, HAMT consists of unimodal transformers for text, history and observation encoding, and a cross-modal transformer to capture long-range dependencies of the history sequence, current observation and instruction. Since our history contains a sequence of all previous observations, its encoding is computationally expensive. To resolve complexity issues, we propose a hierarchical vision transformer as shown in Figure 4.2, which progressively learns representations for a single view, spatial relationships among views within a panorama and, finally, the temporal dynamics across panoramas of the history. In order to learn better visual representations, we propose auxiliary proxy tasks for end-to-end training. Such tasks include single-step action prediction based on imitation learning, self-supervised spatial relationship reasoning, masked language and image predictions and instruction-trajectory matching. We empirically show that our training facilitates the subsequent fine-tuning of our model with RL [225]. We carry out extensive experiments on various VLN tasks, including VLN with *fine-grained instructions* (R2R [14] and RxR [169]), *high-level instructions* (REVERIE [251] and our proposed R2R-Last), *dialogs* [303] as well as *long-horizon VLN* (R4R [139] and our proposed R2R-Back which requires the agent to return back after arriving at the target location). HAMT outperforms state of the art on both seen and unseen environments in all the tasks.

We summarize our contributions as follows: (1) We introduce HAMT to efficiently model long-horizon history of observed panoramas and actions via hierarchical vision transformer; (2) We train HAMT with auxiliary proxy tasks in an end-to-end fashion and use RL to improve the navigation policy; (3) We validate our method and outperform state of the art in a diverse range of VLN tasks, while demonstrating larger gains for long-horizon navigation.

## 4.2 Related work

**Vision-and-language navigation.** Training instruction-following navigation agents has attracted increasing research attention [9, 51, 14, 169, 251, 276]. Anderson *et al*. [14] propose a sequence-to-sequence LSTM baseline for the VLN task. Fried *et al*. [85] extend it with panoramic action space and synthesized instructions. To improve cross-modal alignment, the self-monitoring agent [212] proposes co-grounding and progress estimation, and RelGraph [118] uses graphs to model relationships across scene, objects and directions. Reinforcement learning (RL) is typically used to improve navigation policy. The EnvDrop model [296] mixes imitation learning and A3C [225]. The RCM [321] utilizes intrinsic reward of cross-modal matching in REINFORCE algorithm. Wang *et al*. [317] propose to learn rewards via soft expert distillation. Due to the success of transformer [312], recent works explore transformer architectures in VLN. PRESS [189] replaces LSTM instruction encoder with pretrained BERT [70]. SIA [196] uses transformer for single-step multimodal fusion and LSTM for sequential action prediction. PTA [175] is a transformer VLN model using CNNs to extract visual features [112]. Here we propose the first full transformer architecture for VLN and train it end-to-end.




**(a) Hierarchical history encoding**. It first encodes individual view images with ViT, then models the spatial relation between images in each panorama, and finally captures the temporal relation between panoramas in the history.

**(b) Flattened history encoding**. It encodes spatial and temporal relations at the same time.

**(c) Temporal-only history encoding**. It only considers temporal relation of oriented views.

**Figure 4.2:** A comparison of history encoding methods. Circle nodes in different colors denote view images of panorama at different steps. Darker circle nodes are the oriented view of the agent.

**Memory-based policy for navigation.** LSTMs [117] have been the dominant approach to encode memories for navigation [14, 85, 296, 321]. Condensing all history into one feature vector, however, is prone to the loss of information. Alternative approaches include topological map memory structures [101, 265]. Deng *et al*. [67] use graphs to capture environment layout and enable long-term planing. A similar graph is adopted in [316] with frontier-exploration based decision making. But these works still utilize LSTMs for state tracking. To exploit long-term spatio-temporal dependencies, Fang *et al*. [79] store histories in a sequence encoded with transformer. Recurrent VLN-BERT [122] injects a recurrent unit to encode histories in transformer for VLN. The most similar work to ours is Episodic Transformer (E.T.) [241]. Differently from [241], we propose a hierarchical encoding of the panoramic observation history and optimize the whole model in end-to-end training.

**Multimodal pretraining with transformers.** Recent works show significant progress in vision and language tasks using multimodal pretraining. In particular, transformer architectures such as one-stream [59, 190] and dual-stream [204, 294] achieve state of the art for a number of downstream tasks including visual question answering, image-text retrieval and image captioning. While most previous methods rely on CNN to extract image representations, ViLT [157] adopts Vision Transformer (ViT) [72] and trains it with associated texts in an end-to-end manner thanks to the efficiency of ViT. A few endeavors [107, 216] explore multimodal pretraining for VLN. PREVALENT [107] pretrains a transformer using instructions and single-step observations without referring to trajectory history. VLN-BERT [216] measures the compatibility between an instruction and images in a path but does not support action prediction. This chapter presents the first end-to-end trainable VLN transformer that jointly encodes text, history and observation, and is able to sequentially predict actions.

## 4.3 Method

**Problem definition** The VLN problem [14] is formulated as a partially observable Markov decision process, where future observations are independent of the past conditioning on current state $s_t$. Given an instruction $\mathscr{W}$ containing a sequence of $L$ words $(w_1, w_2, \cdots, w_L)$, an agent should follow the instruction to move in a connectivity graph to reach the goal location. At each step $t$, the agent receives an observation $\mathscr{O}_t$, a panorama of its surrounding environment. The $\mathscr{O}_t$ consists of $K$ single view images split from the panorama $\mathscr{O}_t \triangleq ([v_1^o; a_1^o], \cdots, [v_K^o; a_K^o])$, where $v_i^o$ is the visual feature of the $i$-th view and $a_i^o$ denotes the relative angle to face the view (subscript $t$ is omitted for simplicity). There are $n$ navigable viewpoints among all the $K$ views[1], denoted as $\mathscr{O}_t^c \triangleq ([v_1^c; a_1^c], \cdots, [v_n^c; a_n^c])$. We follow the setup in [85] and use $\mathscr{O}_t^c$ as the decision space, so the agent only needs to select a candidate in $\mathscr{O}_t^c$ at each step. All observations $\mathscr{O}_i$ and performed actions $a_i^h$ before step $t$ form the history $\mathscr{H}_t \triangleq ([\mathscr{O}_1; a_1^h], \cdots, [\mathscr{O}_{t-1}; a_{t-1}^h])$, where $a_i^h$ denotes the turned angles at step $i$. The goal is to learn a policy $\pi$ parametrized by $\Theta$ to predict the next action based on the instruction, history and the current observation, which is $\pi(a_t | \mathscr{W}, \mathscr{H}_t, \mathscr{O}_t, \mathscr{O}_t^c; \Theta)$.

Unlike dominant recurrent approaches to condense $\mathscr{H}_t$ into a fixed-size vector, in this section, we present the History Aware Multimodal Transformer (HAMT) that jointly encodes text, long-horizon history, and observation for sequential action prediction. The model architecture is described in Section 4.3.1. We propose end-to-end training for HAMT in Section 4.3.2 to learn unimodal and multimodal representations, and then use RL to fine-tune the navigation policy in Section 4.3.3.

### 4.3.1 HAMT: History Aware Multimodal Transformer

Figure 4.1 illustrates the model architecture of HAMT. The inputs text $\mathscr{W}$, history $\mathscr{H}_t$ and observation $\mathscr{O}_t$ are first encoded via the corresponding unimodal transformers respectively, and then fed into the cross-modal transformer encoder to capture multimodal relationships.

**Text Encoding.** For each token $i$ in the instruction $\mathscr{W}$, we embed it as the summation of its word embedding $w_i$, position embedding $E_i^P$ and type embedding of text $E_0^T$. Then we employ a transformer with $N_L$ layers to obtain contextual representation $x_i$ following the standard BERT [70].

**Observation Encoding.** For each view $[v_i^o; a_i^o]$ in the panoramic observation $\mathscr{O}_t$, we first represent the relative angle $a_i^o$ as $E_{a_i^o}^A = (\sin\theta_i, \cos\theta_i, \sin\phi_i, \cos\phi_i)$ where $\theta_i$ and $\phi_i$ are the relative heading and elevation angle to the agent's orientation. Then the observation embedding $o_i$ is as follows:

$$o_i = \mathrm{LN}(W_v^o v_i^o) + \mathrm{LN}(W_a^o E_{a_i^o}^A) + E_{o_i}^N + E_1^T \tag{4.1}$$

where $W_v^o, W_a^o$ are learnable weights. The $E_{o_i}^N$ denotes the navigable embedding to differentiate types of views, with $E_0^N$ for non-navigable view, $E_1^N$ for navigable view

[1] A navigable view can lead to one or multiple viewpoints. We follow [122, 296] to use different features for these viewpoints. The viewpoints share the same visual features but differ in angle features.

and $E_2^N$ for stop view (we append a stop token in observation to support stop action). The $E_1^T$ is the type embedding of observation. We omit bias terms for simplicity. The LN denotes layer normalization [23]. Because $a_i^o$ has much lower feature dimensions than $v_i^o$, we apply LN to balance the encoded $a_i^o$ and $v_i^o$.

**Hierarchical History Encoding.** As $\mathscr{H}_t$ consists of all the past panoramic observations $\mathscr{O}_i$ and performed actions $a_i^h$ before step $t$, it is important to encode $\mathscr{H}_t$ efficiently as context. Figures 4.2b-4.2c depict the flattened and temporal-only history encoding approaches used in VLN-BERT [216] and E.T. [241] respectively. The flattened approach treats each view image in $\mathscr{O}_i$ as a token, so the history sequence contains $tK$ tokens. Though it enables to learn relationships among all image views, the computation cost quadratically increases with the sequence length, making it inefficient for long-horizon tasks. In the temporal-only approach, only the oriented view of the agent in each $\mathscr{O}_i$ is taken as inputs instead of the whole panorama, so only $t$ temporal tokens are encoded. However, this approach can lose critical information in past observations. For example, in the instruction *"with the windows on your left, walk through the large room past the sitting areas"*, the object *"window"* does not appear in the oriented view of the agent. Therefore, the encoded history is insufficient to tell whether the agent passed the window or not, making the model confused to take the next action.

In order to balance computational efficiency and information integrity, we propose a hierarchical history encoding approach as illustrated in Figure 4.2a. It hierarchically encodes view images within each panorama and then temporal relationships across panoramas, similar to the factorized spatial-temporal video transformer [18]. For each $\mathscr{O}_i$, its constituent view images are first embeded via ViT and Eq (4.1), and then encoded via a panoramic transformer with $N_h$ layers to learn spatial relationships within the panorama. We apply average pooling to obtain panorama embedding, and add it with the oriented view image feature in residual connection. The parameters in ViT and panoramic transformer are shared for different steps. In this way, each historical observation $\mathscr{O}_i$ is represented as $v_i^h$, and the final temporal token $h_i$ is computed as:

$$h_i = \mathrm{LN}(W_v^h v_i^h) + \mathrm{LN}(W_a^h E^A_{a_i^h}) + E_i^S + E_2^T \tag{4.2}$$

where $E_i^S$ denotes the $i$-th step embedding, $E_2^T$ is the type embedding of history. The computational cost is $O(tK^2 + t^2)$, which significantly reduces from $O(t^2K^2)$ in the flattened approach. To be noted, we add a special token `[cls]` to the start of the history sequence to obtain a global representation. The embedding of `[cls]` is a parameter to learn, which is initialized from a zero vector.

**Cross-modal Encoding.** We concatenate history and observation as the vision modality, and use cross-modal transformer with $N_x$ layers to fuse features from text, history and observation as shown in the right of Figure 4.1. The reason of using such dual-stream architecture rather than one-stream is that the length of different modalities can be highly imbalanced, and the dual-stream architecture can balance the importance of intra- and inter-modal relationships by model design [40]. In each cross-modal layer, a vision-text cross-attention is firstly performed for vision modality to attend relevant text information and vice versa for text modality. Then each modality uses self-attention to learn intra-modal relationship such as interaction between observation and history, followed by a fully-connected neural network. Finally, the HAMT model outputs

**Table 4.1:** Comparison of HAMT and previous VLN transformers.

| Models | Inputs | | | Proxy Tasks | | | | |
|---|---|---|---|---|---|---|---|---|
| | Text | History | Observation | MLM | MRM | ITM | SAP/SAR | SPREL |
| PREVALENT [107] | ✓ | | ✓ | ✓ | | | ✓ | |
| VLN-BERT [216] | ✓ | ✓ | | ✓ | ✓ | ✓ | | |
| HAMT (Ours) | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ |

embeddings $X^{'} = (x'_{\text{cls}}, x'_1, \cdots, x'_L), H'_t = (h'_{\text{cls}}, h'_1, \cdots, h'_{t-1}), O'_t = (o'_1, \cdots, o'_K, o'_{\text{stop}})$ for tokens in text, history and observation respectively.

### 4.3.2 End-to-end training with proxy tasks

As it is difficult to train large-scale transformers with RL due to sparse supervision [239], we propose to first end-to-end train HAMT via several proxy tasks to learn unimodal and multimodal representation.

Table 4.1 compares our HAMT with previous VLN transformers PREVALENT [107] and VLN-BERT [216] in inputs and proxy tasks. As neither PREVALENT nor VLN-BERT jointly encodes text, history and observation, a limited choice of proxy tasks can be applied in training. Our model instead can take advantage of various proxy tasks to learn cross-modal alignment, spatial and temporal reasoning, and history-aware action prediction. Given the input pair $(\mathscr{W}, \mathscr{H}_T)$ where $T$ is the length of full trajectory, we can apply common proxy tasks as in vision-and-language pretraining [204, 216], including Masked Language Modeling (MLM), Masked Region Modeling (MRM) and Instruction Trajectory Matching (ITM). In the following, we introduce new proxy tasks given the triplet input $(\mathscr{W}, \mathscr{H}_t, \mathscr{O}_t)$ specifically for VLN tasks.

**Single-step Action Prediction/Regression (SAP/SAR).** The task deploys imitation learning to predict the next action based on instruction, history from expert demonstration and the current observation. We formulate it as a classification and a regression task respectively. In the SAP classification task, we predict action probability for each navigable view in $\mathscr{O}_t^c$ which is $p_t(o'_i) = \frac{\exp(f_{\text{SAP}}(o'_i \odot x'_{\text{cls}}))}{\sum_j \exp(f_{\text{SAP}}(o'_j \odot x'_{\text{cls}}))}$, where $f_{\text{SAP}}$ is a two-layer fully-connected network, $\odot$ is element-wise multiplication and $x'_{\text{cls}}$ is output embedding of special text token `[cls]`. The objective is to minimize negative log probability of the target view action $o'_*$: $L_{\text{SAP}} = -\log\, p_t(o'_*)$. In SAR regression task, we directly predict the action heading and elevation angles based on the text token `[cls]` which is $\hat{\theta}_t, \hat{\phi}_t = f_{\text{SAR}}(x'_{\text{cls}})$. The loss function is $L_{\text{SAR}} = (\hat{\theta}_t - \theta_t)^2 + (\hat{\phi}_t - \phi_t)^2$. The two proxy tasks enable the model to learn how to make action decision conditioning on instruction and contextual history.

**Spatial Relationship Prediction (SPREL).** Expressions of egocentric and allocentric spatial relations are frequent in navigational instructions, such as *"walk into the room on your left"* and *"enter the bedroom next to the stairs"*. In order to learn spatial relation aware representations, we propose the SPREL self-supervised task to predict relative spatial position of two views in a panorama based on only visual feature, angle feature or both. Assume $[v_i^o; a_i^o]$ and $[v_j^o; a_j^o]$ are two views in $\mathscr{O}_t$, we randomly zero out $v_*^o$ or $a_*^o$

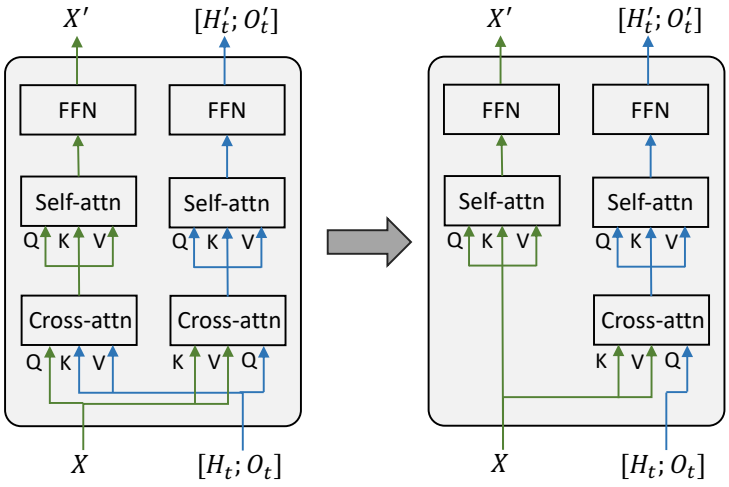



**Figure 4.3:** Comparison of the original cross-modal transformer layer (left) and the encoder-decoder based variant (right).

with probability of 0.3. Their encoded representations are $o_i'$ and $o_j'$, and their relative heading and elevation angles are $\theta_{ij}, \phi_{ij}$. We then predict $\hat{\theta}_{ij}, \hat{\phi}_{ij} = f_{\text{SPREL}}([o_i'; o_j'])$ where $[;]$ denotes vector concatenation and optimize $L_{\text{SPREL}} = (\hat{\theta}_{ij} - \theta_{ij})^2 + (\hat{\phi}_{ij} - \phi_{ij})^2$. The task helps for spatial relationship reasoning in the observation.

**Training Strategy.** Instead of directly training the whole HAMT model at once, we propose to progressively train HAMT in two stages. In the first stage, we freeze ViT pretrained on ImageNet [66] and train the rest of the modules which are randomly initialized. This aims to avoid catastrophic forgetting of the pretrained weights in ViT. Then we unfreeze ViT and train the whole model end-to-end. The learning rate for ViT is set to be higher than for others modules to avoid vanishing gradients and to speedup convergence.

### 4.3.3 Fine-tuning for sequential action prediction

**Structure Variants.** We present two variants of HAMT for action prediction in the following. 1) MLP action head: we directly reuse the action prediction network $f_{\text{SAP}}$ in the SAP task to predict navigable views. We use it as default for VLN tasks. 2) MLP action head based on encoder-decoder structure: the original HAMT model applies cross-modal attention for both vision-to-text and text-to-vision, which is computationally expensive when instructions are long. Therefore, we remove the cross-modal attention from text to vision. In this way, we separate the cross-modal transformer into an encoder which only takes instruction as input, and a decoder that inputs history and observation as query and attends over encoded text tokens.

We present the encoder-decoder variant of HAMT in fine-tuning on the right of Figure 4.3. Compared to the original cross-modal transformer on the left, the variant

removes text-to-vision cross-modal attention. The encoder encodes the texts to obtain textual embeddings. Then the decoder reuses the same text embeddings in vision-to-text attention layer at each navigation step. In this way, the variant is more efficient when instructions are long *e.g.* in R4R and RxR datasets. here $\mu$ is the learning rate, $a_t^*$ is the expert action at step $t$ of the expert trajectory of length $T^*$.

**RL+IL Objective.** We combine Reinforcement Learning (RL) and Imitation Learning (IL) to fine-tune HAMT for sequential action prediction. The IL relies on the SAP loss defined in Section 4.3.2 and follows the expert action at each step while RL samples actions according to the policy $\pi$. Specifically, we use the Asynchronous Advantage Actor-Critic (A3C) RL algorithm [225]. At each step $t$, the agent samples an action based on policy $\pi$: $\hat{a}_t^h \sim \pi(a_t|\mathscr{W}, \mathscr{H}_t, \mathscr{O}_t, \mathscr{O}_t^c)$ and receives an immediate reward $r_t$. For non-stop actions, we set $r_t$ as the reduced distance of taking the action to the target and the increased alignment score [139] compared to expert demonstration as defined in [122]; for the stop action, $r_t = 2$ if the agent successfully arrives at the target otherwise -2. A critic network is trained to estimate the value of each state $s_t$, which is $R_t = \sum_{k=0}^{T-t} \gamma^k r_{t+k}$ where $\gamma$ is discount factor. We implement it as $V_t = f_{\text{critic}}(x'_{\text{cls}} \odot h'_{\text{cls}})$. As the reward signal favors shortest distance, we empirically find it benefits to combine A3C RL with IL weighted by $\lambda$, which is:

$$\Theta \leftarrow \Theta + \underbrace{\mu \frac{1}{T} \sum_{t=1}^{T} \nabla_\Theta \log \pi(\hat{a}_t^h; \Theta)(R_t - V_t)}_{\text{Reinforcement Learning (RL)}} + \underbrace{\lambda \mu \frac{1}{T^*} \sum_{t=1}^{T^*} \nabla_\Theta \log \pi(a_t^*; \Theta)}_{\text{Imitation Learning (IL)}} \tag{4.3}$$

### 4.3.4 Proxy tasks in training

We employ five proxy tasks to train HAMT and introduced SAP/SAR and SPREL in Section 4.3.2. In the following, we present the other three proxy tasks, which are all based on the input pair $(\mathscr{W}, \mathscr{H}_T)$, where $\mathscr{W}$ is the textual instruction and $\mathscr{H}_T$ is the full trajectory with length $T$.

**Masked Language Modeling (MLM).** The task predicts masked words based on contextual words and the full trajectory. We randomly mask out tokens in $\mathscr{W}$ with the probability of 15% with a special token `[mask]` as in BERT, and predict the word distribution $p(w_i|\mathscr{W}_{\setminus i}, \mathscr{H}_T) = f_{\text{MLM}}(x'_i)$ where $\mathscr{W}_{\setminus i}$ is the masked instruction, $x'_i$ is the output embedding of the masked word $w_i$ and $f_{\text{MLM}}$ is a two-layer fully-connected network. The objective is to minimize the negative log-likelihood of original words: $L_{\text{MLM}} = -\log p(w_i|\mathscr{W}_{\setminus i}, \mathscr{H}_T)$. The task is beneficial to learn grounded language representations and cross-modal alignment.

**Masked Region Modeling (MRM).** The task aims to predict semantic labels of masked observations in the trajectory given an instruction and neighboring observations. We zero out observations in $\mathscr{H}_T$ 15% of the time. The target of a masked $\mathscr{O}_i$ is the class probability predicted by an image classification model pretrained on ImageNet. We use ViT-B/16 [72] in this chapter. Suppose $P_i \in \mathbb{R}^{1000}$ is the target class probability for a masked $\mathscr{O}_i$, we predict $\hat{P}_i = f_{\text{MRM}}(o'_i)$ where $o'_i$ is the output embedding of masked $\mathscr{O}_i$, and minimize the KL divergence between the two probability distributions: $L_{\text{MRM}} = -\sum_{j=1}^{1000} P_{i,j} \log \hat{P}_{i,j}$. In order to solve the task, $o'_i$ should capture temporal continuity in the history sequence and align with relevant instructions.

**Instruction Trajectory Matching (ITM).** The task predicts whether a pair of instruction and trajectory is aligned. We predict the alignment score as $s(\mathscr{W}, \mathscr{H}_T) = f_{\text{ITM}}(x'_{\text{cls}} \odot h'_{\text{cls}})$, where $\odot$ is element-wise multiplication and $x'_{\text{cls}}, h'_{\text{cls}}$ are output embeddings for the text `[cls]` token and the history `[cls]` token respectively. We sample 4 negative trajectories for each positive instruction-trajectory pair during training, in which two negative trajectories are randomly selected from other positive pairs in the mini-batch, two are obtained by temporally shuffling the positive trajectory. The objective is the Noisy Contrastive Estimation loss [103]: $L_{\text{ITM}} = -\log \frac{\exp(s(\mathscr{W}, \mathscr{H}_T))}{\exp(s(\mathscr{W}, \mathscr{H}_T)) + \sum_{k=1}^{4} \exp(s(\mathscr{W}, \mathscr{H}^{\text{neg}}_{T,k}))}$. The model is supposed to learn cross-modal alignment and be sensitive to temporal orders of history to solve the task.

## 4.4 Experiments

### 4.4.1 Experimental setup

**Datasets.** We evaluate our method on four VLN tasks (seven datasets): *VLN with fine-grained instructions* (R2R [14], RxR [169]); *VLN with high-level instructions* (REVERIE [251], R2R-Last); *vision-and-dialogue navigation* (CVDN [303]); and *long-horizon VLN* (R4R [139], R2R-Back).

- **R2R** [9] builds upon Matterport3D [43] and includes 90 photo-realistic houses with 10,567 panoramas. It contains 7,189 shortest-path trajectories, each associated with 3 instructions. The dataset is split into train, val seen, val unseen and test unseen sets with 61, 56, 11 and 18 houses respectively. Houses in val seen split are the same as training, while houses in val unseen and test splits are different from training.
- **RxR** [169] is a large multilingual VLN dataset based on Matterport 3D. The instructions are in three different languages (English, Hindi and Telugu). The dataset emphasizes the role of language in VLN by addressing biases in paths and describing more visible entities than R2R.
- **R4R** [139] extends R2R dataset by concatenating two adjacent tail-to-head trajectories in R2R. Therefore, it has longer instructions and trajectories. The trajectories are also less biased as they are not necessarily the shortest-path from start to end location.
- **R2R-Back** is a new VLN setup proposed in this chapter. The agent is required to return to its start location after arriving at the destination. The agent needs to remember its navigation histories to solve the task. We add a return command at the end of each instruction in R2R and a reverse path from the end to start locations as expert demonstration.
- **CVDN** [303] defines a navigation from dialog history task, which requires an agent to arrive at goal regions based on multi-turn question-answering dialogs. Such types of instructions are often ambiguous and under-specified. The lengths of instructions and paths are also long.
- **REVERIE** [251] replaces step-by-step instructions in R2R with high-level instructions, which mainly describe the target location and object. The agent, hence, is

required to navigate to the goal without detailed guidance and depends on its past experiences.

- **R2R-Last** is our proposed VLN setup similar to REVERIE. It only uses the last sentence from the original R2R instructions describing the final destination.

Table 4.2 summarizes details of the dataset split. The proposed R2R-Back and R2R-Last setups consider exactly the same splits as the R2R dataset. We present details to construct R2R-Back and R2R-Last in the following.

**Table 4.2:** Dataset statistics. #traj, #instr denote the number of trajectories and instructions respectively.

| Dataset | Train | | Val Seen | | Val Unseen | | Test Unseen | |
|---|---|---|---|---|---|---|---|---|
| | #traj | #instr | #traj | #instr | #traj | #instr | #traj | #instr |
| R2R [14] | 4,675 | 14,039 | 340 | 1,021 | 783 | 2,349 | 1,391 | 4,173 |
| RxR [169] | 11,077 | 79,467 | 1,244 | 8,813 | 1,517 | 13,652 | - | 11,888 |
| R4R [139] | 25,921 | 233,532 | 115 | 1,035 | 5,026 | 45,234 | - | - |
| R2R-Back | 4,675 | 14,039 | 340 | 1,021 | 783 | 2,349 | - | - |
| CVDN [303] | 4,742 | 4,742 | 382 | 382 | 907 | 907 | 1,384 | 1,384 |
| R2R-Last | 4,675 | 14,039 | 340 | 1,021 | 783 | 2,349 | - | - |
| REVERIE [251] | 4,150 | 10,466 | 515 | 1,423 | 1,328 | 3,521 | 2,304 | 6,292 |

**R2R-Back.** We append a returning command at the end of annotated instructions in R2R to create new instructions for R2R-Back. The returning command is randomly sampled from the following sentences: "walk back to the start", "return by the way you came", "double back to where you start", "backtrack to the start", "back the way you came", "return to the starting point". The original target location is viewed as a middle stop point. The groundtruth trajectory in R2R-Back is the concatenation of the original and its inverse trajectory.

**R2R-Last.** We use spacy toolkit[2] to split sentences for instructions in R2R. We only select the last sentence in each instruction as the new high-level instruction. It mainly describes where the goal location is *e.g.* "stop in front of the vent", requiring the agent to explore houses without step-by-step textual guidance. The groundtruth trajectory is the same as R2R.

**Evaluation metrics.** We adopt standard metrics [9], including (1) Trajectory Length (TL): the agent's navigated path in meters; (2) Navigation Error (NE): the average distance in meters between the agent's final position and the target; (3) Success Rate (SR): the ratio of trajectories reaching the destination with a maximum error of 3 meters to the target; and (4) Success Rate normalized by the ratio between the length of the shortest path and the predicted path (SPL). SPL is more relevant than SR as it balances the navigation accuracy and efficiency. For long-horizon VLN task (R4R and R2R-Back), we further employ three metrics to measure the path fidelity between the predicted path and target path, including (5) Coverage weighted by Length Score (CLS) [139]; (6) the normalized Dynamic Time Warping (nDTW) [134]; and (7) the Success

[2] https://spacy.io/

weighted by nDTW (SDTW).

In R2R, RxR, R4R and R2R-Last datasets, a predicted trajectory is considered to be successful if the agent arrives 3 meters near to the final destination. However, such definition would make a motionless agent achieve 100% success rate (SR) on R2R-Back dataset as the final destination is the same as the starting location. Therefore, in R2R-Back evaluation, we define the success as that an agent firstly arrives 3 meters near to the original destination and then returns 3 meters near to its starting location. The groundtruth length in the SPL metric is also modified as the total traversed distance in groundtruth trajectory rather than the shortest distance between start and target location. As the REVERIE task aims for remote object grounding, the success on REVERIE is defined as arriving at a viewpoint where the target object is visible.

**Implementation details.** For the HAMT model, we set $N_L = 9$ for language transformer, $N_h = 2$ for panoramic transformer in hierarchical history encoding, and $N_x = 4$ for cross-modal transformer. There are $K = 36$ view images in each panoramic observation. We use ViT-B/16 [72] for image encoding if not otherwise specified. In training with proxy tasks, we randomly select proxy tasks for each mini-batch with predefined ratio. We train HAMT for 200k iterations with fixed ViT using learning rate of 5e-5 and batch size of 64 on 4 NVIDIA Tesla P100 GPUs ($\sim$1 day). The whole HAMT model is trained end-to-end for 20k iterations on 20 NVIDIA V100 GPUs with learning rate of 5e-5 for ViT and 1e-5 for the others ($\sim$20 hours). We use R2R training set and augmented pairs from [107] for training unless otherwise noted. In fine-tuning with RL+IL, we set $\lambda = 0.2$ in Eq (4.3) and $\gamma = 0.9$. The model is fine-tuned for 100k iterations with learning rate of 1e-5 and batch size of 8 on a single GPU. Unimodal encoders are fixed by default. The best model is selected according to performance on val unseen split. We use the same augmented data as [122] for R2R for fair comparison, while no augmented data is used for other datasets. Greedy search is applied in inference following the single-run setting.

**Training with proxy tasks.** We sample proxy tasks for each mini-batch to train the HAMT model. The sampling ratio is MLM:MRM:ITM:SAP:SAR:SPREL=5:2:2:1:1:1. The optimizer is AdamW [202]. In the end-to-end training stage, we use image augmentation and regularization techniques to avoid overfitting of the ViT model, including RandAugment [62] and stochastic depth [127].

**Fine-tuning for sequential action prediction.** Due to different goals in various VLN tasks, we design different rewards in reinforcement learning for each downstream VLN dataset. In R2R, RxR and R4R datasets, the reward is introduced in Section 4.3.3 to take both goal distance and path fidelity into account. In R2R-Last, REVERIE and CVDN datasets where the instruction may not describe detailed navigation path, we only use the reduced distance to the goal viewpoints as rewards. We normalize the reduced distance in the same way as in the R2R dataset. In R2R-Back dataset, we use a different fine-tune strategy to avoid trivial motionless solutions. We require the agent to predict stop actions twice for the original destination (midpoint) and its starting point (final destination) respectively. Before arriving at the midpoint, the RL reward is computed based on distances to the midpoint. If the agent predicts a wrong location to stop for the midpoint, the episode is stopped; otherwise the agent continues its task while receiving rewards based on the distance to the final destination for fine-tuning. We

**Table 4.3:** R2R navigation results for alternative methods of history encoding. All methods use Resnet152 visual features and are trained from scratch on R2R dataset.

| History Encoding | Val Seen SR↑ | Val Seen SPL↑ | Val Unseen SR↑ | Val Unseen SPL↑ |
|---|---|---|---|---|
| RecBERT [122] | 62 | 59 | 50 | 46 |
| Recurrent | $60.9_{\pm 1.0}$ | $56.6_{\pm 1.1}$ | $52.2_{\pm 0.7}$ | $47.0_{\pm 0.5}$ |
| Temporal-only | $61.5_{\pm 0.8}$ | $57.7_{\pm 0.7}$ | $53.2_{\pm 0.1}$ | $48.0_{\pm 0.4}$ |
| Hierarchical | $\mathbf{65.5}_{\pm 1.2}$ | $\mathbf{61.3}_{\pm 1.4}$ | $\mathbf{54.4}_{\pm 0.4}$ | $\mathbf{48.7}_{\pm 0.4}$ |

run each experiment twice for ablation study and use the best result on the validation unseen split for the state-of-the-art comparison.

### 4.4.2 Ablation studies

In this section, we evaluate each component in the HAMT model, including: hierarchical history encoding, end-to-end training with proxy tasks, and fine-tuning objectives.

**How important is the history encoding for VLN?** For fair comparison with the state-of-the-art recurrent architecture RecBERT [122], we use the same Resnet152 visual features and train all the models from scratch with RL+IL objectives to avoid the influence of different weight initialization. The models are optimized for 300k iterations end-to-end except for the visual feature. Table 4.3 compares different history encoding approaches on R2R dataset. Our recurrent model slightly differs from RecBERT (no init. OSCAR) [122] in transformer architecture as shown in Figure 4.1. It achieves slightly better performance on val unseen split. The temporal-only model uses transformer to encode agent's oriented visual observations in history sequence, and outperforms the recurrent method by relative gains of 1.9% on SR and 2.1% on SPL for val unseen split. Adding panoramic observations in a hierarchical way results in 4.2% (SR) and 3.6% (SLP) relative improvements on the val unseen split compared to the recurrent method. Even larger improvements are achieved on val seen split as the hierarchical model has a larger capacity to fit the seen environments. This evaluation demonstrates the advantage of our hierarchical history representation compared to the recurrent and temporal-only history representation.

**How much does training with proxy tasks help?** We next evaluate the advantage of training HAMT end-to-end with proxy tasks. In Table 4.4a, the first row uses RL+IL objectives to train HAMT from scratch, while the second row uses proxy tasks for training prior to RL+IL fine-tuning. We can see that it significantly boosts the performance to first train with proxy tasks. It improves on val unseen split with 16.7% and 18.0% relative gains on SR and SPL respectively, indicating that training with auxiliary proxy tasks enables better generalization. In the third row, we replace the visual feature from Resnet152 to ViT. The ViT feature improves the performance on

**Table 4.4:** Ablations for end-to-end HAMT training on R2R dataset using proposed proxy tasks.

**(a)** Comparison of visual features and end-to-end training. The "PT" stands for proxy tasks in training; "e2e" for optimizing the visual representation.

| feature | PT | e2e | Val Seen SR↑ | Val Seen SPL↑ | Val Unseen SR↑ | Val Unseen SPL↑ |
|---|---|---|---|---|---|---|
| Resnet 152 | × | × | $65.5_{\pm 1.2}$ | $61.3_{\pm 1.4}$ | $54.4_{\pm 0.4}$ | $48.7_{\pm 0.4}$ |
| | ✓ | × | $69.3_{\pm 1.0}$ | $64.8_{\pm 1.2}$ | $63.5_{\pm 0.5}$ | $57.5_{\pm 0.5}$ |
| ViT | ✓ | × | $\mathbf{75.7}_{\pm 1.0}$ | $\mathbf{72.5}_{\pm 1.0}$ | $64.4_{\pm 0.3}$ | $58.8_{\pm 0.0}$ |
| | ✓ | ✓ | $75.0_{\pm 0.9}$ | $71.7_{\pm 0.7}$ | $\mathbf{65.7}_{\pm 0.7}$ | $\mathbf{60.9}_{\pm 0.7}$ |

**(b)** Comparison of different proxy tasks. The "SAP(R)" denotes the single step action prediction and regression task, and "SPREL" is the spatial relationship prediction task.

| SAP (R) | SP REL | Val Seen SR↑ | Val Seen SPL↑ | Val Unseen SR↑ | Val Unseen SPL↑ |
|---|---|---|---|---|---|
| × | × | $71.2_{\pm 2.3}$ | $67.2_{\pm 2.0}$ | $62.8_{\pm 1.3}$ | $57.7_{\pm 1.0}$ |
| ✓ | × | $74.7_{\pm 0.6}$ | $71.1_{\pm 0.9}$ | $63.6_{\pm 0.1}$ | $58.1_{\pm 0.4}$ |
| ✓ | ✓ | $\mathbf{75.7}_{\pm 1.0}$ | $\mathbf{72.5}_{\pm 1.0}$ | $\mathbf{64.4}_{\pm 0.3}$ | $\mathbf{58.8}_{\pm 0.0}$ |

**Table 4.5:** Ablations for fine-tuning objectives of sequential action prediction on R2R dataset.

| IL | RL | Val Seen SR↑ | Val Seen SPL↑ | Val Unseen SR↑ | Val Unseen SPL↑ |
|---|---|---|---|---|---|
| × | × | 57.9 | 54.8 | 51.8 | 48.9 |
| ✓ | × | $63.7_{\pm 2.1}$ | $61.7_{\pm 2.2}$ | $57.2_{\pm 0.1}$ | $54.7_{\pm 0.3}$ |
| × | ✓ | $70.5_{\pm 2.9}$ | $65.6_{\pm 2.8}$ | $63.5_{\pm 1.4}$ | $57.5_{\pm 1.1}$ |
| ✓ | ✓ | $\mathbf{75.0}_{\pm 0.9}$ | $\mathbf{71.7}_{\pm 0.7}$ | $\mathbf{65.7}_{\pm 0.7}$ | $\mathbf{60.9}_{\pm 0.7}$ |

both val seen and val unseen splits, showing that more powerful visual representations matter. Finally, training ViT end-to-end obtains 2.1% gains on SPL on val unseen split. This is the first time to show that optimizing visual representations end-to-end is beneficial for VLN tasks. In Table 4.4b, we evaluate the benefit of the two new proxy tasks for frozen ViT features using the other proxy tasks by default. The SAP(R) uses imitation learning to predict actions, which directly influences the navigation policy and improves the performance by a large margin. The SPREL is a self-supervised proxy task that forces the model to learn spatial relationships in panorama and helps generalization in unseen environments.

**What is the impact of the fine-tuning objectives?** Table 4.5 presents results using different objectives in fine-tuning. The first row directly applies HAMT trained by proxy tasks, which achieves lower performance than that after IL fine-tuning, because we mainly use augmented data in proxy task training to increase visual diversity, but such noisy data deteriorates action prediction performance. Previous work [296] has shown that RL alone performs poorly. However, training with proxy tasks stabilizes the followup RL fine-tuning. HAMT optimized by RL achieves much better performance than that when fine-tuning with IL on the SR metric. It indicates that RL is able to learn better exploration strategy on unseen environments. However, as the reward for RL focuses more on shortest paths rather than path fidelity with instructions, the improvement on SPL metric is relatively small compared to SR metric. Moreover, the fluctuation of the pure RL objective is larger than IL. Therefore, mixing the RL and IL achieves the best performance.

**Table 4.6:** Computation time in inference on R2R val unseen split.

| | Inference Time (s) | SR | SPL |
|---|---|---|---|
| RecBERT [122] | 69 | 63 | 57 |
| HAMT | 104 | 66 | 61 |
| HAMT noT2V | 76 | 65 | 60 |

**Table 4.7:** Comparison with state-of-the-art methods on R2R dataset.

| Methods | Validation Seen | | | | Validation Unseen | | | | Test Unseen | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | TL | NE↓ | SR↑ | SPL↑ | TL | NE↓ | SR↑ | SPL↑ | TL | NE↓ | SR↑ | SPL↑ |
| Seq2Seq [14] | 11.33 | 6.01 | 39 | - | 8.39 | 7.81 | 22 | - | 8.13 | 7.85 | 20 | 18 |
| SF [85] | - | 3.36 | 66 | - | - | 6.62 | 35 | - | 14.82 | 6.62 | 35 | 28 |
| PRESS [189] | 10.57 | 4.39 | 58 | 55 | 10.36 | 5.28 | 49 | 45 | 10.77 | 5.49 | 49 | 45 |
| EnvDrop [296] | 11.00 | 3.99 | 62 | 59 | 10.70 | 5.22 | 52 | 48 | 11.66 | 5.23 | 51 | 47 |
| AuxRN [363] | - | 3.33 | 70 | 67 | - | 5.28 | 55 | 50 | - | 5.15 | 55 | 51 |
| PREVALENT [107] | 10.32 | 3.67 | 69 | 65 | 10.19 | 4.71 | 58 | 53 | 10.51 | 5.30 | 54 | 51 |
| RelGraph [118] | 10.13 | 3.47 | 67 | 65 | 9.99 | 4.73 | 57 | 53 | 10.29 | 4.75 | 55 | 52 |
| RecBERT [122] | 11.13 | 2.90 | 72 | 68 | 12.01 | 3.93 | 63 | 57 | 12.35 | 4.09 | 63 | 57 |
| HAMT (Ours) | 11.15 | **2.51** | **76** | **72** | 11.46 | **2.29** | **66** | **61** | 12.27 | **3.93** | **65** | **60** |

**Computation Efficiency.** To assess the influence of history encoding on the inference time, we compare HAMT with RecBERT [122]. The HAMT and RecBERT use the same number of layers in the language transformer and cross-modal transformer. The main difference of two models is in the history encoding and the attended length of history for action prediction. We run each model on the R2R val unseen split (2349 instructions) and report inference times averaged over two runs using a single Tesla P100 GPU. For our method we compare variants with and without Text-to-Vision Attention, denoted here as HAMT and HAMT noT2V respectively. We can see that HAMT and its noT2V variant are only 1.5x and 1.1x slower compared to RecBERT, suggesting that attending to the whole history does not increase the inference time significantly. Moreover, while HAMT noT2V is only 10% slower compared to [122], it still outperforms [122] in SR and SPL on val unseen split.

### 4.4.3 Comparison to state of the art

**VLN with fine-grained instructions: R2R and RxR.** Table 4.7 compares HAMT with previous VLN methods on the R2R benchmark. Our model outperforms state-of-the-art results of RecBERT [122] by relative 5.9% and 7.0% improvements in SPL on val seen and unseen splits respectively. We achieve state-of-the-art performance under the single-run setting on the unseen testing split of the leaderboard[3]. It demonstrates the effectiveness and generalization of our model.

[3]We report the published results on the testing unseen split as shown in https://eval.ai/web/challenges/challenge-page/97/leaderboard/270 (25/10/2021).

| Methods | NE↓ | SR↑ | CLS↑ | nDTW↑ | SDTW↑ |
|---|---|---|---|---|---|
| SF [85] | 8.47 | 24 | 30 | - | - |
| RCM [321] | - | 29 | 35 | 30 | 13 |
| PTA [175] | 8.25 | 24 | 37 | 32 | 10 |
| EGP [67] | 8.0 | 30.2 | 44.4 | 37.4 | 17.5 |
| RelGraph [118] | 7.43 | 36 | 41 | 47 | 34 |
| RecBERT† [122] | 6.67 | 43.6 | 51.4 | 45.1 | 29.9 |
| HAMT (Ours) | **6.09** | **44.6** | **57.7** | **50.3** | **31.8** |

**Table 4.8:** Comparison on R4R val unseen split.

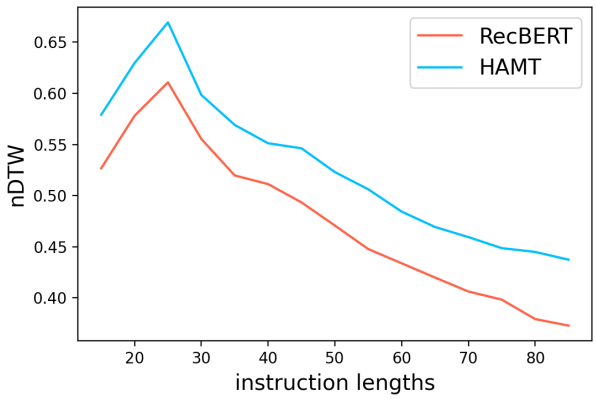



**Figure 4.4:** nDTW with respect to instruction length on R4R val unseen split.

**Long-horizon VLN: R4R and R2R-Back.** Table 4.8 shows navigation results on R4R dataset. As R4R contains longer instructions and trajectories compared to R2R, we use the encoder-decoder variant of HAMT for better efficiency. Our method outperforms previous approaches in all metrics and shows particularly large improvements for the path fidelity related metrics. Compared to RecBERT, HAMT achives 8.2% and 9.5% relative improvement in CLS and nDTW respectively. The large improvements on these path fidelity related metrics indicate that HAMT is better to follow the designated path of the fine-grained instruction. Figure 4.4 evaluates the performance of HAMT and RecBERT with respect to instruction length measured by words. Though the nDTW decreases for longer instructions, the relative improvement of HAMT increases with the instruction length.

The navigation performance on R2R-Back dataset is presented in Table 4.9. We compare with two state-of-the-art recurrent models EnvDrop [296] and RecBERT [122] based on LSTM and transformer respectively (both models are trained on R2R-Back for fair comparison). The improvements are more significant on this task as it requires the agent to remember the way it came to the target to successfully return back. The recurrent state is insufficient to capture such history and leads to inferior performance

compared to the HAMT model.

**Table 4.9:** Comparison of methods on the R2R-Back dataset.

| Methods | Val Seen | | | | | Val Unseen | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| | TL | SR↑ | SPL↑ | nDTW↑ | SDTW↑ | TL | SR↑ | SPL↑ | nDTW↑ | SDTW↑ |
| EnvDrop† [296] | 23.83 | 44.1 | 42.0 | 61.3 | 39.4 | 24.57 | 32.4 | 30.2 | 51.1 | 28.0 |
| RecBERT† [122] | 22.33 | 51.4 | 48.4 | 67.3 | 45.7 | 23.35 | 41.1 | 37.7 | 58.2 | 35.6 |
| HAMT (Ours) | 22.76 | **64.8** | **61.8** | **73.7** | **58.9** | 23.78 | **57.2** | **53.1** | **65.1** | **49.5** |

**Vision-and-Dialog Navigation: CVDN.** The CVDN dataset contains dialogs as instructions and use Goal Progress (GP) in meters as the primary evaluation metric. GP measures the difference between completed distance and left distance to the goal, so the higher the better. There are two types of demonstrations in the dataset. One is shortest-path trajectory and the other is player's navigation trajectory. We mix the two types of demonstrations as supervision in training which has shown to be the most effective in previous works [107, 277, 323]. As navigation paths in CVDN dataset are much longer than R2R dataset, we adopt the encoder-decoder variant of HAMT. As shown in Table 4.10, HAMT outperforms existing recurrent approaches on both seen and unseen environments, and achieves the top position in the leaderboard[4]. It demonstrates that our HAMT model is generalizable to different types of instructions in new VLN tasks.

**VLN with high-level instructions: R2R-Last and REVERIE.** Table 4.11 shows results on the R2R-Last dataset that specifies the goal location and contains no step-by-step instructions. The HAMT model with the hierarchical history encoding is able to better accumulate the knowledge of the environment and achieves 9.8% and 10.5% relative gains on SPL metric on seen and unseen splits respectively compared to RecBERT [122]. The REVERIE dataset also contains high-level instructions but requires object grounding at the target location besides navigation. Our HAMT achieves

[4] https://eval.ai/web/challenges/challenge-page/463/leaderboard/1292 (25/10/2021)

† We run the original implementation of methods released by the authors.

**Table 4.10:** Navigation performance on CVDN dataset.

| | Val Seen | Val Unseen | Test Unseen |
|---|---|---|---|
| PREVALENT [107] | - | 3.15 | 2.44 |
| VISITRON [277] | 5.11 | 3.25 | 3.11 |
| MT-RCM+EnvAg [323] | 5.07 | 4.65 | 3.91 |
| HAMT (Ours) | **6.91** | **5.13** | **5.58** |

**Table 4.11:** Comparison on the R2R-Last dataset.

| Methods | Val Seen | | Val Unseen | |
|---|---|---|---|---|
| | SR↑ | SPL↑ | SR↑ | SPL↑ |
| EnvDrop† [296] | 42.8 | 38.4 | 34.3 | 28.3 |
| RecBERT† [122] | 50.2 | 45.8 | 41.6 | 37.3 |
| HAMT (Ours) | **53.3** | **50.3** | **45.2** | **41.2** |

**Table 4.12:** Navigation performance on RxR test split.

| | PL | SR↑ | SPL↑ | nDTW↑ | SDTW↑ |
|---|---|---|---|---|---|
| Multilingual Baseline [169] | 16.88 | 20.98 | 18.55 | 41.05 | 20.59 |
| Monolingual Baseline [169] | 17.05 | 25.40 | 22.59 | 41.05 | 20.59 |
| CLIP-ViL | 15.43 | 38.34 | 35.17 | 51.10 | 32.42 |
| CLEAR-CLIP | 16.46 | 40.29 | 36.57 | 53.69 | 34.86 |
| Multilingual HAMT | 19.77 | **53.12** | **46.62** | **59.94** | **45.19** |
| Human | 20.78 | 93.92 | 74.13 | 79.48 | 76.90 |

SPL 30.20 and 26.67 on val unseen and test splits respectively, outperforming the state of the art navigation performance [122] by 5.3% and 2.7%.

### RxR dataset

As shown in Table 4.2, RxR dataset contains much more instructions than R2R dataset. Therefore, we directly use RxR in training proxy tasks rather than R2R with augmented data. As there are three different languages in RxR, we take advantage of pretrained multilingual BERT [60] to initialize the unimodal language encoder, so we are able to deal with multilingual instructions using the same HAMT model. We employ the encoder-decoder variant of HAMT for computational efficiency. For fair comparison with other approaches in RxR testing leaderboard[5] which adopt pretrained CLIP [255] features, we use the same visual features without end-to-end optimization. Table 4.12 presents navigation performances on RxR test split. Our multilingual HAMT model achieves 12.83% and 6.25% gains on SR and nDTW respectively than the second place. Nevertheless, there is still a large gap compared to the human performance. We further present results on val seen and val unseen splits in Table 4.13.

### REVERIE dataset

The remote object localization task in REVERIE dataset requires both navigation and object grounding. To support the two subtasks in HAMT, we concatenate object features with original view image features for each viewpoint, and add an object grounding head

[5] https://ai.google.com/research/rxr/competition?active_tab=leaderboard (25/10/2021).

**Table 4.13:** Navigation performances on RxR val seen and val unseen splits.

| | Val Seen | | | | Val Unseen | | | |
|---|---|---|---|---|---|---|---|---|
| | SR↑ | SPL↑ | nDTW↑ | SDTW↑ | SR↑ | SPL↑ | nDTW↑ | SDTW↑ |
| Multilingual Baseline [169] | 25.2 | - | 42.2 | 20.7 | 22.8 | - | 38.9 | 18.2 |
| Monolingual Baseline [169] | 28.8 | - | 46.8 | 23.8 | 28.5 | - | 44.5 | 23.1 |
| Multilingual HAMT | **59.4** | **58.9** | **65.3** | **50.9** | **56.5** | **56.0** | **63.1** | **48.3** |

**Table 4.14:** Navigation and object grounding performances on REVERIE val unseen and test splits.

| Methods | Validation Unseen | | | | | | Test Unseen | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | Navigation | | | | Grounding | | Navigation | | | | Grounding | |
| | TL | SR↑ | OSR↑ | SPL↑ | RGS↑ | RGSPL↑ | TL | SR↑ | OSR↑ | SPL↑ | RGS↑ | RGSPL↑ |
| Seq2Seq [14] | 11.07 | 4.20 | 8.07 | 2.84 | 2.16 | 1.63 | 10.89 | 3.99 | 6.88 | 3.09 | 2.00 | 1.58 |
| RCM [321] | 11.98 | 9.29 | 14.23 | 6.97 | 4.89 | 3.89 | 10.60 | 7.84 | 11.68 | 6.67 | 3.67 | 3.14 |
| SMNA [212] | 9.07 | 8.15 | 11.28 | 6.44 | 4.54 | 3.61 | 9.23 | 5.80 | 8.39 | 4.53 | 3.10 | 2.39 |
| FAST-MATTN [251] | 45.28 | 14.40 | 28.20 | 7.19 | 7.84 | 4.67 | 39.05 | 19.88 | 30.63 | 11.6 | 11.28 | 6.08 |
| SIA [196] | 41.53 | 31.53 | **44.67** | 16.28 | **22.41** | 11.56 | 48.61 | **30.80** | **44.56** | 14.85 | **19.02** | 9.20 |
| RecBERT [122] | 16.78 | 30.67 | 35.02 | 24.90 | 18.77 | 15.27 | 15.86 | 29.61 | 32.91 | 23.99 | 16.50 | **13.51** |
| HAMT | 14.08 | **32.95** | 36.84 | **30.20** | 18.92 | **17.28** | 13.62 | 30.40 | 33.41 | **26.67** | 14.88 | 13.08 |

to predict the target object given output embeddings of all object tokens. We fine-tune HAMT that is end-to-end pretrained on R2R dataset, and use the optimized ViT to extract object features given groundtruth object bounding boxes in REVERIE dataset. As shown in Table 4.14, HAMT achieves better navigation performance (SR and SPL), but the object grounding performance (RGS and RGSPL) on test split is worse than state of the art. Since HAMT can more effectively encode observed visual scenes and actions in the history sequence, it is able to better understand house environments and navigate to target viewpoints more efficiently as shown in the much higher SPL score. However, as we use ViT optimized on R2R dataset to extract object features, the object representation might not be as generalizable as object features used in previous works which are pretrained on large-scale object detection datasets.

### 4.4.4 Additional ablations

**History in training with proxy tasks**

We show that the history input plays a critical role for training with proxy tasks. We compare HAMT with history input and PREVALENT [107] without history. For fair comparison, we re-implement PREVALENT which only takes instruction $\mathscr{W}$ and single-step observation $\mathscr{O}_t$ as input and the other architectures are set the same as HAMT. We train PREVALENT with all proxy tasks except the ITM task because there is no trajectory input in PREVALENT for instruction-trajectory matching. ViT features pretrained on ImageNet are used in this experiment.

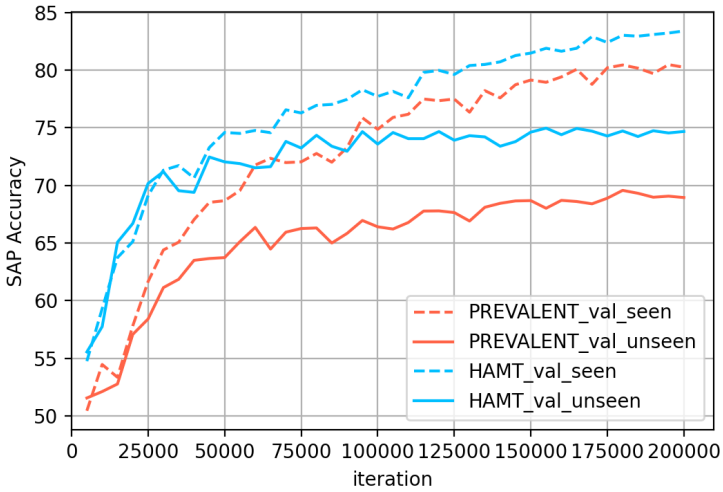



**Figure 4.5:** SAP accuracy of PREVALENT (w/o history) and HAMT (w/ history) on R2R dataset.

In Figure 4.5, we present the single-step action prediction (SAP) accuracy of HAMT and PREVALENT during the training. The SAP accuracies on val seen split are similar for the two models, however, PREVALENT performs much worse on the val unseen split than HAMT. Due to the capacity of large-scale transformer, PREVALENT is likely to memorize the map structure of seen houses, and thus achieves comparable performance to HAMT. However, such knowledge cannot be transferred to unseen houses because the structure and visual observations are distinct for seen and unseen houses. Feeding history as inputs avoids the model simply cramming the structure of seen houses, and enables it to align the history with an instruction to predict actions for better generalization. After fine-tuning the two models on R2R dataset, we obtain SPL 57.5 on val unseen split for HAMT, while 52.7 for PREVALENT without history input. As the same proxy tasks are used in training, the large gains of our HAMT model contribute to the history encoding. Therefore, **the proposed history encoding can largely improve the navigation performance on top of training proxy tasks.**

### Visual features in training with proxy tasks

Table 4.15 provides an additional experiment in the third row compared to Table 4.4a. It demonstrates that ViT features outperform ResNet152 features with and without training proxy tasks. Comparing the last two rows in Table 4.15, end-to-end feature optimization improves SPL by 2.1% on val unseen split but decreases SPL by 0.8% on val seen split. Note that we follow previous VLN works [122, 296] to select the best model based on val unseen and use the same model for val seen split. We observe that the performance on val seen split can be improved with longer training time. After optimizing visual representations, HAMT converges faster on val unseen split and achieves the best performance at earlier iterations. Therefore, the performance on val seen split is slightly worse than no end-to-end optimization. If training longer, the

**Table 4.15:** Comparison of features (same notations as Table 4.4a).

| Features | PT | e2e | Val Seen SR | Val Seen SPL | Val Unseen SR | Val Unseen SPL |
|---|---|---|---|---|---|---|
| Resnet | × | × | 65.5 | 61.3 | 54.4 | 48.7 |
| 152 | ✓ | × | 69.3 | 64.8 | 63.5 | 57.5 |
| ViT | × | × | 68.8 | 66.1 | 56.3 | 52.5 |
| | ✓ | × | **75.7** | **72.5** | 64.4 | 58.8 |
| | ✓ | ✓ | 75.0 | 71.7 | **65.7** | **60.9** |

**Table 4.16:** Comparison of different proxy tasks in end-to-end optimization.

| SAP(R) | SPREL | Val Seen SR | Val Seen SPL | Val Unseen SR | Val Unseen SPL |
|---|---|---|---|---|---|
| × | × | 70.1 | 65.9 | 63.3 | 57.7 |
| ✓ | × | 72.5 | 69.2 | 64.5 | 59.4 |
| ✓ | ✓ | **75.0** | **71.7** | **65.7** | **60.9** |

performance with optimized ViT features on val seen split can be higher.

#### Different proxy tasks in end-to-end training

In Table 4.4b, we fix ViT features to ablate contributions of different proxy tasks in training. We further present the ablation results in a fully end-to-end training setup in Table 4.16, where different proxy tasks are used to train HAMT including the ViT features. The results show the same trend as Table 4.4b, where our proposed two new proxy tasks (SAP/R and SPREL) are beneficial. Moreover, we can see that the end-to-end ViT features are superior to fixed ViT features in Table 4.4b on val unseen split for all the three proxy task combinations.

#### Two-stage end-to-end (e2e) training strategy

We compare our two-stage e2e training strategy with a single-stage e2e training of HAMT. However, single-stage e2e training achieves inferior performance to the two-stage training or even no e2e training. When trained for 25k iterations and evaluated on the val unseen split, the single-stage e2e training of HAMT results in SPL 53.5 while no e2e training achieves SPL 56.5. We hypothesize that the single-stage e2e training is less effective for VLN given (a) the limited training data available for the VLN task and (b) the higher complexity of VLN compared to common vision and language tasks.

#### History encoding in long-horizon VLN task

We compare different history encoding approaches on the R2R-Back dataset to show that the history information is more beneficial for the long-horizon VLN task. Table 4.17

presents navigation results. All the models are initialized from weights after training with proxy tasks. In order to successfully return back, the agent should remember the way it comes to the targets. The recurrent state is insufficient to capture all the information and achieves the worst navigation performance. Encoding agent's oriented view at each step in temporal-only model improves over the recurrent approach. However, as the oriented view of the agent in backward trajectory is different from the view in forward trajectory, temporal-only model does not take advantage of the full memory in previous exploration and performs inferior to our hierarchical history encoding model. It demonstrates the effectiveness of our proposed method in long-horizon VLN task that requires long-term dependency. We also show that using the end-to-end trained ViT features further benefits the navigation performance.

**Table 4.17:** Navigation results for R2R-Back dataset.

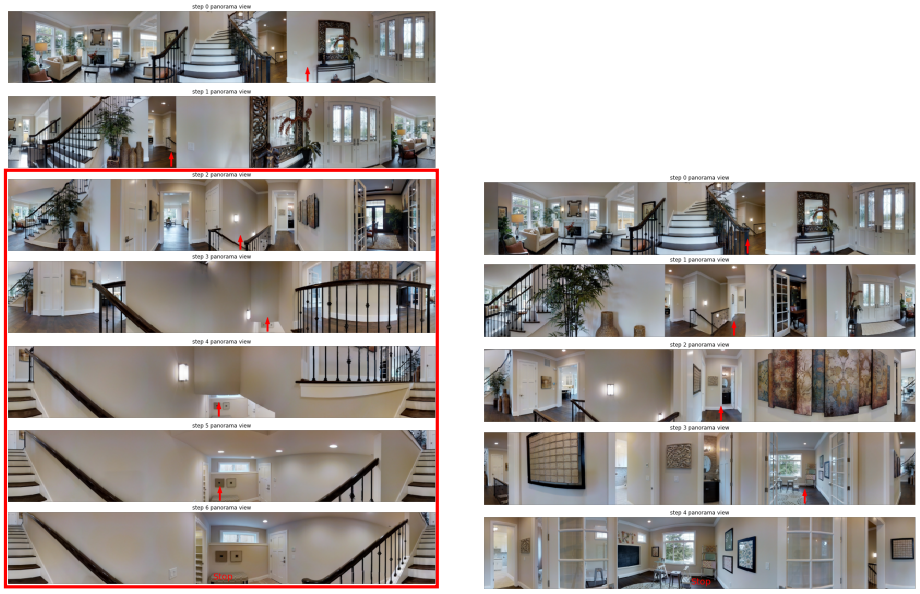
| History Encoding | e2e | Val Seen | | | | | Val Unseen | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|
| | | TL | SR↑ | SPL↑ | nDTW↑ | SDTW↑ | TL | SR↑ | SPL↑ | nDTW↑ | SDTW↑ |
| Recurrent | × | 22.33 | 51.4 | 48.4 | 67.3 | 45.7 | 23.35 | 41.1 | 37.7 | 58.2 | 35.6 |
| Temporal-only | × | 22.70 | 51.6 | 49.6 | 67.8 | 46.7 | 22.93 | 45.1 | 42.9 | 62.7 | 40.2 |
| Hierarchical | × | 23.52 | **66.8** | **63.5** | **73.8** | **60.4** | 24.58 | 56.5 | 51.7 | 63.6 | 48.4 |
| Hierarchical | ✓ | **22.76** | 64.8 | 61.8 | 73.7 | 58.9 | **23.78** | **57.2** | **53.1** | **65.1** | **49.5** |

### Structure variants in fine-tuning

Our model reuses the $f_{\text{SAP}}(o'_i \odot x'_{\text{cls}})$ in training proxy tasks to sequentially predict action in fine-tuning. In Table 4.18, we compare using different input tokens for the action prediction in $f_{\text{SAP}}$, including different combinations of the observation token $o'_i$, global history token $h'_{\text{cls}}$ and special text token $x'_{\text{cls}}$. We can see that the performance varies little on the val unseen split, which indicates that the cross-modal transformer in our model is able to effectively fuse different modalities so that the performance is influenced little by tokens used in prediction.

**Table 4.18:** Comparison of using different tokens in $f_{\text{SAP}}$ in fine-tuning.

| Action Prediction Token | | | Val Seen | | Val Unseen | |
|---|---|---|---|---|---|---|
| obs | txt | hist | SR↑ | SPL↑ | SR↑ | SPL↑ |
| ✓ | × | × | 76.1 | 72.8 | **66.0** | 60.3 |
| ✓ | ✓ | × | 75.0 | 71.7 | 65.7 | **60.9** |
| ✓ | × | ✓ | **78.0** | **75.9** | 65.5 | 60.2 |
| ✓ | ✓ | ✓ | 76.3 | 73.4 | 65.5 | 60.9 |



**(a)** Predicted trajectory by RecBERT [122] (failed). **(b)** Predicted trajectory by HAMT (succeed).

**Figure 4.6:** Examples in R2R val unseen split. Navigation steps inside red box are incorrect. The instruction is "Walk to the right of the stairs. Continue past and to the right of the stairs that go down. Turn right and stop in the doorway of the double glass doors." (id: 697_0). The RecBERT misunderstands the instruction and goes down the stairs instead of turning right. Our HAMT is better to understand the instruction and spatial relation related to the stairs to turn to the right of the stairs.

### 4.4.5 Qualitative results

Figures 4.6-4.9 illustrate trajectories obtained by our HAMT model and compare them to results of the state-of-the-art RecBERT [122] model. We can see that HAMT enables to better interpret instructions (Figure 4.6), recognize the scene (Figure 4.7), follow the correct direction (Figure 4.8), and align the current observation with the instruction (Figure 4.9). We also provide some failure cases in Figures 4.10-4.11, where the HAMT model still needs improvements on scene and object recognition.

## 4.5 Conclusion

This chapter presents the first end-to-end transformer for vision-and-language navigation, denoted as History Aware Multimodal Transformer (HAMT). Our method efficiently encodes long-horizon history and combines it with instructions and observations to derive multimodal action prediction. The HAMT is first trained with proxy tasks in an end-to-end manner, and is then fine-tuned with RL to improve the navigation policy. We achieve state-of-the-art navigation performance on a diverse range of challenging


**(a)** Predicted trajectory by RecBERT [122] (failed). **(b)** Predicted trajectory by HAMT (succeed).

**Figure 4.7:** Examples in R2R val unseen split. Navigation steps inside red box are incorrect. The instruction is "Walk into the kitchen area. Walk by the sink and oven. Walk straight into the hallway. Turn right into the little room. Turn left and walk into the bedroom. Stop by the corner of the bed." (id: 155_0). The RecBERT fails to recognize the kitchen area and navigates back and forth in wrong locations. Our HAMT correctly recognizes the kitchen and follows the instruction.

**(a)** Predicted trajectory by RecBERT [122] (failed). **(b)** Predicted trajectory by HAMT (succeed).

**Figure 4.8:** Examples in R2R val unseen split. Navigation steps inside red box are incorrect. The instruction is "Walk straight until you get to a room that has a black table on the left with flowers on it. Wait there." (id: 4182_2). The RecBERT takes the wrong direction at the first step, while our HAMT follows the instruction and successfully stops.

**(a)** Predicted trajectory by RecBERT [122] (failed). **(b)** Predicted trajectory by HAMT (succeed).

**Figure 4.9:** Examples in R2R val unseen split. Navigation steps inside red box are incorrect. The instruction is "Walk out of the bathroom and turn right. Turn left and walk down the hallway. Turn right and stop by the end table." (id: 5153_0). The RecBERT correctly performs the first two turns but fails to track the third turn right action and stops incorrectly. Our HAMT is better to align the current state with the instruction to correctly perform the third turn right action.

**(a)** groundtruth trajectory. **(b)** Predicted trajectory by HAMT (failed).

**Figure 4.10:** Failure cases in R2R val unseen split. The instruction is "Go stand underneath the stairs, next to the liquor shelf. " (id: 36968_2). Though HAMT correctly goes towards the direction, it fails to recognize the liquor shelf and results in exploring further the room until reaching the maximum number of navigation steps.

VLN tasks, demonstrating improved accuracy and generalization of our approach compared to the dominant recurrent methods. Future work could extend our history-aware transformer to VLN with continuous actions [164] and could benefit from pretraining on larger navigation datasets.


**(a)** Groundtruth trajectory.

**(b)** Predicted trajectory by HAMT (failed).

**Figure 4.11:** Failure cases in R2R val unseen split. The instruction is "With the low stone or concrete barrier behind you, walk parallel to the board covering the floor and turn left before reaching the end. Move forward to leave the wooden flooring and when on the stone flooring, turn right and stand in front of the doors leading out of the room." (id: 5873_1). As the scene is unusual, HAMT fails to locate itself in the correct direction at the first step.

# Chapter 5

# Instruction-driven history-aware policies for robotic manipulations

In human environments, robots are expected to accomplish a variety of manipulation tasks given simple natural language instructions. In Chapters 3 and 4, the action space was simplified, as we predicted the next node location of the robot, and then use a lower-level controller to move the robot from node to node. This simplification does not hold anymore when tackling manipulation-based tasks, as it requires to control fine-grained motors. Furthermore, robotic manipulation requires as well long-term memory as well as a generalization to previously unseen tasks and environments.

To address these challenges, we propose a unified transformer-based approach that takes into account multiple inputs. In particular, our transformer architecture integrates (i) natural language instructions and (ii) multi-view scene observations while (iii) keeping track of the full history of observations and actions. Such an approach enables learning dependencies between history and instructions and improves manipulation precision using multiple views.

We evaluate our method on the challenging RLBench benchmark and on a real-world robot. Notably, our approach scales to 74 diverse RLBench tasks and outperforms the state-of-the-art. We also address instruction-conditioned tasks and demonstrate excellent generalization to previously unseen variations.

## 5.1 Introduction

People can naturally follow language instructions and manipulate objects to accomplish a wide range of tasks from cooking to assembly and repair. It is also easy to generalize to new tasks by building upon skills learned from previously seen tasks. Hence, one of the long-term goals for robotics is to create generic instruction-following agents that can generalize to multiple tasks and environments.

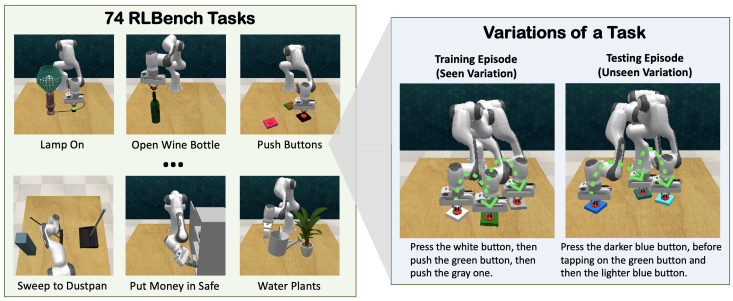



**Figure 5.1: Left:** Hiveformer can adapt to perform 74 tasks from RLBench [143] given language instructions. **Right:** Multiple variations of the *push buttons* task.

Thanks to significant advances in learning generic representations for vision and language [69, 111, 204, 254], recent work has made great progress towards this goal [211, 218, 219, 275, 145]. For example, CLIPort [275] exploits CLIP models [254] to encode single-step visual observations and language instructions and to learn a single policy for 10 simulated tasks. BC-Z [145] uses a pre-trained sentence encoder [339] to generalize to multiple manipulation tasks. However, several challenges remain underexplored. One important challenge is that sequential tasks require to track object states that may be hidden from current observations, or to remember previously executed actions. This behaviour is hard to model with recent methods that mainly rely on current observations [275, 145].

Another challenge concerns manipulation tasks that require precise control of the robot end-effector to reach target locations. Such tasks can be difficult to solve with single-view approaches [142], especially in situations with visual occlusions and objects of different sizes, *e.g.* see *put money in safe* Figure 5.1 (left). While several recent approaches combine views from multiple cameras by converting multi-view images into a unified 2D/3D space [350, 144] or through a late fusion of multi-view predictions [200], learning representations for multiple camera views is an open research problem. Furthermore, cross-modal alignment between vision, action, and text is challenging, in particular when training and test tasks differ in terms of objects and the order of actions, see Figure 5.1 (right). Most of existing methods [275, 145, 270, 230] condense instructions into a global vector to condition policies [244] and are prone to lose fine-grained information about different objects.

To address the above challenges, we introduce *Hiveformer* - a **H**istory-aware **i**nstruction-conditioned multi-**vie**w trans**former**. It converts instructions into language tokens given a pre-trained language encoder [254], and combines visual tokens for both past and current visual observations and proprioception. These tokens are concatenated and fed into a multimodal transformer which jointly models dependencies between the current and past observations, spatial relations among views from multiple cameras, as well as fine-grained cross-modal alignment between vision and instruction. Based on the output representations from our multimodal transformer, we predict 7-DoF actions, *i.e.* , position, rotation and state of the gripper, with a UNet [260] decoder.

We carry out extensive experiments on RLBench [143] in three setups: single-task

learning, multi-task learning, and multi-variation generalization[1]. Our Hiveformer significantly outperforms state-of-the-art models for all three settings, demonstrating the effectiveness of encoding instruction, history and views from multiple cameras with the proposed transformer. Moreover, we evaluate our model on 74 tasks of RLBench, which goes beyond the 10 tasks used by Liu *et al*. [200]. We manually group all the tasks into 9 categories according to their main challenges and analyze results per category for a better understanding. Hiveformer not only excels in the multiple task setting with seen instructions in training, but also enables generalization to new instructions that represent different variations of the task, even with human-written language instructions. Finally, we evaluate our model deployed on a real robot and show excellent performance. Interestingly, pretraining the model in the RLBench simulator results in significant performance gains when only a small number of real robot demonstration is available.

To summarize, our contributions are three-fold:

- We introduce a new model Hiveformer to solve various challenges in robotics tasks. It jointly models instruction, multiple views, and history via a multimodal transformer for action prediction in robotic manipulation.
- We perform extensive ablations of our model on RLBench with 74 tasks grouped into 9 distinct categories. The history improves long-term tasks and the multi-view setting is most helpful for tasks requiring high precision or in the presence of visual occlusions.
- We demonstrate that Hiveformer outperforms the state of the art in three RLBench setups, namely single-task, multi-task and multi-variation. A single Hiveformer trained with synthetic instructions is able to solve multiple tasks and task variations, can generalize to unseen human-written instructions and shows excellent performance on a real robot after finetuning.

Our code, pre-trained models and additional results are available from the project webpage [99].

## 5.2 Related work

**Vision-based robotic manipulation.** While earlier methods for solving robotics tasks such as visual servoing [115, 47] were designed manually, the need to cope with large variations of objects and environments led to the emergence of learning-based neural approaches [288, 178, 25, 83]. Deep neural networks [152, 172] have achieved impressive results in manipulation for single tasks [4], and recently led to more challenging setups such as multi-task learning [41, 7, 68, 281]. Different multi-task approaches are explored by discovering which tasks should be trained together [200, 284], determining shared features across tasks [305, 49], meta-learning [81, 346, 347], goal-conditioned learning [82, 149], or inverse reinforcement learning [48]. These approaches can be generally split in two categories according to the training algorithm: reinforcement learning (RL) methods [292, 104, 245, 240] which learn policies from rewards provided by environments and behavioral cloning methods [308, 116, 210] that learn from demon-

[1]We follow definitions in RLBench [143] for tasks and variations. A task can be composed of multiple variations that share the same skills but differ in objects, attributes or order as shown in Figure 5.1 (right).

strations using supervised learning. Demonstrations can be obtained from humans [246], robots [288, 125] or play interactions [210]. The emergence of robotic simulators, such as Gym [36], manipulaTHOR [74], dm_control [309], Sapien [334], CausalWorld [2], and RLBench [143], also greatly accelerated the development of manipulation methods. In this chapter, we use behavioral cloning to train policies given scripted demonstrations from RLBench [143] which covers many challenging manipulation tasks.

**Instruction-driven vision-based robotic manipulation** has received growing attention for manipulations in 2D planar [223, 285] or recent 3D environments [219, 114, 299], and has been transferred to the real world [275, 145]. As grounding the language in visual scenes is important, existing works have focused on challenges in object grounding, such as localizing objects based on referring expressions [243, 233, 94] and grounding spatial relationships [218, 300, 201]. Since language describes high-level actions, several works [299, 131, 89] consider a hierarchical approach to decompose a task into sub-goals. Because natural language is rich and diverse, while training resources are limited, further works learn from collected offline data with instructions [145, 230] or leverage pre-trained vision-language models [204, 254] for action prediction [275, 156]. To further improve the precision of manipulation skills, Mees *et al*. [219] align instructions with multiple cameras by fusing input images with known camera parameters. Most of these works [211, 219, 275, 145] are stateless, since they only employ current observations to predict next actions. Instead, this chapter proposes to jointly model language instructions, history, and multi-view observations.

**Transformers** [313] have led to significant gains in natural language processing [69], computer vision [72] and related fields [204, 254, 294]. They have also been used in the context of supervised reinforcement learning, such as Decision Transformer [54] or Trajectory Transformer [249]. Recent works in Vision-and-Language Navigation (VLN) [55, 242, 100] further demonstrate that the transformer allows to better leverage previous observations to improve multi-modal action prediction. Transformers are also used to build a multi-modal, multi-task, multi-embodiment generalist agent, GATO [257]. Inspired by the success of transformers, we explore the transformer architecture for instruction-driven and history-aware robotic manipulation.

## 5.3 Problem Definition

Our goal is to train a policy $\pi\left(a_{t+1}|\{x_l\}_{l=1}^n, \{o_i\}_{i=1}^t, \{a_i\}_{i=1}^t\right)$ for robotic manipulation conditioned on a natural language instruction $\{x_l\}_{l=1}^n$, visual observations $\{o_i\}_{i=1}^t$, and previous actions $\{a_i\}_{i=1}^t$ where $n$ is the number of words in the instruction and $t$ is the current step. For robotic control, we use macro steps [142] – key turning points in action trajectories where the gripper changes its state (open/close) or velocities of joints are close to zero. We employ an inverse-kinematics based controller to find a trajectory between macro-steps. In this way, the sequence length of an episode is significantly reduced from hundreds of small steps to typically less than 10 macro steps.

The **observation** $o_t$ at step $t$ consists of RGB images $I_t$ and point clouds $P_t$ aligned with the RGB images. $I_t$ is composed of $\{I_t^k\}_{k=1}^K$ RGB images from $K$ cameras, with each $I_t^k$ being of size $H \times W \times 3$ (height, width, 3 channels). Following [200], we use

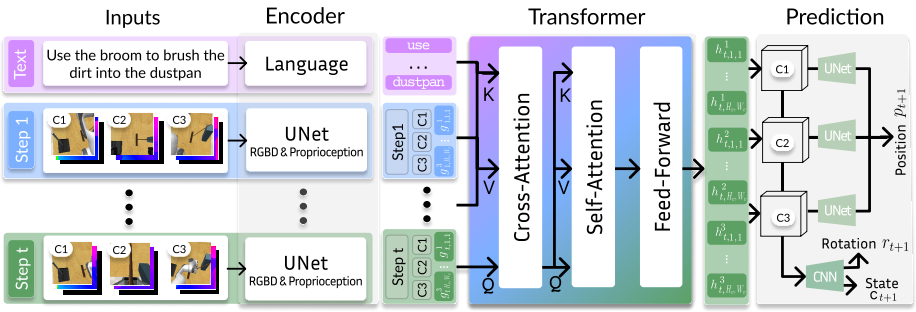



**Figure 5.2:** Hiveformer jointly models instructions, views from multiple cameras, and past actions and observations with a multimodal transformer for robotic manipulation.

$K = 3$ with cameras on the wrist, left shoulder and right shoulder of the agent, and $H = W = 128$. Similarly, $P_t$ represents point clouds $\{P_t^k\}_{k=1}^K$ from $K = 3$ cameras. A point cloud $P_t^k \in \mathbb{R}^{H \times W \times 3}$ is obtained by projecting a single channel depth image $H \times W$ from the $k$-th camera in world coordinates using known camera intrinsics and extrinsics. Each point in $P_t^k$ has thus 3D coordinates and is aligned with a pixel in $I_t^k$.

The **action space** $a_t$ consists of the gripper pose and its state following the standard setup in RLBench [142]. The gripper pose is composed of the Cartesian coordinates $p_t = (x_t, y_t, z_t)$ and its rotation described by a quaternion $q_t = (q_t^0, q_t^1, q_t^2, q_t^3)$ relative to the base frame. The gripper's state $c_t$ is boolean and indicates whether the gripper is open or closed. An object is grasped when it is located in between the gripper's two fingers and the gripper is closing its grasp. The execution of an action is achieved by a motion planner in RLBench.

## 5.4 Our Model: Hiveformer

We propose a unified architecture for robotic tasks called Hiveformer (**H**istory-aware **i**nstruction-conditioned multi-**v**i**e**w trans**former**), see Figure 5.2 for an overview. It consists of three modules: feature encoding, multimodal transformer and action prediction. The feature encoding module (Sec. 5.4.1) generates token embeddings for instructions $\{x_l\}_{l=1}^n$, visual observations $\{o_i\}_{i=1}^t$ and previous actions $\{a_i\}_{i=1}^t$. Then, the multimodal transformer (Sec. 5.4.2) learns relationships between the instruction, current multi-camera observations and history. Finally, the action prediction module (Sec. 5.4.3) utilizes a convolutional network (CNN) to predict the next rotation $q_{t+1}$ and gripper state $c_{t+1}$, and adopts a UNet decoder [260] to predict the next position $p_{t+1}$.

### 5.4.1 Feature Encoding

We encode the instruction, visual observations, and actions as a sequence of tokens.

**Instructions.** We employ a pre-trained language encoder to tokenize and encode the sentence instruction. Specifically, we use the language encoder in the CLIP model [254].

Thanks to its vision-and-language pre-training, it is better at differentiating vision-related semantics such as colors compared to pure language-only pre-trained models like BERT [69], see Table 5.5.

We freeze the pre-trained language encoder and use a linear layer on top of it to obtain embeddings $\hat{x}_l \in \mathbb{R}^d$ for each word token:

$$\hat{x}_l = \text{LN}(W_x \tilde{x}_l) + E_T^x, \tag{5.1}$$

with $\tilde{x}_l$ the $l$-th embedding output by the language encoder, LN layer normalization [24], $W_x$ a projection matrix, and $E_T^x$ a type embedding which differentiates instructions from visual observations.

**Observations and Proprioception.** We encode the RGB image $I_t^k$, point clouds $P_t^k$, and proprioception $A_t^k$ for each camera $k$ separately. $A_t^k \in \{0,1\}^{H \times W}$ is a binary attention map used to encode the position of the gripper $p_t$. It takes value one at the location of the gripper center and zeroes elsewhere. We concatenate $I_t^k$ and $A_t^k$ in the channel dimension and use a UNet encoder to obtain a feature map $\hat{F}_t^k \in \mathbb{R}^{H^v \times W^v \times d_v}$, where $H^v, W^v, d_v$ are the height, width, and the number of channels of the feature map. Next, we concatenate $\hat{F}_t^k$ with point cloud representations in the channel dimension to indicate the spatial location of each patch in the feature map. To match the size of $P_t^k$ and $\hat{F}_t^k$, we apply mean-pooling to $P_t^k$. The final encoded feature map $F_t^k \in \mathbb{R}^{H^v \times W^v \times (d_v+3)}$ is computed as follows:

$$F_t^k = \left[\text{CNN}([I_t^k; A_t^k]); \text{MeanPool}(P_t^k)\right]. \tag{5.2}$$

**UNet encoder for image encoding.** The CNN in Eq 5.2 is composed of 6 convolutional layers. The first two layers use kernels of size 3x3, strides of size 1, and output channels of sizes 8 and 16 respectively with the LeakyReLU activation function. The remaining four layers use 3x3 kernels, strides of size 2, and output channels of size 16 followed by group normalization and LeakyReLU activation function. Therefore, an image of size $H \times W \times 3$ is encoded by a feature map of size $\frac{H}{16} \times \frac{W}{16} \times 16$.

**UNet decoder for position prediction.** The decoder uses a sequence of convolutional and upsampling layers to generate a heatmap on the point clouds. Specifically, the convolutional layer is fed with the output from the previous layer and the residual connection from the corresponding layer in the UNet encoder. Its output channel size is 16, kernel size is 3, and stride size is 1. The upsampling layer uses a scale factor of 2 and bilinear sampling. We stack 4 blocks of the layers to recover the original image size $H \times W$.

**Patches.** We use patches $f_{t,h,w}^k \in F_t^k, h \in [1, H^v], w \in [1, W^v]$ as separate visual tokens. We further encode $f_{t,h,w}^k$ using embeddings of the camera id $E_C^k$, of the step id $E_S^t$, and of the patch location $E_L^{h,w}$ as well as an embedding to indicate the visual nature of the tokens $E_T^v$ as follows:

$$g_{t,h,w}^k = \text{LN}(W_f f_{t,h,w}^k) + E_C^k + E_S^t + E_L^{h,w} + E_T^v. \tag{5.3}$$

The encoded visual tokens of the $k$-th camera at step $t$ are denoted as $G_t^k = \{g_{t,h,w}^k\}_{h=1,w=1}^{H^v,W^v} \in \mathbb{R}^{H^v \times W^v \times d}$. We concatenate the encoded tokens for all cameras as $G_t = (G_t^1, \cdots, G_t^K)$.

### 5.4.2 Multimodal transformer

Given the encoded tokens at the current macro step $t$, the multimodal transformer aims to obtain a contextualized representation for $G_t$ conditioned on the encoded instruction $\{\hat{x}_l\}_{l=1}^{n}$ and history $\{G_i\}_{i=1}^{t-1}$. This enables learning relationships among views from multiple cameras, the current observations and instructions, and between the current observations and history for action prediction.

We use the transformer's attention mechanism [312] to learn such relationships:

$$\text{Attn}(Q,K,V) = \text{Softmax}\left(\frac{W_Q Q (W_K K)^T}{\sqrt{d}}\right) W_V V, \tag{5.4}$$

where $W_Q, W_K, W_V$ are learnable parameters. Unlike previous work [242] that uses self-attention layers to capture all relationships, we employ different attention layers to capture different types of relationships, in order to reinforce the importance of the context. First, we use a cross-attention layer to learn the inter-modal relationships between $G_t$ and its conditioned contexts $C_t$ consisting of tokens in the instruction $\{\hat{x}_l\}_{l=1}^{n}$ and history $\{G_i\}_{i=1}^{t-1}$, which is:

$$\tilde{G}_t = \text{CA}(G_t, C_t) = \text{Attn}\left(G_t, C_t, C_t\right). \tag{5.5}$$

Then we learn the intra-modal relationships among patch tokens obtained from the views from multiple cameras through a self-attention layer, *i.e.* $\text{SA}(\tilde{G}_t) = \text{Attn}(\tilde{G}_t, \tilde{G}_t, \tilde{G}_t)$. Finally, a feed-forward network consisting of two linear layers $W_1$ and $W_2$ is applied as follows:

$$\hat{G}_t = \text{LN}\left(W_2\ \text{GeLU}\left(W_1\ \text{SA}(\tilde{G}_t)\right)\right). \tag{5.6}$$

### 5.4.3 Action Prediction

We concatenate the output embeddings of the transformer $\hat{G}_t$ in Eq (5.6) and the original encoded visual representations $\hat{F}_t$ in Sec. 5.4.1 in the channel dimension and reshape the flattened sequence into a feature map $H_t \in \mathbb{R}^{K \times H^v \times W^v \times (d+d_v)}$ to predict the next action $a_{t+1} = [p_{t+1}; q_{t+1}; c_{t+1}]$. As some RLBench tasks require accurate fine-grained positioning, different from the rotation $q_{t+1}$ and gripper state $c_{t+1}$, the position $p_{t+1}$ is predicted through a separate module that uses point clouds $P_t$.

**Rotation and gripper's state.** We transform $H_t$ into $\mathbb{R}^{H^v \times W^v \times K(d+d_v)}$ and feed it into a CNN decoder. We then apply average pooling across spatial dimensions and employ a linear layer to regress a 5-dimension vector $[q_{t+1}; c_{t+1}]$.

**Position.** The prediction of the gripper position $p_{t+1}$ is decomposed into an expected point on point clouds $p^e_{t+1}$ and an offset $p^o_{t+1}$, *i.e.* $p_{t+1} = p^e_{t+1} + p^o_{t+1}$. The offset allows us to predict a virtual point outside the convex hull of the point cloud, *e.g.* when a robotic arm reaches first above the object and then touch the object. For each camera $k$, a CNN with an upsampling layer predicts an attention map $B^k_t \in \mathbb{R}^{H \times W}$ over the point clouds $P^k_t$. Each value $B^k_{t,h,w} \in B^k_t$ corresponds to the probability of reaching the point $P^k_{t,h,w} \in P^k_t$. Therefore, we compute $p^e_{t+1}$ as the expected position over all cameras:

$$p^e_{t+1} = \sum_{k,h,w} \left(B^k_{t,h,w} \cdot P^k_{t,h,w}\right). \tag{5.7}$$

The offset $p^o_{t+1}$ is computed from the instruction and the current step id. Let $E_O \in \mathbb{R}^{N_\tau \times T \times 3}$ be a learnable embedding, where $N_\tau$ is the number of tasks and $T$ is the maximum length of episodes. We predict the task id from the instruction: $\Pr(m) = \text{Softmax}\left(W_m \frac{1}{n}\sum_{l=1}^{n} \tilde{x}_l\right)$, where $\Pr(m) \in [0,1]^{N_\tau}$, and we obtain the offset as: $p^o_{t+1} = \sum_m Pr(m) \cdot E_O(m,t,:)$.

### 5.4.4 Training and Inference

**Losses.** We use behavioral cloning to train the models. In RLBench, we generate $\mathsf{D}$, a collection of $N$ successful demonstrations for each task. Each demonstration $\delta \in \mathsf{D}$ is composed of a sequence of (maximum) $T$ macro-steps with observations $\{o^\delta_i\}_{i=1}^T$, actions $\{a^*_i\}_{i=1}^T$, task $m^*$ and instruction $\{x_l\}_{l=1}^n$. We minimize a loss function $\mathscr{L}$ over a batch of demonstrations $\mathsf{B} = \{\delta_j\}_{j=1}^{|B|} \subset \mathsf{D}$. The loss function is the sum of two losses: a mean-square error (MSE) on the gripper's action and a cross-entropy (CE) over the task classification:

$$\mathscr{L} = \frac{1}{|B|} \sum_{\delta \in \mathsf{B}} \left[ \sum_{t \leq T} \text{MSE}\left(a_t, a^*_t\right) + \text{CE}\left(\Pr(m), m^*\right) \right]. \tag{5.8}$$

**Masking current observation.** To ensure that the model uses past information $\{o_i\}_{i=1}^{t-1}, \{a_i\}_{i=1}^{t-1}$ instead of only relying on the current observation $o_t$, we randomly mask the current observation with a probability of 0.1. The masking zeros out randomly selected patch features in the current observation. Therefore, even if the unmasked current observations contain sufficient information, the model still requires to complete the masked observations from the history for action prediction.

## 5.5 Experiments

In this section we present experiments on RLBench [143] tasks to demonstrate the effectiveness of our Hiveformer model in three settings: single-task, multi-task, and multi-variation. In the single-task setup, a separate model is trained and tested for each task with no variations of the task. Multi-task refers to a setting where one model is trained for multiple tasks (but each task has a unique variation). In the multi-variation case we train a single model to solve multiple variations of a single task and test it on new variations of the task unseen during training.

### 5.5.1 Experimental Setup

**Dataset setups.** RLBench [143] is a benchmark of robotic tasks. To compare our method with previous work [200], we use the same 10 tasks with 100 demonstrations for training unless stated otherwise. We further evaluate our model on 74 tasks for which RLBench provides successful demonstrations. Although RLBench[2] currently contains

[2] https://github.com/stepjam/RLBench/tree/master/rlbench/tasks

106 supported tasks, we had difficulties to produce demonstrations for 32 of them due to issues with the scripts and the motion planner. To analyze the performance of our model applied to different types of tasks, we manually group 74 tasks into 9 categories according to their key challenges. The 9 task groups are defined as follows:

- The **Planning** group contains tasks with multiple sub-goals (*e.g.* picking a basket ball and then throwing the ball). The included tasks are: basketball in hoop, put rubbish in bin, meat off grill, meat on grill, change channel, tv on, tower3, push buttons, stack wine.
- The **Tools** group is a special case of planning where a robot must grasp an object to interact with the target object. The included tasks are: slide block to target, reach and drag, take frame off hanger, water plants, hang frame on hanger, scoop with spatula, place hanger on rack, move hanger, sweep to dustpan, take plate off colored dish rack, screw nail.
- The **Long term** group requires more than 10 macro-steps to be completed. The included tasks are: wipe desk, stack blocks, take shoes out of box, slide cabinet open and place cups.
- The **Rotation-invariant** group can be solved without changes in the gripper rotation. The included tasks are: reach target, push button, lamp on, lamp off, push buttons, pick and lift, take lid off saucepan.
- The **Motion planner** group requires precise grasping. As observed in [141] such tasks often fail due to the motion planner. The included tasks are: toilet seat down, close laptop lid, open box, open drawer, close drawer, close box, phone on base, toilet seat up, put books on bookshelf.
- The **Multimodal** group can have multiple possible trajectories to solve a task due to a large affordance area of the target object (*e.g.* the edge of a cup). The included tasks are: pick up cup, turn tap, lift numbered block, beat the buzz, stack cups.
- The **Precision** group involves precise object manipulation. The included tasks are: take usb out of computer, play jenga, insert onto square peg, take umbrella out of umbrella stand, insert usb in computer, straighten rope, pick and lift small, put knife on chopping board, place shape in shape sorter, take toilet roll off stand, put umbrella in umbrella stand, setup checkers.
- The **Screw** group requires screwing an object. The included tasks are: turn oven on, change clock, open window, open wine bottle.
- The **Visual Occlusion** group involves tasks with large objects and thus there are occlusions from certain views. The included tasks are: close microwave, close fridge, close grill, open grill, unplug charger, press switch, take money out safe, open microwave, put money in safe, open door, close door, open fridge, open oven, plug charger in power supply.

**Table 5.1:** Success rate on the single-task setting. Variance is obtained with demonstrations generated with 5 random seeds.

| | Inputs | | | Transformer | | | Training | SR |
|---|---|---|---|---|---|---|---|---|
| | Visual Tokens | Point Clouds | Gripper Position | Multi-View | History | Attn | Mask Obs | |
| R1 | × | × | × | × | × | × | × | $73.2 \pm 3.0$ |
| R2 | Channel | × | × | ✓ | × | Self | × | $74.3 \pm 4.5$ |
| R3 | Channel | ✓ | × | ✓ | × | Self | × | $76.7 \pm 5.7$ |
| R4 | Channel | ✓ | ✓ | ✓ | × | Self | × | $77.3 \pm 5.6$ |
| R5 | Channel | ✓ | ✓ | ✓ | ✓ | Self | × | $81.8 \pm 5.2$ |
| R6 | Channel | ✓ | ✓ | ✓ | ✓ | Self | ✓ | $82.3 \pm 6.3$ |
| R7 | Patch | ✓ | ✓ | ✓ | ✓ | Self | ✓ | $84.5 \pm 6.4$ |
| R8 | Patch | ✓ | ✓ | ✓ | ✓ | Cross | ✓ | $88.3 \pm 5.1$ |

We evaluate models by measuring the per task success rate for 500 unseen episodes.

**Implementation details.** We use the Adam optimizer with a learning rate of $5 \times 10^{-5}$. Each batch consists of 32 demonstrations. Models were trained for 100,000 iterations. We apply data augmentation in training including jitter over RGB images $I_t^k$, and a random crop of $I_t^k$, $P_t^k$, and $A_t^k$ while keeping them aligned. Models are trained on one NVIDIA Tesla V100 SXM2 GPU using a Singularity container with headless rendering. Auto-$\lambda$ [200] uses a UNet network and applies late fusion to predictions from multiple views.

**Collection of human-written instructions.** In addition to synthetic instructions used for training, we collect human-written natural language instructions for testing. We collect 162 human-written instructions and measure the success rate for each instruction on 10 episodes with random object locations. 8 native English speakers participated in the dataset collection, leading to 63 instructions of 51 testing variations for the push buttons task, and 99 instructions for 99 testing variations for the tower task. Human-written instructions are more varied than the synthetic ones. They contain unseen verbs (e.g. "Tap on the green button, then the grey button and end up pressing the pink button"), unseen formulations (e.g. "Press the green, cyan and pink buttons in that order"), longer sentences (e.g. "Press the white button and then you go to green button and press it and finally press the black button") or unseen color references (e.g. "Press the darker blue button, then the gray one and finally the lighter blue button.").

**Motion Planner.** We modified the default motion planner in RLBench, as it sometimes fails to reach a target pose even though there exist successful trajectories in the 3D space. To reduce the impact of the imperfect motion planner, we run the motion planner with different seeds up to 10 times until it finds a trajectory to the target.

### 5.5.2 Ablations

To demonstrate the effectiveness of the proposed model architecture, we ablate the impact of its components in Table 5.1. The model in R1 (row 1) is a UNet architecture

**Table 5.2:** Comparison with state-of-the-art methods on 10 tasks. We report success rate (%).

| | Pick & Lift | Pick-Up Cup | Push Button | Put Knife | Put Money | Reach Target | Slide Block | Stack Wine | Take Money | Take Umbr. | Avg. |
|---|---|---|---|---|---|---|---|---|---|---|---|
| *Single-task learning* | | | | | | | | | | | |
| ARM [142] | 70 | 80 | - | - | - | 100 | - | 70 | - | 70 | - |
| Auto-$\lambda$ [200] | 82 | 72 | 95 | 36 | 31 | 100 | 36 | 23 | 38 | 37 | 55.0 |
| Ours | 93.8 | 82.7 | 99.5 | 69.9 | 96.3 | 100.0 | 95.3 | 82.1 | 82.3 | 90.7 | 88.3 |
| *Multi-task learning* | | | | | | | | | | | |
| Auto-$\lambda$ [200] | 87 | 78 | 95 | 31 | 62 | 100 | 77 | 19 | 64 | 80 | 69.3 |
| Ours (w/o inst) | 84.0 | 13.8 | 97.6 | 41.8 | 54.2 | 98.8 | 36.0 | 68.8 | 74.6 | 72.6 | 64.6 |
| Ours | 88.8 | 93.0 | 100.0 | 75.0 | 58.0 | 100.0 | 79.8 | 70.4 | 79.0 | 89.2 | 83.3 |

**Table 5.3:** Comparison with state-of-the-art on 74 RLBench tasks grouped into 9 categories. We report success rate (%) for the single-task setting. *The performance of Auto-$\lambda$ is obtained by running their code.

| | Planning | Tools | Long Term | Rot. Invar. | Motion Planning | Screw | Multi Modal | Precision | Visual Occlusion | Avg |
|---|---|---|---|---|---|---|---|---|---|---|
| Num. of tasks | 9 | 11 | 4 | 7 | 9 | 4 | 5 | 11 | 14 | 74 |
| Auto-$\lambda$ [200]* | 58.9 | 20.0 | 2.3 | 73.1 | 66.7 | 48.2 | 47.6 | 34.6 | 40.6 | 44.0 |
| Ours (w/o hist) | 78.9 | 46.7 | 10.0 | 84.6 | 73.3 | 72.6 | 60.0 | 63.8 | 57.9 | 60.9 |
| Ours (one view) | 57.7 | 23.2 | 12.3 | 57.8 | 63.2 | 35.6 | 40.7 | 33.7 | 37.1 | 40.1 |
| Ours | 81.6 | 53.0 | 16.9 | 84.2 | 72.7 | 80.9 | 67.1 | 64.7 | 60.2 | 65.4 |

similar to Auto-$\lambda$ [200] except that it is conditioned on instructions rather than task ids. This baseline only uses visual observations at the current step and already achieves promising results with a success rate of 73.2%. On top of R1's architecture, a multimodal transformer with self-attention is added in R2 to improve the modeling of multi-view images. Visual tokens $\{G_t^i\}_{i=1}^K$ are different channels in the feature map instead of spatial patches used in our final model. In R3 and R4, we further add point clouds $P_t$ and gripper position $A_t$ in the feature encoding, which leads to 3% improvement in total. The impact of history, *i.e.* the use of observations from previous steps, ($\{G_j^i\}_{i=1,j=1}^{K,t-1}$) is studied in R5 and R6. The history information brings 4.5% absolute gains and the masking of observations during training further improves the performance by 0.5%. In R7, we replace the tokenization of feature maps from channels $g_{t,c}^k$ to patches $g_{t,h,w}^k$, and obtain another 2.2% gain. This improvement can be attributed to patch tokens that help encode fine-grained spatial information. Finally, we use cross-attention instead of self-attention (Eq. 5.5) to condition on the instruction and history context. It further boosts the performance with a 3.8% gain.

### 5.5.3 Comparison with State of the Art

**Single-task evaluation.** The upper block in Table 5.2 presents results of different models on 10 single tasks in RLBench. We compare our model with ARM [142]

**Table 5.4:** Comparison with LanCon-Learn [279] on 10 tasks.

| | Hist. | Pick & Lift | Pick-Up Cup | Push Button | Put Knife | Put Money | Reach Target | Slide Block | Stack Wine | Take Money | Take Umbr. | Avg. |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| *Single-task learning* | | | | | | | | | | | | |
| LanCon-Learn | - | 20.2 | 25.2 | 96.2 | 57.8 | 91.4 | 99.6 | 60.2 | 57.0 | 58.4 | 73.0 | 63.9 |
| LanCon-Learn | ✓ | 64.8 | 56.8 | 96.4 | 59.4 | 90.6 | 98.7 | 63.4 | 56.6 | 67.8 | 74.8 | 72.9 |
| Ours | ✓ | 93.8 | 82.7 | 99.5 | 69.9 | 96.3 | 100.0 | 95.3 | 82.1 | 82.3 | 90.7 | 88.3 |
| *Multi-task learning* | | | | | | | | | | | | |
| LanCon-Learn | - | 18.2 | 23.2 | 80.2 | 28.8 | 59.6 | 100.0 | 38.8 | 25.2 | 58.2 | 45.6 | 47.8 |
| LanCon-Learn | ✓ | 52.6 | 44.2 | 81.5 | 32.2 | 75.6 | 100.0 | 42.2 | 24.6 | 70.2 | 50.8 | 57.4 |
| Ours | ✓ | 88.8 | 93.0 | 100.0 | 75.0 | 58.0 | 100.0 | 79.8 | 70.4 | 79.0 | 89.2 | 83.3 |

and Auto-$\lambda$ [200], two state-of-the-art methods on RLBench and observe a consistent improvement for all tasks.

**Extending tasks in a single-task evaluation setup.** In Table 5.3, we further compare Auto-$\lambda$ and Hiveformer's variants across 74 RLBench tasks grouped into 9 categories. The variant without history removes the history tokens in Hiveformer, while the variant with one view only uses one camera at each step (we take the best among the 3 cameras for each task). The full Hiveformer achieves consistently better performance compared to Auto-$\lambda$ [200] on all types of tasks. Among them, the Long-term, Tools and Planning task groups assess the use of history, where our model brings improves significantly over the variant without history. Compared to the one view variant, our full model performs significantly better on tasks requiring fine-grained control or with large occlusions such as Screw, Precision and Visual Occlusion categories. Yet, our method performs relatively poorly for Long-term tasks with more than 10 steps, such as "take shoes out of box". As Long-term tasks have an average number of steps 2-4 times higher than others, they are more prone to distribution shift issues and accumulated errors. Hierarchical modeling or better training algorithms such as reinforcement learning and dagger could be helpful, but are left as future work.

### 5.5.4 Comparison with Additional State-of-the-Art Approach

LanCon-Learn [279] is a recent instruction-conditioned multi-task approach. It takes as input the gripper state and the object state, namely the ground truth pose of each object in the scene, instead of raw visual observations as ours. It encodes instructions with GloVe embeddings and a bi-directional LSTM. It predicts the next pose of the gripper based on a modular architecture conditioned on encoded text features. We run experiments on RLBench with the code provided by the authors. Since some RLBench tasks require identifying the colors of an object, we modify their object state to include a RGB reference of each object. Moreover, we complete their method with a history mechanism, where the gripper state is concatenated with the gripper state from the previous step.

As described in Table 5.4, we obtained an average success rate of 63.9% with their original method (vs. 72.9% with history vs. 88.3% for our approach) in our single-task

**Figure 5.3:** The tasks used in multi-variation setting. Left: push buttons task. Right: tower task.

setting and 47.8% (vs. 57.4% with history vs. 83.8% for our approach) in our multi-task setting.

**Multi-task evaluation.** The lower half in Table 5.2 shows the results in a multi-task setting. Notably, Auto-$\lambda$ uses a training algorithm that dynamically adjusts the weights of different tasks, while our model simply treats all tasks with equal weights. Nevertheless, our model outperforms Auto-$\lambda$ by 14%, demonstrating the improvements due to our architecture. We further compare our model with a variant without instructions in the input sequence (since $p^o_{t+1}$ is predicted from instructions, we modify the model such as it is predicted from $H^k_t$). The results show that instructions are important in the multi-task setting. Moreover, the performance of our *single* model trained for all tasks is only slightly worse than the performance of individual models for each task.

**Task Setup in Multi-variation Setting.** For the multi-variation setting we choose tasks with as many variations as possible. We hence select the push buttons and tower tasks, for which we can easily construct new variations, as illustrated in Figure 5.3. For each of these tasks we use 100 variations for training and 100 different variations for testing.

- The **Push Buttons** task has three buttons with unique colors in the scene. The robot should press some or all of the buttons according to the order in an instruction. Variations of the task are defined by the different order and different colors of buttons. RLBench provides three sentence templates to generate synthetic instructions with changing button colors such as "push the red button, and then push the cyan one".
- The **Tower** task is inspired by the "stack block" task. The robot must stack some of the three colored cubes at a target location following the color order provided by the instruction. We generate synthetic instructions for each variation, such as "Stack the red, blue, green blocks", or "Stack the yellow block. Stack the purple block on top of it, then add the cyan cube".

### 5.5.5 Additional Ablations on Multi-variation Setting

In the multi-variation setting in Table 5.5, the gap between LanCon-Learn and our approach is more significant than in Table 5.4, since GloVe embeddings differentiate

**Table 5.5:** Comparison with LanCon-Learn [279] and ablation of the instruction encoding in the multi-variation setting for seen or unseen variations and synthetic, corrupted or real instructions.

| Method | Hist. | Instructions | | | Visual | Push buttons | | | | Tower | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | Format | Encoder | Emb. $E_T^x$ | Emb. $E_T^v$ | Seen | Unseen | | | Seen | Unseen | | |
| | | | | | | Synt. | Synt. | Corr. | Real | Synt. | Synt. | Corr. | Real |
| LanCon-Learn | No | Cat. | GloVe | - | - | 25.6 | 12.1 | 0.3 | 0.1 | 21.3 | 9.1 | 0.1 | 0.0 |
| LanCon-Learn | Yes | Cat. | GloVe | - | - | 37.8 | 16.7 | 1.6 | 0.9 | 34.9 | 14.2 | 1.2 | 0.8 |
| Ours | No | Seq. | CLIP | ✓ | ✓ | 8.6 | 3.6 | 0.3 | 0.1 | 7.1 | 4.5 | 0.2 | 0.0 |
| Ours | Yes | Avg. | CLIP | ✓ | ✓ | 9.1 | 1.1 | 0.0 | 0.0 | 5.3 | 0.2 | 0.0 | 0.0 |
| Ours | Yes | Seq. | CLIP | - | ✓ | 100 | 83.2 | 81.1 | 71.3 | 77.1 | 53.2 | 51.3 | 21.3 |
| Ours | Yes | Seq. | CLIP | - | - | 86.2 | 65.2 | 56.4 | 49.8 | 54.9 | 34.8 | 29.8 | 24.7 |
| Ours | Yes | Seq. | BERT | ✓ | ✓ | 54.6 | 40.2 | 15.6 | 21.8 | 42.9 | 28.9 | 8.2 | 10.2 |
| Ours | Yes | Seq. | OHE | ✓ | ✓ | 100 | 3.1 | 0.1 | 0.0 | 96.8 | 3.8 | 0.4 | 0.0 |
| Ours | Yes | Seq. | CLIP | ✓ | ✓ | 100 | 86.3 | 85.6 | 74.2 | 77.4 | 56.2 | 53.6 | 24.1 |

poorly colors: for the "push buttons" task on unseen variations and synthetic instructions, the performance reaches only 1.7% (vs. 86.3% with our approach). This also happens when replacing CLIP embeddings with BERT in our model (40.2%).

The important role of history for long-term planning tasks such as "pushing buttons" is confirmed when comparing LanCon-Learn or our model with and without history in the Table 5.5. The model without history can only use its current observation to predict the next action. Therefore, it is hard to infer which buttons have been pressed and which button is the next target, leading to poor performance on the task.

We found that removing $E_T^x$ decreases the performance only by 3.1%, but removing both $E_T^x$ and $E_T^v$ decreases the performance by 21.1%. Moreover, replacing the instructions with one-hot encoding of the variation index increases the performance for seen variations (by 19.4% on the tower task), but prevents the model from generalizing to unseen variations.

We performed ablations with corrupted instructions on unseen variations. Corrupted instructions were created from synthetic instructions by replacing color references with synonyms that have not been seen during training. For example, the color "azure" is replaced with "light blue", and the color "maroon" with "dark red". Baselines using CLIP as a language encoder have a much smaller drop of performance than any other encoder.

We also test our model with a global language embedding (average over word tokens) as in [275] and observe a significant drop in performance. The main reason is that the averaged embeddings do not represent well different action orders, *e.g.* we have obtained the average cosine similarity of 0.97 for instructions corresponding to same actions in different orders.

**Generalization to multi-variations.** Table 5.6 shows results of Hiveformer trained on different variations of the two tasks *Tower* and *Push Buttons*. The *Tower* (resp. *Push Buttons*) task requires the robot to sequentially stack colored cubes (resp. push colored buttons) using the order specified in the instruction, see Figure 5.1 (right). We use 100 variations in training and test models for both the 100 seen variations and 100 unseen variations. In this setting, instructions are necessary to generalize to unseen

**Table 5.6:** Success rate (%) in the multi-variation setting for seen or unseen variations and synthetic or real instructions.

| # Demos Per Variation | Push Buttons Seen Synt. | Push Buttons Unseen Synt. | Push Buttons Unseen Real | Tower Seen Synt. | Tower Unseen Synt. | Tower Unseen Real |
|---|---|---|---|---|---|---|
| 10 | 96.4 | 71.1 | 65.7 | 71.6 | 49.8 | 19.4 |
| 50 | 99.4 | 83.1 | 70.9 | 74.3 | 52.1 | 20.6 |
| 100 | 100 | 86.3 | 74.2 | 77.4 | 56.2 | 24.1 |

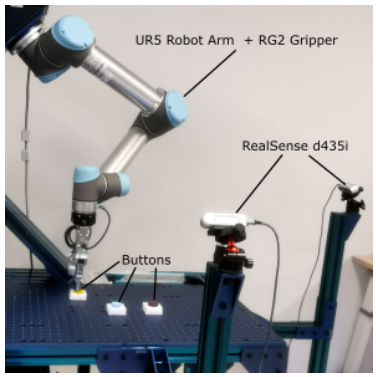



**Figure 5.4:** The robot scene with two RGB-D cameras and a UR5 robotics arm with an RG2 gripper.

variations (*e.g.* it is impossible to distinguish the order of pushing buttons red-green-blue vs. blue-red-green by only looking at the scene). We compare the models trained with different numbers of demonstrations per variation. Even in the most challenging case where only 10 demonstrations are available per variation, Hiveformer achieves a success rate of 71.1% for the *push buttons* task and 49.8% for the *tower* task in unseen variations. Furthermore, besides tests on synthetic instructions (Synt), we also test the generalization to real instructions. Despite being only trained on synthetic instructions with limited vocabulary and diversity, our model performs well on instructions generated by humans (Real). Finetuning Hiveformer on human instructions [242] is expected to result in further improvements.

### 5.5.6 Real-robot Experiments

**Setup.** We conduct real-robot experiments for the *push buttons* task on a 6-DoF UR5 robotic arm equipped with a 2-finger Robotiq RG2 gripper and two cameras on each side of the scene. As there exists a large difference between simulated and real environments, we finetune the simulator-trained policy on real-robot demonstrations. We use 10 variations of the task and 10 real-robot demonstrations per variation.

The cameras are Intel RealSense RGB-D cameras mounted on a fixed support as illustrated in Figure 5.4. We adapt our model to use $K = 2$ cameras. The resolution

**Table 5.7:** Success rate of push buttons task on real robots.

| Pretrain | Seen Vars | Unseen Vars |
|---|---|---|
| - | 86.7 | 13.3 |
| ✓ | 92.2 | 85.7 |

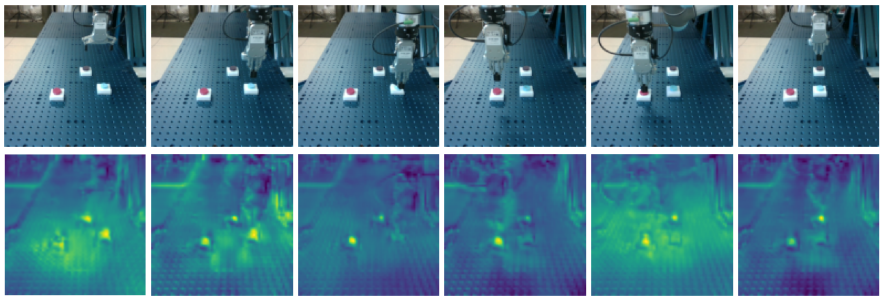


**Figure 5.5:** The instruction is "Press the cyan button, and then press the rose one, and then press the purple one". Top row: sequence of observations from one of the two side cameras in the robot scene. Bottom row: sequence of predicted attention maps by our model that indicate the gripper's position for the next step.

of the captured images is at a resolution of $1280 \times 720$, we apply center crop and downsampling to obtain images of size $128 \times 128$, which is the input to our model. We use nearest approximation to downsample depth images and bilinear approximation for RGB images. We use intrinsic parameters provided by Intel, and perform extrinsic calibration between the camera and the robot base-frame using an AprilTag marker [235]. We built 10 buttons using white cellulose foams: we manually cut them into $5 \times 5$ cm squares and attached to each square a painted rounded foam. The button bases and buttons have an average size of $4.95 \pm 0.1$ cm and $3.16 \pm 0.22$ cm respectively.

To collect demonstrations with the real robot, we design a script that automatically solves the task provided ground truth locations of buttons and the correct sequence of actions. In each demonstration objects are placed at random locations on the workspace and actions are executed at 10 Hz. We finetune the model for 8k iterations using the same training setup as that in the simulator.

**Qualitative Results.** Figure 5.5 shows a successful example from our real robot experiments. The attention maps reveal that the robot correctly attends to the next buttons. Thanks to the history of previous observations and actions, the model is confident to not press a button that has already been pressed before (for example the cyan in the fourth column).

In Figure 5.6, we analyze the robustness of our model for to unseen variations in more challenging situations. The instruction of the variation is written by human: *"Press the yellow button and then press the black button and finish with the white button"*. Our model successfully pressed the buttons in the correct order for all situations in Figure 5.6.

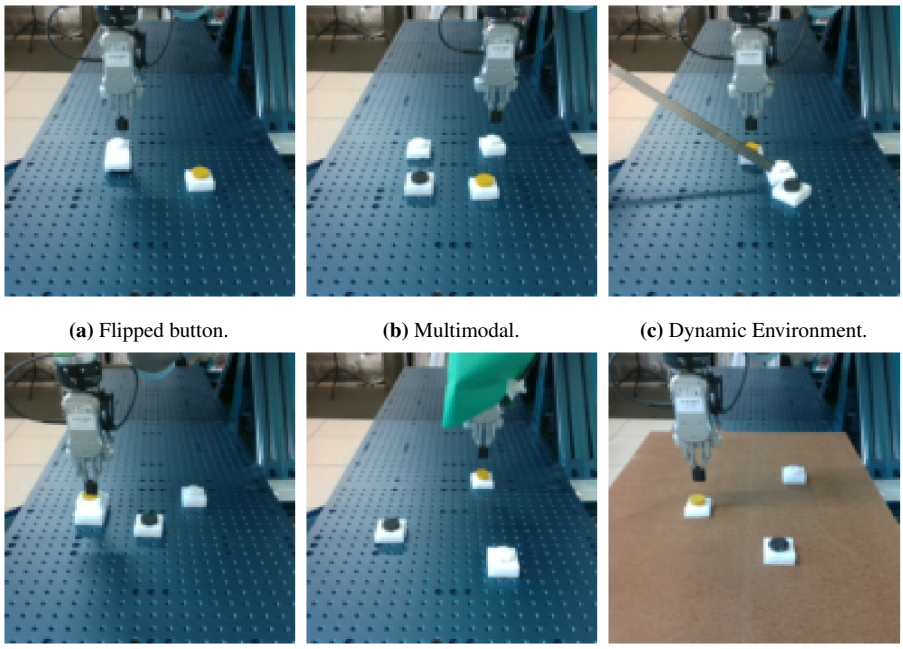
- Figure 5.6a: Since the foam buttons have low friction with the table, the gripper has accidentally flipped the black button, providing two buttons looking white. However, thanks to its history component, the robot is able to successfully press the right white button instead of the flipped button.
- Figure 5.6b: Two white buttons are present in the scene. This is a multi-modal example, in which the robot might predict a mean position between the two white buttons, whereas our robot can cope with this challenge.
- Figure 5.6c: We use a ruler to move the location the white button in the scene after the robot pushed the yellow button. Although such perturbations have never been used in training sequences, the robot remains robust to this dynamic environment.
- Figure 5.6d: We change the shape of the button by increasing the height of the yellow button.
- Figure 5.6e: We add occlusion to the gripper.
- Figure 5.6f: We change the appearance of the table. Our model is robust to the above visual modifications.

More details and video demonstrations of our real-robot experiments are available from the project webpage [99].

## 5.6 Conclusion

We introduced Hiveformer, a multimodal transformer that jointly models instructions, views from multiple cameras, and history for instruction-driven robotics manipulation. We evaluated the model on RLBench in three settings: single-task learning, multi-task learning, and multi-variation generalization, and we demonstrated its effectiveness outperforming state-of-the-art. We deployed our model on a real robot that is able to generalize to unseen variations and human-written instructions.

**Limitations.** The computational cost quadratically increases with the input sequence length due to the transformer. Furthermore, our model is trained with behavioral cloning, which may suffer from exposure bias. Future works could improve the efficiency of long-term tasks with hierarchical models and also incorporate reinforcement learning. Moreover, our model is trained on only synthetic instructions and performs worse on human-written instructions. Training on human-written automatically generated instructions could help improve performance.



**(a)** Flipped button. **(b)** Multimodal. **(c)** Dynamic Environment.

**(d)** Unseen Button. **(e)** Unseen Gripper. **(f)** Unseen Table.

**Figure 5.6:** Robustness of the learned policy on an unseen variation: *"Press the yellow button and then press the black button and finish with the white button"*.

# Chapter 6

# Perspectives

In this Chapter, we summarize the main contributions of this thesis in Section 6.1, and we discuss open research problems and directions for future work in Section 6.2.

## 6.1 Contributions

This thesis has studied how robots can follow instructions. This research field, language-guided robotics, has its foundations in computer vision, natural language processing, and task planning. We have addressed two different tasks of language-guided robotics.

The first task is called vision-and-language navigation. It consists in asking a robot to find a target location, described by provided instructions. We performed experiments in the Matterport3D simulator [44] and considered several benchmarks: some instructions are fine-grained [14, 139, 170], meaning that step-by-step guidance is provided, but some other instructions are coarse-grained [251, 361], in which case only a description of the target location is provided. We have also tested our approaches on recorded dialogs [302].

Since collecting instructions and environments is particularly costly, models suffer from data scarcity. In Chapter 3, we have addressed this issue by collecting BnB, which consists of more than 1 million images and captions on a rental marketplace. We have suggested several directions to efficiently pre-train discriminative and generative models. In addition, we have proposed a new training loss, dubbed as the shuffle loss, to enhance the learning of temporal causality in the transformer. Airbert, our model trained with the BnB dataset and the shuffling loss, obtained best results in the REVERIE [251] and in R2R [14], which is still ranked first two years later [77]. Furthermore, this work has suggested a new challenging setup where the training dataset is reduced to a single environment. This work has inspired other researchers: in particular, Qiao *et al*. [253] uses our BnB dataset and an improvement of our shuffle loss to obtain state-of-the-art results over 4 datasets; Hahn and Rehg [105] demonstrates the significant improvement brought by our shuffle loss.

While Airbert is using pre-computed visual features, in Chapter 4 we proposed an

end-to-end approach dubbed HAMT. In this approach, a transformer-based architecture is aligning instructions to the panoramic-view observations obtained for all previous steps. The panoramic views are reshaped into thousands of tokens, which is problematic for Transformer models, whose algorithmic complexity is polynomial with respect to the number of tokens. That is why, we considered a hierarchical approach: a first Transformer architecture encodes visual features, while a second architecture learns the alignment. We have outperformed state-of-the-art over four datasets, and our model has been used as a backbone of multiple follow-up works, notably CLEAR [185], EnvEdit [186], An *et al*. [8], Iterative VLN [163], ULN [80], CSAP [330], and Kamath *et al*. [150].

The second axis of this thesis considers tasks based on manipulation. In Chapter 5, we have provided instructions to a robotics arm, requesting it to manipulate objects. Manipulation is a different task, because (i) the environment is entirely observable, but (ii) the action space is continuous. We have offered a new variant of the transformer to deal with 74 tasks on RLBench [143], outperforming existing methods on single-task and multi-task settings. We have also suggested a new challenging setting, referred to as multi-variation, where the robot learns several variations on a single task and must generalize to unseen variations. We tested this challenging setting on a real-world robot. Finally, we tested that our robot can generalize to human-written instructions while being trained on synthetic instructions.

## 6.2 Future research directions

While this thesis achieves new state-of-the-art results over a dozen of benchmarks, our proposed methods suffer from several limitations. In particular, this thesis deals separately with navigation and manipulation, and one could wonder how to combine them. Below we suggest several directions to address current limitations.

**Pre-training.** In Chapter 3, we show that pre-training on the BnB dataset can significantly improve performance. Nevertheless, it still tends to overfit training environments and would benefit from additional training data. While large-scale task-specific data collection is prohibitive, alternative approaches exist. Notably, in Auto-VLN [57], embodied agents are pre-trained on 1000 unlabelled 3D environments, and we show that their performance would even benefit from a higher number of pre-trained environments. But how to obtain a much higher number of environments? For example, one could use the images of the BnB dataset to generate environments, as in Hosseini and Furukawa [123]. Even though the quality of such environments might be lowered, they could enhance any embodied agents. Pre-training could also benefit from videos. Numerous YouTube videos present walking tours, whereas the large-scale Something-Something dataset [96] contains 220k short videos of human manipulation. Those videos have successfully been employed to learn robotics tasks [246], and could potentially enhance language-guided robots.

**Exposure bias.** Models trained to reproduce demonstrations have not been exposed to any new or unseen data points and may not be able to accurately predict outcomes for novel inputs. This issue is a typical limitation of behavioral cloning, used in

Hiveformer 5. In Chapters 3 and 4, reinforcement learning has successfully mitigated this issue, because it enhances the exploration of environments during training. But in the context of continuous actions, reinforcement learning is sample inefficient [207, 228], and future works could investigate other strategies to reduce the exposure bias of Hiveformer.

**Motion-planning.** In this thesis, we have assumed the existence of a lower-level controller, a motion planner, which enabled us to focus on a higher-level controller, a task planner. In the case of navigation, we supposed that such a controller would allow an agent to move from one location to another, whereas in the case of manipulation, we employed inverse kinematics to predict the trajectory of the gripper. However, James *et al*. [141] showed that learning-based controllers could outperform inverse-kinematics in RLBench, as it allows a robot to learn the affordance of objects. For navigation, recent works [164, 135] studied how motion planners and task planners can be learned at the same time. Further works could extend it to the context of manipulation.

Robots able to understand natural language utterances help the evaluation of their cognitive abilities and can be used by anthropologists to gain more insights about robotics. In particular, it has led to the development of innovative methods, such as "integrative social robotics" [268].

**Environmental concerns.** Robots pose also an environmental risk [97]. Sustainability is affected by several factors. First, raw materials to build a robot have a significant environmental cost [333], while the vast majority of manufacturers do not conceive and build in a circular economy paradigm. Second, the greenhouse gas emissions of the scope 1 and 2 (direct emissions) of a robot are significant, mostly due to greenhouse gases emitted during the fabrication of a robot [187], and emissions used by cloud computing [229]. Last but not least, the scope 3 (indirect emissions) [238] is likely to become the main concern of robotics since new applications emerging with robots are expected to increase the demand for goods and services [278].

More than 200 000 V100 GPU hours have been used during this thesis, consuming roughly 100 MWh. Given the carbon intensity of France, the used electricity has produced around 10 tCO2eq. However, the environmental impact of this thesis is difficult to estimate. If the environmental impact can be estimated with the life cycle assessment, other phases such as the fabrication, transport, and end-of-life of the computer cluster, must be taken into account and they may represent around 50% of the carbon footprint [31]. But the attributional framework of life cycle assessment might hide real impacts impeded to the consequence of this thesis [191]: the emergence of robots solving household tasks could significantly increase the amount of energy and materials required to build such robots. Being able to improve the impacts of our research should be made a priority to analyze the costs and benefits of our research.

# Bibliography


[1] Panos Achlioptas, Ahmed Abdelreheem, Fei Xia, Mohamed Elhoseiny, and Leonidas Guibas. ReferIt3D: Neural listeners for fine-grained 3d object identification in real-world scenes. In *European Conference on Computer Vision*, pages 422–440. Springer, 2020. 16, 17

[2] Ossama Ahmed, Frederik Träuble, Anirudh Goyal, Alexander Neitz, Yoshua Bengio, Bernhard Schölkopf, Manuel Wüthrich, and Stefan Bauer. CausalWorld: A robotic manipulation benchmark for causal structure and transfer learning. *arXiv preprint arXiv:2010.04296*, 2020. 80

[3] Airbnb. Airbnb fast facts. Accessed: 2021-03-13 at https://news.airbnb.com/fast-facts/. 24

[4] Ilge Akkaya, Marcin Andrychowicz, Maciek Chociej, Mateusz Litwin, Bob McGrew, Arthur Petron, Alex Paino, Matthias Plappert, Glenn Powell, Raphael Ribas, et al. Solving rubik's cube with a robot hand. *arXiv preprint arXiv:1910.07113*, 2019. 7, 79

[5] Chris Alberti, Jeffrey Ling, Michael Collins, and David Reitter. Fusion of detected objects in text for visual question answering. In *EMNLP*, 2019. 22

[6] Md Amirul Islam, Mrigank Rochan, Neil DB Bruce, and Yang Wang. Gated feedback refinement network for dense image labeling. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 3751–3759, 2017. 16

[7] Haitham Bou Ammar, Eric Eaton, Paul Ruvolo, and Matthew Taylor. Online multi-task learning for policy gradient methods. In *ICML*, pages 1206–1214. PMLR, 2014. 79

[8] Dong An, Zun Wang, Yangguang Li, Yi Wang, Yicong Hong, Yan Huang, Liang Wang, and Jing Shao. 1st place solutions for rxr-habitat vision-and-language navigation competition (cvpr 2022). *arXiv preprint arXiv:2206.11610*, 2022. 96

[9] Peter Anderson, Angel Chang, Devendra Singh Chaplot, Alexey Dosovitskiy, Saurabh Gupta, Vladlen Koltun, Jana Kosecka, Jitendra Malik, Roozbeh Mottaghi, Manolis Savva, et al. On evaluation of embodied navigation agents. *arXiv preprint arXiv:1807.06757*, 2018. 10, 18, 21, 23, 49, 51, 58, 59

[10] Peter Anderson, Xiaodong He, Chris Buehler, Damien Teney, Mark Johnson, Stephen Gould, and Lei Zhang. Bottom-up and top-down attention for image captioning and visual question answering. In *CVPR*, pages 6077–6086, 2018. 10

[11] Peter Anderson, Xiaodong He, Chris Buehler, Damien Teney, Mark Johnson, Stephen Gould, and Lei Zhang. Bottom-up and top-down attention for image captioning and visual question answering. In *CVPR*, 2018. 31, 34, 35

[12] Peter Anderson, Ayush Shrivastava, Devi Parikh, Dhruv Batra, and Stefan Lee. Chasing ghosts: Instruction following as bayesian state tracking. *NeurIPS*, 32:371–381, 2019. 19

[13] 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 *CVPR*, 2018. 18, 19, 21, 23, 25, 26, 28, 33, 35, 38, 39, 41

[14] 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*, pages 3674–3683, 2018. 50, 51, 52, 53, 58, 59, 63, 67, 95

[15] Lisa Anne Hendricks, Oliver Wang, Eli Shechtman, Josef Sivic, Trevor Darrell, and Bryan Russell. Localizing moments in video with natural language. In *Proceedings of the IEEE international conference on computer vision*, pages 5803–5812, 2017. 16

[16] Stanislaw Antol, Aishwarya Agrawal, Jiasen Lu, Margaret Mitchell, Dhruv Batra, C Lawrence Zitnick, and Devi Parikh. VQA: Visual Question Answering. In *ICCV*, pages 2425–2433, 2015. 16, 18, 24

[17] Archive. Search the history of over 739 billion web pages on the internet. Searchthehistoryofover739billionwebpagesontheInternet. Access:2022-09-28. 5

[18] Anurag Arnab, Mostafa Dehghani, Georg Heigold, Chen Sun, Mario Lučić, and Cordelia Schmid. Vivit: A video vision transformer. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 6836–6846, 2021. 14, 54

[19] Yoav Artzi and Luke Zettlemoyer. Weakly supervised learning of semantic parsers for mapping instructions to actions. *Transactions of the Association for Computational Linguistics*, 1:49–62, 2013. 16

[20] Dilip Arumugam, Siddharth Karamcheti, Nakul Gopalan, Edward C Williams, Mina Rhee, Lawson LS Wong, and Stefanie Tellex. Grounding natural language instructions to semantic goal representations for abstraction and generalization. *Autonomous Robots*, 43(2):449–468, 2019. 16

[21] Minoru Asada, Hiroaki Kitano, Itsuki Noda, and Manuela Veloso. Robocup: Today and tomorrow—what we have learned. *Artificial Intelligence*, 110(2):193–214, 1999. 17

[22] Karl J Åström and Björn Wittenmark. *Adaptive control*. Courier Corporation, 2013. 14

[23] Jimmy Lei Ba, Jamie Ryan Kiros, and Geoffrey E Hinton. Layer normalization. *arXiv preprint arXiv:1607.06450*, 2016. 54

[24] Jimmy Lei Ba, Jamie Ryan Kiros, and Geoffrey E Hinton. Layer normalization. *arXiv preprint arXiv:1607.06450*, 2016. 82

[25] Quentin Bateux, Eric Marchand, Jürgen Leitner, François Chaumette, and Peter Corke. Training deep neural networks for visual servoing. In *ICRA*, pages 1–8. IEEE, 2018. 79

[26] Khaled Bayoudh, Raja Knani, Fayçal Hamdaoui, and Abdellatif Mtibaa. A survey on deep multimodal learning for computer vision: advances, trends, applications, and datasets. *The Visual Computer*, 38(8):2939–2970, 2022. 13

[27] Charles Beattie, Joel Z Leibo, Denis Teplyashin, Tom Ward, Marcus Wainwright, Heinrich Küttler, Andrew Lefrancq, Simon Green, Víctor Valdés, Amir Sadik, et al. Deepmind lab. *arXiv preprint arXiv:1612.03801*, 2016. 18

[28] Bahram Behzadian, Pratik Agarwal, Wolfram Burgard, and Gian Diego Tipaldi. Monte carlo localization in hand-drawn maps. In *2015 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 4291–4296. IEEE, 2015. 15

[29] Tony Belpaeme, James Kennedy, Aditi Ramachandran, Brian Scassellati, and Fumihide Tanaka. Social robots for education: A review. *Science robotics*, 3(21):eaat5954, 2018. 5

[30] Samy Bengio, Oriol Vinyals, Navdeep Jaitly, and Noam Shazeer. Scheduled sampling for sequence prediction with recurrent neural networks. In *Advances in Neural Information Processing Systems*, volume 28, 2015. 50

[31] Francoise Berthoud, Bruno Bzeznik, Nicolas Gibelin, Myriam Laurens, Cyrille Bonamy, Maxence Morel, and Xavier Schwindenhammer. *Estimation de l'empreinte carbone d'une heure. coeur de calcul*. PhD thesis, UGA-Université Grenoble Alpes; CNRS; INP Grenoble; INRIA, 2020. 97

[32] 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. 15

[33] Alfred H Bloom. The impact of chinese linguistic structure on cognitive style. *Current anthropology*, 20(3):585–586, 1979. 15

[34] Valts Blukis, Dipendra Misra, Ross A Knepper, and Yoav Artzi. Mapping navigation instructions to continuous control actions with position-visitation prediction. In *CoRL*, pages 505–518. PMLR, 2018. 19

[35] Robert Bogue. The role of robots in firefighting. *Industrial Robot: the international journal of robotics research and application*, 2021. 4

[36] Greg Brockman, Vicki Cheung, Ludwig Pettersson, Jonas Schneider, John Schulman, Jie Tang, and Wojciech Zaremba. OpenAI Gym. *arXiv preprint arXiv:1606.01540*, 2016. 80

[37] Simon Brodeur, Ethan Perez, Ankesh Anand, Florian Golemo, Luca Celotti, Florian Strub, Jean Rouat, Hugo Larochelle, and Aaron Courville. Home: A household multimodal environment. *arXiv preprint arXiv:1711.11017*, 2017. 18

[38] Eduardo F Camacho and Carlos Bordons Alba. *Model predictive control*. Springer science & business media, 2013. 14

[39] Cătălina Cangea, Eugene Belilovsky, Pietro Liò, and Aaron Courville. Videonavqa: Bridging the gap between visual and embodied question answering. *arXiv preprint arXiv:1908.04950*, 2019. 18

[40] Jize Cao, Zhe Gan, Yu Cheng, Licheng Yu, Yen-Chun Chen, and Jingjing Liu. Behind the scene: Revealing the secrets of pre-trained vision-and-language models. In *European Conference on Computer Vision*, pages 565–580. Springer, 2020. 54

[41] Rich Caruana. Learning many related tasks at the same time with backpropagation. *NeurIPS*, 7, 1994. 79

[42] Charlotte Caucheteux and Jean-Rémi King. Brains and algorithms partially converge in natural language processing. *Communications biology*, 5(1):1–10, 2022. 5

[43] Angel Chang, Angela Dai, Thomas Funkhouser, Maciej Halber, Matthias Niebner, Manolis Savva, Shuran Song, Andy Zeng, and Yinda Zhang. Matterport3d: Learning from rgb-d data in indoor environments. In *2017 International Conference on 3D Vision (3DV)*, pages 667–676. IEEE, 2017. 2, 8, 18, 58

[44] Angel Chang, Angela Dai, Thomas Funkhouser, Maciej Halber, Matthias Niessner, Manolis Savva, Shuran Song, Andy Zeng, and Yinda Zhang. Matterport3D: Learning from RGB-D data in indoor environments. *3DV*, 2017. 18, 24, 26, 42, 95

[45] Huiwen Chang, Han Zhang, Lu Jiang, Ce Liu, and William T Freeman. Maskgit: Masked generative image transformer. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 11315–11325, 2022. 14

[46] Devendra Singh Chaplot, Ruslan Salakhutdinov, Abhinav Gupta, and Saurabh Gupta. Neural topological SLAM for visual navigation. In *CVPR*, pages 12875–12884, 2020. 19

[47] François Chaumette and Seth Hutchinson. Visual servo control. i. basic approaches. *IEEE Robotics & Automation Magazine*, 13(4):82–90, 2006. 14, 79

[48] Annie S Chen, Suraj Nair, and Chelsea Finn. Learning generalizable robotic reward functions from "in-the-wild" human videos. *RSS*, 2021. 79

[49] Bryan Chen, Alexander Sax, Gene Lewis, Iro Armeni, Silvio Savarese, Amir Zamir, Jitendra Malik, and Lerrel Pinto. Robust policies via mid-level visual representations: An experimental study in manipulation and navigation. *CoRL*, 2020. 79

[50] Dave Zhenyu Chen, Angel X Chang, and Matthias Nießner. ScanRefer: 3d object localization in rgb-d scans using natural language. In *European Conference on Computer Vision*, pages 202–221. Springer, 2020. 16, 17

[51] Howard Chen, Alane Suhr, Dipendra Misra, Noah Snavely, and Yoav Artzi. Touchdown: Natural language navigation and spatial reasoning in visual street environments. In *CVPR*, pages 12538–12547, 2019. 18, 23, 49, 50, 51

[52] Jingyuan Chen, Xinpeng Chen, Lin Ma, Zequn Jie, and Tat-Seng Chua. Temporally grounding natural sentence in video. In *Proceedings of the 2018 conference on empirical methods in natural language processing*, pages 162–171, 2018. 16

[53] Kevin Chen, Junshen K Chen, Jo Chuang, Marynel Vázquez, and Silvio Savarese. Topological planning with transformers for vision-and-language navigation. In *CVPR*, pages 11276–11286, 2021. 19

[54] Lili Chen, Kevin Lu, Aravind Rajeswaran, Kimin Lee, Aditya Grover, Misha Laskin, Pieter Abbeel, Aravind Srinivas, and Igor Mordatch. Decision transformer: Reinforcement learning via sequence modeling. In *NeurIPS*, 2021. 80

[55] Shizhe Chen, Pierre-Louis Guhur, Cordelia Schmid, and Ivan Laptev. History aware multimodal transformer for vision-and-language navigation. *Advances in Neural Information Processing Systems*, 34, 2021. 80

[56] Shizhe Chen, Pierre-Louis Guhur, Makarand Tapaswi, Cordelia Schmid, and Ivan Laptev. Language conditioned spatial relation reasoning for 3d object grounding. *NIPS*, 2022. 11

[57] Shizhe Chen, Pierre-Louis Guhur, Makarand Tapaswi, Cordelia Schmid, and Ivan Laptev. Learning from unlabeled 3d environments for vision-and-language navigation. In *European Conference on Computer Vision*, pages 638–655. Springer, 2022. 96

[58] Shizhe Chen, Pierre-Louis Guhur, Makarand Tapaswi, Cordelia Schmid, and Ivan Laptev. Think global, act local: Dual-scale graph transformer for vision-and-language navigation. In *CVPR*, 2022. 10, 19

[59] Yen-Chun Chen, Linjie Li, Licheng Yu, Ahmed El Kholy, Faisal Ahmed, Zhe Gan, Yu Cheng, and Jingjing Liu. Uniter: Universal image-text representation learning. In *ECCV*, pages 104–120. Springer, 2020. 22, 24, 52

[60] Alexis Conneau, Kartikay Khandelwal, Naman Goyal, Vishrav Chaudhary, Guillaume Wenzek, Francisco Guzmán, Édouard Grave, Myle Ott, Luke Zettlemoyer, and Veselin Stoyanov. Unsupervised cross-lingual representation learning at scale. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 8440–8451, 2020. 66

[61] Antonia Creswell, Tom White, Vincent Dumoulin, Kai Arulkumaran, Biswa Sengupta, and Anil A Bharath. Generative adversarial networks: An overview. *IEEE signal processing magazine*, 35(1):53–65, 2018. 1

[62] Ekin D Cubuk, Barret Zoph, Jonathon Shlens, and Quoc V Le. Randaugment: Practical automated data augmentation with a reduced search space. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops*, pages 702–703, 2020. 60

[63] Angela Dai, Angel X Chang, Manolis Savva, Maciej Halber, Thomas Funkhouser, and Matthias Nießner. Scannet: Richly-annotated 3d reconstructions of indoor scenes. In *CVPR*, pages 5828–5839, 2017. 16

[64] Abhishek Das, Samyak Datta, Georgia Gkioxari, Stefan Lee, Devi Parikh, and Dhruv Batra. Embodied question answering. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*, pages 1–10, 2018. 18

[65] Sudeep Dasari, Frederik Ebert, Stephen Tian, Suraj Nair, Bernadette Bucher, Karl Schmeckpeper, Siddharth Singh, Sergey Levine, and Chelsea Finn. Robonet: Large-scale multi-robot learning. *arXiv preprint arXiv:1910.11215*, 2019. 17

[66] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. ImageNet: A large-scale hierarchical image database. In *2009 IEEE conference on computer vision and pattern recognition*, pages 248–255. IEEE, 2009. 13, 56

[67] Zhiwei Deng, Karthik Narasimhan, and Olga Russakovsky. Evolving graphical planner: Contextual global planning for vision-and-language navigation. In *Advances in Neural Information Processing Systems*, volume 33, 2020. 19, 50, 52, 64

[68] Coline Devin, Abhishek Gupta, Trevor Darrell, Pieter Abbeel, and Sergey Levine. Learning modular neural network policies for multi-task and multi-robot transfer. In *ICRA*, pages 2169–2176. IEEE, 2017. 79

[69] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. BERT: Pre-training of deep bidirectional transformers for language understanding. In *NAACL-HLT*, 2019. 22, 24, 31, 39, 78, 80, 82

[70] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, 2019. 50, 51, 53

[71] Fethiye Irmak Doğan, Sinan Kalkan, and Iolanda Leite. Learning to generate unambiguous spatial referring expressions for real-world environments. In *2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 4992–4999. IEEE, 2019. 16

[72] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, et al. An image is worth 16x16 words: Transformers for image recognition at scale. *ICLR*, 2020. 14, 52, 57, 60, 80

[73] Zi-Yi Dou and Nanyun Peng. Foam: A follower-aware speaker model for vision-and-language navigation. *arXiv preprint arXiv:2206.04294*, 2022. 19

[74] Kiana Ehsani, Winson Han, Alvaro Herrasti, Eli VanderBilt, Luca Weihs, Eric Kolve, Aniruddha Kembhavi, and Roozbeh Mottaghi. ManipulaTHOR: A framework for visual object manipulation. In *CVPR*, pages 4497–4506, 2021. 17, 80

[75] Clemens Eppner, Sebastian Höfer, Rico Jonschkowski, Roberto Martín-Martín, Arne Sieverling, Vincent Wall, and Oliver Brock. Lessons from the amazon picking challenge: Four aspects of building robotic systems. In *Robotics: science and systems*, 2016. 17

[76] Andre Esteva, Katherine Chou, Serena Yeung, Nikhil Naik, Ali Madani, Ali Mottaghi, Yun Liu, Eric Topol, Jeff Dean, and Richard Socher. Deep learning-enabled medical computer vision. *NPJ digital medicine*, 4(1):1–9, 2021. 1

[77] Eval.ai. Leadearboard evalai - room-to-room dataset. https://eval.ai/web/challenges/challenge-page/97/leaderboard/270. Access:2022-10-11. 21, 95

[78] Mark Everingham, Luc Van Gool, Christopher KI Williams, John Winn, and Andrew Zisserman. The pascal visual object classes (voc) challenge. In *International journal of computer vision*, volume 88, pages 303–338. Springer, 2010. 13

[79] Kuan Fang, Alexander Toshev, Li Fei-Fei, and Silvio Savarese. Scene memory transformer for embodied agents in long-horizon tasks. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 538–547, 2019. 50, 52

[80] Weixi Feng, Tsu-Jui Fu, Yujie Lu, and William Yang Wang. Uln: Towards underspecified vision-and-language navigation. In *EMNLP*, 2022. 96

[81] Chelsea Finn and Sergey Levine. Deep visual foresight for planning robot motion. In *ICRA*, pages 2786–2793. IEEE, 2017. 79

[82] Chelsea Finn, Tianhe Yu, Tianhao Zhang, Pieter Abbeel, and Sergey Levine. One-shot visual imitation learning via meta-learning. In *CoRL*, pages 357–368. PMLR, 2017. 15, 79

[83] Pete Florence, Corey Lynch, Andy Zeng, Oscar A Ramirez, Ayzaan Wahid, Laura Downs, Adrian Wong, Johnny Lee, Igor Mordatch, and Jonathan Tompson. Implicit behavioral cloning. In *CoRL*, pages 158–168. PMLR, 2022. 17, 79

[84] Terry Kit fong Au. Counterfactuals: In reply to alfred bloom. *Cognition*, 17(3):289–302, 1984. 15

[85] Daniel Fried, Ronghang Hu, Volkan Cirik, Anna Rohrbach, Jacob Andreas, Louis-Philippe Morency, Taylor Berg-Kirkpatrick, Kate Saenko, Dan Klein, and Trevor Darrell. Speaker-Follower models for vision-and-language navigation. In *NeurIPS*, pages 3318–3329, 2018. 19, 23, 24, 39, 41, 50, 51, 52, 53, 63, 64

[86] Chuang Gan, Jeremy Schwartz, Seth Alter, Martin Schrimpf, James Traer, Julian De Freitas, Jonas Kubilius, Abhishek Bhandwaldar, Nick Haber, Megumi Sano, et al. Threedworld: A platform for interactive multi-modal physical simulation. *arXiv preprint arXiv:2007.04954*, 2020. 18

[87] Jiyang Gao, Chen Sun, Zhenheng Yang, and Ram Nevatia. Tall: Temporal activity localization via language query. In *Proceedings of the IEEE international conference on computer vision*, pages 5267–5275, 2017. 16

[88] Xiaofeng Gao, Qiaozi Gao, Ran Gong, Kaixiang Lin, Govind Thattai, and Gaurav S Sukhatme. Dialfred: Dialogue-enabled agents for embodied instruction following. *arXiv preprint arXiv:2202.13330*, 2022. 18

[89] Divyansh Garg, Skanda Vaidyanath, Kuno Kim, Jiaming Song, and Stefano Ermon. LISA: Learning interpretable skill abstractions from language. *arXiv preprint arXiv:2203.00054*, 2022. 80

[90] Andreas Geiger, Philip Lenz, and Christoph Stiller. Vision meets robotics: The kitti dataset. *The International Journal of Robotics Research*, 32(11):1231–1237, 2013. 13

[91] Dileep George and Jeff Hawkins. A hierarchical bayesian model of invariant pattern recognition in the visual cortex. In *Proceedings. 2005 IEEE International Joint Conference on Neural Networks, 2005.*, volume 3, pages 1812–1817. IEEE, 2005. 5

[92] Ross Girshick. Fast R-CNN. In *Proceedings of the IEEE international conference on computer vision*, pages 1440–1448, 2015. 16

[93] Ross Girshick, Jeff Donahue, Trevor Darrell, and Jitendra Malik. Rich feature hierarchies for accurate object detection and semantic segmentation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 580–587, 2014. 16

[94] Walter Goodwin, Sagar Vaze, Ioannis Havoutis, and Ingmar Posner. Semantically grounded object matching for robust robotic scene rearrangement. *ICRA*, 2021. 80

[95] Nakul Gopalan, Dilip Arumugam, Lawson LS Wong, and Stefanie Tellex. Sequence-to-sequence language grounding of non-markovian task specifications. In *Robotics: Science and Systems*, 2018. 16

[96] 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. 96

[97] María Amparo Grau Ruiz and Fiachra O'Brolchain. Environmental robotics for a sustainable future in circular economies. *Nature Machine Intelligence*, 4(1):3–4, 2022. 97

[98] Giorgio Grisetti, Gian Diego Tipaldi, Cyrill Stachniss, Wolfram Burgard, and Daniele Nardi. Fast and accurate slam with rao–blackwellized particle filters. *Robotics and Autonomous Systems*, 55(1):30–38, 2007. 15

[99] Pierre-Louis Guhur. Hiveformer webpage. https://guhur.github.io/hiveformer/. Access:2022-09-18. 79, 93

[100] Pierre-Louis Guhur, Makarand Tapaswi, Shizhe Chen, Ivan Laptev, and Cordelia Schmid. Airbert: In-domain pretraining for vision-and-language navigation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 1634–1643, 2021. 80

[101] Saurabh Gupta, James Davidson, Sergey Levine, Rahul Sukthankar, and Jitendra Malik. Cognitive mapping and planning for visual navigation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2616–2625, 2017. 52

[102] Tanmay Gupta, Arash Vahdat, Gal Chechik, Xiaodong Yang, Jan Kautz, and Derek Hoiem. Contrastive learning for weakly supervised phrase grounding. In *ECCV*, 2020. 38

[103] Michael Gutmann and Aapo Hyvärinen. Noise-contrastive estimation: A new estimation principle for unnormalized statistical models. In *Proceedings of the Thirteenth International Conference on Artificial Intelligence and Statistics*, pages 297–304. JMLR Workshop and Conference Proceedings, 2010. 58

[104] Tuomas Haarnoja, Aurick Zhou, Kristian Hartikainen, George Tucker, Sehoon Ha, Jie Tan, Vikash Kumar, Henry Zhu, Abhishek Gupta, and Pieter Abbeel. Soft actor-critic algorithms and applications. *arXiv preprint arXiv:1812.05905*, 2018. 79

[105] Meera Hahn and James M Rehg. Which way is 'right'?: Uncovering limitations of vision-and-language navigation models, 2022. 95

[106] Weituo Hao, Chunyuan Li, Xiujun Li, Lawrence Carin, and Jianfeng Gao. Towards learning a generic agent for vision-and-language navigation via pre-training. In *CVPR*, 2020. 19, 22, 23, 24, 35, 39, 40, 41

[107] Weituo Hao, Chunyuan Li, Xiujun Li, Lawrence Carin, and Jianfeng Gao. Towards learning a generic agent for vision-and-language navigation via pre-training. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 13137–13146, 2020. 50, 52, 55, 60, 63, 65, 67

[108] Clyde L Hardin and Luisa Ed Maffi. Color categories in thought and language. In *Color Categories, Oct, 1992, Asilomar, CA, US; This volume is based on the" Color Categories" conference, held in Asilomar, California, Oct 25–28, 1992.* Cambridge University Press, 1997. 15

[109] Yana Hasson, Gul Varol, Dimitrios Tzionas, Igor Kalevatykh, Michael J Black, Ivan Laptev, and Cordelia Schmid. Learning joint reconstruction of hands and manipulated objects. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 11807–11816, 2019. 13

[110] Dailan He, Yusheng Zhao, Junyu Luo, Tianrui Hui, Shaofei Huang, Aixi Zhang, and Si Liu. Transrefer3d: Entity-and-relation aware transformer for fine-grained 3d visual grounding. In *ACM MM*, pages 2344–2352, 2021. 17

[111] Kaiming He, Haoqi Fan, Yuxin Wu, Saining Xie, and Ross Girshick. Momentum contrast for unsupervised visual representation learning. In *CVPR*, 2020. 78

[112] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *CVPR*, pages 770–778, 2016. 13, 51

[113] Keji He, Yan Huang, Qi Wu, Jianhua Yang, Dong An, Shuanglin Sima, and Liang Wang. Landmark-rxr: Solving vision-and-language navigation with fine-grained alignment supervision. *Advances in Neural Information Processing Systems*, 34:652–663, 2021. 18

[114] Karl Moritz Hermann, Felix Hill, Simon Green, Fumin Wang, Ryan Faulkner, Hubert Soyer, David Szepesvari, Wojciech Marian Czarnecki, Max Jaderberg, Denis Teplyashin, Marcus Wainwright, Chris Apps, Demis Hassabis, and Phil Blunso. Grounded language learning in a simulated 3D world. *NeurIPS Workshop*, 2017. 80

[115] John Hill. Real time control of a robot with a mobile camera. In *9th Int. Symp. on Industrial Robots, 1979*, pages 233–246, 1979. 79

[116] Jonathan Ho and Stefano Ermon. Generative adversarial imitation learning. *Advances in neural information processing systems*, 29, 2016. 79

[117] Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural computation*, 9(8):1735–1780, 1997. 52

[118] Yicong Hong, Cristian Rodriguez, Yuankai Qi, Qi Wu, and Stephen Gould. Language and visual entity relationship graph for agent navigation. *Advances in Neural Information Processing Systems*, 33:7685–7696, 2020. 50, 51, 63, 64

[119] Yicong Hong, Cristian Rodriguez, Qi Wu, and Stephen Gould. Sub-instruction aware vision-and-language navigation. In *EMNLP*, 2020. 26

[120] Yicong Hong, Cristian Rodriguez-Opazo, Qi Wu, and Stephen Gould. Sub-instruction aware vision-and-language navigation. *arXiv preprint arXiv:2004.02707*, 2020. 19

[121] Yicong Hong, Qi Wu, Yuankai Qi, Cristian Rodriguez-Opazo, and Stephen Gould. A recurrent vision-and-language BERT for navigation. *arXiv preprint arXiv:2011.13922*, 2021. 23, 24, 31, 33, 34, 35, 38, 39, 40, 41

[122] Yicong Hong, Qi Wu, Yuankai Qi, Cristian Rodriguez-Opazo, and Stephen Gould. Vln bert: A recurrent vision-and-language bert for navigation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 1643–1653, 2021. 19, 49, 52, 53, 57, 60, 61, 63, 64, 65, 66, 67, 68, 71, 72, 73

[123] Sepidehsadat Hosseini and Yasutaka Furukawa. Extreme floorplan reconstruction by structure-hallucinating transformer cascades. *arXiv preprint arXiv:2206.00645*, 2022. 96

[124] Thomas M Howard, Stefanie Tellex, and Nicholas Roy. A natural language planner interface for mobile manipulators. In *2014 IEEE International Conference on Robotics and Automation (ICRA)*, pages 6652–6659. IEEE, 2014. 16

[125] Yordan Hristov, Daniel Angelov, Michael Burke, Alex Lascarides, and Subramanian Ramamoorthy. Disentangled relational representations for explaining and learning from demonstration. In *CoRL*, 2019. 18, 80

[126] Yordan Hristov, Daniel Angelov, Michael Burke, Alex Lascarides, and Subramanian Ramamoorthy. Disentangled relational representations for explaining and learning from demonstration. In *Conference on Robot Learning*, pages 870–884. PMLR, 2020. 17

[127] Gao Huang, Yu Sun, Zhuang Liu, Daniel Sedra, and Kilian Q Weinberger. Deep networks with stochastic depth. In *European conference on computer vision*, pages 646–661. Springer, 2016. 60

[128] Haoshuo Huang, Vihan Jain, Harsh Mehta, Alexander Ku, Gabriel Magalhaes, Jason Baldridge, and Eugene Ie. Transferable representation learning in vision-and-language navigation. In *ICCV*, 2019. 22, 24, 33

[129] Pin-Hao Huang, Han-Hung Lee, Hwann-Tzong Chen, and Tyng-Luh Liu. Text-guided graph neural networks for referring 3d instance segmentation. In *AAAI*, volume 35, pages 1610–1618, 2021. 17

[130] Shijia Huang, Yilun Chen, Jiaya Jia, and Liwei Wang. Multi-view transformer for 3d visual grounding. In *CVPR*, 2022. 17

[131] Wenlong Huang, Pieter Abbeel, Deepak Pathak, and Igor Mordatch. Language models as zero-shot planners: Extracting actionable knowledge for embodied agents. *arXiv preprint arXiv:2201.07207*, 2022. 80

[132] Drew A Hudson and Christopher D Manning. GQA: A new dataset for real-world visual reasoning and compositional question answering. In *CVPR*, pages 6700–6709, 2019. 16

[133] 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. 15

[134] Gabriel Ilharco, Vihan Jain, Alexander Ku, Eugene Ie, and Jason Baldridge. General evaluation for instruction conditioned navigation using dynamic time warping. *arXiv preprint arXiv:1907.05446*, 2019. 19, 59

[135] Muhammad Zubair Irshad, Chih-Yao Ma, and Zsolt Kira. Hierarchical cross-modal agent for robotics vision-and-language navigation. In *ICRA*, pages 13238–13246, 2021. 18, 19, 97

[136] Andrew Jaegle, Sebastian Borgeaud, Jean-Baptiste Alayrac, Carl Doersch, Catalin Ionescu, David Ding, Skanda Koppula, Daniel Zoran, Andrew Brock, Evan Shelhamer, et al. Perceiver io: A general architecture for structured inputs & outputs. *arXiv preprint arXiv:2107.14795*, 2021. 6, 14

[137] Sandeep Jagtap, Farah Bader, Guillermo Garcia-Garcia, Hana Trollman, Tobi Fadiji, and Konstantinos Salonitis. Food logistics 4.0: Opportunities and challenges. *Logistics*, 5(1):2, 2020. 4

[138] Ayush Jain, Nikolaos Gkanatsios, Ishita Mediratta, and Katerina Fragkiadaki. Looking outside the box to ground language in 3d scenes. *arXiv preprint arXiv:2112.08879*, 2021. 17

[139] Vihan Jain, Gabriel Magalhaes, Alex Ku, Ashish Vaswani, Eugene Ie, and Jason Baldridge. Stay on the path: Instruction fidelity in vision-and-language navigation. *arXiv preprint arXiv:1905.12255*, pages 1862–1872, 2019. 49, 51, 57, 58, 59, 95

[140] Vihan Jain, Gabriel Magalhaes, Alexander Ku, Ashish Vaswani, Eugene Ie, and Jason Baldridge. Stay on the path: Instruction fidelity in vision-and-language navigation. In *ACL*, 2019. 18, 19

[141] Stephen James and Pieter Abbeel. Coarse-to-fine Q-Attention with learned path ranking. *arXiv preprint arXiv:2204.01571*, 2022. 85, 97

[142] Stephen James and Andrew J Davison. Q-Attention: Enabling efficient learning for vision-based robotic manipulation. *RA-L*, 2022. 78, 80, 81, 87

[143] Stephen James, Zicong Ma, David Rovick Arrojo, and Andrew J Davison. RLBench: The robot learning benchmark & learning environment. *RA-L*, 5(2):3019–3026, 2020. 2, 17, 78, 79, 80, 84, 96

[144] Stephen James, Kentaro Wada, Tristan Laidlow, and Andrew J Davison. Coarse-to-fine Q-Attention: Efficient learning for visual robotic manipulation via discretisation. In *CVPR*, 2022. 78

[145] Eric Jang, Alex Irpan, Mohi Khansari, Daniel Kappler, Frederik Ebert, Corey Lynch, Sergey Levine, and Chelsea Finn. BC-Z: Zero-shot task generalization with robotic imitation learning. In *CoRL*, pages 991–1002. PMLR, 2022. 17, 78, 80

[146] Yunseok Jang, Yale Song, Youngjae Yu, Youngjin Kim, and Gunhee Kim. Tgif-qa: Toward spatio-temporal reasoning in visual question answering. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2758–2766, 2017. 16

[147] Matthew Johnson, Katja Hofmann, Tim Hutton, and David Bignell. The malmo platform for artificial intelligence experimentation. In *Ijcai*, pages 4246–4247. Citeseer, 2016. 18

[148] V. Joshi, Matthew E. Peters, and Mark Hopkins. Extending a parser to distant domains using a few dozen partially annotated examples. In *ACL*, 2018. 38

[149] Dmitry Kalashnikov, Jacob Varley, Yevgen Chebotar, Benjamin Swanson, Rico Jonschkowski, Chelsea Finn, Sergey Levine, and Karol Hausman. MT-Opt: Continuous multi-task robotic reinforcement learning at scale. *arXiv preprint arXiv:2104.08212*, 2021. 79

[150] Aishwarya Kamath, Peter Anderson, Su Wang, Jing Yu Koh, Alexander Ku, Austin Waters, Yinfei Yang, Jason Baldridge, and Zarana Parekh. A new path: Scaling vision-and-language navigation with synthetic instructions and imitation learning. *arXiv preprint arXiv:2210.03112*, 2022. 96

[151] Sertac Karaman, Matthew R Walter, Alejandro Perez, Emilio Frazzoli, and Seth Teller. Anytime motion planning using the rrt. In *2011 IEEE International Conference on Robotics and Automation*, pages 1478–1483. IEEE, 2011. 14

[152] Artúr István Károly, Péter Galambos, József Kuti, and Imre J Rudas. Deep learning in robotics: Survey on model structures and training strategies. *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, 51(1):266–279, 2020. 79

[153] Sahar Kazemzadeh, Vicente Ordonez, Mark Matten, and Tamara Berg. ReferItGame: Referring to objects in photographs of natural scenes. In *EMNLP*, 2014. 16, 24

[154] Michał Kempka, Marek Wydmuch, Grzegorz Runc, Jakub Toczek, and Wojciech Jaśkowski. Vizdoom: A doom-based ai research platform for visual reinforcement learning. In *2016 IEEE conference on computational intelligence and games (CIG)*, pages 1–8. IEEE, 2016. 18

[155] 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. 19

[156] 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. 80

[157] Wonjae Kim, Bokyung Son, and Ildoo Kim. Vilt: Vision-and-language transformer without convolution or region supervision. In *Proceedings of the 38th International Conference on Machine Learning*, pages 5583–5594, 2021. 52

[158] Thomas Kollar, Stefanie Tellex, Deb Roy, and Nicholas Roy. Toward understanding natural language directions. In *2010 5th ACM/IEEE International Conference on Human-Robot Interaction (HRI)*, pages 259–266, 2010. 16

[159] Eric Kolve, Roozbeh Mottaghi, Winson Han, Eli VanderBilt, Luca Weihs, Alvaro Herrasti, Daniel Gordon, Yuke Zhu, Abhinav Kumar Gupta, and Ali Farhadi. AI2-THOR: An interactive 3d environment for visual ai. *ArXiv*, abs/1712.05474, 2017. 18

[160] Chen Kong, Dahua Lin, Mohit Bansal, Raquel Urtasun, and Sanja Fidler. What are you talking about? text-to-image coreference. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 3558–3565, 2014. 16

[161] Gert Kootstra, Mila Popović, Jimmy Alison Jørgensen, Danica Kragic, Henrik Gordon Petersen, and Norbert Krüger. Visgrab: A benchmark for vision-based grasping. *Paladyn*, 3(2):54–62, 2012. 17

[162] Mario Köppen. The curse of dimensionality. In *5th online world conference on soft computing in industrial applications (WSC5)*, volume 1, pages 4–8, 2000. 6

[163] Jacob Krantz, Shurjo Banerjee, Wang Zhu, Jason Corso, Peter Anderson, Stefan Lee, and Jesse Thomason. Iterative vision-and-language navigation. *arXiv preprint arXiv:2210.03087*, 2022. 96

[164] Jacob Krantz, Erik Wijmans, Arjun Majumdar, Dhruv Batra, and Stefan Lee. Beyond the nav-graph: Vision-and-language navigation in continuous environments. In *European Conference on Computer Vision*, pages 104–120. Springer, 2020. 18, 74, 97

[165] Jacob Krantz, Erik Wijmans, Arjun Majumdar, Dhruv Batra, and Stefan Lee. Beyond the nav-graph: Vision-and-language navigation in continuous environments. In *ECCV*, 2020. 23

[166] 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. *IJCV*, 123(1):32–73, 2017. 16

[167] 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. *IJCV*, 123:32–73, 2017. 16, 28

[168] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. *Communications of the ACM*, 60(6):84–90, 2017. 13, 16

[169] Alexander Ku, Peter Anderson, Roma Patel, Eugene Ie, and Jason Baldridge. Room-across-room: Multilingual vision-and-language navigation with dense spatiotemporal grounding. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 4392–4412, 2020. 18, 50, 51, 58, 59, 66, 67

[170] Alexander Ku, Peter Anderson, Roma Patel, Eugene Ie, and Jason Baldridge. Room-Across-Room: Multilingual vision-and-language navigation with dense spatiotemporal grounding. In *EMNLP*, 2020. 23, 95

[171] Serdar Kucuk and Zafer Bingul. *Robot kinematics: Forward and inverse kinematics*. INTECH Open Access Publisher London, UK, 2006. 14

[172] Yann Labbé, Justin Carpentier, Mathieu Aubry, and Josef Sivic. Cosypose: Consistent multi-view multi-object 6d pose estimation. In *ECCV*, pages 574–591. Springer, 2020. 79

[173] Yann Labbé, Sergey Zagoruyko, Igor Kalevatykh, Ivan Laptev, Justin Carpentier, Mathieu Aubry, and Josef Sivic. Monte-carlo tree search for efficient visually guided rearrangement planning. *IEEE Robotics and Automation Letters*, 5(2):3715–3722, 2020. 15

[174] Zhenzhong Lan, Mingda Chen, Sebastian Goodman, Kevin Gimpel, Piyush Sharma, and Radu Soricut. Albert: A lite bert for self-supervised learning of language representations. *arXiv preprint arXiv:1909.11942*, 2019. 14

[175] Federico Landi, Lorenzo Baraldi, Marcella Cornia, Massimiliano Corsini, and Rita Cucchiara. Perceive, transform, and act: Multi-modal attention networks for vision-and-language navigation. *arXiv preprint arXiv:1911.12377*, 2019. 51, 64

[176] Yann LeCun. A path towards autonomous machine intelligence version 0.9. 2, 2022-06-27. *Open Review*, 2022. 1

[177] Yann LeCun, Léon Bottou, Yoshua Bengio, and Patrick Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998. 13, 16

[178] Alex X Lee, Sergey Levine, and Pieter Abbeel. Learning visual servoing with deep features and fitted q-iteration. In *ICLR*, 2017. 79

[179] Jie Lei, Licheng Yu, Mohit Bansal, and Tamara L Berg. Tvqa: Localized, compositional video question answering. In *Empirical Methods in Natural Language Processing*, 2018. 16

[180] Jürgen Leitner, Adam W Tow, Niko Sünderhauf, Jake E Dean, Joseph W Durham, Matthew Cooper, Markus Eich, Christopher Lehnert, Ruben Mangels, Christopher McCool, et al. The acrv picking benchmark: A robotic shelf picking benchmark to foster reproducible research. In *2017 IEEE International Conference on Robotics and Automation (ICRA)*, pages 4705–4712. IEEE, 2017. 17

[181] Beatriz León, Stefan Ulbrich, Rosen Diankov, Gustavo Puche, Markus Przybylski, Antonio Morales, Tamim Asfour, Sami Moisio, Jeannette Bohg, James Kuffner, et al. Opengrasp: a toolkit for robot grasping simulation. In *International conference on simulation, modeling, and programming for autonomous robots*, pages 109–120. Springer, 2010. 17

[182] Stephen C Levinson. Studying spatial conceptualization across cultures: Anthropology and cognitive science. *Ethos*, 26(1):7–24, 1998. 15

[183] Chengshu Li, Cem Gokmen, Gabrael Levine, Roberto Martín-Martín, Sanjana Srivastava, Chen Wang, Josiah Wong, Ruohan Zhang, Michael Lingelbach, Jiankai Sun, et al. Behavior-1k: A benchmark for embodied ai with 1,000 everyday activities and realistic simulation. In *6th Annual Conference on Robot Learning*, 2022. 18

[184] Chengshu Li, Fei Xia, Roberto Martín-Martín, Michael Lingelbach, Sanjana Srivastava, Bokui Shen, Kent Elliott Vainio, Cem Gokmen, Gokul Dharan, Tanish Jain, et al. igibson 2.0: Object-centric simulation for robot learning of everyday household tasks. In *Conference on Robot Learning*, pages 455–465. PMLR, 2022. 18

[185] Jialu Li, Hao Tan, and Mohit Bansal. CLEAR: Improving vision-language navigation with cross-lingual, environment-agnostic representations. *arXiv preprint arXiv:2207.02185*, 2022. 19, 96

[186] Jialu Li, Hao Tan, and Mohit Bansal. Envedit: Environment editing for vision-and-language navigation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 15407–15417, 2022. 19, 96

[187] Luyao Li, Xiaoyi He, Gregory A Keoleian, Hyung Chul Kim, Robert De Kleine, Timothy J Wallington, and Nicholas J Kemp. Life cycle greenhouse gas emissions

for last-mile parcel delivery by automated vehicles and robots. *Environmental Science & Technology*, 55(16):11360–11367, 2021. 97

[188] Xiujun Li, Chunyuan Li, Qiaolin Xia, Yonatan Bisk, Asli Celikyilmaz, Jianfeng Gao, Noah A Smith, and Yejin Choi. Robust navigation with language pretraining and stochastic sampling. In *EMNLP*, pages 1494–1499, 2019. 19, 22, 39, 41

[189] Xiujun Li, Chunyuan Li, Qiaolin Xia, Yonatan Bisk, Asli Celikyilmaz, Jianfeng Gao, Noah A Smith, and Yejin Choi. Robust navigation with language pretraining and stochastic sampling. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 1494–1499, 2019. 50, 51, 63

[190] Xiujun Li, Xi Yin, Chunyuan Li, Pengchuan Zhang, Xiaowei Hu, Lei Zhang, Lijuan Wang, Houdong Hu, Li Dong, Furu Wei, et al. Oscar: Object-semantics aligned pre-training for vision-language tasks. In *ECCV*, pages 121–137. Springer, 2020. 22, 24, 35, 52

[191] Anne-Laure Ligozat, Julien Lefèvre, Aurélie Bugeau, and Jacques Combaz. Unraveling the hidden environmental impacts of ai solutions for environment life cycle assessment of ai solutions. *Sustainability*, 14(9):5172, 2022. 97

[192] Bingqian Lin, Yi Zhu, Zicong Chen, Xiwen Liang, Jianzhuang Liu, and Xiaodan Liang. Adapt: Vision-language navigation with modality-aligned action prompts. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 15396–15406, 2022. 19

[193] Chuang Lin, Yi Jiang, Jianfei Cai, Lizhen Qu, Gholamreza Haffari, and Zehuan Yuan. Multimodal transformer with variable-length memory for vision-and-language navigation. *arXiv preprint arXiv:2111.05759*, 2021. 19

[194] Tsung-Yi Lin, Piotr Dollár, Ross Girshick, Kaiming He, Bharath Hariharan, and Serge Belongie. Feature pyramid networks for object detection. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2117–2125, 2017. 16

[195] 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 *European conference on computer vision*, pages 740–755. Springer, 2014. 13

[196] Xiangru Lin, Guanbin Li, and Yizhou Yu. Scene-intuitive agent for remote embodied visual grounding. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 7036–7045, 2021. 50, 51, 67

[197] Chong Liu, Fengda Zhu, Xiaojun Chang, Xiaodan Liang, Zongyuan Ge, and Yi-Dong Shen. Vision-language navigation with random environmental mixup. In *ICCV*, pages 1644–1654, 2021. 19

[198] Haolin Liu, Anran Lin, Xiaoguang Han, Lei Yang, Yizhou Yu, and Shuguang Cui. Refer-it-in-RGBD: A bottom-up approach for 3d visual grounding in rgbd images. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 6032–6041, 2021. 16

[199] Li Liu, Wanli Ouyang, Xiaogang Wang, Paul Fieguth, Jie Chen, Xinwang Liu, and Matti Pietikäinen. Deep learning for generic object detection: A survey. *International journal of computer vision*, 128(2):261–318, 2020. 13

[200] Shikun Liu, Stephen James, Andrew J Davison, and Edward Johns. Auto-Lambda: Disentangling dynamic task relationships. *TMLR*, 2022. 17, 78, 79, 80, 84, 86, 87, 88

[201] Weiyu Liu, Chris Paxton, Tucker Hermans, and Dieter Fox. StructFormer: Learning spatial structure for language-guided semantic rearrangement of novel objects. *ICRA*, 2021. 80

[202] Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. In *7th International Conference on Learning Representations*, 2019. 60

[203] Cewu Lu, Ranjay Krishna, Michael Bernstein, and Li Fei-Fei. Visual relationship detection with language priors. In *European conference on computer vision*, pages 852–869. Springer, 2016. 16

[204] Jiasen Lu, Dhruv Batra, Devi Parikh, and Stefan Lee. ViLBERT: Pretraining task-agnostic visiolinguistic representations for vision-and-language tasks. In *NeurIPS*, volume 32, 2019. 14, 19, 22, 24, 31, 33, 52, 55, 78, 80

[205] Jiasen Lu, Vedanuj Goswami, Marcus Rohrbach, Devi Parikh, and Stefan Lee. 12-in-1: Multi-task vision and language representation learning. In *CVPR*, 2020. 22

[206] John A Lucy. Linguistic relativity. *Annual review of anthropology*, pages 291–312, 1997. 15

[207] Jelena Luketina, Nantas Nardelli, Gregory Farquhar, Jakob N Foerster, Jacob Andreas, Edward Grefenstette, Shimon Whiteson, and Tim Rocktäschel. A survey of reinforcement learning informed by natural language. In *IJCAI*, 2019. 19, 97

[208] Huaishao Luo, Lei Ji, Ming Zhong, Yang Chen, Wen Lei, Nan Duan, and Tianrui Li. Clip4clip: An empirical study of clip for end to end video clip retrieval. *arXiv preprint arXiv:2104.08860*, 2021. 16

[209] Junyu Luo, Jiahui Fu, Xianghao Kong, Chen Gao, Haibing Ren, Hao Shen, Huaxia Xia, and Si Liu. 3d-sps: Single-stage 3d visual grounding via referred point progressive selection. In *CVPR*, 2022. 17

[210] Corey Lynch, Mohi Khansari, Ted Xiao, Vikash Kumar, Jonathan Tompson, Sergey Levine, and Pierre Sermanet. Learning latent plans from play. In *CoRL*, pages 1113–1132. PMLR, 2020. 17, 79, 80

[211] Corey Lynch and Pierre Sermanet. Language conditioned imitation learning over unstructured data. *RSS*, 2021. 17, 78, 80

[212] 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 *ICLR*, 2019. 19, 24, 38, 39, 50, 51, 67

[213] Chih-Yao Ma, Zuxuan Wu, Ghassan AlRegib, Caiming Xiong, and Zsolt Kira. The Regretful Agent: Heuristic-aided navigation through progress estimation. In *CVPR*, pages 6732–6740, 2019. 19, 24

[214] Yiwei Ma, Guohai Xu, Xiaoshuai Sun, Ming Yan, Ji Zhang, and Rongrong Ji. X-clip: End-to-end multi-grained contrastive learning for video-text retrieval. *arXiv preprint arXiv:2207.07285*, 2022. 16

[215] Arjun Majumdar, Ayush Shrivastava, Stefan Lee, Peter Anderson, Devi Parikh, and Dhruv Batra. Improving vision-and-language navigation with image-text pairs from the web. In *ECCV*, 2020. 22, 23, 24, 31, 33, 34, 35, 36, 37, 38, 39

[216] Arjun Majumdar, Ayush Shrivastava, Stefan Lee, Peter Anderson, Devi Parikh, and Dhruv Batra. Improving vision-and-language navigation with image-text pairs from the web. In *European Conference on Computer Vision*, pages 259–274. Springer, 2020. 52, 54, 55

[217] Cecilia Mauceri, Martha Palmer, and Christoffer Heckman. Sun-spot: an rgb-d dataset with spatial referring expressions. In *Proceedings of the IEEE/CVF International Conference on Computer Vision Workshops*, pages 0–0, 2019. 16

[218] Oier Mees and Wolfram Burgard. Composing pick-and-place tasks by grounding language. In *ISER*, 2021. 78, 80

[219] Oier Mees, Lukas Hermann, Erick Rosete-Beas, and Wolfram Burgard. CALVIN: A benchmark for language-conditioned policy learning for long-horizon robot manipulation tasks. *RA-L*, 2022. 17, 78, 80

[220] Antoine Miech, Dimitri Zhukov, Jean-Baptiste Alayrac, Makarand Tapaswi, Ivan Laptev, and Josef Sivic. HowTo100M: Learning a text-video embedding by watching hundred million narrated video clips. In *ICCV*, 2019. 5, 24

[221] Milot Mirdita, Konstantin Schütze, Yoshitaka Moriwaki, Lim Heo, Sergey Ovchinnikov, and Martin Steinegger. Colabfold: making protein folding accessible to all. *Nature Methods*, pages 1–4, 2022. 1

[222] Piotr Mirowski, Andras Banki-Horvath, Keith Anderson, Denis Teplyashin, Karl Moritz Hermann, Mateusz Malinowski, Matthew Koichi Grimes, Karen Simonyan, Koray Kavukcuoglu, Andrew Zisserman, et al. The streetlearn environment and dataset. *arXiv preprint arXiv:1903.01292*, 2019. 18

[223] Dipendra Misra, John Langford, and Yoav Artzi. Mapping instructions and visual observations to actions with reinforcement learning. *ACL*, 2017. 18, 80

[224] Dipendra K Misra, Jaeyong Sung, Kevin Lee, and Ashutosh Saxena. Tell me dave: Context-sensitive grounding of natural language to manipulation instructions. *The International Journal of Robotics Research*, 35(1-3):281–300, 2016. 17

[225] Volodymyr Mnih, Adria Puigdomenech Badia, Mehdi Mirza, Alex Graves, Timothy Lillicrap, Tim Harley, David Silver, and Koray Kavukcuoglu. Asynchronous methods for deep reinforcement learning. In *International conference on machine learning*, pages 1928–1937. PMLR, 2016. 51, 57

[226] Abhinav Moudgil, Arjun Majumdar, Harsh Agrawal, Stefan Lee, and Dhruv Batra. Soat: A scene-and object-aware transformer for vision-and-language navigation. *NeurIPS*, 34, 2021. 19

[227] Omar Mubin, Catherine J Stevens, Suleman Shahid, Abdullah Al Mahmud, and Jian-Jie Dong. A review of the applicability of robots in education. *Journal of Technology in Education and Learning*, 1(209-0015):13, 2013. 4

[228] Michael Murray and Maya Cakmak. Following natural language instructions for household tasks with landmark guided search and reinforced pose adjustment. *IEEE Robotics and Automation Letters*, 2022. 97

[229] David Mytton. Hiding greenhouse gas emissions in the cloud. *Nature Climate Change*, 10(8):701–701, 2020. 97

[230] Suraj Nair, Eric Mitchell, Kevin Chen, Silvio Savarese, Chelsea Finn, et al. Learning language-conditioned robot behavior from offline data and crowd-sourced annotation. In *CoRL*, pages 1303–1315. PMLR, 2022. 78, 80

[231] Khanh Nguyen and Hal Daumé III. Help, Anna! Visual navigation with natural multimodal assistance via retrospective curiosity-encouraging imitation learning. In *ACL*, 2019. 18, 23

[232] Khanh Nguyen, Debadeepta Dey, Chris Brockett, and Bill Dolan. Vision-based navigation with language-based assistance via imitation learning with indirect intervention. In *CVPR*, 2019. 18, 23

[233] Thao Nguyen, Nakul Gopalan, Roma Patel, Matt Corsaro, Ellie Pavlick, and Stefanie Tellex. Robot object retrieval with contextual natural language queries. *RSS*, 2020. 19, 80

[234] Lucas Nogueira. Comparative analysis between gazebo and v-rep robotic simulators. *Seminario Interno de Cognicao Artificial-SICA*, 2014(5):2, 2014. 8

[235] Edwin Olson. Apriltag: A robust and flexible visual fiducial system. In *ICRA*, 2011. 92

[236] Vicente Ordonez, Girish Kulkarni, and Tamara Berg. Im2Text: Describing images using 1 million captioned photographs. In *NeurIPS*, 2011. 22, 24

[237] 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. 18

[238] Ariadni Papadopoulou, Niraj S Kumar, Anne Vanhoestenberghe, and Nader K Francis. Environmental sustainability in robotic and laparoscopic surgery: Systematic review. *British Journal of Surgery*, 109(10):921–932, 2022. 97

[239] Emilio Parisotto, Francis Song, Jack Rae, Razvan Pascanu, Caglar Gulcehre, Siddhant Jayakumar, Max Jaderberg, Raphael Lopez Kaufman, Aidan Clark, Seb Noury, et al. Stabilizing transformers for reinforcement learning. In *International Conference on Machine Learning*, pages 7487–7498. PMLR, 2020. 50, 55

[240] Alexander Pashevich, Danijar Hafner, James Davidson, Rahul Sukthankar, and Cordelia Schmid. Modulated policy hierarchies. *NeurIPS Deep RL Workshop*, 2018. 79

[241] Alexander Pashevich, Cordelia Schmid, and Chen Sun. Episodic transformer for vision-and-language navigation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 15942–15952, 2021. 19, 52, 54

[242] Alexander Pashevich, Cordelia Schmid, and Chen Sun. Episodic transformer for vision-and-language navigation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 15942–15952, 2021. 80, 83, 91

[243] Rohan Paul, Jacob Arkin, Nicholas Roy, and Thomas M Howard. Efficient grounding of abstract spatial concepts for natural language interaction with robot manipulators. In *Robotics: Science and Systems Foundation*, 2016. 80

[244] Ethan Perez, Florian Strub, Harm De Vries, Vincent Dumoulin, and Aaron Courville. FiLM: Visual reasoning with a general conditioning layer. In *AAAI*, 2018. 78

[245] Jan Peters, Sethu Vijayakumar, and Stefan Schaal. Reinforcement learning for humanoid robotics. In *Proceedings of the third IEEE-RAS international conference on humanoid robots*, pages 1–20, 2003. 79

[246] Vladimír Petrík, Makarand Tapaswi, Ivan Laptev, and Josef Sivic. Learning object manipulation skills via approximate state estimation from real videos. In *CoRL*, 2020. 80, 96

[247] Rolf Pfeifer and Josh Bongard. *How the body shapes the way we think: a new view of intelligence*. MIT press, 2006. 7

[248] Pedro Pinheiro and Ronan Collobert. Recurrent convolutional neural networks for scene labeling. In *International conference on machine learning*, pages 82–90. PMLR, 2014. 16

[249] Aaron Louie Putterman, Kevin Lu, Igor Mordatch, and Pieter Abbeel. Pretraining for language conditioned imitation with transformers. In *NeurIPS*, 2021. 80

[250] Yuankai Qi, Zizheng Pan, Shengping Zhang, Anton van den Hengel, and Qi Wu. Object-and-action aware model for visual language navigation. In *ECCV*, pages 23–28. Springer, 2020. 24

[251] 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 *CVPR*, pages 9982–9991, 2020. 18, 19, 23, 25, 26, 33, 35, 38, 50, 51, 58, 59, 67, 95

[252] Yanyuan Qiao, Chaorui Deng, and Qi Wu. Referring expression comprehension: A survey of methods and datasets. *IEEE Transactions on Multimedia*, 23:4426–4440, 2020. 16

[253] Yanyuan Qiao, Yuankai Qi, Yicong Hong, Zheng Yu, Peng Wang, and Qi Wu. Hop: History-and-order aware pre-training for vision-and-language navigation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 15418–15427, 2022. 95

[254] 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 *ICML*, pages 8748–8763. PMLR, 2021. 6, 19, 78, 80, 81

[255] 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 *Proceedings of the 38th International Conference on Machine Learning*, pages 8748–8763. PMLR, 2021. 24, 66

[256] Marc'Aurelio Ranzato, Sumit Chopra, Michael Auli, and Wojciech Zaremba. Sequence level training with recurrent neural networks. *International Conference on Learning Representations*, 2016. 50

[257] Scott Reed, Konrad Zolna, Emilio Parisotto, Sergio Gomez Colmenarejo, Alexander Novikov, Gabriel Barth-Maron, Mai Gimenez, Yury Sulsky, Jackie Kay, Jost Tobias Springenberg, et al. A generalist agent. *arXiv preprint arXiv:2205.06175*, 2022. 80

[258] Sebastian Risi and Mike Preuss. From chess and atari to starcraft and beyond: How game ai is driving the world of ai. *KI-Künstliche Intelligenz*, 34(1):7–17, 2020. 1

[259] Junha Roh, Karthik Desingh, Ali Farhadi, and Dieter Fox. Languagerefer: Spatial-language model for 3d visual grounding. In *CoRL*, pages 1046–1056. PMLR, 2021. 17

[260] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. UNet: Convolutional networks for biomedical image segmentation. In *International Conference on Medical image computing and computer-assisted intervention*, pages 234–241. Springer, 2015. 13, 78, 81

[261] S Rooban, Shaik Dilawar Suraj, Shaik Babji Vali, and Nagandla Dhanush. Coppeliasim: Adaptable modular robot and its different locomotions simulation framework. *Materials Today: Proceedings*, 2021. 17

[262] Stéphane Ross, Geoffrey Gordon, and Drew Bagnell. A reduction of imitation learning and structured prediction to no-regret online learning. In *Proceedings of the fourteenth international conference on artificial intelligence and statistics*, pages 627–635. JMLR Workshop and Conference Proceedings, 2011. 6, 17, 50

[263] Edward Sapir. The status of linguistics as a science. *Language*, pages 207–214, 1929. 15

[264] Barbara Saunders. Revisiting basic color terms. *Journal of the Royal Anthropological Institute*, 6(1):81–99, 2000. 15

[265] Nikolay Savinov, Alexey Dosovitskiy, and Vladlen Koltun. Semi-parametric topological memory for navigation. *International Conference on Learning Representations*, 2018. 52

[266] Manolis Savva, Angel X Chang, Alexey Dosovitskiy, Thomas Funkhouser, and Vladlen Koltun. Minos: Multimodal indoor simulator for navigation in complex environments. *arXiv preprint arXiv:1712.03931*, 2017. 18

[267] 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 *ICCV*, pages 9339–9347, 2019. 18

[268] Johanna Seibt, Malene Flensborg Damholdt, and Christina Vestergaard. Five principles of integrative social robotics. In *Robophilosophy/TRANSOR*, pages 28–42, 2018. 97

[269] Dhruv Shah, Blazej Osinski, Brian Ichter, and Sergey Levine. Lm-nav: Robotic navigation with large pre-trained models of language, vision, and action. *arXiv preprint arXiv:2207.04429*, 2022. 19

[270] Lin Shao, Toki Migimatsu, Qiang Zhang, Karen Yang, and Jeannette Bohg. Concept2robot: Learning manipulation concepts from instructions and human demonstrations. *The International Journal of Robotics Research*, 40(12-14):1419–1434, 2021. 78

[271] Piyush Sharma, Nan Ding, Sebastian Goodman, and Radu Soricut. Conceptual Captions: A cleaned, hypernymed, image alt-text dataset for automatic image captioning. In *ACL*, 2018. 22, 24, 25, 26, 33

[272] 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. 18

[273] Sheng Shen, Liunian Harold Li, Hao Tan, Mohit Bansal, Anna Rohrbach, Kai-Wei Chang, Zhewei Yao, and Kurt Keutzer. How much can CLIP benefit vision-and-language tasks? *arXiv preprint arXiv:2107.06383*, 2021. 19

[274] Marlena H Shin, Jaye McLaren, Alvin Ramsey, Jennifer L Sullivan, and Lauren Moo. Improving a mobile telepresence robot for people with alzheimer disease and related dementias: Semistructured interviews with stakeholders. *JMIR aging*, 5(2):e32322, 2022. 4

[275] Mohit Shridhar, Lucas Manuelli, and Dieter Fox. CLIPort: What and where pathways for robotic manipulation. In *CoRL*, pages 894–906. PMLR, 2022. 17, 18, 78, 80, 90

[276] 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 *CVPR*, pages 10740–10749, 2020. 18, 23, 51

[277] Ayush Shrivastava, Karthik Gopalakrishnan, Yang Liu, Robinson Piramuthu, Gokhan Tür, Devi Parikh, and Dilek Hakkani-Tür. Visitron: Visual semantics-aligned interactively trained object-navigator. *arXiv preprint arXiv:2105.11589*, 2021. 65

[278] P.R. Shukla, J. Skea, R. Slade, A. Al Khourdajie, R. van Diemen, D. McCollum, M. Pathak, S. Some, P. Vyas, R. Fradera, M. Belkacemi, A. Hasija, G. Lisboa, S. Luz, and J. Malley. Climate change 2022: Climate change 2022: Mitigation of climate change. contribution of working group III to the sixth assessment report of the intergovernmental panel on climate change. *IPCC Sixth Assessment Report*, 2022. 97

[279] Andrew Silva, Nina Moorman, William Silva, Zulfiqar Zaidi, Nakul Gopalan, and Matthew Gombolay. Lancon-learn: Learning with language to enable generalization in multi-task manipulation. *IEEE Robotics and Automation Letters*, 7(2):1635–1642, 2021. 88, 90

[280] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014. 13

[281] Shagun Sodhani, Amy Zhang, and Joelle Pineau. Multi-task reinforcement learning with context-based representations. In *International Conference on Machine Learning*, pages 9767–9779. PMLR, 2021. 79

[282] Shawn Squire, Stefanie Tellex, Dilip Arumugam, and Lei Yang. Grounding english commands to reward functions. In *Robotics: Science and Systems*, 2015. 16

[283] 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. PMLR, 2022. 18

[284] Trevor Standley, Amir Zamir, Dawn Chen, Leonidas Guibas, Jitendra Malik, and Silvio Savarese. Which tasks should be learned together in multi-task learning? In *International Conference on Machine Learning*, pages 9120–9132. PMLR, 2020. 79

[285] Elias Stengel-Eskin, Andrew Hundt, Zhuohong He, Aditya Murali, Nakul Gopalan, Matthew Gombolay, and Gregory Hager. Guiding multi-step rearrangement tasks with natural language instructions. In *Conference on Robot Learning*, pages 1486–1501. PMLR, 2021. 80

[286] Frode J Strømnes. Memory models and language comprehension. *Scandinavian Journal of Psychology*, 15(1):26–32, 1974. 15

[287] Robin Strudel, Ricardo Garcia, Ivan Laptev, and Cordelia Schmid. Segmenter: Transformer for semantic segmentation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 7262–7272, 2021. 14

[288] Robin Strudel, Alexander Pashevich, Igor Kalevatykh, Ivan Laptev, and Cordelia Schmid. Learning to combine primitive skills: A step towards versatile robotic manipulation. In *ICRA*, pages 4637–4643. IEEE, 2020. 15, 17, 79, 80

[289] Robin Strudel, Ricardo Garcia Pinel, Justin Carpentier, Jean-Paul Laumond, Ivan Laptev, and Cordelia Schmid. Learning obstacle representations for neural motion planning. In *Conference on Robot Learning*, pages 355–364. PMLR, 2021. 15

[290] Weijie Su, Xizhou Zhu, Yue Cao, Bin Li, Lewei Lu, Furu Wei, and Jifeng Dai. Vl-bert: Pre-training of generic visual-linguistic representations. In *ICLR*, 2019. 22

[291] Sanjay Subramanian, Will Merrill, Trevor Darrell, Matt Gardner, Sameer Singh, and Anna Rohrbach. Reclip: A strong zero-shot baseline for referring expression comprehension. In *ACL*, pages 5198–5215, 2022. 17

[292] Richard S Sutton and Andrew G Barto. *Reinforcement learning: An introduction*. MIT press, 2018. 15, 19, 20, 50, 79

[293] Christian Szegedy, Wei Liu, Yangqing Jia, Pierre Sermanet, Scott Reed, Dragomir Anguelov, Dumitru Erhan, Vincent Vanhoucke, and Andrew Rabinovich. Going deeper with convolutions. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1–9, 2015. 13

[294] Hao Tan and Mohit Bansal. Lxmert: Learning cross-modality encoder representations from transformers. In *EMNLP*, pages 5103–5114, 2019. 14, 24, 35, 52, 80

[295] Hao Tan, Licheng Yu, and Mohit Bansal. Learning to navigate unseen environments: Back translation with environmental dropout. In *NAACL*, 2019. 19, 20, 24, 26, 33, 37, 38, 39, 41, 42

[296] Hao Tan, Licheng Yu, and Mohit Bansal. Learning to navigate unseen environments: Back translation with environmental dropout. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 2610–2621, 2019. 50, 51, 52, 53, 62, 63, 64, 65, 66, 68

[297] Sinan Tan, Weilai Xiang, Huaping Liu, Di Guo, and Fuchun Sun. Multi-agent embodied question answering in interactive environments. In *European Conference on Computer Vision*, pages 663–678. Springer, 2020. 18

[298] Makarand Tapaswi, Yukun Zhu, Rainer Stiefelhagen, Antonio Torralba, Raquel Urtasun, and Sanja Fidler. Movieqa: Understanding stories in movies through question-answering. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4631–4640, 2016. 13, 16

[299] DeepMind Interactive Agents Team, Josh Abramson, Arun Ahuja, Arthur Brussee, Federico Carnevale, Mary Cassin, Felix Fischer, Petko Georgiev, Alex Goldin, Tim Harley, et al. Creating multimodal interactive agents with imitation and self-supervised learning. *arXiv preprint arXiv:2112.03763*, 2021. 80

[300] Stefanie Tellex, Thomas Kollar, Steven Dickerson, Matthew R Walter, Ashis Gopal Banerjee, Seth Teller, and Nicholas Roy. Understanding natural language commands for robotic navigation and mobile manipulation. In *Twenty-Fifth AAAI Conference on Artificial Intelligence*, 2011. 15, 17, 80

[301] M Therasa and G Mathivanan. Survey of machine reading comprehension models and its evaluation metrics. In *2022 6th International Conference on Computing Methodologies and Communication (ICCMC)*, pages 1006–1013. IEEE, 2022. 1

[302] Jesse Thomason, Michael Murray, Maya Cakmak, and Luke Zettlemoyer. Vision-and-dialog navigation. In *CoRL*, 2020. 18, 23, 95

[303] Jesse Thomason, Michael Murray, Maya Cakmak, and Luke Zettlemoyer. Vision-and-dialog navigation. In *Conference on Robot Learning*, pages 394–406. PMLR, 2020. 50, 51, 58, 59

[304] Sebastian Thrun, Wolfram Burgard, and Dieter Fox. *Probabilistic robotics*. MIT press, 2005. 15, 19

[305] Sebastian Thrun and Joseph O'Sullivan. Discovering structure in multiple learning tasks: The TC algorithm. In *ICML*, volume 96, pages 489–497, 1996. 79

[306] Long Tian, Zhigang Tu, Dejun Zhang, Jun Liu, Baoxin Li, and Junsong Yuan. Unsupervised learning of optical flow with cnn-based non-local filtering. *IEEE Transactions on Image Processing*, 29:8429–8442, 2020. 13

[307] Horaţiu George Todoran and Markus Bader. Extended kalman filter (ekf)-based local slam in dynamic environments: A framework. In *Advances in Robot Design and Intelligent Control*, pages 459–469. Springer, 2016. 15

[308] Faraz Torabi, Garrett Warnell, and Peter Stone. Behavioral cloning from observation. In *Proceedings of the 27th International Joint Conference on Artificial Intelligence*, pages 4950–4957, 2018. 6, 17, 79

[309] Saran Tunyasuvunakool, Alistair Muldal, Yotam Doron, Siqi Liu, Steven Bohez, Josh Merel, Tom Erez, Timothy Lillicrap, Nicolas Heess, and Yuval Tassa. dm_control: Software and tasks for continuous control. *Software Impacts*, 6:100022, 2020. 80

[310] Gul Varol, Duygu Ceylan, Bryan Russell, Jimei Yang, Ersin Yumer, Ivan Laptev, and Cordelia Schmid. Bodynet: Volumetric inference of 3d human body shapes. In *Proceedings of the European conference on computer vision (ECCV)*, pages 20–36, 2018. 13

[311] Arun Balajee Vasudevan, Dengxin Dai, and Luc Van Gool. Talk2nav: Long-range vision-and-language navigation with dual attention and spatial memory. *International Journal of Computer Vision*, 129(1):246–266, 2021. 18

[312] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *NeurIPS*, volume 30, pages 5998–6008, 2017. 14, 22, 24, 51, 83

[313] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. *Advances in neural information processing systems*, 30, 2017. 17, 80

[314] Oriol Vinyals, Meire Fortunato, and Navdeep Jaitly. Pointer networks. *arXiv preprint arXiv:1506.03134*, 2015. 30

[315] Oriol Vinyals, Alexander Toshev, Samy Bengio, and Dumitru Erhan. Show and tell: Lessons learned from the 2015 MSCOCO image captioning challenge. *PAMI*, 39:652–663, 2016. 24

[316] Hanqing Wang, Wenguan Wang, Wei Liang, Caiming Xiong, and Jianbing Shen. Structured scene memory for vision-language navigation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 8455–8464, 2021. 19, 50, 52

[317] Hu Wang, Qi Wu, and Chunhua Shen. Soft expert reward learning for vision-and-language navigation. In *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part IX 16*, pages 126–141. Springer, 2020. 20, 51

[318] Hu Wang, Qi Wu, and Chunhua Shen. Soft expert reward learning for vision-and-language navigation. In *ECCV*, 2020. 24

[319] Ke-Jyun Wang, Yun-Hsuan Liu, Hung-Ting Su, Jen-Wei Wang, Yu-Siang Wang, Winston Hsu, and Wen-Chin Chen. Ocid-ref: A 3d robotic dataset with embodied language for clutter scene grounding. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 5333–5338, 2021. 16

[320] Liwei Wang, Yin Li, and Svetlana Lazebnik. Learning deep structure-preserving image-text embeddings. In *CVPR*, 2016. 24

[321] Xin Wang, Qiuyuan Huang, Asli Celikyilmaz, Jianfeng Gao, Dinghan Shen, Yuan-Fang Wang, William Yang Wang, and Lei Zhang. Reinforced cross-modal matching and self-supervised imitation learning for vision-language navigation. In *CVPR*, pages 6629–6638, 2019. 20, 24, 38, 39, 50, 51, 52, 64, 67

[322] Xin Wang, Wenhan Xiong, Hongmin Wang, and William Yang Wang. Look before you leap: Bridging model-free and model-based reinforcement learning for planned-ahead vision-and-language navigation. In *ECCV*, pages 37–53, 2018. 24

[323] Xin Eric Wang, Vihan Jain, Eugene Ie, William Yang Wang, Zornitsa Kozareva, and Sujith Ravi. Environment-agnostic multitask learning for natural language grounded navigation. In *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part XXIV 16*, pages 413–430. Springer, 2020. 65

[324] William Weir. Robots help children with autism improve social skills. https://news.yale.edu/2018/08/22/robots-help-children-autism-improve-social-skills. Access:2022-08-26. 4

[325] Jinming Wen, Li He, and Fumin Zhu. Swarm robotics control and communications: Imminent challenges for next generation smart logistics. *IEEE Communications Magazine*, 56(7):102–107, 2018. 4

[326] James CR Whittington, Joseph Warren, and Timothy EJ Behrens. Relating transformers to models and neural representations of the hippocampal formation. *arXiv preprint arXiv:2112.04035*, 2021. 5

[327] Benjamin Lee Whorf. *Language, thought, and reality: Selected writings of Benjamin Lee Whorf*. MIT press, 2012. 15

[328] Erik Wijmans, Samyak Datta, Oleksandr Maksymets, Abhishek Das, Georgia Gkioxari, Stefan Lee, Irfan Essa, Devi Parikh, and Dhruv Batra. Embodied Question Answering in Photorealistic Environments with Point Cloud Perception. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019. 18

[329] Terry Winograd. Understanding natural language. *Cognitive psychology*, 3(1):1–191, 1972. 15

[330] Siying Wu, Xueyang Fu, Feng Wu, and Zheng-Jun Zha. Cross-modal semantic alignment pre-training for vision-and-language navigation. In *Proceedings of the 30th ACM International Conference on Multimedia*, pages 4233–4241, 2022. 96

[331] Yi Wu, Yuxin Wu, Georgia Gkioxari, and Yuandong Tian. Building generalizable agents with a realistic and rich 3d environment. *arXiv preprint arXiv:1801.02209*, 2018. 18

[332] Yi Wu, Yuxin Wu, Georgia Gkioxari, and Yuandong Tian. Building generalizable agents with a realistic and rich 3d environment. *arXiv preprint arXiv:1801.02209*, 2018. 18

[333] Heather Wyatt, Allan Wu, Rami Thomas, and Yuelei Yang. Life cycle analysis of double-arm type robotic tools for lcd panel handling. *Machines*, 5(1):8, 2017. 97

[334] Fanbo Xiang, Yuzhe Qin, Kaichun Mo, Yikuan Xia, Hao Zhu, Fangchen Liu, Minghua Liu, Hanxiao Jiang, Yifu Yuan, He Wang, et al. Sapien: A simulated part-based interactive environment. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 11097–11107, 2020. 18, 80

[335] Huijuan Xu, Kun He, Bryan A Plummer, Leonid Sigal, Stan Sclaroff, and Kate Saenko. Multilevel language and vision integration for text-to-clip retrieval. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pages 9062–9069, 2019. 16

[336] Yufei Xu, Qiming Zhang, Jing Zhang, and Dacheng Tao. Vitae: Vision transformer advanced by exploring intrinsic inductive bias. *Advances in Neural Information Processing Systems*, 34:28522–28535, 2021. 14

[337] Antoine Yang, Antoine Miech, Josef Sivic, Ivan Laptev, and Cordelia Schmid. Tubedetr: Spatio-temporal video grounding with transformers. *CVPR*, 2022. 5

[338] Yajue Yang, Jia Pan, and Weiwei Wan. Survey of optimal motion planning. *IET Cyber-Systems and Robotics*, 1(1):13–19, 2019. 14

[339] Yinfei Yang, Daniel Cer, Amin Ahmad, Mandy Guo, Jax Law, Noah Constant, Gustavo Hernandez Abrego, Steve Yuan, Chris Tar, Yun-Hsuan Sung, et al. Multilingual universal sentence encoder for semantic retrieval. In *ACL*, pages 87–94, 2020. 78

[340] Zhengyuan Yang, Songyang Zhang, Liwei Wang, and Jiebo Luo. Sat: 2d semantics assisted training for 3d visual grounding. In *ICCV*, pages 1856–1866, 2021. 17

[341] Khalid Yousif, Alireza Bab-Hadiashar, and Reza Hoseinnezhad. An overview to visual odometry and visual slam: Applications to mobile robotics. *Intelligent Industrial Systems*, 1(4):289–311, 2015. 15

[342] YouTube. 500+ hours of content uploaded every minute. https://blog.youtube/press/. Access:2022-09-28. 5

[343] Leijian Yu, Erfu Yang, Peng Ren, Cai Luo, Gordon Dobie, Dongbing Gu, and Xiutian Yan. Inspection robots in oil and gas industry: a review of current solutions and future trends. In *2019 25th International Conference on Automation and Computing (ICAC)*, pages 1–6. IEEE, 2019. 4

[344] Licheng Yu, Xinlei Chen, Georgia Gkioxari, Mohit Bansal, Tamara L Berg, and Dhruv Batra. Multi-target embodied question answering. In *CVPR*, pages 6309–6318, 2019. 18

[345] Licheng Yu, Zhe Lin, Xiaohui Shen, Jimei Yang, Xin Lu, Mohit Bansal, and Tamara L Berg. Mattnet: Modular attention network for referring expression comprehension. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1307–1315, 2018. 50

[346] Tianhe Yu, Chelsea Finn, Annie Xie, Sudeep Dasari, Tianhao Zhang, Pieter Abbeel, and Sergey Levine. One-shot imitation from observing humans via domain-adaptive meta-learning. *RSS*, 2018. 79

[347] Tianhe Yu, Deirdre Quillen, Zhanpeng He, Ryan Julian, Karol Hausman, Chelsea Finn, and Sergey Levine. Meta-world: A benchmark and evaluation for multi-task and meta reinforcement learning. In *CoRL*, pages 1094–1100. PMLR, 2020. 79

[348] Zhihao Yuan, Xu Yan, Yinghong Liao, Ruimao Zhang, Sheng Wang, Zhen Li, and Shuguang Cui. Instancerefer: Cooperative holistic understanding for visual grounding on point clouds through instance multi-level contextual referring. In *ICCV*, pages 1791–1800, 2021. 17

[349] John M Zelle and Raymond J Mooney. Learning to parse database queries using inductive logic programming. In *Proceedings of the national conference on artificial intelligence*, pages 1050–1055, 1996. 16

[350] Andy Zeng, Pete Florence, Jonathan Tompson, Stefan Welker, Jonathan Chien, Maria Attarian, Travis Armstrong, Ivan Krasin, Dan Duong, Vikas Sindhwani, et al. Transporter networks: Rearranging the visual world for robotic manipulation. In *CoRL*, pages 726–747. PMLR, 2021. 78

[351] Luke Zettlemoyer and Michael Collins. Online learning of relaxed ccg grammars for parsing to logical form. In *Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP-CoNLL)*, pages 678–687, 2007. 16

[352] Tianwei Zhang, Huayan Zhang, Yang Li, Yoshihiko Nakamura, and Lei Zhang. Flowfusion: Dynamic dense rgb-d slam based on optical flow. In *2020 IEEE International Conference on Robotics and Automation (ICRA)*, pages 7322–7328. IEEE, 2020. 15

[353] Yubo Zhang, Hao Tan, and Mohit Bansal. Diagnosing the environment bias in vision-and-language navigation. In *IJCAI*, 2020. 22

[354] Yubo Zhang, Hao Tan, and Mohit Bansal. Diagnosing the environment bias in vision-and-language navigation. *International Joint Conferences on Artificial Intelligence*, 2020. 49, 50

[355] Lichen Zhao, Daigang Cai, Lu Sheng, and Dong Xu. 3dvg-transformer: Relation modeling for visual grounding on point clouds. In *ICCV*, pages 2928–2937, 2021. 17

[356] Wenshuai Zhao, Jorge Peña Queralta, and Tomi Westerlund. Sim-to-real transfer in deep reinforcement learning for robotics: a survey. In *2020 IEEE Symposium Series on Computational Intelligence (SSCI)*, pages 737–744. IEEE, 2020. 15

[357] Zhou Zhao, Jinghao Lin, Xinghua Jiang, Deng Cai, Xiaofei He, and Yueting Zhuang. Video question answering via hierarchical dual-level attention network learning. In *Proceedings of the 25th ACM international conference on Multimedia*, pages 1050–1058, 2017. 16

[358] Kaizhi Zheng, Xiaotong Chen, Odest Chadwicke Jenkins, and Xin Eric Wang. Vlmbench: A compositional benchmark for vision-and-language manipulation. *arXiv preprint arXiv:2206.08522*, 2022. 17

[359] Bolei Zhou, Agata Lapedriza, Aditya Khosla, Aude Oliva, and Antonio Torralba. Places: A 10 million image database for scene recognition. *PAMI*, 40:1452–1464, 2017. 25, 27, 28, 35

[360] Chen Zhu, Wei Ping, Chaowei Xiao, Mohammad Shoeybi, Tom Goldstein, Anima Anandkumar, and Bryan Catanzaro. Long-short transformer: Efficient transformers for language and vision. *Advances in Neural Information Processing Systems*, 34:17723–17736, 2021. 14

[361] Fengda Zhu, Xiwen Liang, Yi Zhu, Qizhi Yu, Xiaojun Chang, and Xiaodan Liang. Soon: Scenario oriented object navigation with graph-based exploration. In *CVPR*, pages 12689–12699, 2021. 18, 95

[362] Fengda Zhu, Yi Zhu, Xiaojun Chang, and Xiaodan Liang. Vision-language navigation with self-supervised auxiliary reasoning tasks. In *CVPR*, 2020. 19, 39

[363] Fengda Zhu, Yi Zhu, Xiaojun Chang, and Xiaodan Liang. Vision-language navigation with self-supervised auxiliary reasoning tasks. In *CVPR*, pages 10012–10022, 2020. 63

[364] Wang Zhu, Hexiang Hu, Jiacheng Chen, Zhiwei Deng, Vihan Jain, Eugene Ie, and Fei Sha. Babywalk: Going farther in vision-and-language navigation by taking baby steps. *arXiv preprint arXiv:2005.04625*, 2020. 19

## RÉSUMÉ

Les progrès réalisés dans le domaine de l'apprentissage automatique ont permis des percées importantes, notamment en vision par ordinateur, en traitement du langage naturel et en robotique. Peut-on aller plus loin et combiner ces domaines de recherche ? Cela développera de nouvelles applications, comme la robotique guidée par le langage, où un robot doit suivre les instructions fournies par un opérateur.
Alors que les humains apprennent à suivre des instructions dès leur enfance, la même tâche est difficile pour des robots, et cela pour plusieurs raisons: (i) le manque de données d'entraînements, (ii) les raisonnements faits sur multiples niveaux d'abstraction, et (iii) l'espace d'actions ayant une haute dimension.
L'objectif de cette thèse est d'améliorer la robotique guidée par le langage en relevant ces défis. Nous décomposons la difficulté de la robotique guidée par le langage en considérant deux types de tâches : (i) un robot mobile doit se rendre à un endroit cible décrit par des instructions ; (ii) les instructions décrivent une séquence d'actions qu'un bras robotique doit opérer sur des objets placés sur une table.
Nos contributions sont les suivantes : (i) pour résoudre le manque de données d'entraînement, nous avons développé une procédure efficace de pré-entraînement basé sur le nouveau jeu de données BnB, (ii) nous avons construit de nouvelles architectures neuronales basées sur une approche hiérarchique pour encoder plusieurs niveaux d'abstractions, et (iii) nous avons proposé une nouvelle méthode pour prédire des actions continus et en sur plusieurs dimensions pour résoudre un grand nombre de tâches.

## MOTS CLÉS

Robotique, vision par ordinateur, traitement du langage naturel, transformer

## ABSTRACT

Recent progress in machine learning has enabled groundbreaking improvements notably in computer vision, natural language processing, and robotics. Can we go one step further and combine these research fields? This would allow new applications, such as language-guided robotics, where a robot must follow instructions provided by an operator.
While people learn to follow natural language instructions from their childhood, the same task is difficult for robots. Current challenges include (i) the limited amount of training data, (ii) the multiple levels of reasoning, and (iii) the multi-dimensional continuous action space.
The goal of this thesis is to improve language-guided robotics by addressing these challenges. We break down the difficulty of language-guided robotics by considering two types of tasks: (i) vision-and-language navigation, where a mobile robot must go to a target location, and (ii) vision-and-language manipulation, where a robotic arm should manipulate objects on a tabletop.
Our contributions are the following: (i) we address the scarcity of training data and develop an efficient pre-training procedure based on the new BnB dataset, (ii) we propose a hierarchical approach based on the Transformer architecture to encode several layers of abstractions, and (iii) we propose a new method predicting continuous and multi-dimensional actions for solving a large number of robotics tasks on a tabletop. Methods developed in this thesis have been tested in photo-realistic simulators and on a real-world robot. They have outperformed the state-of-the-art performance on a dozen of benchmarks.

## KEYWORDS

Robotics, computer vision, natural language processing, transformer