Learning to Understand Spatial Language for Robotic Navigation and Mobile Manipulation

by

Thomas Kollar

B.S., University of Rochester (2004)
S.M., Massachusetts Institute of Technology (2007)

Submitted to the Department of Electrical Engineering and Computer Science
in partial fulfillment of the requirements for the degree of

Doctor of Philosophy

at the

MASSACHUSETTS INSTITUTE OF TECHNOLOGY

June 2011

Author .....
Department of Electrical Engineering and Computer Science
May 20, 2011

Certified by... .....
Nicholas Roy
Associate Professor
Thesis Supervisor

Accepted by ....
Professor Leslie A. Kolodziejski
Chair, Department Committee on Graduate Students

Learning to Understand Spatial Language for Robotic Navigation and Mobile Manipulation

by

Thomas Kollar

Submitted to the Department of Electrical Engineering and Computer Science
on May 20, 2011, in partial fulfillment of the
requirements for the degree of
Doctor of Philosophy

Abstract

This thesis focuses on understanding task-constrained natural language commands, where a person gives a natural language command to the robot and the robot infers and executes the corresponding plan. Understanding natural language is difficult because a system must infer the location of landmarks such as "the computer cluster," and actions corresponding to spatial relations such as "to" or "around" and verbs such as "put" or "take," each of which may be composed in complex ways. In addition, different people may give very different types of commands to perform the same action.

The first chapter of this thesis focuses on simple natural language commands such as "Find the computer," where a person commands the robot to find an object or place and the robot must infer a corresponding plan. This problem would be easy if we constrained the set of words that the robot might need to reason about. However, if a person says, "find the computer," and the robot has not previously detected a "computer," then it is not clear where the robot should look. We present a method that uses previously detected objects and places in order to bias the search process toward areas of the environment where a previously unseen object is likely to be found. The system uses a semantic map of the environment together with a model of contextual relationships between objects to infer this plan, which finds the query object with minimal travel time. The contextual relationships are learned from the captions of a large dataset of photos downloaded from Flickr. Simulated and real-world experiments show that a small subset of detectable objects and scenes are able to predict the location of previously unseen objects and places.

In the second chapter, we take steps toward building a robust spatial language understanding system for three different domains: route directions, visual inspection, and indoor mobility. We take as input a natural language command such as "Go through the double doors and down the hallway," extract a semantic structure called a Spatial Description Clause (SDC) from the language, and ground each SDC in a partial or complete semantic map of the environment. By extracting a flat sequence of SDCs, we are able to ground the language by using a probabilistic graphical model that is factored into three key components. First, a landmark component grounds novel noun phrases such as "the computers" in the perceptual frame of the robot by exploiting object co-occurrence statistics between unknown noun phrases and known

perceptual features. These statistics are learned from a large database of tagged images such as Flickr, and build off of the model developed in the first component of the thesis. Second, a spatial reasoning component judges how well spatial relations such as "past the computers" describe the path of the robot relative to a landmark. Third, a verb understanding component judges how well spatial verb phrases such as "follow", "meet", "avoid" and "turn right" describe how an agent moves on its own or in relation to another agent. Once trained, our model requires only a metric map of the environment together with the locations of detected objects in order to follow directions through it. This map can be given *a priori* or created on the fly as the robot explores the environment.

In the final chapter of the thesis, we focus on understanding mobile manipulation commands such as, "Put the tire pallet on the truck." The first contribution of this chapter is the Generalized Grounding Graph (G^3), which connects language onto grounded aspects of the environment. In this chapter, we relax the assumption that the language has fixed and flat structure and provide a method for constructing a hierarchical probabilistic graphical model that connects each element in a natural language command to an object, place, path or event in the environment. The structure of the G^3 model is dynamically instantiated according to the compositional and hierarchical structure of the command, enabling efficient learning and inference. The second contribution of this chapter is to formulate the problem as a discriminative learning problem that maps from language directly onto a robot plan. This probabilistic model is represented as a conditional random field (CRF) that learns the correspondence of robot plans and the language and is able to learn the meanings of complex verbs such as "put" and "take," as well as spatial relations such as "on" and "to."

Thesis Supervisor: Nicholas Roy
Title: Associate Professor

Acknowledgments

First and foremost, I would like to thank my parents Mary and Robert Kollar. They have always believed strongly in the value of education. They let me choose my path, which eventually led me to enroll as an undergraduate at the University of Rochester, to perform robotics research with Jon Schmid and Professor Chris Brown, and then to become a graduate student at MIT. This path is a tribute to their support, inspiration, immaculate ability to listen and unending love, which has always given clarity to my life.

I also would like to thank my wife, Beth Kollar, who is the love of my life and the foundation on which I stand. She has kept me (mostly) out of the thesis repulsor field. Her unending love, devotion, positive outlook and helpful comments have given me perspective and kept me focused.

I would also like to thank my thesis adviser, Nicholas Roy, for never denying an opportunity for me to learn and grow as a researcher. He has encouraged exploration and has always been a mentor. The many days spent in his office and the many iterations of papers, theses and presentations have always made them stronger. His advice in all areas has proven invaluable.

Much of this thesis has benefited from a collaboration with Stefanie Tellex. Because of this collaboration, the thesis as a whole is bigger than sum of its parts. Chapter 5 has benefited significantly from the efforts of Steven Dickerson, who even though he was only a part of the project for a short time has contributed immeasurably to its outcome. I also have to thank all of my collaborators: Ashis Banerjee, Matthew Walter, Emma Brunskill, Sachi Hemachandra, Pablo Espinace, Albert Huang, Abraham Bachrach, Alexander Shkolnik and Finale Doshi. Without their efforts and input, this thesis either would have taken a number of additional years or would simply not have happened. Finally, I would like to thank our administrative assistant, Sophia Hasenfus, for all of her help; the thesis process would not have gone smoothly without her.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

Imagine for the moment the capabilities of the robots in Figure 1-1: a robotic wheelchair could help patients in hospitals and homes regain independence, a micro-air vehicle could aid soldiers in the surveillance of hostile areas or help maintenance workers inspect faulty equipment, a humanoid robot could help rescuers find victims during disasters or aid them in performing chores in the home, and a robotic forklift could help warehouse staff move supplies more efficiently. Although each of these scenarios may sound decades from application, we currently have prototype systems in the laboratory that are beginning to achieve these tasks. Even still, real-world interfaces to systems in the field are still very basic: an operator has to be highly trained, often sits at a laptop or desktop computer and commands the robot by giving it low-level commands that manipulate individual joints or motors. What we would really like is for an operator in the field to be able to give high-level commands to the robot and to be able to keep his hands and eyes free so that they might be able to achieve other tasks.

This is the promise of speech, which is a natural, flexible and intuitive way for humans to command complex systems. This is evidenced by the human-computer interaction literature, where Ochsman and Chapanis [1974] have found that speech provides speed benefits in human-human teams on collaborative tasks that compared written, typed, speech, and fully-expressive communication modalities. The key difference in the speed at which subjects achieved the task came with the introduction of the speech modality. Building on this result Martin [1989] gave both a review article and a set of experiments that showed significant speed advantages to using speech for a wide range of human-computer interaction domains. Further, speed is not the only important criterion. Rudnicky [1993] found that there was a strong preference for speech even when other modalities, such as a scroller, keyboard, and mouse, were available.

However, most of these results have been from the *human-computer* interaction literature, whereas the goal is to determine when *human-robot* interaction can benefit from speech interfaces. Since robots have limited interfaces and operators often need to keep their hands and eyes free for other tasks, one would expect a robust language understanding system to be the preferred interface. Craparo [2004] has studied a natural language interface for controlling unmanned aerial vehicles (UAVs) in an air-

(a) Humanoid Robot (b) Robotic Helicopter (c) Robotic Forklift (d) Robotic Wheelchair

Figure 1-1: Robot platforms that can take advantage of the ability to understand natural language commands.

traffic control tower. She focused on setup, ingress, re-targeting, and egress tasks for the UAV and found that not only was speech preferred for many of the tasks, there was also consistent decrease in mental workload, physical demand and effort, indicating that operators are also able to achieve more tasks in a smaller amount of time.

Given the evidence that *human-robot* teams could benefit in a significant way from the introduction of a speech interface, why have we not seen a greater growth in speech interfaces to robotic systems?

1.1 The Grounding Problem

Certainly there are many reasons that speech is difficult, but in this thesis I argue that at least one reason we have not seen wider use of language to command robots is because the grounding problem is extremely challenging. The grounding problem, which addresses the *symbol-grounding problem* [Harnad, 1990], refers to the tenuous connection that logical systems in artificial intelligence (AI) have with the environment, often leading to systems that have little or no connection to perceptual features of the physical world. In this work we will refer to the *grounding* problem as the problem of taking natural language input and converting it into a plan that the robot can execute in the environment. This conversion requires mapping from linguistic terms for objects such as "the doors" or "the elevators," places such as "on the truck," paths such as "past a whiteboard" or events such as "put the tire pallet on the truck," onto the corresponding object, place, path or event (plan) that executes the language. For example, a command to follow a route might be:

> With your back to the windows, walk straight through the door near the elevators. Continue to walk straight, going through one door until you come to an intersection just past a whiteboard. Turn left, turn right, and

enter the second door on your right (sign says "Administrative Assistant").

This conversion requires the robot to reason not only about the spatial and semantic features of real-world environments, but also the meaning of its own actions, even when there is a high degree of perceptual and physical complexity. This is challenging, because we want robots to operate in environments like the one below:

There is additional complexity introduced if we put few constraints on the language, which is necessary if we want to be flexible to the different ways that people give commands. In this thesis, we have constrained people only by the task. For example, if you ask people to give commands that can enable another person to achieve a goal destination in the environment (route following), then you get a wide variety of responses. Each command will refer to different objects or places in the environment and different ways that the agent can achieve the goal. In Table 1.1, we can see the variety of commands people give for the same route.

Toward this end, we study commands from five different domains. The first three (object finding, route following, and visual inspection) can reasonably be captured under the title *route commands*, each of which include commands that move a robot from one location to another in the environment. The other category (indoor mobility and mobile manipulation) can be captured by the title *general commands*, which include understanding and reasoning about action verbs such as *meet*, *bring*, *put* or *pick up*, and which may not only require reasoning about the path of an agent and static objects, but also about other dynamic agents and the motions of objects over time. Examples from four of these domains can be seen in Table 1.2.

1.2 The State of the Art

Enabling robots to understand natural language commands requires taking text, converting it into a structure that represents the components of the command, and converting each element in the structure into an action that the robot can execute. This requires 1) a structure, 2) the specification of the conversion process from the command onto this structure, 3) a conversion from the structure onto a set of actions,

Commands for the same route.
-Look for a long corridor, travel down it until you reach a wall (dead end) take a left down a corridor until you reach a wall / office, turn right, stop at second opening on left, enter door, you have arrived.
-Stand beside the spiral staircase. turn to face the three nearby elevators. Notice the double doors immediately to the right of where you're facing. Go through them and walk straight down the hall. There is a gray door in front of you and stairs to your left. Go through the door and take the first left through the blue double doors. Go straight. You should pass a supply room on your right. Turn right around the corner, following the hall, and go straight until you pass the wall folders on the left. Opposite the whiteboard on the right is room 36-872.
-Walk thru glass doors, continue forward approximately 30 feet, passing bathroom and continue till get to door with phone, walk thru door. Take 5 steps forward, will see nitrogen tank, take left. Keep tan walls on your right, continue forward 9 steps. Make right. First blue door on left. Stop.

Table 1.1: Commands from the route following corpus that are given by different people, but have the exact same start and end location.

and 4) a set of actions that the robot can execute. In this thesis, we build on the state of the art in two significant ways.

First, we provide a flexible semantic structure that enables our systems to understand unconstrained spatial natural language commands. The representation is inspired both by the structure of spatial language [Denis et al., 1999] and by the need to keep a flexible representation that still has the ability to capture the breadth of the language in cases where the language is un-grammatical or challenging to parse. The semantic structure presented in this work builds on the work of Jackendoff [1983], Landau and Jackendoff [1993], Tversky and Lee [1998] and Talmy [2005], providing a computational instantiation of their formalisms. Further, we automatically learn the mapping from the language onto this semantic structure.

Secondly, we provide models that learn the mapping from the semantic structure onto an unconstrained action space, which enables the system to reason about a wide variety of spatial language. Inspired by Regier [1992], who learned a specific set of spatial relations words like "to" or "toward," we expand the set of applicability to spatial discourse. The models described in this thesis predict the state sequence of the robot (e.g., its path) given a natural language command, which means that an action specification is no longer in symbolic form as in previous work [Ge and Mooney, 2005, Dzifcak et al., 2009]. The specification of an action is now in terms of the language and states of the robot (e.g., "to" in "to the kitchen," means that the physical path of the robot ends near the kitchen), which enables the system to learn the meanings of words in terms of spatial and semantic aspects of the environment,

Commands from the corpus	
Route Following	With your back to the windows, walk straight through the door near the elevators. Continue to walk straight, going through one door until you come to an intersection just past a white board. Turn left, turn right, and enter the second door on your right (sign says "Administrative Assistant").
Visual Inspection	Go forward until you are able to make a left. Then move ahead until you reach the opposite wall, then make a right. Go straight, past one staircase and to the next staircase. At the top of this staircase you will find a fire alarm on your left at approximately 7ft up.
Indoor Mobility	Follow the person to the kitchen. Then move toward the bathroom. Next go down the hall to the lounge.
Mobile Manipulation	Pick up the pallet of boxes in the middle and place them on the trailer to the left.

Table 1.2: Example commands from each of the corpora we have collected.

instead of abstracting actions into a fixed, discrete set.

Finally, taking inspiration from Bugmann et al. [2004] and Levit and Roy [2007], we take an empirical approach by evaluating our system on corpora of commands, enabling us to quantify the robustness of the system.

1.3 Technical Approach

The general approach that we take to understanding language from these domains is to take as input the language and a map of the environment, and compute the minimum cost plan for the robot. More formally, if we have a set of directions Z:

> With your back to the windows, walk straight through the door near the elevators. Continue to walk straight, going through one door until you come to an intersection just past a whiteboard. Turn left, turn right, and enter the second door on your right (sign says "Administrative Assistant").

and we have a map of the environment m, then the goal is to find the lowest cost plan, Γ, for the robot to execute:

$$\underset{\Gamma}{\operatorname{argmin}}\, C(\Gamma|Z, m) \tag{1.1}$$

The first thing that you might notice is that given language of the type described above, it is not at all clear how to compute this cost function C. Mapping from arbitrary spatial language seems as if it might require arbitrary dependencies between terms. One of the key ideas of this thesis is to decompose the problem according to the linguistic structure of the command and learn a cost function over robot plans

that is likely to correspond to a specific natural language command given to the robot. At training time, we will assume access to a corpus of language paired with robot plans, while at inference time, we will take advantage of the structure of the problem in order to find the corresponding robot plan.

1.4 Environmental Semantics

In this thesis, we want to work in real-world environments. In order to map from language onto structures that the robot can reason about, there must be overlap between the representations that humans use and those that robots use. In this work, we assume that a robot is able to acquire semantics about the environment: in this case the name (e.g., "monitor") and geometric location/shape of an object or place (e.g. a polygon representing the physical shape of the object). This is not an unreasonable assumption, since computer vision techniques as a part of the PASCAL Visual Object Classes (VOC) challenge [Everingham et al., 2010] and the Semantic Robot Vision Challenge [Meger et al., 2008] have shown promising results on very challenging datasets and in real-world environments.

We have explored several methods for obtaining these semantics. The first method is to visually detect objects and then place them into a map of the environment. We have used a visual object detector from Felzenszwalb et al. [2008], which uses a mixture of multiscale deformable parts models and has been shown to represent highly variable object classes and achieves state-of-the art results on the PASCAL object detection challenges.

However, the object names that are extracted from a visual object classifier may not give a human-centric conceptualization of the environment. In order to explore more human-centric conceputalizations, we have also explored a second method, which involves giving the robot a tour of the environment [Hemachandra et al., 2011]. In this scenario, a robot follows a person, who tells the robot the names of spaces in the environment. By correlating spoken place names with a space in the map, we can acquire the human-centric conceptualizations of space such as "CSAIL Headquarters" instead of specific object types (e.g., "monitor"). Others have been working to extract human-centric semantics from the environment by extracting text from signs [Posner et al., 2010]. We have also annotated the semantic map of the environment with the location and name of objects and places.

The resulting map of the environment along with the semantics is called a *semantic map* (Figure 1-2). The semantic map consists of an occupancy grid map of the environment [Thrun et al., 2008] along with the detections (textual names) and spatial layout of each of the objects or regions.

1.5 Thesis Statement

Inferring the mapping between spatial language commands and robot actions is best expressed as the minimization of a cost function that is represented as a learned probabilistic graphical model.

Figure 1-2: A semantic occupancy grid map that contains the locations and names of some objects that a robot can detect using a visual object detector.

1.6 Contributions and Organization of Chapters

The remainder of this thesis is organized as follows:

Chapter 2

In Chapter 2 we discuss background work on human-computer interaction using speech. We find that speech has been shown to increase speed, is often preferred to other interfaces and can reduce workload on people in human-robot interaction domains.

We identify grounding as one of the missing components to enabling robust speech interfaces and describe the grounding problem. We categorize previous work on grounding into three categories in the context of human-robot interaction. The first approach is to define a procedural specification of robot action primitives. These approaches tend to focus on the parsing of language into an intermediate representation that can then be converted into action primitives. Most do not involve learning. The second approach is to perform grounding directly in the sensorimotor space of the robot, which requires learning directly from language input correlated with the motor control system of the robot. A final approach is to use learning either as a method to convert from language into a symbolic representation (which would enable flexible language to be used) or to learn the mapping from linguistic clauses into the action space of the robot or perceptual features of the environment (or the environment

itself).

Chapter 3

In Chapter 3, we focus on simple natural language commands such as "Find the computer," where a person commands the robot to find an object or place and the robot must infer a corresponding plan. This problem would be easy if we constrained the set of words that the robot might need to reason about. However, if a person says, "find the computer," and the robot has not previously detected a "computer," then it is not clear where the robot should look. We present a method that uses previously detected objects and places in order to bias the search process toward areas of the environment where a previously unseen object is likely to be found. The system uses a semantic map of the environment together with a model of contextual relationships between objects to infer this plan, which finds the query object with minimal travel time. The contextual relationships are learned from the captions of a large dataset of photos downloaded from the Flickr photosharing website.

Simulated and real-world experiments are performed, showing that a small subset of detectable objects and places can be used to predict the location of previously unseen objects or places. We also show that the system generates better plans than a greedy approach for finding certain objects or places.

Chapter 4

In Chapter 4, we take steps toward building a robust spatial language understanding system for three different domains: route directions, visual inspection, and indoor mobility. We take advantage of the structure of spatial language in order to break a command into clauses, and to decompose those clauses into components. For the command, "Go through the double doors and down the hallway," the component clauses include "go through the double doors" and "go down the hallway," each of which further breaks down into fields of a semantic structure. This decomposition enables the system to understand each component independently, in order to build a plan for the robot.

A key contribution of this section is to learn the mapping between the language and robot plans. Instead of pre-specifying the meaning of words such as "the doors," "to," or "follow" in the robot's representation, we learn the mapping between the two. This allows people to use arbitrary combinations of words in order to command the robot and does not require pre-specification of word meaning. In this section, a human teacher shows a robot examples of how to perform a task such as "go through the doors," and the robot learns the mapping between this language and a robot plan.

The learning in this chapter takes the form of a probabilistic graphical model that is factored into three key components. The first component grounds novel noun phrases such as "the computers" in the perceptual frame of the robot by exploiting object co-occurrence statistics between unknown noun phrases and known perceptual features using the model from Chapter 3. Second, a spatial reasoning component judges how well spatial relations such as "past the computers" describe the path of the

robot relative to a landmark. Third, a verb understanding component judges how well spatial verb phrases such as "follow", "meet", "avoid" and "turn right" describe how an agent moves on its own or in relation to another agent. Once trained, our model requires only a metric map of the environment together with the locations of detected objects in order to follow directions through it. This map can be given *a priori* or created on the fly as the robot explores the environment. We have demonstrated our system on both a robotic wheelchair and a micro-air vehicle.

Chapter 5

In the final chapter of the thesis, we focus on understanding mobile manipulation commands such as, "Put the tire pallet on the truck." The first contribution of this chapter is the Generalized Grounding Graph (G^3), which connects language onto grounded aspects of the environment. Unlike Chapter 4, we relax the assumption that the language has fixed and flat structure and provide a method for constructing a hierarchical probabilistic graphical model that connects each element in a natural language command to an object, place, path or event in the environment. The structure of the G^3 model is dynamically instantiated according to the compositional and hierarchical structure of the command, enabling efficient learning and inference.

The second contribution of this chapter is to formulate the learning problem as a discriminative learning problem that maps from language directly onto a robot plan. Because standard parameterizations of the plan space would lead to an environment-specific model, we instead formulate the learning problem as one of determining whether the language and plan correspond. This probabilistic model is represented as a conditional random field (CRF) that learns the correspondence of robot plans and the language and is able to learn the meanings of complex verbs such as "put" and "take," as well as spatial relations such as "on" and "to." We have demonstrated the system on a corpus collected for a robotic forklift, and run demonstrations on a robotic forklift platform.

Chapter 6

Chapter 6 concludes with the contributions of the thesis and future directions for the work.

Appendix A

Appendix A gives a review and comparison of the task-constrained corpora that we have collected, along with the structure of spatial language that is contained in these corpora for route following, visual inspection, indoor mobility and mobile manipulation commands.

Chapter 2

Related Work

This thesis draws on rich basis of work in robotics, machine learning, natural language processing, cognitive science and artificial intelligence.

2.1 Speech for Human-Robot Interaction

Speech has a strong advantage over other interaction modalities since humans do not need special training and operators can keep their hands and eyes free for other tasks. In the human-computer interaction literature, Cohen and Oviatt [1995] characterized the situations where speech can benefit an operator. He listed five main scenarios: when the operator's hands and eyes are busy, when only a limited interface is available, when the operator is disabled, when pronunciation is the matter of computer use, and when natural language interaction is preferred. With the exception of the fourth criterion, all of these could be said to apply to robots. He proposes a research direction which applies to this thesis particularly well: "How to select relatively 'closed' domains, for which the vocabulary and linguistic constructs can be acquired through iterative training and testing on a large corpus of user input." In this thesis, we have collected four sets of corpora in relatively closed domains.

The benefits of using speech to interact with a computational system go back to Ochsman and Chapanis [1974], who found that speech has a significant advantage over typing and that its introduction is the critical component to speeding collaborative tasks between humans. When pairs of people were working to achieve a task via typing, video, voice, or a "communication rich" mode, the biggest improvement in speed to problem solution came with the introduction of voice. Building on this work, Martin [1989] found that, when compared to other human-computer interfaces, speech input was both faster than typed input and that it increased user productivity by providing an additional response channel. Further, subjects were able to complete more tasks when speech input was available, and typing increased by 30% the time to execute a command sequence. In addition to efficiency, Rudnicky [1993] showed that speech can also be a preferred interface to achieve a task, even when other modalities (scroller, keyboard, or mouse) are present and could achieve the task more efficiently.

However, for many of these early studies, the authors were comparing text to

speech. For speech interfaces to be applied in practice, they must provide a more intuitive or flexible interface than the alternative: a graphical user interface. In these domains, speech has faced challenges; Cohen et al. [2000] hypothesized that this is because people are being asked to speak when other modalities of communication may be more appropriate. To address this they evaluated a multi-modal interface that takes advantage not only of speech, but also of other input modalities, and found a 3.5-fold speed improvement in the interaction time.

It seems that applying these insights to robotics should be straightforward; robots have limited interfaces, operators often need to keep their hands and eyes free for other tasks, and one would expect a robust language understanding system to be preferred to other interfaces. Craparo [2004] focused on the growing need to provide air-traffic controllers a natural language interface to unmanned aerial vehicles (UAVs). For this domain, she focused on setup, ingress, re-targeting, and egress tasks for the UAV. In experiments with a prototype system, she found that there was a consistent decrease in mental workload, physical demand and effort ratings for controlling UAVs with speech.

This leads to the conclusion that speech is a promising interface for humans to interact with robots. In this thesis, we explore the grounding problem in the context of spatial language understanding. In this thesis, we use the term *spatial language* to refer to language where all parts can reasonably be grounded in some aspect of the environment. To make this more concrete, spatial language will include the domain of route directions (commands to move from one location in an environment to another) in conjunction with motion verbs that involve reasoning about the action of other agents (e.g., "meet") and verbs that require reasoning about action sequences involving other objects (e.g., "put").

2.2 The Grounding Problem in Human-Robot Interaction

Mapping from language onto spatial, semantic, and perceptual features is often termed the *grounding* problem, because it addresses the symbol-grounding problem [Harnad, 1990]. The symbol-grounding problem occurs when meaning is represented symbolically: a symbol is defined in terms of other symbols, which leads to circular definitions of the meanings of words or concepts. To make this idea more concrete, Harnad [1990] uses Searle's Chinese Room Experiment experiment to ask the question: If a person is only given a Chinese/Chinese dictionary and is asked to learn Chinese, could they do so? This person would be able to start at one word in the dictionary and move to the next. However, eventually there would be a set of foundational words that a person could not understand without reference to the environment. In this case, the person would pass endlessly from one meaningless definition to another, never being able to grasp the perceptual meaning of any of the words. The perceptual aspect is the critical missing piece for these systems; each of the words (symbols) must be converted onto different aspects of the environment.

There are broadly three different ways that researchers generally perform language grounding for robotics. The first is to convert the natural language text into a semantic representation (by a pre-specified grammar), which is then manually mapped to pre-specified robot actions and environmental features. For example, in Levit and Roy [2007], the authors defined "to," to be:

> TO: in a straight line, approach the closest point of a reference object.

This means that anytime a robot sees the word "to" in the language, it should use this template to define the meaning of "to." In addition, on the language side Dzifcak et al. [2009] specify a grammar for spatial commands, assuming that the instructions are complete, grammatical and in the robot's lexicon.

A second approach is to perform grounding directly in the sensorimotor space of the robot. This entails giving the robot examples of a sensorimotor task paired with language and learning a controller that achieves a given task. For example, Sugita and Tani [2005] treat words as a sensory input along with the joint angles of the robot and learn a recursive controller for the meaning of certain actions in the space of words and motor primitives. Although this approach shows the promise of providing a perceptual representation of language thereby enabling a robot to connect language directly to sensory aspects of the world, these approaches tend not leverage the linguistic regularities of spatial language.

A final approach is to perform grounding by learning how to extract the basic clauses from the command and learning the meaning of each clause in the perceptual representation of the robot. This has the advantage both that the space of commands need not be pre-specified and that the symbols are grounded in perceptual features of the environment. At the same time, the learning tends not to be in the direct sensorimotor space of the robot, but instead in a representation of the physical world, which enables the system to take advantage of the linguistic regularities while at the same time abstracting away from the "blooming buzzing confusion" of direct sensory input.

In the next few sections, we give an overview of systems that have implemented each of these approaches to understanding spatial language. Many of the papers are discussed here, but the interested reader may want to consult the following references as well [Roy and Reiter, 2005, Roy, 2005].

2.2.1 Procedural Approaches to Grounding

Beginning with SHRDLU [Winograd, 1970], a computational system that provided a dialog interface to a simulated manipulator robot, researchers have dreamed of a system where humans could command robots via natural language. At an abstract level SHRDLU performed reasoning in much the same way as modern systems: language is input to the system, is converted onto a semantic structure that represents the relevant meaning for a robot, and this semantic structure is manually mapped to robot actions. Even though the mechanism of SHRDLU was fairly complex, it was one of the most promising AI systems of the time. Even the Lighthill report,

which was particularly critical of AI, cited it with great optimism as a very promising demonstration of the potential of artificial intelligence [Lighthill, 1973]. However, there were concerns about the generality of the approach, which led Winograd to start a new research direction based on a completely different approach [Wilks, 1974].

One of the main challenges of applying logical systems to understand language is to map between each symbol in the representation and some element of the physical environment (the symbol-grounding problem). Often, very sophisticated language understanding systems are created that map onto a pre-defined or fixed action space. In this section, we focus on systems that take strong advantage of the structure of spatial language, but at the same time use little or no learning, little perceptual feedback, and often have a fixed action space. These approaches usually involve specifying an action for each new word or phrase that is present in the natural language command.

Action Primitives for Spatial Language

A number of authors have focused on defining action primitives that, when available, will enable a robot to execute natural language commands. Müller et al. [2000] presents a system that breaks a command down into a starting point, reorientation commands, path/progression landmarks, and finally the goal. The authors focus not on translating route directions as given by people, but on route specifications. For each component in the specification, there is a fixed mapping from each component of the intermediate representation onto a behavior module that achieves the command. If the route specification is a landmark, then the behavior is to go straight and not branch into other rooms. If it is a reorientation component, then the system looks for the orientation action to execute (e.g., "turn right"). When there are no more elements in specification, then the goal is assumed to have been reached.

Bugmann et al. [2004] identify 15 sensorimotor primitives in a corpus of spoken natural language directions such as "go," "location_is," "go_until," and "enter_roundabout, " that a robot must have in order to follow route directions. A corresponding program that enabled the robot to execute the action was implemented for each primitive. The authors found that their primitives are not a closed class, which means that more action primitives would need to be written for any new terms.

Roy et al. [2003] and Hsiao et al. [2003] describe a system in which each word, such as "blue" is defined by hand in a fuzzy way as a scalar value that indicates how strongly the color of the object matches the expected canonical value of blue. Similarly, "touch" is defined as the reaching gesture that terminates when the touch sensors are activated and the visual system reports that the target is in view. Building on this, Levit and Roy [2007] performed an analysis of verbal commands in terms of elementary semantic units, and created a set of *navigational informational units* NIUs, that break down instructions into components. NIUs include moving around objects, moving in absolute directions, turning and verifying closeness to a specific landmark. They manually design prototypes for each rule corresponding to an NIU, which are handcrafted structures combined with data-driven parameter adaptation.

Natural Language and Formal Approaches

There are also a set of approaches that focus more on the language aspect of the grounding problem. In particular, some will either convert onto a semantic structure or will convert the problem to a temporal logic representation for planning. In particular, Craparo [2004] proposed a natural language interface for air traffic controllers to coordinate unmanned aerial traffic in the area surrounding an airport. They take a natural language query, parse it using a context free grammar and then use verb frames to give each command a semantic representation that is specific to the air-traffic control domain (e.g., the generated verb frames output headings based on the verb) and then use a pre-defined mapping from each element onto a database that contains the current state of the system (rejecting any inconsistent action).

Kruijff et al. [2007] focus on the task of labeling the spatial organization of the environment (e.g., *human-augmented mapping*). The authors do this by representing the spatial organization using spatial entities (e.g., rooms, areas and floors), spatial aspects (e.g., connected areas), and functional aspects (e.g., a kitchen is used for cooking). Each command is parsed into a combinatorial categorical grammar (CCG), and each element in the CCG is mapped onto a set of behaviors that the robot knows how to execute.

Kress-Gazit et al. [2008] provide a system that converts from "structured" English onto low-level robot controllers that can execute the corresponding command by converting language into linear temporal logic via hand-designed rules. The logical representation can then be converted into a hybrid controller for the robot by assuming that the robot has perfect sensors and access to all necessary information.

Dzifcak et al. [2009] focus on understanding natural language commands by inferring two meanings for each command: the specification of goal states and the means of achieving the goal state. They develop an incremental parser that takes lexical items from English with syntactic annotations from a CCG and semantic annotations from a temporal and dynamic logic and maps them onto dynamic logic expressions that represent the goals and actions specified by the natural language directive. To convert from the goals and actions into a plan, the authors, "had an annotated map of the whole environment and were able to associate locations like 'breakroom' with particular areas on the map." In addition, they assume that instructions are complete and grammatical and are in the robot's lexicon, so that words that do not appear in the logic are unable to be understood. Cantrell et al. [2010] builds on the work of Dzifcak et al. [2009] by incrementally extracting both goals and actions at the same time incrementally.

Integrated Approaches

There have been systems that take advantage of both linguistic structures and action primitives to provide an integrated approach to following commands. The first such system, MARCO, is an agent that is able to follow free-form natural language route instructions [MacMahon et al., 2006, MacMahon, 2007]. MARCO consists of six components: a *syntax parser* models the surface structure, a *content framer* interprets

the surface meaning, an *instruction modeler* combines information across phrases and sentences, an *executor* acts to gain knowledge about the environment, a *robot controller* executes an action and the *view description matcher* checks the expected model with the observed model. When the content framer must convert from the surface meaning of the instruction onto a *procedural specification* of the command, each verb frame in the content frame is associated with a hand-coded procedure based on the frame arguments and idioms, which is then translated via the executor to a corresponding plan. Each action is executed via the robot controller using the simple actions of move, turn, verify, and declare-goal, plus a set of pre- and post-conditions. For landmarks, a view description models the object's type and location (which side and how far) within the view relative to the observer, which is matched to a symbolic representation of that object.

Skubic et al. [2004] focus primarily on a dialog between the robot and a person about the spatial structure of the environment instead of inferring a plan for the robot given a natural language command. Spatial modeling is accomplished using a *histogram of forces*, which represents the relative spatial position between an object and the robot. These histograms are converted into a set of features, which are then converted via a set of hand-designed rules into a three-part linguistic spatial description: a primary direction, a secondary direction and an assessment of the description, along with a fourth part which describes the Euclidean distance between the robot and object.

2.2.2 Sensorimotor Approaches to Grounding

The opposite perspective on grounding language is that language is inseparable from its sensorimotor experience. The essential premise of Sugita and Tani [2005] is that semantics are an inseparable process in the process of embodied cognition. In this work, they present a novel connectionist model that learns the semantics of simple language through behavioral experiences of the robot. This means that no symbolic or structural representations were provided *a priori*, such that the lexicon, syntax and semantics are treated the same as other input modalities. The types of commands that they learn are two word sentences consisting of a verb followed by a noun, such as, "hit center." They show that the structures of situated semantics can organize on their own using a recurrent neural network with parametric bias nodes (RNNPB). They perform an off-line training phase that takes the full state of the robot paired with language and learns the parameters of an RNNPB that minimizes the error of the training set. Because the learning takes place very close to the percepts, there is almost no representation of the command other than the words as input (e.g., semantics, syntax or otherwise).

In Marocco et al. [2010], the authors study how a humanoid robot can learn to understand the meaning of action words by physically acting in the environment and linking the effects of its own actions to the behavior of the objects in the environment. Specifically, the authors study object manipulation and argue that the dynamics of manipulating an object can be characterized by the action performed on the object and the activation of its sensors during the movement and physical interaction with

the object. They explore a simulated humanoid robot, which is controlled by a recurrent artificial neural network trained on examples via the back-propagation-through-time algorithm. The learning takes place in the space of linguistic input and joint encoders. The results indicate that the robot is able to extract sensorimotor contingencies based on a particular interaction with an object and reproduce its dynamics by acting on the environment. They show that the robot is able to categorize the linguistic label correctly given sensorimotor input and the system appears to have learned a notion of the force-dynamics of interacting with the robot directly from the sensorimotor space of the robot. In this work, they only learn over three different linguistic terms directly from the sensorimotor input.

Modayil and Kuipers [2007] describe how a robot can learn about objects from its own autonomous experience with the continuous world. The authors present a formalism that represents the ontology of objects and actions, a learning algorithm and an evaluation with a physical robot. The authors assume that the robot has already learned the basic structure of its sensorimotor system and has the ability to construct and use local maps of the static environment. The ontology of objects is an abstraction of the low level continuous experience of the robot. The symbolic abstraction consists of trackers, perceptual functions, concepts, and actions. Trackers track a cluster of sensory experiences as it evolves over time, a perceptual function generates a percept which represents a property of a tracker over time (e.g., distance, location, color), a concept is an implicitly defined set of percepts, and an action contains a description of its effects on the object's percepts, the context where it is reliable and a control law. Actions are learned via the effects of motor babbling. In order to understand the high-level task such as, "Place a recycle bin in the center of the room" an associated goal state where the recycle bin is in the center of the room is generated and backchaining is used to create reactive plans to change a percept to a goal value.

2.2.3 Learning Approaches to Grounding

In Section 2.2.1, we saw work where a parser was created that mapped natural language onto a semantic structure, which was in turn mapped to a discrete action space for the robot. Each of the actions corresponding to linguistic terms were hand-coded. In Section 2.2.2, the exact opposite approach was taken: the meanings of a very limited number of words were learned, but were inextricably tied directly to the direct sensorimotor experience of the robot. This means that robot plans were necessarily represented at the sensorimotor level, but at the same time led to an inability to understand a wide variety of concepts.

In this section, we give examples of work that lie in the middle of these two. For these approaches there is generally still some connection to the environment, but it may be slightly abstracted from the sensorimotor system of the robot. In general, these approaches have employed learning of action space or learning for the conversion from language into a semantic representation. This generally has the promise to make the system more flexible, since the system can learn the mapping from language onto its own action space or onto perceptual features of the environment, which we would

hope generalize over environments and tasks.

Ge and Mooney [2005] introduces an approach to understanding RoboCup coaching commands. In this approach, the system learns a semantic parser that converts from natural language onto a formal meaning representation, which can be used to coach a RoboCup team. The meaning representation language (MRL) is a detailed, formal, representation that can be used to interpret commands to coach a RoboCup soccer team in a simulated soccer domain. By augmenting a state-of-the-art statistical parsing model to include semantic information, their system is able to integrate syntactic and semantic clues for robust interpretation. However, a true understanding of the linguistic concepts requires capturing the connection between language and perceptual aspects of the environment [Mooney; 2008].

Matuszek et al. [2010] builds off of the work of Wong and Mooney [2006], by learning the mapping between natural language and the physical environment, as represented by a topology that is directly extracted from sensor measurements. Their approach breaks down a natural language command into clauses and learns the mapping between each clause and the *map layer* of the environment using a *path description language*. This language is an unambiguous synchronous context-free grammar (SCFG) that describes the basic structure of the problem and does not represent *landmarks* or *context*. Given this SCFG and examples of language/path description language pairs, the authors use the WASP parser to learn the mapping from natural language onto a path represented in the path description language [Wong and Mooney, 2006]. Features used in the learning are a function of the parse tree and include the number of times a rule has been used in the parse, the number of times that a word is generated from a word gap, and the total number of words generated from word gaps. To follow a command, the authors break the command down into clauses (each clause was created by splitting on pre-defined keywords) and then search in the learned model for a robot path in the path description language of maximum probability. The authors evaluate their system on a corpus of commands, assuming that they have a way to determine when the destination has been reached once a path has been traversed. On the subset of *followable* commands, they show that their system is able to interpret the commands 71% of the time.

Similar to Matuszek et al. [2010], Shimizu and Haas [2009] proposes the goal of creating a system that can understand instructions given in unrestricted natural language. They assume access to a topological representation of the world and the current pose of the robot. Using a path (represented symbolically as a sequence of actions) paired with linguistic clauses, they learn the mapping between the two using a conditional random field (CRF). In this work, they treat the grounding of a command as a sequence labeling problem. The input observations are each word in the language and the output is an action frame, which describes the action the robot should take at that moment. Action frames are composed of four slots: whether to go down the hall and turn at the end or enter the doorway, whether the location is a hallway or a door, whether to turn left, right or go straight and which ordinal number from one to four describes where to take the action (e.g., the third door). Instead of concatenating the values of the frame slots to form one label for each token, they factorize each action frame and the slots associated with it into a segment, calling this the frame-segment

model. Using only linguistic features, they train the conditional random field (CRF) and show better performance than a linear-chain CRF on a corpus that they have collected.

There are approaches that use non-linguistic features, using aspects of the environment in order to understand linguistic terms. Regier [1992] learns a partially-structured connectionist model for the acquisition of lexical semantics for specific spatial terms in a visually-grounded domain. The partially-structured nature of the network attempts to both enable a human designer to build knowledge of the domain into the architecture, while at the same time enabling the system to learn which features of the environment might be relevant. The set of terms includes spatial relations such as "above," "below," "to the left of," "to the right of," "through," or "around." For those lexical terms involving motion, the authors use a static feature detecting module, the output of which is fed into a motion module that judges how well the example is described by a particular lexical term up to and including the current frame of the video. The authors extend their work to include the Attention Vector Sum (AVS) feature, which characterizes the relative "goodness" of spatial relations such as "above" [Regier and Carlson, 2001].

Finally, we might want to learn to follow commands with less supervision. Branavan et al. [2009] develop a system that uses reinforcement learning to map natural language instructions onto sequences of executable actions. During training, the learner constructs action sequences for a set of documents, executes those actions and then observes the resulting reward. Policy gradient is used to estimate the parameters of a log-linear model of action selection, which they use to infer actions in both a Windows troubleshooting domain and a game tutorial domain. Finally, they show that their system can rival supervised learning techniques, while requiring few or no annotated training examples.

Building on the work of Branavan et al. [2009], Vogel and Jurafsky [2010] present a reinforcement learning system which learns to interpret route directions from Map-Task without requiring semantic annotation. At training time they take as input a set of dialogs, which are segmented into a sequence of utterances and a map of the environment, which is composed of a set of named landmarks. The state of the system is composed of a named landmark, a cardinal direction, and the current utterance. The action space consists of a landmark together with a cardinal direction. The reward function consists of a hand-crafted combination of features that relate to how similar an utterance looks like a landmark, whether the expert path moves between two corresponding landmarks, and whether the system goes to the correct side of the landmark. Instead of directly representing the state of the world, they convert it into a set of features so that it might generalize to new environments and then learn the state-action value function using Q-learning.

2.3 Conclusions

As Mooney [2008] said:

> I believe the time is ripe for exploring the task of learning the connec-

tion between language and perception.

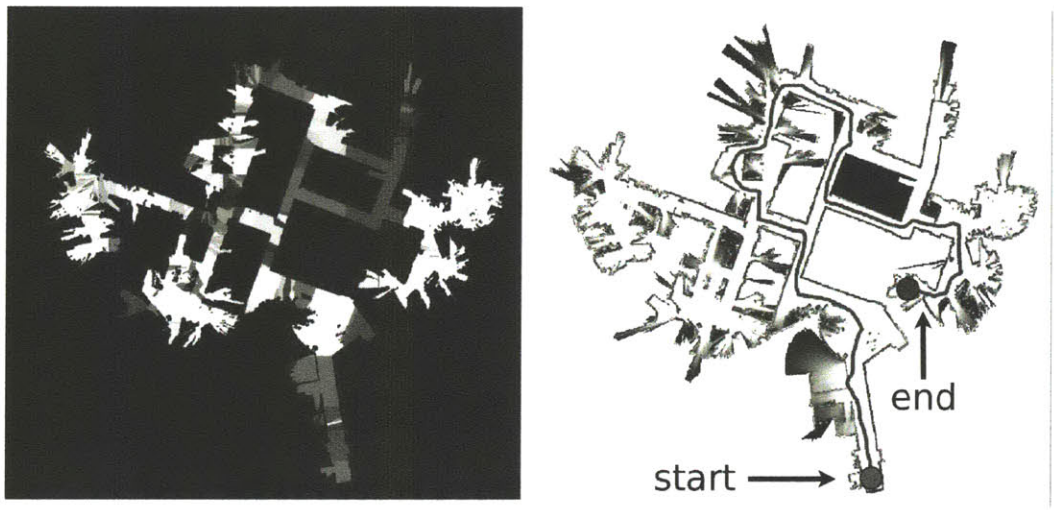
In this thesis, we explore the connection between language and low-level features of the environment. Much of our work is based on features of the physical environment paired with linguistic structures from spatial language. Unlike previous approaches, we do not constrain the set of actions to a predefined set and build up the set of commands that the system can learn, from landmarks to spatial relations to motion verbs. Since the inference space is no longer discrete (e.g., it is over paths and events), inference is more difficult and approximations must be made. Since there are few constraints on the action space, our system is also able to capture the wide variability in the meanings of spatial language commands. At the same time, we are not trying to learn directly from the sensorimotor perceptions of a robot, which gives us hope that our system will generalize over a wide variety of commands. Finally, we take inspiration from prior work and evaluate our system on various corpora of natural language commands [Bugmann et al., 2004, Levit and Roy, 2007, Branavan et al., 2009, Vogel and Jurafsky, 2010].

Chapter 3

A Model for Grounding Object-Finding Commands

The goal of this chapter is to understand natural language interactions such as "Find the computer." where a person commands the robot and the robot must infer a plan that finds an object or place in the environment. This problem would be easy if we constrained the set of words that the robot might need to reason about. For example, if the robot had previously detected a monitor and someone said, "find the monitor," then the nearest location with a monitor would be a good candidate. However, if a person says, "find the computer," and the robot has not previously detected a "computer," then it is not clear where the robot should look. One approach might be for the robot to search arbitrarily through the environment looking for a computer cluster (assuming access to a "computer" detector). However, this search does not utilize prior information that the robot might have about the environment. For example, if the robot is searching for a computer and has previously detected monitors, then we might want to bias the search process toward those areas of the environment where monitors have been found (since monitors and computers tend to be spatially correlated).

In this chapter, we propose a system that takes as input a free-form natural language command such as "Find the computer," extracts a landmark word (e.g., "computer") and, assuming access to an object or place detector, uses a semantic map of the environment together with a model of contextual relationships between objects to infer a plan that finds the query object with minimal travel time. This process consists of first generating a probability distribution over the likely locations of a "computer" (as in Figure 3-1) and then planning a path which, although it may not be the shortest, is the one most likely find the object or place corresponding to the landmark word. A database of over a million image/caption pairs from the Flickr photosharing website has been downloaded and the captions have been found to have tags that are spatially co-located. This enables the system build a model of context and convert from any of the 25,000+ concrete nouns in the English language onto locations in the environment that are likely to see the corresponding object or place



(a) Probability Map (b) Plan

Figure 3-1: In (a) we show the probability map for the previously unseen object, "computer." White is a high probability region where a computer may be found and dark is low probability. Based on the predictions in (a), we can generate a plan that minimizes the expected time to find a computer in (b).

category[1].

Simulated and real-world experiments are performed, showing that a small subset of detectable objects and places can be used to predict the location of previously unseen objects or places with a precision of 95% and a recall of 74% over 10 unknown object and place types. We also show that the system generates better plans than a greedy approach for finding certain objects or places. Finally, we provide a demonstration of the system with automatic object detection, showing that the system can correctly plan a path to find previously unseen objects or places in real-world environments.

Object and place words are one component of spatial language that we study in Chapters 4 and 5. This chapter, aside from providing a cost function that biases the search for an object or place, also provides a foundational component for locating objects or places in the environment that will be useful in subsequent chapters.

3.1 Approach

Recall from Chapter 1.3 that the general approach that we take to understanding commands is to take a natural language command and a semantic map of the environment, and compute the minimum cost plan that corresponds to the language.

[1]The number of objects and places in the English language was estimated to be 25,000 by taking all the concrete nouns from the WordNet database [Miller et al., 1990]. Biederman [1987] estimated the number of common object categories to be between 1,500 and 3,000.

In this case, the plan, Γ, will correspond to the path of the robot. We represent the path Γ as a sequence of poses (x, y, θ) that describe places that the robot visits, where x and y are the x and y coordinates in a 2D map and θ is the orientation. More formally, if we have a set of directions Z that commands the robot to find an object:

Find the computer.

and a semantic map of the environment m, then the goal is to find the lowest cost path Γ that will allow the robot to detect the computer:

$$\underset{\Gamma}{\operatorname{argmin}}\, C(\Gamma|Z, m) \tag{3.1}$$

Instead of arbitrarily searching for a previously unseen object, the goal of this cost function is to bias the search toward areas that are more promising, using an object detector to determine when it has found the object or place from the command. Thus, we would like a cost function that will take advantage of prior knowledge that the robot has about the environment.

We will therefore define the cost function as the expected length of the path under the probability of seeing the object along the path. This means that the system will pick paths that are more likely to find the object earlier, rather than arbitrarily searching for the object or greedily picking the most likely location and driving there. If Γ is the path of the robot, L_Γ is the length of the path, and Γ_l is the part of the path up to length l, then the cost function is:

$$C(\Gamma|Z, m) \triangleq E[L_\Gamma] = \sum_{l=1}^{L} p(\mathrm{Z}|\Gamma_l, m) \times l \tag{3.2}$$

In order to compute the expectation in Equation 3.2, we need a way to connect objects and places from the language to a particular path through the environment. Noting that the probability of an object or place is approximately conditionally independent of location, since observations that are far away from a location do not effect the probability of detecting an object or place at the current location. If $\gamma_i \in \Gamma_l$ is the ith location in the path of length l, Z is the object or place word, and $\phi_i(Z)$ is true when the an object or place is visible at location i and false otherwise, then we can model this as:

$$p\left(Z|\Gamma_l, m\right) = p\left(\phi_l(Z) = T|\gamma_l, m\right) \times \prod_{i \neq l} p\left(\phi_i(Z) = F|\gamma_i, m\right) \tag{3.3}$$

Thus, we model the probability of a object or place along a path as the probability of the robot seeing the object or place at the final location that it visits (since it has already visited all locations l on the way to the destination and not found the object). In this chapter, we are focused on learning the correlations between objects or places the robot can see and objects and places that are unknown. We will focus our effort on a model that does not incorporate sensor error since we use this model directly

in Chapter 4. For models that incorporate sensor error, the interested reader may consult Kollar and Roy [2009] and Espinace et al. [2010].

3.1.1 Context

In order to connect prior information that a robot may have with previously unseen objects, we need a model of context. In Equation 3.3, we can see that we need to model the probability of objects or places given a particular location in the environment. However, this distribution involves a previously unseen object or place word Z and an arbitrary location γ and map m. Let us define the following terms:

- $\phi_i(Z)$ true if the object or place corresponding to the landmark word Z is visible at the i th location in Γ_l
- $s_j(\gamma, m) \in S(\gamma, m)$, true if object or place type j is detected at location γ in map m

Instead of using learning directly over locations, we will compute a set of features corresponding to visible objects from a given location. Given γ_i is a location in the environment, we can now rewrite this distribution as:

$$p(\phi_i(Z)|\gamma_i, m) = p(\phi_i(Z)|S(\gamma_i, m)) \tag{3.4}$$

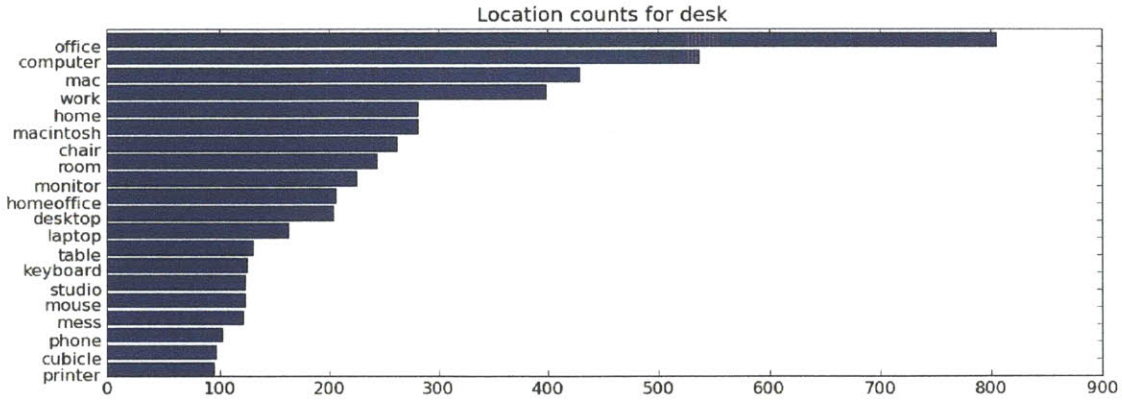
If we instantiate variables ϕ for a particular linguistic term $Z =$ "computer," and assume that the robot has detected a "monitor" and has not detected a "zebra" in the environment, then we can write:

$$p\left(\phi_i(\text{computer}) = T|s_1(\gamma_i, m) = \text{monitor det.}, s_2(\gamma_i, m) = \text{zebra not det.}\right) \tag{3.5}$$

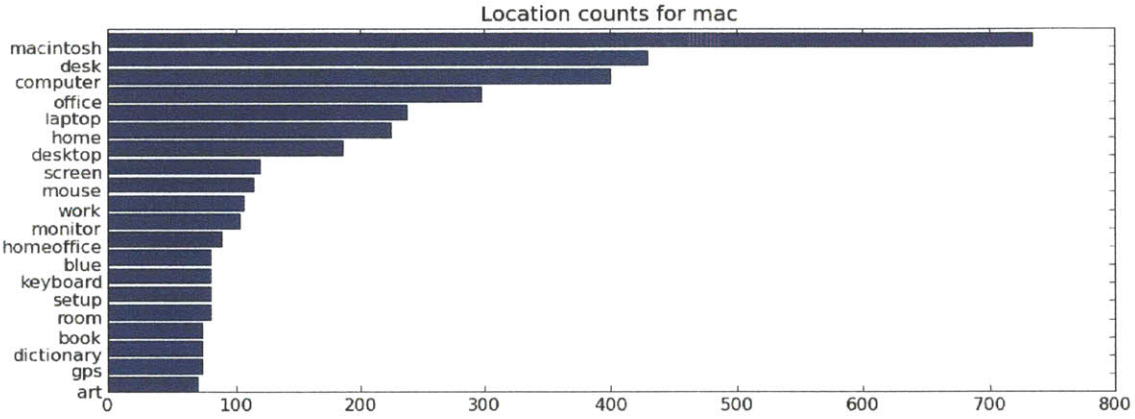
This is the probability that the object corresponding to the landmark word computer is visible given that it can see a monitor and not a zebra. Note that we condition on all objects and place terms that the robot has detected in the environment, so that the system can use negative information such as the fact that the robot has not detected a "zebra" in order to infer that the robot is not outdoors. The model in Equation 3.5 therefore turns into a problem of learning the contextual relationships between the existence of an object or place corresponding to a landmark word and objects detected in the environment.

3.1.2 Learning

One might ask how we are able to learn the probability distribution in Equation 3.4 over the unknown class Z of 25,000+ object or place (landmark) words in the English language. In order to address this issue, we use the idea of context. In Figure 3-3, if we have an object detector for the category "monitor," this can tell the system something about the environment. For example, by detecting a "monitor," we have extracted a landmark word that corresponds to an object or place in the environment. We might want to relate this landmark word to other landmark words such as a "computer,"



(a) desk



(b) mac

Figure 3-2: Two query words and the corresponding counts for the most related categories from the Flickr database. On the horizontal axis are the raw counts for the number of times *desk* or *mac* appeared with each category on the vertical axis in the Flickr database. In (b) we are able to query a challenging word *mac* (a computer made by Apple), which is often found near desks, screens, keyboards, etc. Further, it is clear that both query words are spatially co-located with the most related categories and further that a detectable object (monitor) is strongly related to both query words, indicating that our system should be able to understand commands containing them.

"keyboard," or "mouse." But each of these landmark words is related to objects or places that are likely to be spatially co-located. In order to detect the object "computer," we need both an object detector and also a way of relating things that the robot has detected to places or objects in the environment. The latter is the focus of this section: is there a way to acquire relationships between objects or places that correspond to previously unseen objects or places that the robot was not previously able to detect to objects or places that the robot can detect? And can we do this on the scale of all the concrete nouns in the English language?

In order to explore whether whether we could acquire semantics on a large scale, we

Figure 3-3: An example Flickr image: tags are *desktop, monitor, computer, keyboard and mouse*

turned to large datasets on the World Wide Web. Specifically, we had the hypothesis that images on the photosharing website Flickr had tags that corresponded to objects and places that were spatially co-located, as in Figure 3-3. We download over a million images with their corresponding captions, performing a dense sampling of the concrete nouns defined in the WordNet semantic network (e.g., soap, hallway, office). For each concrete noun, a search was performed on the Flickr site, and the top 1000 hits were downloaded. While not all concrete nouns are landmarks that people use in the corpora we have collected, the set of concrete nouns does provide a superset of the set of terms that need to be understood for giving directions.

We hope to see that when a query landmark word corresponded to one of the tags, then other tags in the image would correspond to objects or places that should be spatially correlated in the image. In Figure 3-2 we can the results for the *desk* and the *mac* classes. On the vertical axis are the top 20 object classes that co-occur with the base category (*desk* or *mac*) and on the horizontal axis are top 20 most-frequently occurring tag words. Near the top of the list for *desk* are *computer, keyboard, mouse, printer, lamp*, all things that humans would expect to find with a *desk*. In addition, we are not limited to a strict vocabulary. This can be seen by looking at *mac* in Figure 3-2(b). *Mac* refers to a Macintosh computer, and as expected we find that it co-occurs with desks, computers, chairs and printers.

We use a dataset of tag correlations downloaded from the Flickr photosharing website in order to learn the distribution in Equation 3.4. We downloaded one million image captions, which contain a set of tags such as, "computer," "monitor," and

"keyboard." Given a query word, such as "computer," we will remove the query landmark word Z (e.g., "computer") from the examples. A positive example for the object word "computer" is "monitor" and "keyboard." A negative example of the object word "computer" may be "microwave." Using a balanced subset of captions from Flickr of positive and negative examples of the query word (filtered for objects that the robot is able to detect), we train the model in Equation 3.4 as a naive Bayes classifier. The model is able to correlate previously unseen words such as "computer" in the language with locations in the environment using contextual relationships with detectable objects such as a "monitor."

3.1.3 Planning

We are now able to use this model in order to compute a plan that minimizes the expected travel distance to find objects or places that correspond to a given landmark word. In order to do this, we perform breadth-first search. Because the enumeration of all paths in a full occupancy grid map of the environment is intractable, we operate in the medial-axis transform of the map [Blum, 1967, Ballard and Brown, 1982], which reduces an occupancy grid map into a skeleton [Zhang and Suen, 1984, Gonzalez and Woods, 1992]. By searching to a fixed depth[2] in the medial-axis transform of the map, the algorithm is able to only expand one or two neighboring locations during the search. To further reduce search complexity, we only allow paths that backtrack when a candidate path reaches a dead-end (e.g., at the end of a corridor). Backtracking enables the planned path for the robot to go into a room, explore it, and then come back out. In Kollar and Roy [2009], we simulated measurements as the robot moved, taking into account sensor error and incorporating the value of additional measurements as the robot backtracks to new locations. If the robot goes into a room, but finds that it is unlikely to see an object in a given room after a few measurements, then the probability in Equation 3.3 will be low, leading to little need for additional information about locations in that room.

Given a robot starting location, the robot position is registered to the closest location on the medial axis, its neighbors are iteratively expanded, the expected path length is evaluated, and the path that minimizes the expected path length to find the object or place corresponding to a landmark word from Equation 3.2 is returned.

3.2 Results

There are two ways to evaluate the approach. The first is a component-wise evaluation of how well the system predicts previously unseen objects or places over the entire environment. A second method is to run end-to-end evaluations of the system in order to evaluate the expected length to find a particular object.

[2] A depth of 1000 in our experiments.

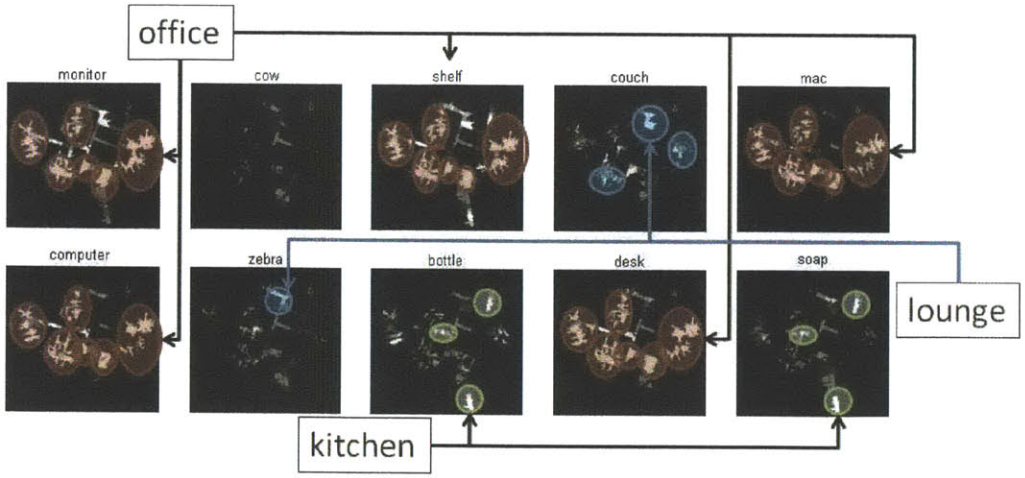


Figure 3-4: The above image shows probability maps for ten different unknown object categories: monitor, cow, shelf, couch, mac, computer, zebra, bottle, desk and soap. White is higher probability, darker is lower probability for each of these objects. For each of the objects, we have highlighted the locations where the algorithm correctly predicted these objects for office, lounge and kitchen areas. For example, we can see that the algorithm properly locates the soap in the kitchen (we did not map bathrooms) and that computers are correctly located in office areas.

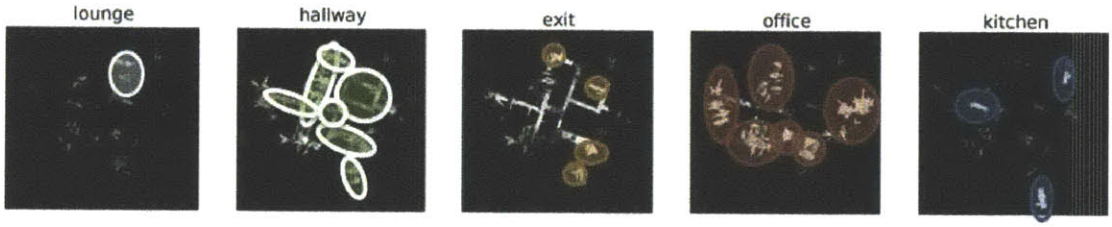


Figure 3-5: The above image shows probability maps for five different unknown place categories: lounge, hallway, exit, office and kitchen. White is higher probability, darker is lower probability for each of these places. For each of the places, we have highlighted the locations where the algorithm correctly predicted these objects for each place category. For example, we can see that the algorithm properly locates kitchens and offices, but is unable to locate the exits.

3.2.1 Contextual Model

In order to perform a component-wise analysis of the contextual model, we have collected a semantic map of the third floor of a building at MIT by manually labeling a fixed number of objects in the environment. In this particular environment, we labeled approximately one hundred object types. For each of the categories in Figure 3-4, we have removed the corresponding object types from the semantic map and evaluated

the probability of a previously unseen object or place Z given a location γ_i over the entire map according to Equation 3.4.

In Figure 3-4, we can see queries for the probability of an object or place corresponding to ten previously unseen object or place Z: monitor, cow, shelf, couch, mac, computer, zebra, bottle, desk and soap. We can see that the *mac* object is usually associated with offices, which seems intuitively correct and there is zero probability of finding a *cow* in an office-type environment, as expected. *zebra* shows a slight probability of appearing in the lounge because the Flickr database had a number of examples of a "zebra couch".

We can also query the model for the most-probable place type for five different previously unseen place categories: lounge, hallway, exit, office and kitchen (as in Figure 3-5). Again, white is higher probability, darker is lower probability for each of these places. We have highlighted the locations where the algorithm correctly predicted the place types. For example, we can see that the algorithm properly locates kitchens and offices, but is unable to locate the exits.

If we run the model on simplified examples, we can see what objects will predict the unknown objects and places and gain insight into the model. For example, in Table 3.1, we can see that a monitor predicts a computer, a sofa predicts tables and living rooms, a microwave predicts food, and a sofa predicts a couch and an armchair. These predictions tend to match our intuitions.

p(computer)	monitor	sofa	microwave
0.72	True	False	False
0.02	False	True	False
0.10	False	False	True

p(table)	monitor	sofa	microwave
0.28	True	False	False
0.42	False	True	False
0.12	False	False	True

p(food)	monitor	sofa	microwave
0.03	True	False	False
0.07	False	True	False
0.70	False	False	True

p(living room)	monitor	sofa	microwave
0	True	False	False
1	False	True	False
0	False	False	True

p(couch)	monitor	sofa	microwave
0.01	True	False	False
0.99	False	True	False
0.00	False	False	True

p(armchair)	monitor	sofa	microwave
0.0	True	False	False
1.0	False	True	False
0.0	False	False	True

Table 3.1: Example of the probability of an object corresponding to the "computer," "table," "food," "living room," "table," and "armchair," given that the robot is able to detect the existence (or not) of a monitor, sofa and microwave.

We have also performed an experiment to determine which objects or places corresponding to a landmark word were predicted well using the model of context. We did this by discretizing the environment (from Figures 3-4) into 21 regions and using the model to categorize when an object or place was present. If the model predicted that

an object or place was in a region of the map above a threshold and the object or place was truly present, then we would say that this was a correct prediction (otherwise it was incorrect). We did this for 10 unknown object categories (e.g., monitor, cow, shelf, couch, mac, computer, zebra, bottle, desk and soap). We found that the model correctly predicted that an object was present 95% of the time (e.g., precision), and also correctly predicted that the object was present in regions that truly contained the object 74% of the time (e.g., recall). If we exclude the bottle class, the recall becomes 88%. The precision and recall on a per-class basis can be seen in Table 3.2, where we find that many unknown objects were predicted with 100% precision. The model misses a number of bottles and couches (low recall). The model likely misses bottles because they can be found in many locations, which leads to a diffuse probability mass over the entire environment. The model likely misses couches because they tend to be found in unlikely locations in the test environment (e.g., offices).

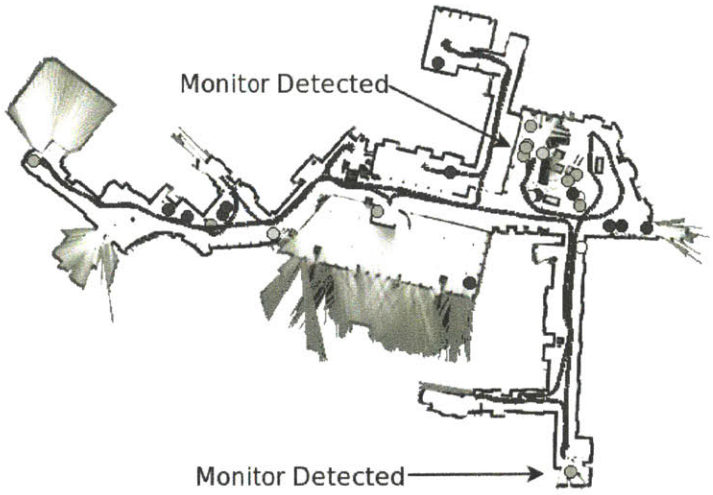
	monitor	cow	shelf	couch	mac	computer	zebra	bottle	desk	soap
precision	0.92	1	0.93	1	1	1	1	1	0.92	1
recall	0.92	1	0.87	0.20	0.91	1	1	0.07	1	1

Table 3.2: Precision and recall for each of the object types. In general, the objects are well-predicted, although in some cases we miss the object when it should be predicted (e.g., low recall). This happened for the bottle and couch class, where both classes were missed a number of times. For the bottle class, this is likely because bottles occur in many locations, so that it is unlikely to find it in any specific space. For couches, some of the areas in the environment were office-type environments, so it is not surprising that these were difficult to predict.

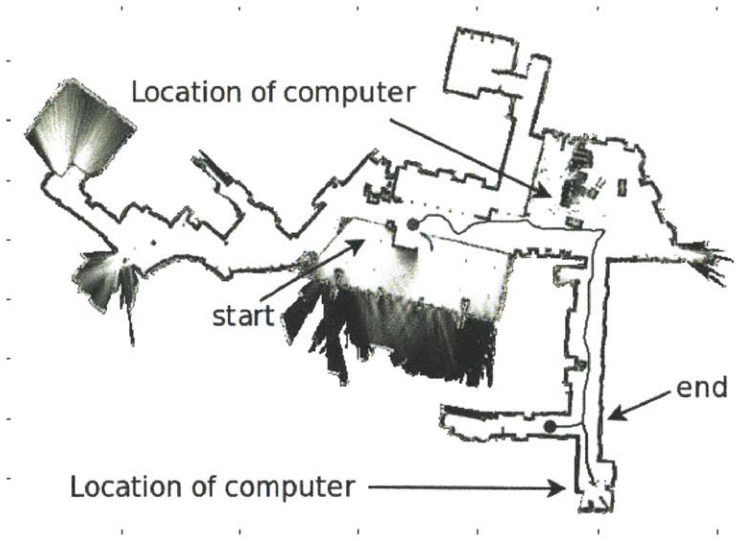
3.2.2 Path optimization results

In this section, we used the approach developed in this chapter to find unknown objects of various types. We picked a greedy baseline with which to compare our system, which picks the most likely location over the full map and generates the corresponding shortest-path plan to that location. In the first set of experiments we placed the robot at a starting location and computed a plan from the starting location to find various types of objects. Looking at Table 3.3, we found that the generated paths using our approach were shorter than those of the greedy approach to find the object.

Finally, we randomly picked 30 starting locations for the robot and generated a plan, performing an end-to-end test for the query objects in from Figure 3-4. We found that our approach had a shorter time to find to find the object for the landmark phrase "desk" 13% of the time and to the object corresponding to the landmark phrase "refrigerator" 68% of the time. In the rest of the cases the *greedy* approach and our approach had equal objective values, since the shortest-path for greedy approach was also the one that minimized the expected length of the path, which led the same path to be generated for both approaches.



(a) Location of known objects registered to the map



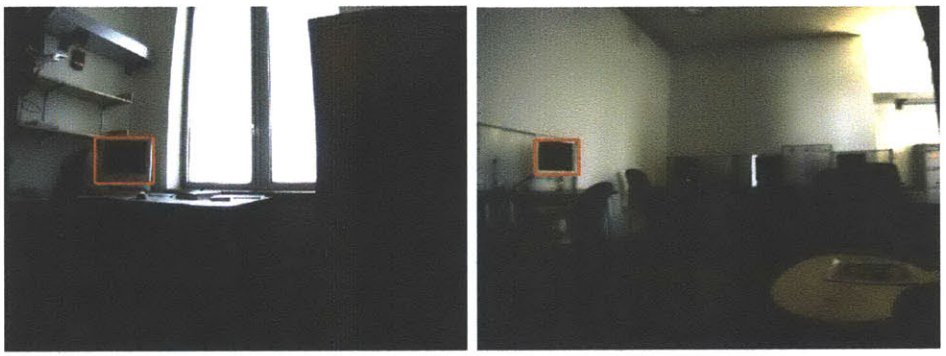
(b) Planned path for an unknown object, *computer*

Figure 3-6: In (a) we can see the location of the detected known objects registered to the map. The path shown in this map is the path that the robot took during data collection. The two locations where monitors were detected are denoted by the arrows. In (b) we can see the path that our system infers in order to find the unknown object, computer. The path passes by the places where two computers reside in the environment.

We also evaluated our techniques on another floor of a building at MIT using a visual object detector from Felzenszwalb et al. [2008]. We detected three objects as the robot moved around the floor: *chairs*, *bicycles*, and *monitors*. Some examples of the classifier output are shown in Figure 3-7. The locations of the object detections were automatically added to the map, according to where the robot was located as

	path length (our approach)	path length (greedy)
bottle	11.60m	39.90m
computer	11.53m	23.56m
soap	37.76m	40.39m
couch	18.45m	20.19m
monitor	11.07m	11.10m

Table 3.3: Distance in meters before finding an unknown object or place assuming access to a detector that can recognize when an object is found. There is a trend indicating that our approach produces shorter paths.



(a) True positive (Monitor) (b) True positive (Monitor)

(c) True positive (Chair) (d) False positive (Chair)

Figure 3-7: Using the approach in Felzenszwalb et al. [2008], we classify a number of object categories in an office building at MIT. Above are some of the images classified correctly and one instance of a false positive. In (a/b) are the two locations where a monitor was detected in the environment. In (c) is a true positive of a chair while in (d) is a false positive of a chair.

shown in Figure 3-6(a). Out of a trajectory of approximately 5000 images, there were 13 false positives, and 64 true positives over all categories. The chair detector incorrectly detected 13 chairs, while the bicycle detector missed no bicycles (there were two in the environment) and the monitor detector falsely detected no monitors

(there were two detections).

Based on these three detectors alone, we were able to predict the the location for the computer and keyboard words, each of which looked qualitatively reasonable. In Figure 3-6(a) we can see the locations where a monitor is known to be visible based on the object detections and in Figure 3-6(b) we can see the resulting search path for a novel object, specifically a computer; the path goes past locations where the monitor is likely to be, minimizing the expected time to find a computer.

3.3 Conclusions

In this chapter we have focused on simple commands of the form, "Find the computer," where we enable a robot to infer when an object or place will correspond to previously unseen landmark phrases. One contribution of this work is a model of context that is learned from the captions of a large dataset of photos downloaded from Flickr. We demonstrate both simulated and real-world experiments that use a small subset of detectable objects and places in order to robustly predict the location of objects or places that correspond to landmark words. In addition, we have demonstrated that in certain circumstances the proposed approach finds objects more quickly than the baseline approach of greedily selecting the most likely location in the map.

Chapter 4

A Model for Grounding Route and Mobility Commands

In Chapter 3, the focus was on understanding where object and places were likely to be found in natural environments. In this chapter, we build on this work in order to understand spatial discourse such as, "Go down the hallway and past the elevators," which, in addition to landmarks such as "hallway," also involve spatial relations such as "past" and motion verbs such as "go." The spatial language discussed in this chapter is challenging because it has only been constrained by the task. This means that each person may give very different language, the quality of which depends on a person's linguistic and spatial capabilities (see Figure 1.1) [Denis et al., 1999, Vanetti and Allen, 1988]. Directions are also difficult because some information tends to be missing, such as a start or end point and path/progression information [Tversky and Lee, 1998]. Finally, there are 80 to 100 relation words such as "to" or "around" [Landau and Jackendoff, 1993], 47 motion verbs such as "meet" or "follow," [Slobin, 1996], and 10,000 and 30,000 landmark words [Biederman, 1987], such as "the doors" or "the hallway." A functional system will need to understand a significant subset of these words in order to follow commands.

In this chapter we take steps toward building a robust spatial language understanding system for three different domains: route following, visual inspection and indoor mobility. In order to understand spatial language, the system exploits the fact that spatial language breaks down into component clauses that consist of actions the agent is meant to execute and landmarks that the agent is meant to see [Denis et al., 1999]. For example, for the command "Go through the double doors and go down the hallway." there are two clauses: "go through the double doors" and "go down the hallway." Each of these breaks down into four components: a *figure*, which is often implicitly "you", a *landmark* such as "the hallway," a *spatial relation* such as "down" and a verb such as "go." Each component in the command is then formalized into a semantic structure that we call a Spatial Description Clause (SDC).

In this chapter, understanding a spatial language command involves grounding

Parts of this chapter were performed in collaboration with Stefanie Tellex, Sachi Hemachandra, Emma Brunskill, Albert Huang and Abraham Bachrach.

a command in a path of the robot and possibly other agents in the environment. One of the contributions of this chapter is a probabilistic graphical model that has three key components. The first component grounds novel noun phrases such as "the computers" in the perceptual frame of the robot by exploiting object co-occurrence statistics between unknown noun phrases and known perceptual features. These statistics are computed from a large database of tagged images such as Flickr (as described in Chapter 3). Second, a spatial reasoning component grounds spatial relations such as "past." Third, a verb understanding component grounds motion verbs such as "follow," "meet," "avoid," and "turn right." Once trained, our model requires only a metric map of the environment together with the locations of detected objects in order to follow directions through it. This map can be given *a priori* or created on the fly as the robot explores the environment.

We evaluate the approach using three corpora. First, we evaluate the system on over 150 natural language route instructions on two different floors of a building. Second, we evaluate on forty-nine 3-D inspection commands for a micro-air vehicle. Finally, we evaluate on an indoor mobility dataset containing verbs of motion such as "meet", "avoid" and "follow." In route directions, the system can successfully follow 67% and 68% of the directions for two different environments, significantly outperforming a baseline that uses only landmark phrases. We also tested our approach with an exploration-based algorithm that does not have a map of the environment *a priori*, showing that spatial relations improve performance in unknown environments. In the inspection domain, the highest performing model successfully followed 40% of the directions and up to 70% for two direction-givers. In the indoor mobility domain, the system followed 70% of the commands and 60% with automatic object recognition.

4.1 Approach

Recall from Chapter 1.3 that the general approach that we take to understanding commands is to take as input the language and a map of the environment, and compute the minimum cost plan that corresponds to this language. More formally, if we have a set of directions Z that command the robot to navigate through the environment:

> With your back to the windows, walk straight through the door near the elevators. Continue to walk straight, going through one door until you come to an intersection just past a white board. Turn left, turn right, and enter the second door on your right (sign says "Administrative Assistant").

and a map of the environment m, then the goal is to find the lowest cost path, Γ, that corresponds to a plan for the robot that obeys the command Z:

$$\underset{\Gamma}{\operatorname{argmin}}\, C(\Gamma|Z, m) \tag{4.1}$$

In this chapter, we use a probabilistic model in order to ground the language Z in a path Γ through a map m:

$$C(\Gamma|Z,m) \triangleq -\log p(\Gamma|Z,m) \tag{4.2}$$

Instead of learning directly in the space of paths Γ, which would require parametrizing the set of paths that the robot can take, we use Bayes rule to rewrite the problem as the probability of the language Z given the grounding Γ. This enables the system to learn over a discrete set of output classes Z given a path Γ (i.e., features of the path), which leads to a model that can learn over a continuous valued Γ. By rewriting this distribution using Bayes rule, we can define the Generative Grounding Model (GGM) (because we are learning a generative model of the language) as:

$$p(\Gamma|Z,m) \propto p(Z|\Gamma,m) \times p(\Gamma|m) \tag{4.3}$$

The question is now how to take an arbitrary command Z with task-constrained words, an arbitrary map of the environment, and a path Γ and learn Equation 4.3 over the arbitrary words and phrases in spatial language.

4.2 The Structure of Spatial Language

In order to model arbitrary spatial commands Z, the system will decompose the learning and inference problems into independent components, and build these independent components into an inference about the most likely plan for a robot. This decomposition uses the the structure of spatial language, breaking an arbitrary spatial command into clauses that correspond to landmark descriptions and actions that the agent is meant to execute [Denis et al., 1999]. For example, for the command "Go through the double doors and go down the hallway." there are two clauses that represent the actions the agent is meant to execute and the descriptive landmarks that a robot is meant to see: "go through the double doors" and "go down the hallway." Each of these clauses breaks down into a figure, verb, spatial relation, and a landmark [Tversky and Lee, 1998]. For example, a command such as "go down the hallway" would have a figure that is implicitly "(you)", a verb "go," a spatial relation "down," and landmark "the hallway." Imperative verbs such as "go," "put," "turn right" or "turn left" tell a person what to do or how to do it. For the route following domain, these verbs were generally variants of "go straight," "turn right" or "turn left" (see Figure 4-1). Non-static spatial relations such as "past" and "through" describe how an agent plan should appear relative to these landmarks. Figure 4.1 shows a list of the spatial relations in English. Landmarks are objects or places that a person is meant to see along the path to the destination region.

The spatial language collected as a part of this thesis is challenging because the language has only been constrained to the task and because people have different linguistic capabilities and representations of space. These aspects lead people to give very different language for the same task [Denis et al., 1999], and which is generally either understandable or almost unintelligible depending on these factors (people are

either notoriously good or bad at giving directions). People with good spatial abilities seem to be generally able to give efficient route directions, which have fewer directional errors, hesitations and requests for assistance [Denis et al., 1999, Vanetti and Allen, 1988].

In addition, there are three aspects of spatial language that are extremely helpful for decomposing the model [Denis et al., 1999]. The first is that the natural language description and the described route share a sequential structure, which means that the sequence of actions in the language corresponds directly to a sequence of actions that are meant to happen temporally in the environment[1]. Secondly, even though descriptive and declarative statements are intertwined, they remain functionally distinct, which means that each component of the command can be understood independent of the other components (e.g., "turn right" and "at the intersection" are functionally separate). This is a form of conditional independence, which will enable us to perform learning and inference independently for components of the command. Finally, landmarks are a significant part of route directions, which means that a system that cannot reason broadly about landmarks is unlikely to be widely applicable.

4.2.1 Spatial Description Clauses

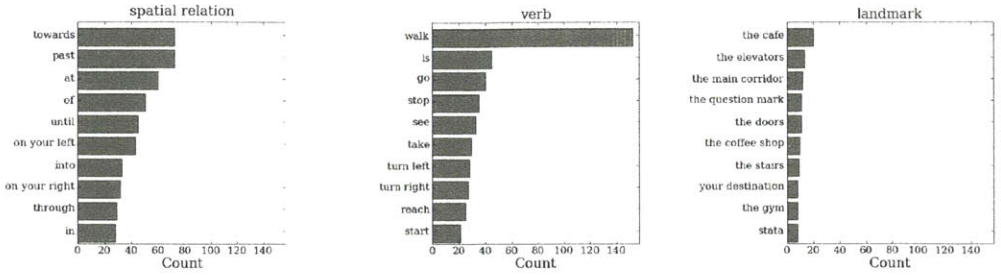
In order to exploit this structure, we have developed a semantic formalism called a spatial description clause (SDC). Following the structure stated before, each SDC corresponds to a clause in the language and itself consists of a *figure* (the subject of the sentence), a *verb* (an action to take), a *landmark* (an object or place in the environment), and a *spatial relation* (a geometric relation between the landmark and the figure). Any of these fields can be unlexicalized and therefore only specified implicitly. For example, in the sentence "Go down the hallway," the figure is an implicit "you," the verb is "go," the spatial relation is "down" and the landmark is "the hallway."

SDCs are also hierarchical. For the sentence "Go through the set of double doors by the red couches," the top level SDC has a verb, "go," a spatial relation, "through," and a landmark, "the set of double doors by the red couches," while the landmark contains a nested SDC with figure "the set of double doors," spatial relation "by" and landmark "the red couches." Figure 4-2(a) shows the hierarchy of SDCs for a sentence in our corpus.

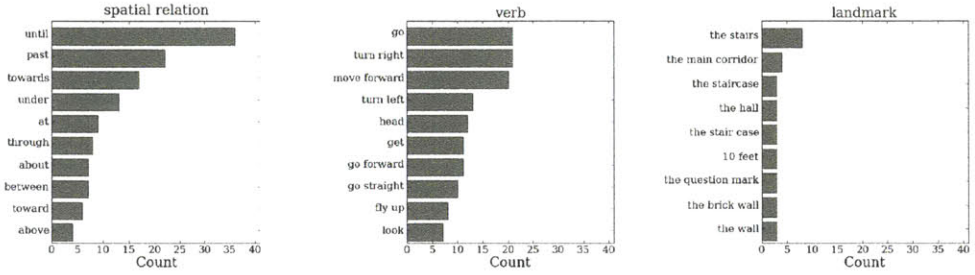
We annotated the text of 150 directions in our corpus with the SDCs in order to verify that they are capable of capturing the linguistically expressed structure of the directions and found that by enabling the system to represent both actions and landmark descriptions, the meaning of each of the clauses was correctly preserved [Denis et al., 1999]. In particular, the annotated SDCs corresponded to the intended meaning of the command with very few orphaned words (7.29%), virtually all stop words.

Figure 4-1 shows the top ten words that appear in each field of an SDC for the three corpora presented in this chapter, showing the variety of words used in the corpus.

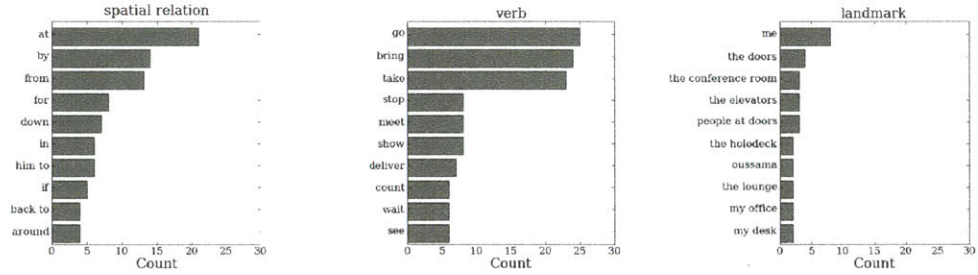
[1] Appendix A describes the structure of spatial language in our corpus in greater detail. We have found that the actions/language are sequential in 99.6% of the instances in one of our corpora.



(a) Route Following



(b) Inspection



(c) Indoor Mobility

Figure 4-1: A histogram of the most frequent words in each of three corpora for the spatial relation, verb, and landmark of the SDCs. In (a) is route following, in (b) is visual inspection and in (c) is indoor mobility. See Appendix A for examples from each of the corpora.

For route instructions, we can see that motion verbs such as "go," "turn right," and "turn left" are most frequent. In addition, we can see that people frequently refer to a "hallway" and "doors." In the visual inspection domain, which involved commanding a micro-air vehicle, we can see that commands involved new prepositions such as "under," and "above" as well as verbs such as "fly up," which require reasoning about the three-dimensional structure of the environment. For the indoor mobility domain, which involved more general commands, we can see motion verbs that involve other agents, such as "meet," "bring" and "take."

Some clauses could not be parsed into the SDC formalism, such as commands not

to do something, multi-argument verbs, and ambiguous prepositional phrase attachment. Although a number of these limitations will be addressed in Chapter 5, we have found that SDCs capture important parts of the semantics of route commands and are able to be extracted efficiently (e.g., a sentence can be parsed in under a second).

$SENTENCE(SDC(v = \texttt{Continue to walk},$
$\quad r = \texttt{straight}),$
$\quad SDC(v = \texttt{going},$
$\quad\quad r = \texttt{through},$
$\quad\quad l = \texttt{one door}),$
$\quad SDC(v = \texttt{going},$
$\quad\quad r = \texttt{until},$
$\quad\quad l = SDC(f = \texttt{you},$
$\quad\quad\quad l = SDC(v = \texttt{come},$
$\quad\quad\quad\quad r = \texttt{to},$
$\quad\quad\quad\quad l = SDC(f = \texttt{an intersection},$
$\quad\quad\quad\quad\quad r = \texttt{just past},$
$\quad\quad\quad\quad\quad l = \texttt{a whiteboard})))))$

(a) Ground Truth

$SENTENCE(SDC(v = \texttt{Continue to walk straight}),$
$\quad SDC(v = \texttt{going},$
$\quad\quad r = \texttt{through},$
$\quad\quad l = \texttt{one door}),$
$\quad SDC(r = \texttt{until},$
$\quad\quad l = \texttt{you come to an intersection}),$
$\quad SDC(r = \texttt{just past},$
$\quad\quad l = \texttt{a whiteboard}))$

(b) Automatic

Figure 4-2: Ground-truth and automatically extracted SDCs for the sentence, "Continue to walk straight, going through one door until you come to an intersection just past a white board." Here, *SENTENCE* is the entire sentence, *SDC* is a spatial description clause, F is the figure, V is the verb, SR is the spatial relation, and L is a landmark.

4.2.2 Parser

We would like to develop a parser that converts from natural language into the parse structure of SDCs. In order to keep the parsing process relatively simple and robust, we make the approximation that a spatial language command can be represented as a flat sequence of SDCs (rather than the hierarchy). This is reasonable, because spatial language tends to have a sequential structure, which means that the sequence

of actions in the language corresponds directly to a sequence of actions that are meant to happen temporally in the environment [Denis et al., 1999, Tversky and Lee, 1998]. Although we may lose some meaning, as when a single landmark element contains the text "the door near the elevators", we still retain the meaning that a robot is meant to go to a place that relates doors and elevators. Figure 4-2(b) shows the flat sequence of SDCs generated by the parser for one sentence. Although it lacks the hierarchical structure of the annotated data (as in Figure 4-2(a)), it segments the key components of each phrase.

In order to extract SDCs from a natural language command, we have trained a conditional random field (CRF) chunker [Kudo, 2009]. The CRF labels each word in a command with one of the four possible fields (*figure*, *verb*, *spatial relation* and *landmark*), or none. A greedy algorithm groups continuous chunks together into SDCs. The CRF was trained on a different corpus of route instructions from the one used in our evaluation. On the test corpus, 60% of the parsed SDCs were found to correspond exactly to the hand-annotated ground truth. To measure inter-annotator agreement, a second person annotated the SDCs in our corpus, and also had 60% agreement. When the automatic algorithm makes a mistake, it is usually a boundary problem, for example including the spatial relation and landmark, but excluding the verb. In these cases, the annotations still contain structured information that can be used to follow the directions.

4.3 The Generative Grounding Model (GGM)

We can now formally define the approach. If we have a set of directions Z, such as one to command the robot to move from one location to another:

> With your back to the windows, walk straight through the door near the elevators. Continue to walk straight, going through one door until you come to an intersection just past a white board. Turn left, turn right, and enter the second door on your right (sign says "Administrative Assistant").

and a map of the environment m, then the goal is to find the lowest cost path Γ that minimizes a cost function C:

$$\underset{\Gamma}{\operatorname{argmin}}\, C(\Gamma|Z,m) \tag{4.4}$$

As before, we can now ground the language Z in a path Γ through a map m as follows:

$$C(\Gamma|Z,m) \triangleq -\log p(\Gamma|Z,m) \tag{4.5}$$

However, we can now leverage the structure of spatial language, specifically that an unstructured natural language command Z breaks down into a sequence of SDCs: $SDC_1 \ldots SDC_N$, where each component SDC_i will correspond to a part of the overall

path Γ, each of which we will call a sub-path γ_i. This means that we can break the overall path Γ into a sequence of sub-paths γ_i, writing:

$$p(\Gamma|Z,m) \propto p(Z|\Gamma,m) \times p(\Gamma|m) \tag{4.6}$$

$$= \prod_i p(\text{SDC}_i|\gamma_i,m) \times p(\Gamma|m) \tag{4.7}$$

We define the following variables:

- λ_i^f The text of the figure field of the i^{th} SDC.
- λ_i^v The text of the verb field of the i^{th} SDC.
- λ_i^r The text of the spatial relation field of the i^{th} SDC.
- λ_i^l The text of the landmark field of the i^{th} SDC.
- Γ The path that the robot takes in map m.
- $\gamma_i \in \Gamma$ The sub-path of the path Γ associated the i^{th} SDC.

For a phrase such as "through the door," λ_i^r is "through," λ_i^l is "the door" and γ_i is the the path of the robot that goes "through" the some object corresponding to "the door".

As we noted in Section 4.2, even though descriptive and declarative statements are intertwined they remain functionally distinct, which means that each component of the command can be understood independent of the other components (e.g., "turn right" and "at the intersection" are functionally separate). We can therefore reason that each field of the SDC is conditionally independent of the others given the grounding γ_i (e.g., that the grounding of "go" is independent of the grounding of "through the door"). This enables us to factor this distribution as:

$$p(\text{SDC}_i|\gamma_i,m) = p(\lambda_i^f|\gamma_i,m) \times p(\lambda_i^v|\gamma_i,m) \times p(\lambda_i^r|\gamma_i,\lambda_i^l,m) \times p(\lambda_i^l|\gamma_i,m) \tag{4.8}$$

The model has four parts, each corresponding to each field of the SDC, plus the prior probability on $p(\Gamma|m)$. We model this prior as uniform among connected viewpoints in the topological map, together with a constraint that disallows backtracking. This constraint means that the path is not allowed to revisit any location that it has previously visited. The following sections describe the other terms.

4.4 Grounding

To implement our model, we need to learn the probability distributions for each of the components of Equation 4.8. The following sections describe how we compute the probability of figure, verb, spatial relation, and landmark fields of the SDC given a sub-path γ_i of the grounding Γ.

Figure 4-3: A map of the environment used to collect our corpus of natural language directions, together with the automatically extracted roadmap. Each circle is a node in the topological map.

It is worth noting that proposal groundings γ_i are created by searching in a semantic map of the environment that includes the location and shape of objects and places. In order to make inference more tractable we reduced the space of robot plans by inducing a topology on the environment. The system first creates a topological roadmap from the gridmap of the environment and then searches for a path within this graph. The roadmap is created by segmenting spaces based on visibility and detected objects and then extracting a topology of the environment from this segmentation, building on techniques described by Brunskill et al. [2007]. Figure 4-3 shows a floorplan of the environment used in our corpus, together with the nodes and edges in this roadmap. As a robot path extends through each of the nodes, it may take on any of the four cardinal directions, which leads to connections in the topological map that include the Cartesian product of the original topological map connections and the four cardinal directions. This enables the system to use features of the orientation of the robot along a path to differentiate the next correct action. For example, "turn right" might be differentiated from "go straight" only by the fact that the orientation at the end of the path differs by approximately 90 degrees in each case.

4.4.1 Grounding the figure and landmark fields

People refer to a wide variety of landmarks in natural language directions, and use diverse expressions to describe them. In the route following corpus, people utilized more than 150 types of objects and places in the landmark field of the SDC, ranging

from "the door near the elevators" to "a beautiful view of the domes." Other corpora used landmarks such as "the stairs" or "the conference room." (Figure 4-1).

In order to learn the probability of a landmark word given a grounding sub-path, we use the idea of context, as in Chapter 3. By taking a semantic map with the locations of 21 types of known objects, we can compute the set of visible objects from the current sub-path γ_i. This set of objects can then be used as features to predict the location of objects or places corresponding to unknown landmark terms. For example, by seeing an object such as "the monitor," the system can infer that the landmark word "computer" is likely to appear.

In order to learn the first component in Equation 4.8, we assume we have a set of features S that includes features s_i for both detected and undetected objects. These features are computed with respect to the end of the sub-path γ_i. Given a sub-path γ_i of the overall grounding path Γ, and $\phi_i(Z)$ that is true if the object or place corresponding to the landmark phrase Z is visible at the final location in the sub-path γ_i, we can rewrite the problem as:

$$p(\lambda_i^l|\gamma_i, m) = p(\phi(\lambda_i^l)|S(\gamma_i, m)) \tag{4.9}$$

$$= p(\phi(\lambda_i^l)|s_1(\gamma_i, m)\ldots s_K(\gamma_i, m)) \tag{4.10}$$

Note that this is exactly Equation 3.4, where the language Z is replaced with the text of the landmark field λ_i^l or λ_i^f.

In Chapter 3, we showed that tags in the Flickr database often corresponded to objects and places that are correlated in space. This distribution can be learned over the subset of over a million Flickr captions tags that are detectable in the environment. By treating each s_k as a feature and setting $\phi(\lambda_i^l)$ to be true when this language corresponds to the features, we can learn a this distribution. In practice, we have using formulated the learning problem as one of learning a naive Bayes classifier. We have also approximated this distribution by picking the single feature s_k that maximizes the probability in equation 4.10. This is a greedy approximation that enables the system to reason about the single visible object that is most strongly related to the query landmark word.

4.4.2 Grounding the spatial relation field

Spatial relations are a closed class of geometric relations between a landmark and the figure (as in Figure 4-4). The set of spatial relations in English according to Landau and Jackendoff [1993] can be seen in Table 4.1. To model the spatial relation component, we need to compute how well the spatial relation text λ_i^r such as "past" is described by a particular grounding sub-path γ_i. Here we focus on spatial prepositions that describe the properties of a path (e.g., "to" or "through") as opposed to static spatial prepositions (e.g., "at" or "on") that localize an object, since almost all of the most frequent spatial prepositions in the route following corpus describe a path (Figure 4-1). Since the command is usually imperative, the command references the agent as the figure. For example, in "go down the hallway," the figure is implicitly "(you)," as in "you go down the hallway."

about	between	outside
above	betwixt	over
across	beyond	past
after	by	through
against	down	throughout
along	from	to
alongside	in	toward
amid(st)	inside	under
among(st)	into	underneath
around	near	up
at	nearby	upon
atop	off	via
behind	on	with
below	onto	within
beneath	opposite	without
beside	out	

Table 4.1: The set of spatial relations in the English language that are not compound, intransitive, or non-spatial [Landau and Jackendoff, 1993].

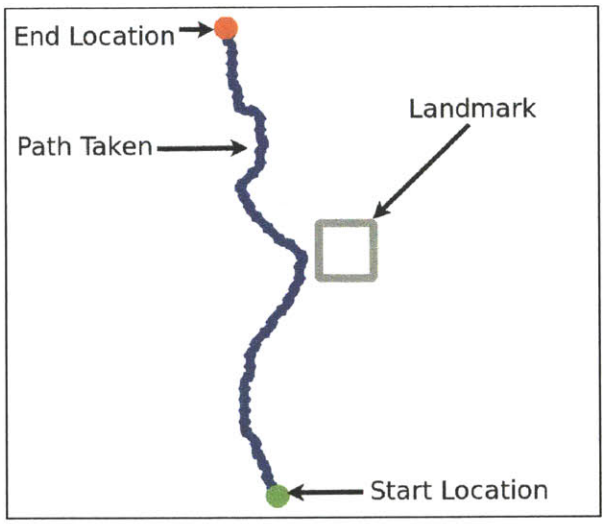


Figure 4-4: An example used to learn the spatial relation "past."

We will model spatial relations by using the path of the agent as the figure relative to a (static) landmark, which is represented as a polygon. For example, in Figure 4-4, the agent takes a path from the start location to the end location relative to the landmark on the right. This is an example of the spatial relation "past."

However, the grounding sub-path γ_i only contains the candidate path of the robot. Therefore, in order to compute the spatial relation component, we must marginalize over candidate landmarks. The probability of the spatial relation text λ_i^r given the grounding sub-path γ_i, a map m and landmark text λ_i^l can be expanded to marginalize over the location and type (e.g. "door") of a candidate landmark, o_j:

$$p(\lambda_i^r = \text{past}|\gamma_i, \lambda_i^l = \text{door}, m) = \sum_{o_j} p(\lambda_i^r = \text{past}|\gamma_i, o_j, m) \times p(o_j|\lambda_i^l = \text{door}, \gamma_i, m) \tag{4.11}$$

The marginalization operation here is meant to pick out good candidate landmarks that correspond to the landmark field text λ_i^l, in addition to picking out paths relative to this landmark that are described well by λ_i^r. We do this because the system does not know which physical door is referred to by the phrase "the door." To speed the inference, the marginalization operation is only performed over the objects or places o_k that are visible from the current sub-path γ_i. To model the second term in Equation 4.11, we again leverage the Flickr dataset.

To model the first term in Equation 4.11, we introduce a set of features S that capture the semantics of spatial prepositions and a correspondence variable ϕ which is true if the path γ_i and object or place o_j corresponds to the language λ_i^r and false otherwise:

$$p(\lambda_i^r|\gamma_i, o_j, m) = p(\phi(\lambda_i^r)|S(\gamma_i, o_j, m)) \tag{4.12}$$

The features for spatial relations are functions of the geometry of the path and landmark. The major axis is the axis that corresponds to the line that is defined by the robot start and end location. The minor axis is defined perpendicular to this and with the origin at the center of the line segment that defines the major axis. Therefore, some example features include:

distFigureEndToGround The distance from the end of the robot path to the closest point on the landmark.

distFigureEndToGroundCentroid The distance from the end of the robot path to the centroid of the landmark.

distFigureStartToGround The distance from the start of the robot path to the centroid of the landmark.

distFigureStartToGroundCentroid The distance from the start of the robot path to the closest point on the landmark.

distFigureEndToGround The distance from the end of the robot path to the closest point on the landmark.

displacementFromGround The difference in distance between the start point of the robot path and the centroid of the ground, and the end point of the robot path and the centroid of the ground (high if the robot is going away from the landmark and low otherwise).

centroidToAxesOrigin The distance between the origin of the axes and the centroid of the landmark.

ratioFigureToAxes The ratio of the distance between the start and end points of the figure and the axes it imposes on the landmark.

angleFigureToAxes The angle between the linearized figure and the line perpendicular to the major axis.

axesLength The length of the axes.

All features are normalized, for example, by the area of the bounding box. More features can be found in Tellex and Roy [2009].

As before, using a set of features s_k we can learn the the parameters of the model in Equation 4.12. Although this learning could take other forms, we learned this distribution as a naive Bayes classifier. The training data comes from a dataset of manually collected examples that represent the spatial relations "across", "along," "through," "past," "around," "to," "out," "towards," "down,", "away from," and "until." ("Until" is not a spatial relation in general, but we modeled it as one here because it almost always refers to an arrival event in the route following corpus, as in "until you come to an intersection just past a whiteboard."). Each example contains a robot path that matched a natural language description such as "past the door" (as in Figure 4-4). Positive examples of one spatial relation were taken to be negative examples of others. Some pairs of spatial relations, such as "to" and "towards," were excluded from each other's training sets.

If we treat the resulting distribution as a classifier for a particular spatial relation, then we can evaluate the performance on a held-out test set from this hand-collected dataset of spatial relations. We can see the performance of the system in Figure 4-5. In general, the system seems to have learned "past," "to," "towards," "across," "until," and "away from." For "along," the system has some difficulty, but is still better than chance (the line in the figure). Further, we can see both high and low-scoring examples for a set of spatial relations in Figure 4-6.

4.4.3 Grounding the verb field

For the route direction and inspection domains there were very few verbs that were not referring to a variant of "go", "turn right" or "turn left." However, when you ask people for more general commands (e.g., the indoor mobility domain) we find that people want robots to interact and reason about other agents. For example, in Figure 4-1, we can see that, in addition to "go," people want the robot to "meet," "bring," and "take" people around the environment. This section is about simple

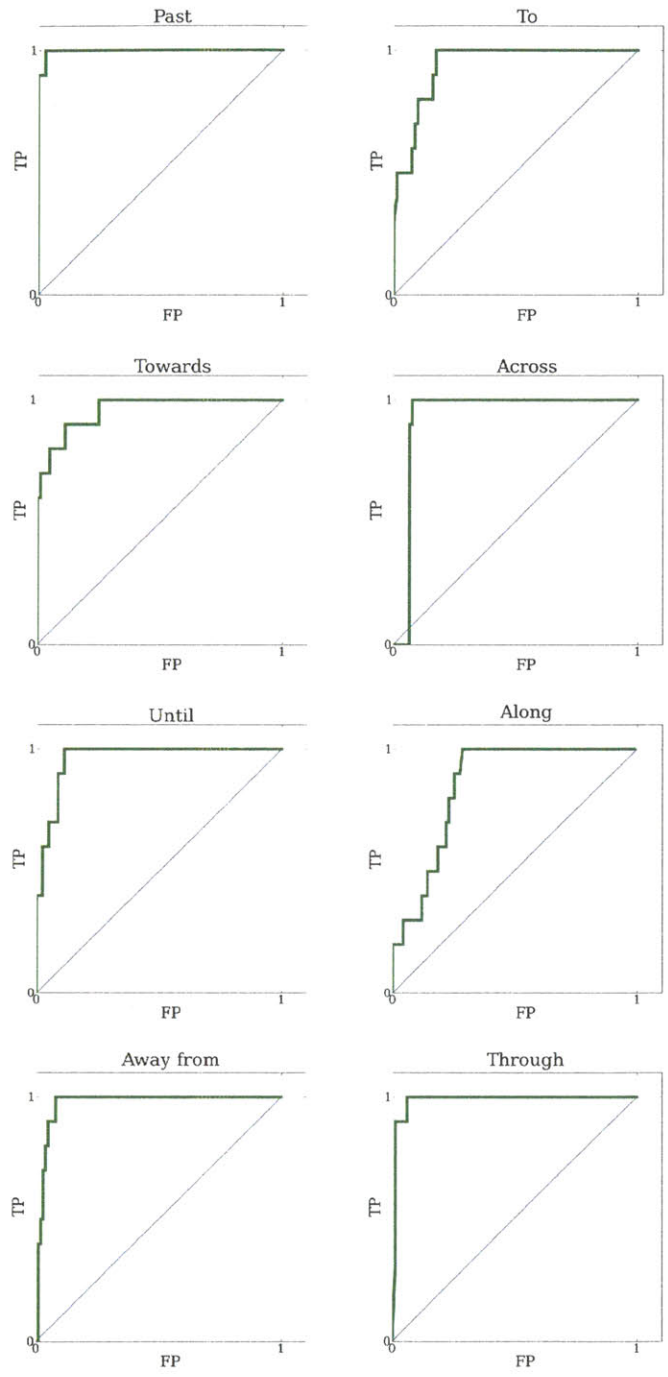


Figure 4-5: The performance on a held-out corpus of examples for the spatial relation component of the model. On the horizontal axis of the ROC curve is the false positive rate (FP) and on the vertical axis is the true positive rate (TP).

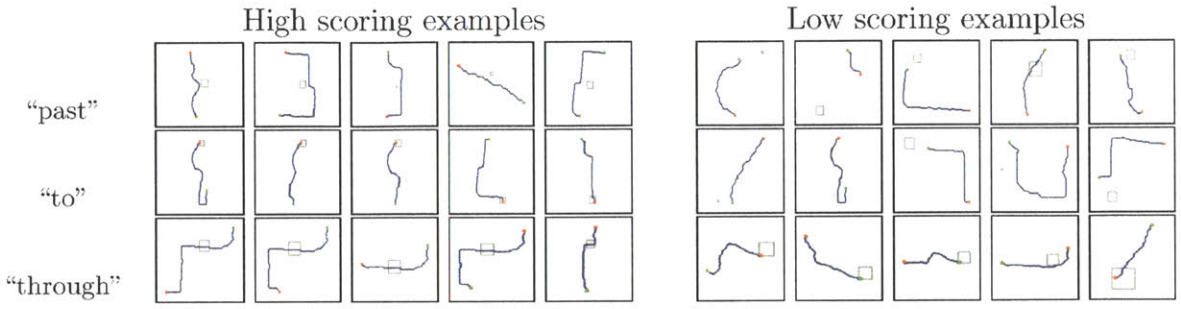


Figure 4-6: Five high scoring and five low scoring examples that were found in our data set for several spatial prepositions.

verbs from the route directions domain and additionally about motion verbs that relate to another agent moving around the environment.

If a verb, such as "go" or "turn right," is only with respect to a single agent (e.g., the action can be understood just in terms of the robot), then we can perform the same learning as before, by extracting a set of features S and introducing a correspondence variable ϕ that determines if the language λ_i^v corresponds to the motion of the robot in the sub-path γ_i, then we can learn this distribution from examples of each motion verb:

$$p(\lambda_i^v = \text{go}|\gamma_i, m) = p(\phi(\lambda_i^v = \text{go})|S(\gamma_i, m)) \tag{4.13}$$

However, when verbs refer to the path of both the person and the robot in a dynamic environment, we must reason about the path of the other agent. For example, if the verb field of the SDC contains text such as "follow," then the grounding γ_i must include both a robot path γ_i^r and an agent path γ_i^a. In this case, we introduce a correspondence variable ϕ that is true when agent and robot sub-paths correspond to the language λ_i^v and false otherwise:

$$p(\lambda_i^v = \text{follow}|\gamma_i^a, \gamma_i^r, m) = p(\phi(\lambda_i^v = \text{follow})|S(\gamma_i^a, \gamma_i^r, m)) \tag{4.14}$$

In order to compute features $S(\gamma_i^a, \gamma_i^r)$, we take a sequence of windows and compute low-level features with respect to these windows. Each feature is computed with respect to the robot paths and person paths for a one-second window with center at time t. These features include:

MovingTowards(γ_i^a, γ_i^r, time) MovingTowards$(\gamma_i^a, \gamma_i^r, t)$ is true when the agent is moving toward the robot at time t

MovingTowards(γ_i^r, γ_i^a, time) MovingTowards$(\gamma_i^r, \gamma_i^a, t)$ is true when the robot is moving toward the agent at time t

IsVisible(γ_i^r, γ_i^a, time) IsVisible$(\gamma_i^r, \gamma_i^a, t)$ is true when γ_i^r is visible from γ_i^a at time t.

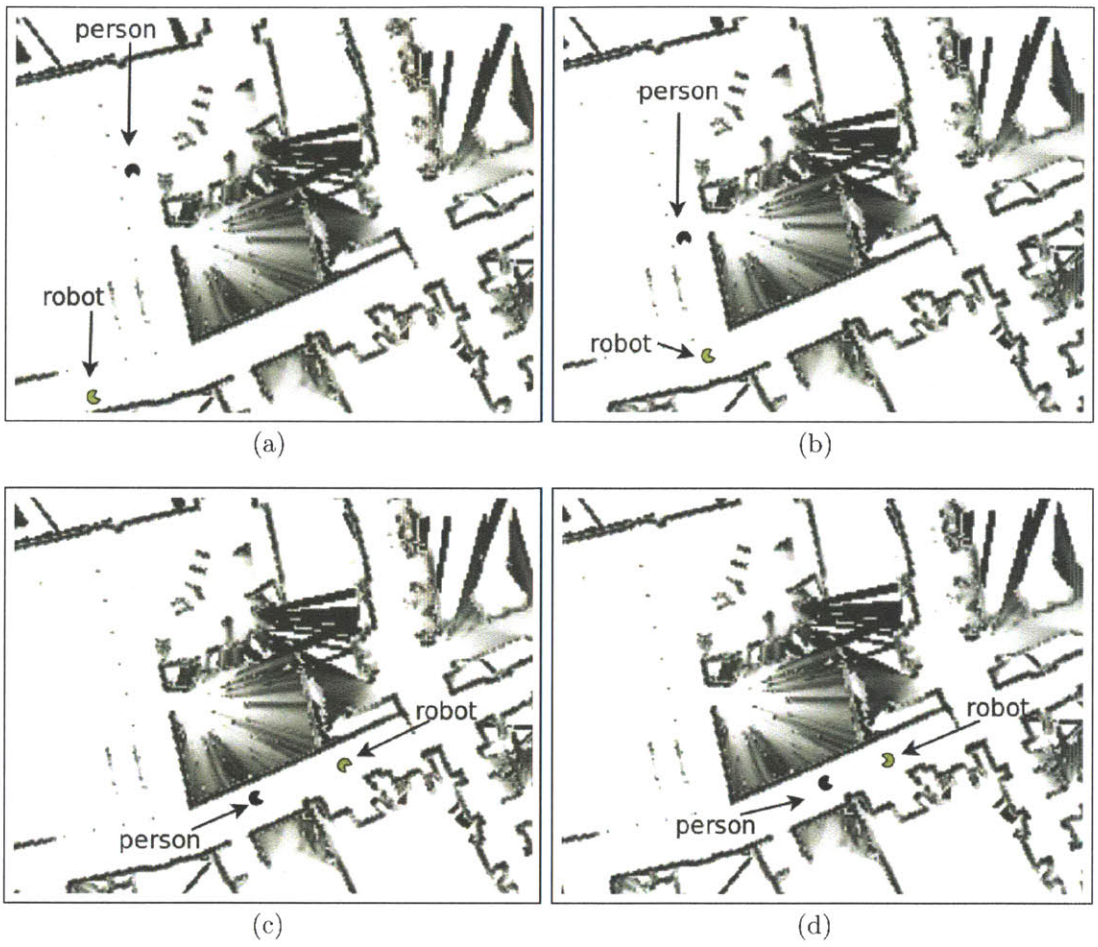


Figure 4-7: An example used to learn the verb "meet."

IsClose(γ_i^r, γ_i^a, time) IsClose$(\gamma_i^r, \gamma_i^a, t)$ is true when the distance between γ_i^r and γ_i^a is less than a set threshold at time t.

IsMoving(γ_i^a, time) IsMoving(γ_i^a, t) is true when γ_i^a is moving at time t.

IsMoving(γ_i^r, time) IsMoving(γ_i^r, t) is true when γ_i^r is moving at time t.

Computing these low-level features results in a vector of feature values, one value for each time $0 < t < T$. We have computed additional low-level features that involve the conjunction of two of the low-level features. Finally, in order to compute features s_i used in the learning, we take the mean of each of the low-level Boolean features g_i (e.g., $g_1(\gamma_i^r, \gamma_i^a, t) = \text{IsClose}(\gamma_i^r, \gamma_i^a, t)$) over all times t:

$$s_i(\gamma_i^a, \gamma_i^r, m) = \frac{1}{T} \sum_{t=1}^{T} g_i(\gamma_i^a, \gamma_i^r, t) \tag{4.15}$$

A corpus of labeled examples for each verb, together with a set S of features s_i, was used to train a naive Bayes classifier. Models were learned for the verbs "meet,"

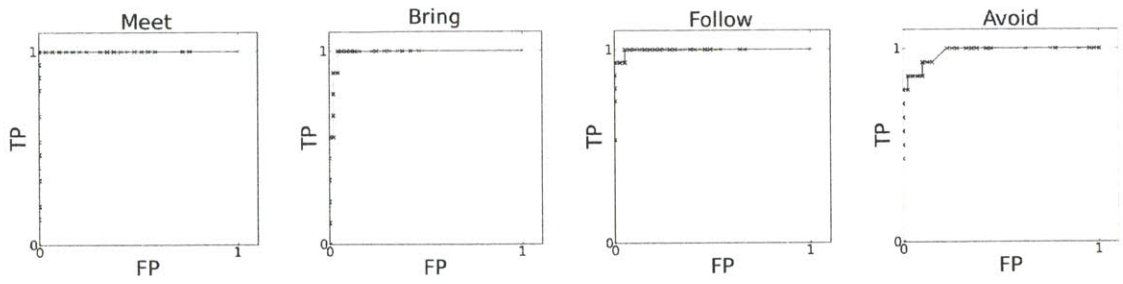


Figure 4-8: ROC curves showing the performance of distributions for various verbs when they are treated as classifiers. TP is true positive rate, and FP is false positive rate. We can see that the system appears to learn the meaning of meet, bring, follow and avoid in our dataset.

"bring," "avoid," "go" and "follow." An example for the verb "meet" can be seen in Figure 4-7. We have also explored the meaning of "straight," "turn left," and "turn right" in terms of the geometry of the path, computing a hand-crafted distribution for route directions. Performance curves for the verbs trained on a corpus of labeled examples are shown in Figure 4-8, showing that we can learn models for each of the verbs in our corpus. In addition, since the features used to learn the verbs are scale-invariant, we expect that this component is likely to generalize to new environments and paths.

4.4.4 Performing Inference

Once the model is trained, we expect that our system should be able to infer paths through any environment since the learning is over domain-independent features. If the robot has explored the entire area *a priori* and has access to a map of the environment, *global* inference searches through all possible paths to find the global maximum of the joint distribution. When a full map is unavailable, the robot uses a greedy *local* inference algorithm that searches for paths using only local information.

Global inference has been performed using a dynamic programming algorithm [Viterbi, 1967] that finds the most probable sequence of viewpoints (nodes the topological map with an orientation) corresponding to a given sequence of SDCs. The algorithm takes as input a set of starting viewpoints, a map of the environment with some labeled objects, and the sequence of SDCs extracted from the directions. It outputs a series of viewpoints through the environment, using the model described above to compute the probability of a transition between two viewpoints. For verbs of motion, since the predictions that the robot may have about the agent may change over time, we must recompute the inferred plan whenever the agent does something unexpected. Candidate paths for the agent and robot are generated using breadth-first search, starting from the current location of both and were evaluated according to Equation 4.14.

The local inference algorithm has access only to parts of the map that it has explored. This algorithm takes as input a partial map of the environment with some labeled objects, a subset of the SDCs in a command and a starting viewpoint. It

outputs a series of viewpoints through the environment that correspond to the subset of the command. In this mode, the system starts with first SDC and a partial map of the environment (its local surroundings) The system then plans the path corresponding to the first part of the command. When an SDC leads to a part of the partial map that the robot has not seen before, it explores this region, incrementally growing the map of the environment. The system performs this search, incrementally adding SDCs, until all the SDCs have been satisfied. Finally, the system picks the best path in the current partial map of the environment to be the path commanded and drives to the final destination. We expect global inference to perform better because it searches through all possible paths to find the one that best matches the descriptions. However, the local inference is more practical for a real robot, because it does not require the robot to have built a complete map of the environment before following directions.

4.5 Evaluation on Route Directions

In order to evaluate the robustness of our system, we measured its performance at following natural language route directions from our corpus. Because subjects used many different starting orientations for a given route, we gave the system access to a set of all reasonable starting orientations that subjects used for each route. We evaluated how close the system got to the final destination at the final location along the route, counting a command as correct if the final location was less than 10 meters from the destination. These results are shown in Table 4.2. Ten meters was selected because it is qualitatively close to the final destination and allows us to compare to human performance on this dataset.

	% correct
Human Performance	85%
With prior map (global)	67%
Wei et al. [2009]	53%
Last SDC only	48%
Without prior map (local)	40%
Random	0%

Table 4.2: The performance of various models at 10 meters on 150 directions from the first half of the study, which was collected on the 8th floor of the Stata Center at MIT.

The first thing we notice is that humans are not perfect at giving commands. In both datasets we found that approximately 15% of the commands were un-followable. In addition, we found that people were either very good or very poor at giving route directions (see Appendix A.1.5). The first baseline we implemented was (*Random*), which is the distance between the true destination and a randomly selected viewpoint.

The second (*Last SDC*) returns the location that best matches the last SDC in the directions. The third baseline (*Landmarks Only*) corresponds to the method described by Wei et al. [2009], which performs global inference using landmarks visible from any orientation in a region, and no spatial relations or verbs. Our global inference model significantly outperforms these baselines, while the local inference model still has a large gap between its performance and that of the global and human performance.

We have also tested the system in a different environment (see Figure A-1 for the different floorplans) using exactly the same model as for another floor. We found that the system is able to follow directions with approximately the same performance as the first environment in Table 4.3. Although our system is approaching human performance, there is still a gap.

	% correct	
	Environment 1	Environment 2
Human Performance	85%	86%
System Performance (global)	67%	68%

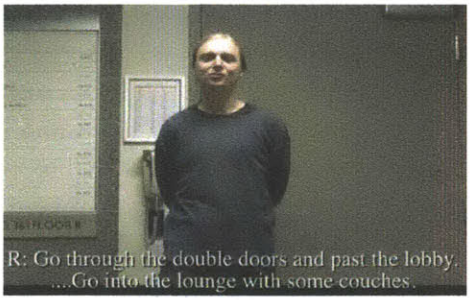
Table 4.3: The performance of our models at 10 meters over all commands on the 8th (Environment 1) and 1st floor (Environment 2) of the Stata Center at MIT. Here we compare both datasets to human performance.

The most significant improvement in performance over the system from Wei et al. [2009] comes from the model of verbs as described in Section 4.4.3, suggesting that the combination of verbs and landmarks is critical for understanding natural language directions. We were surprised that a simple model of verbs involving only left, right, and straight, caused a large improvement compared to the effect of spatial relations. We have also demonstrated our system on a robotic wheelchair in Figure 4-9.

4.6 Evaluation on Inspection

We have also evaluated the system on inspection commands (see Appendix A.1.2). In these experiments, people were asked to give commands to a micro-air vehicle (MAV) in order to inspect certain objects in the environment. This involved commanding the robot to navigate from a starting location and to a position where the robot could see the object.

We evaluate how well the system performed by the proportion of inferred paths that terminate within a certain 2D distance of the target object. Figure 4-10 shows this metric for each subject as the distance is increased. Performance varied widely for different subjects: the system successfully followed 71% of the directions from two subjects to within 10 meters of the object (subjects A and B), but none from another (subject G). For comparison, we also show the overall performance of an algorithm that randomly chooses a final destination.



(a) giving directions

(b) inferring double doors

(c) going into lounge

(d) past the stairs

(e) at destination

Figure 4-9: An example interaction with our system for the navigation command, "Go through the double doors and past the lobby. Go into the lounge with some couches. Enjoy the view over there. Go past the spiral staircase. Continue down the hallway with the cubby holes, but don't go down the hallway. Instead take a right into the kitchen" The system uses *global* inference in this interaction.

Examining the differences in the types of language produced by the subjects shows that the system performed best when the subjects referred to distinctive landmarks in the environment. Some landmarks can be resolved unambiguously by the system because that type of landmark appears only once in the environment. Others refer to relatively unambiguous landmarks for a human, but are not correctly identified by our system due to its inability to reason about adjectives or ordinal numbers such as "square" and "concrete" in "the square concrete column" or "circular" in "circular table." In addition, subjects used multiple scales, which is not currently understood by the system. For example, in every instance where the system failed on subjects A

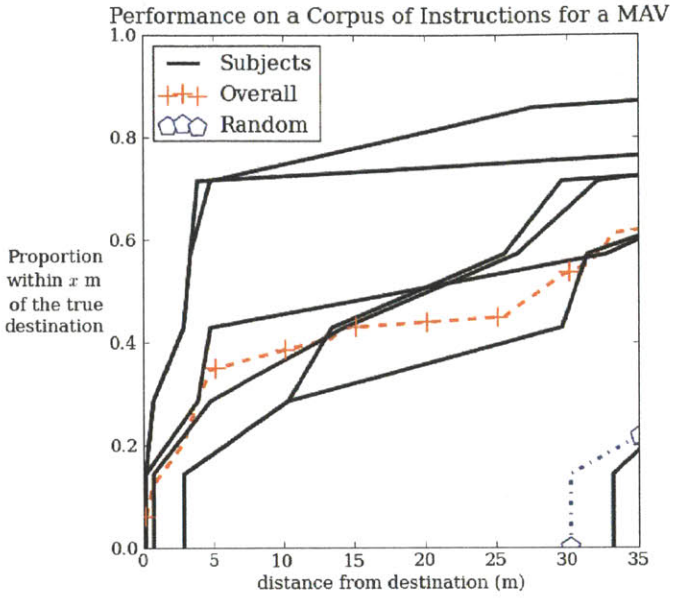


Figure 4-10: System performance for each subject (black) and across all users (dashed red) on a corpus of natural language commands collected for the MAV. Each point represents the proportion of directions followed to within x meters of the true destination. Performance of a system that randomly chooses a destination is shown in blue for comparison.

and B, the directions included phrases such as "Go... the full length of the building" or "Proceed all the way down the hallway."

In the case of subject G, the language used was ungrammatical, had no punctuation, and contained many typographical errors. For example, "you will a starcase to your right [*sic*]." In addition, the language lacked distinctive landmarks; when distinctive landmarks existed, they were often misspelled. (e.g., "Queston Mark [*sic*].") Most landmarks from this subject were distinguished by color and metrical distances such as "between the yellow and red wall" and "2 meters," which are not understood by the current system. Finally, the subject would say "right" when the correct turn direction was "left." Like humans, our system had trouble following directions with this type and frequency of ambiguity and errors.

4.6.1 MAV Demonstration

To demonstrate the overall system, we developed an interface that enabled a person to give directions to the MAV using a speech recognition system or by typing a textual string. Paths computed by the natural language direction understanding module were then executed autonomously by the MAV. For this set of experiments, we assume that the robot has been told about some space names. In particular, for rooms such as "room 124" or "MIT Libraries," we assume that the robot has been told that the name of a space is "124" and that there is a "library" in the environment. Examples

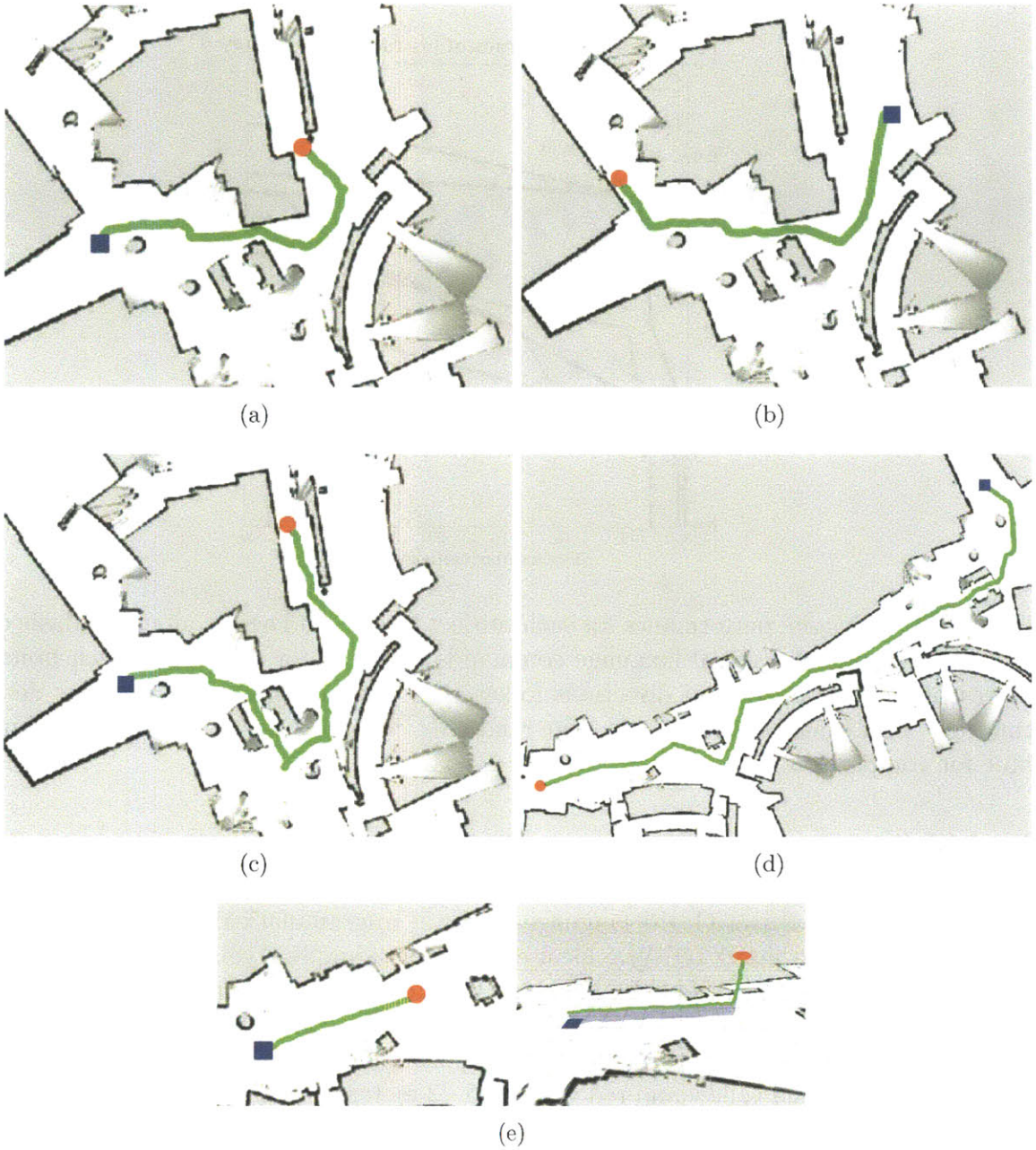


Figure 4-11: Example paths (green) executed by the MAV superimposed on a map. Path start points are marked by blue squares, and end points are marked by red circles. In (a)-(d) the path is shown from an overhead viewpoint. In (e) both an overhead and a perspective view are given to illustrate the elevation change of the MAV. Natural language directions corresponding to each path are given in the text.

of successfully executed paths are shown in Figure 4-11. The directions corresponding to these paths are:

(a) Go past the library and tables till you see a cafe to the left. Fly past the cafe

Figure 4-12: (left) Photographs of the MAV executing an interactive series of instructions. (right) Imagery from the on-board camera, transmitted to the operator as the MAV flies. The commands issued to the MAV are given in the text.

and there will be other eateries. Head into this area.

(b) Stand with your back to the exit doors. Pass the cafe on your right. Make a right directly after the cafe, and into a seating area. Go towards the big question mark.

(c) Go straight away from the door that says CSAIL, passing a room on your right with doors saying MIT Libraries. Turn left, going around the cafe and walk towards the cow.

(d) Turn right and fly past the libraries. Keep going straight and on the left near the end of the hallway there is a set of doors that say Children's technology center. You are at the destination.

(e) Fly to the windows and then go up.

In addition to following individual sets of directions, the vehicle can accept directions interactively, while relaying on-board camera images and LIDAR measurements back to the user in real-time. In one such interaction, the vehicle was commanded to

fly past a classroom, collect some video, and return to the user. As the MAV carries out the directions, on-board sensor data is continuously transmitted back to the user (as seen in Figure 4-12). The commands given were:

Fly past room 124 and then face the windows.
Go up.
Go back down.
Come back towards the tables and chairs.

4.7 Evaluation on Indoor Mobility

In order to evaluate our system on verbs that involve multiple agents in the environment, we used the indoor mobility corpus (see Appendix A.1.3). We evaluated our system on directives that were constructed around combinations of five motion verbs: "go," "meet," "follow," "bring," and "avoid." For each verb, we collected an evaluation corpus that consisted of approximately ten primitive natural language directives such as, "Follow the person to the kitchen." and compound commands involving several phrases such as, "Follow the person to the kitchen. Then move towards the bathroom. Next go down the hall to the lounge."

On held-out data from this corpus, we used our inference algorithm to control the robot's activity, given the same information a person had when creating the corpus. When the behavior of the robot was judged to have followed the natural language command, then we counted it as correct. When any part of the robot's behavior did not correspond to the command, then the behavior was judged to be incorrect. A breakdown of the performance over different verbs can be seen in Table 4.4. In addition, using annotated object detections, we evaluated the system on 10 compound commands, and found that comparable performance (60%) on commands involving two or more verbs.

verb	% correct
Go	90%
Follow	80%
Avoid	78%
Meet	70%
Bring	29%

Table 4.4: Breakdown of the performance on specific verbs for on 46 commands and annotated object detections.

To include the effects of potentially noisy object recognition, we also present results for creating this semantic map automatically. For the automatically generated detections, we used the wheelchair robot depicted in Figure 1-1(d), equipped with a camera and LIDAR to drive around the environment. Then we used an object detector [Felzenszwalb et al., 2008] to detect six types of objects to seed the map.

Figure 4-13: An object detection of a microwave from the environment.

Figure 4-13 shows the object detector automatically detecting a microwave in the environment. Results comparing an annotated semantic map to one where visual object recognition is used are shown in Table 4.5.

map type	% correct
Hand	69%
Auto	60%

Table 4.5: Percentage of time that our algorithm followed the language exactly for various command sets. *Hand* shows an experiment with 46 commands and annotated object detections. *Auto* shows results for 10 commands and a fully automatic object detector.

The verb "bring" performs much worse than the others in our test set. This disparity occurs because "bring" events are longer and contain significant non-Markov information. We hypothesize that one reason is because our algorithm cannot search deeply enough to find a complete "bring" event in the time available. Optimizing and parallelizing our system to enable it to search deeper in the plan space could alleviate this problem.

4.8 Conclusion

In Chapter 4, we have take steps toward building a robust spatial language understanding system for three different domains: route directions, visual inspection, and indoor mobility. We take advantage of the structure of spatial language in order to decompose it into clauses and to decompose those clauses into components. One of the contributions of this chapter has been that by using a decomposition of the spatial language into spatial description clauses (SDCs), we can enable the system to understand each component of spatial language independently, building a plan for the robot from its component parts.

A second key contribution of this section is to learn the mapping between the language and robot plans. Instead of manually specifying the meaning of "the doors," "to," or "follow" in the robot's representation, we learn the mapping between the two. This allows people to use arbitrary combinations of words in order to command the robot and does not require manual specification of word meaning. The learning in this chapter takes the form of a probabilistic graphical model that is factored into three key components. The first component grounds novel noun phrases such as "the computers" in the perceptual frame of the robot by exploiting object co-occurrence statistics between unknown noun phrases and known perceptual features using the model from Chapter 3. Second, a spatial reasoning component judges how well spatial relations such as "past the computers" describe a plan, including spatial prepositions that are necessary for 3-D environments. Third, a verb understanding component judges how well spatial verb phrases such as "follow", "meet", "avoid" and "turn right" describe a plan. Once trained, our model requires only a metric map of the environment together with the locations of detected objects in order to follow directions through it. This map can be given *a priori* or created on the fly as the robot explores the environment. We have demonstrated our system on both a robotic wheelchair and a micro-air vehicle.

We have evaluated the approach using three different corpora. First, we have evaluated the system on over 150 natural language route instructions on two different floors of a building. Second, we evaluated the system on forty-nine 3-D inspection commands for a micro-air vehicle. Finally, we have evaluated on an indoor mobility dataset containing verbs of motion such as "meet", "avoid" and "follow." In route directions, the system can successfully follow 67% and 68% of the directions in two different environments, significantly outperforming a baseline that uses only landmark phrases. We also tested our approach with an exploration-based algorithm that does not have a map of the environment *a priori*, showing that spatial relations improve performance in unknown environments. In the inspection domain, the highest performing model successfully followed 40% of the directions and up to 70% for two direction-givers. In the indoor mobility domain, the system followed 70% of the commands and 60% with automatic object recognition. We found that our system is able to understand verbs of motion such as "follow," "avoid," "meet," "bring," and "go."

Chapter 5

A Model for Grounding Mobile Manipulation Commands

In this chapter, we expand the set of language that the system can understand to mobile manipulation commands such as, "put the tire pallet on the truck." These commands are difficult to understand because they may involve more than one argument (e.g., "the tire pallet" and "on the truck") and need to be understood hierarchically. The reader may recall that in Chapter 4, although spatial description clauses (SDCs) were introduced hierarchically, at inference time the system used a flat sequence of SDCs to understand a natural language command. This led to a flat model that grounded each component of spatial language. For example "put the tire pallet on the truck" might be parsed to the spatial description clause: "V:put, L:the tire pallet on the truck." This would lead to the wrong meaning of this phrase, since the verb "put" does not operate on "the tire pallet on the truck," it operates on "the tire pallet" and moves it to "the truck".

The first contribution of this chapter is the Generalized Grounding Graph (G^3), which connects language onto grounded aspects of the environment. Unlike Chapter 4, we relax the assumption that the language has fixed and flat structure and provide a method for constructing a hierarchical probabilistic graphical model that connects each element in a natural language command to an object, place, path or event in the environment. The structure of the G^3 model is dynamically instantiated according to the compositional and hierarchical structure of the command, enabling efficient learning and inference.

The second contribution of this chapter is a discriminative instantiation of the Generalized Grounding Graph. In Chapter 4, Equation 4.3, the learning problem was re-written using Bayes rule, which led to a generative model of the language given the plan. In this chapter, we propose a discriminative model that learns the mapping from language directly onto a robot plan. By introducing a correspondence vector which determines when a component of the language and the plan correspond, the system is able to ground commands in the robot's representation without requiring a

Parts of this chapter were performed with Stefanie Tellex, Steven Dickerson, Matthew Walter, and Ashis Banerjee

Commands from the corpus
- Go to the first crate on the left and pick it up.
- Pick up the pallet of boxes in the middle and place them on the trailer to the left.
- Go forward and drop the pallets to the right of the first set of tires.
- Pick up the tire pallet off the truck and set it down.

Table 5.1: Sample commands from the domain created by untrained human annotators. Our system is able to successfully follow these commands.

discrete plan space.

The G^3 model described in this chapter is trained on a corpus of natural language commands paired with groundings for each part of the command, enabling the system to automatically learn meanings for words in the corpus, including complex verbs such as "put" and "take." We evaluate our model in the domain of mobile manipulation and navigation commands given to a robotic forklift, although approach generalizes to any domain where linguistic constituents can be associated with specific actions and environmental features. Some examples of commands which our approach correctly follows can be seen in Table 5.1.

5.1 Approach

Recall from Chapter 1.3 that the general approach we take to understanding commands is to take as input the language and a semantic map of the environment, and compute the minimum cost plan that corresponds to the language. More formally, if we have a set of directions Z that command the robot to manipulate and move an object:

Put the tire pallet on the truck.

and a semantic map of the environment m, then the goal is to find the lowest cost plan, Γ, that obeys the command Z:

$$\underset{\Gamma}{\operatorname{argmin}}\, C(\Gamma|Z,m) \tag{5.1}$$

As in Chapter 4, we use a probabilistic model to ground the language Z in a plan Γ through a semantic map m:

$$C(\Gamma|Z,m) \triangleq -\log p(\Gamma|Z,m) \tag{5.2}$$

In this chapter, we would like to learn $p(\Gamma|Z,m)$ directly instead of using Bayes rule (as in Chapter 4, Equation 4.3). Learning $p(\Gamma|Z,m)$ directly is challenging because it involves learning in an arbitrary, continuous plan space. Others have converted the

plan space to a discrete set by defining an action space for the robot [Shimizu and Haas, 2009]. However, this requires a discrete, hand-selected set of actions the robot can execute. Another parametrization of the plan space is to compute a fixed set of plans corresponding to a given environment. However, this leads to environment-specific learning and inference. What is needed is a discriminative formulation of the problem that does not discretize the plan space, but at the same time is able to handle new environments that have a unique spatial layout and arrangement of objects.

We address these issues by introducing a correspondence vector Φ that true when a linguistic term corresponds to the robot plan and false otherwise. By learning over a correspondence vector instead of a discrete plan space, the system is able to extract an arbitrary set of domain-independent semantic and geometric features from the language and plan, which enables the system to perform inference over objects, places, paths and events in the state space of the robot. This alleviates the need for an intermediate action space or an environment-specific plan space and promises to generalize to new environments, since the features used by the system are invariant to scale and orientation (as in Chapter 4).

More formally, the correspondence vector Φ is a Boolean vector that determines when each component of a plan Γ and language Z in a given semantic map m correspond:

$$p(\Gamma|Z, m) \approx p(\Phi|\Gamma, Z, m) \tag{5.3}$$

In order to perform learning and inference using this model, we will need to take advantage of the structure of the the natural language command Z. As in Chapter 4, we will break down spatial language into a semantic structure called a Spatial Description Clause (SDC) and use this as a part of the learning and inference.

5.1.1 Spatial Description Clause

Although Spatial Description Clauses from Chapter 4 capture much of the semantics of spatial language, they do not capture the type of the grounding, which is critical for understanding manipulation commands such as those in Table 5.1. In this chapter, we build off of the SDCs from Chapter 4 by collapsing the *relation* and *verb* fields and adding object types. An SDC now corresponds to a linguistic clause with a *figure* f, a *relation* r, and a variable number of *landmarks* l_i with one of the following types [Jackendoff, 1983]:

- **EVENT** Something that takes place (or should take place) in the world (e.g., "Move the tire pallet").
- **OBJECT** A thing in the world. This category includes people and the robot as well as physical objects (e.g., "Forklift," "the tire pallet," "the truck," "the person").
- **PLACE** Places in the world (e.g., "on the truck," or "next to the tire pallet").
- **PATH** Paths through the world (e.g., "past the truck," or "toward receiving").

$SDC(r = \texttt{Put},$
$\quad l = \texttt{the pallet on the truck})$

(a) Flat Parse

$EVENT(r = \texttt{Put},$
$\quad l = OBJ(f = \texttt{the pallet}),$
$\quad l2 = PLACE(r = \texttt{on},$
$\quad\quad l = OBJ(f = \texttt{the truck})))$

(b) Hierarchical Parse

Figure 5-1: A hierarchical and flat parse tree for the command "Put the pallet on the truck." EVENT corresponds to an EVENT SDC with relation text "put." There are two OBJECT SDCs (OBJ): one is the first argument to the EVENT SDC and has a figure field that contains the text "the pallet." The other has a figure field that contains the text "the truck." Finally, there is a PLACE SDC that is the second argument to the EVENT SDC.

Each EVENT and PATH SDC contains a relation with one or more core arguments. Since almost all relations (e.g., verbs) take two core arguments or less, we use at most two landmark fields l_1 and l_2 for the rest of the chapter.

Given this definition, a general natural language command is represented as a sequence of SDC trees. An SDC tree for the command "Put the pallet on the truck" appears in Figure 5-1(b). Leaf SDCs in the tree contain only text in the figure field, such as "the pallet." Internal SDCs contain text in the relation field and child SDCs in the figure and landmark fields. For example "next to" in "the pallet next to the truck" would correspond to an internal OBJECT SDC with relation "next to" and child OBJECT SDCs with the figure filled for "the pallet" and "the truck."

Although SDCs in Chapter 4 supported hierarchical parses, a flat parse was used at inference time. A flat parse for a command can be seen in Figure 5-1(a). This parse clearly has the wrong interpretation, since "put" does not operate on "the pallet on the truck," it operates on "the pallet" and moves it to the place represented by "on the truck." A flat structure was reasonable for route directions because there was a sequential nature to the commands, which contained only landmarks that the robot was meant to see and actions that the robot was meant to take. For mobile manipulation commands, a hierarchical structure is required to enable the system to represent the relationships between objects, places, paths and events and manipulate and move objects around the environment. A hierarchical parse for the command, "Put the pallet on the truck" can be seen in Figure 5-1(b). We have built and use an automatic hierarchical SDC parser that converts Stanford dependencies into a sequence of SDC trees [Marneffe et al., 2006].

5.1.2 Factor Graphs

In this chapter, we will use a graphical representation of functions called a *factor graph* to visualize the factorization of the probability distribution in Equation 5.3. Factor graphs are an intuitive way to visually expose the dependencies in a probability distribution, even for undirected models [Kschischang et al., 2001].

Assuming where $\Psi_j(X_j)$ is a function taking the elements of $X_j \subset \{x_1 \dots x_n\}$ as arguments and that we have a function:

$$p(x_1 \dots x_n) = \prod_{j \in J} \Psi_j(X_j) \tag{5.4}$$

Then a factor graph for this function can be represented as a bipartite graph having a variable node for each variable x_i and a factor node for each function Ψ_j if and only if x_i is an argument of Ψ_j.

For example we can take the classic Bayesian network that models the joint probability of a sprinkler being on, rain falling outside and the grass being wet:

$$\begin{aligned} &p(\text{sprinkler,rain, grass wet}) \\ &\quad = p(\text{grass wet}|\text{sprinkler, rain}) \times p(\text{sprinkler}|\text{rain}) \times p(\text{rain}) \end{aligned} \tag{5.5}$$

Looking at Figure 5-2, we can see the factor graph for this distribution. This factor graph has three variable nodes, one for each of "sprinkler," "rain," and "grass wet," and three factor nodes. The prior probability $p(\text{rain})$ is represented as the factor node with a single incoming edge that is connected to the variable node "rain." The probability $p(\text{sprinkler}|\text{rain})$ is represented as the factor node with two incoming edges from "sprinkler" and "rain." Finally, $p(\text{grass wet}|\text{sprinkler, rain})$ is represented as a factor node with incoming edges from "grass wet," "sprinkler," and "rain."

Given this factor graph, we can infer that a relationship exists between the incoming variables to a factor node. For example, we might derive the intuition that the grass is wet because its either raining or the sprinkler was on. Alternatively, we might infer that the sprinkler is not on if it is raining. Although a factor graph is not as direct at representing causal relationships as a directed Bayesian network, it is more intuitive when representing relationships between variables in an undirected graphical model (as in this chapter).

5.1.3 Generalized Grounding Graph

We can now present an algorithm for constructing the Generalized Grounding Graph (G^3) according to the linguistic structure defined by a tree of SDCs. We define a plan Γ to be the set of all groundings γ_i for a given command. In order to allow for uncertainty in the groundings, we introduce a binary correspondence vector Φ; each $\phi_i \in \Phi$ is true if $\gamma_i \in \Gamma$ is correctly mapped to part of the natural language command. The induced G^3 for a given command factorizes the distribution from Equation 5.3

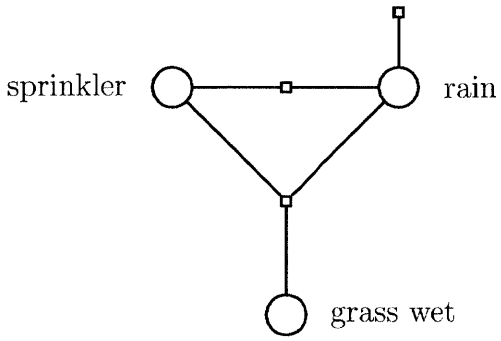


Figure 5-2: A simple factor graph that represents the joint probability distribution from Equation 5.5. The first element corresponds to the factor in the center of the graph. The second element corresponds to the factor between sprinkler and rain. The third element corresponds to a prior over rain and is at top right.

by creating a factor Ψ for each SDC and candidate grounding in the hierarchy:

$$\begin{aligned} p(\Phi|Z,\Gamma,m) =& p(\Phi|\text{SDCs},\Gamma,m) \\ =& \frac{1}{Z}\prod_i \Psi_i(\phi_i,\text{SDC}_i,\Gamma,m) \end{aligned} \tag{5.6}$$

where Z is the normalization constant.

We define the variables in the model as follows:

- ϕ_i True if the grounding γ_i corresponds to i^{th} SDC.
- λ_i^f The text of the figure field of the i^{th} SDC.
- λ_i^r The text of the relation field of the i^{th} SDC.
- $\gamma_i^f, \gamma_i^{l1}, \gamma_i^{l2} \in \Gamma$ The groundings associated with the corresponding field of the i^{th} SDC: the robot or object state sequence or a location in the semantic map.

Looking at Equation 5.6, we can see that the model has a factor for each SDC in the parse. For each leaf SDC or internal SDC, the model has a factor that will take different arguments according to its location in the tree and the number of landmark fields in the SDC. The dynamically generated factors Ψ fall into two types:

- $\Psi(\phi_i, \lambda_i^f, \gamma_i)$ for leaf SDCs.
- $\Psi(\phi_i, \lambda_i^r, \gamma_i^f, \gamma_i^{l1})$ or $\Psi(\phi_i, \lambda_i^r, \gamma_i^f, \gamma_i^{l1}, \gamma_i^{l2})$ for internal SDCs.

Leaf factors always correspond to an OBJECT or PLACE SDC and operate over the correspondence variable ϕ_i, the figure text λ_i^f and a unique grounding γ_i. An internal factor corresponds to an OBJECT, PLACE, PATH, or EVENT SDC which always has text in the relation field. These operate over the correspondence variable

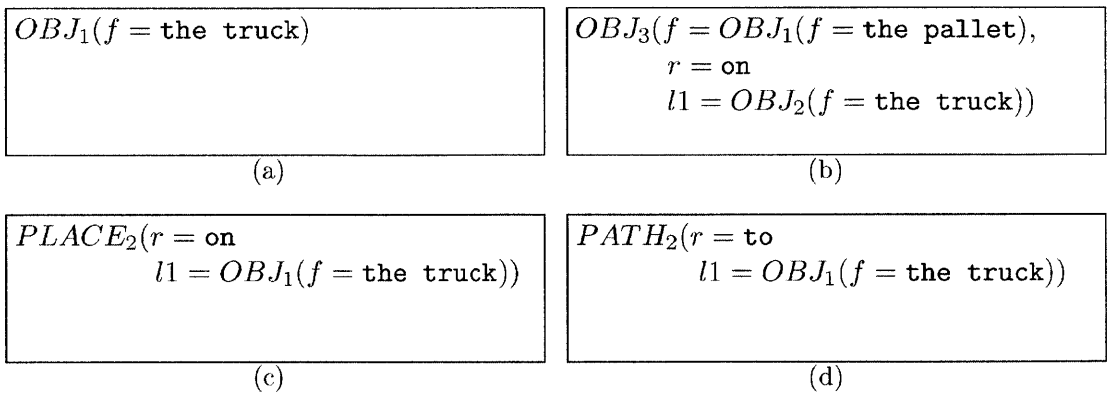


Figure 5-3: Four SDC parses. In 5-3(a), the language is "the truck," in 5-3(b) the language is "the pallet on the truck," in 5-3(c) the language is "on the truck," and in 5-3(d) the language is "to the truck." The corresponding generalized grounding graphs for each parse are in Figure 5-4(a-d), respectively.

ϕ_i, relation text λ_i^r, and the candidate groundings γ_i^f and γ_i^{l1} (and optionally γ_i^{l2}) corresponding to the figure and landmark fields of an SDC. Each of the factors from Equation 5.6 is modeled as a conditional random field in which each factor Ψ takes the following form [Lafferty et al., 2001]:

$$\Psi_i(\phi_i, \text{SDC}_i, \Gamma) = \exp\left(\sum_k \mu_k s_k(\phi_i, \text{SDC}_i, \Gamma)\right) \tag{5.7}$$

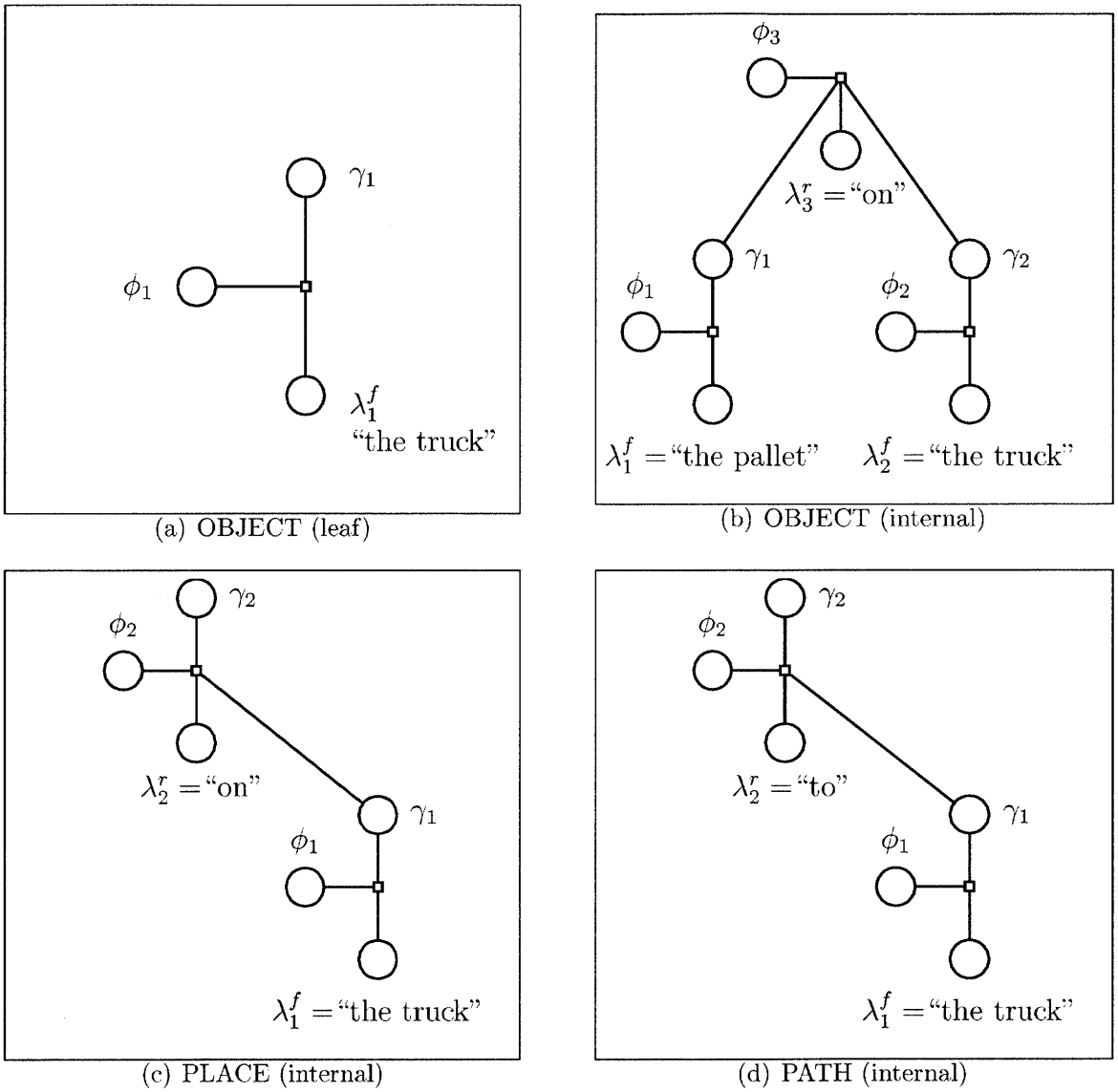
Here, s_k are feature functions that take as input the correspondence variable, an SDC and a set of groundings and output a binary decision. The μ_k are the weights corresponding to the output of a particular feature function.

At training time, we observe SDCs, groundings Γ, and the output variable Φ. In order to learn the parameters μ_k that maximize the likelihood of the training dataset, we compute the gradient and use L-BFGS to optimize the parameters of the model via gradient descent. When inferring a plan, we optimize over Γ given fixed values for the correspondence vector Φ and the SDCs.

5.1.4 Examples

In Figure 5-4 are four examples of generalized grounding graphs for OBJECT, PLACE, and PATH SDC trees from Figure 5-3. Figure 5-4(a) exhibits a leaf factor, where λ_1^f corresponds to the text "the truck," γ_1 corresponds to a physical object in the robot's representation (e.g., the physical truck), and ϕ_1 is true when the grounding is the same as the text (e.g., the language λ_1^f is "the truck" and γ_1 is the object that corresponds to the truck).

In Figure 5-4(b), the command is "the pallet on the truck." In this case, we have two leaf OBJECT SDCs (as in Figure 5-4(a)). Here λ_3^r, "on," is a part of a factor that relates the groundings for these leaf OBJECT SDCs, γ_1 and γ_2, which are two



(a) OBJECT (leaf)

(b) OBJECT (internal)

(c) PLACE (internal)

(d) PATH (internal)

Figure 5-4: Four different generalized grounding graphs for the four commands from Figure 5-3.

objects corresponding to a pallet and a truck, respectively. Thus, if ϕ_3 is true, then γ_1 and γ_2 should be in a configuration that corresponds to the text "on."

In Figure 5-4(c), the command is "on the truck." In this case, the G^3 model is represented as the place in the environment (γ_2) that is "on" the object named "the truck." We have an OBJECT SDC as in Figure 5-4(a) to ground the truck. Then, λ_2^r is the text "on," γ_2 represents a place in the environment corresponding to "on," and the ϕ_2 is true when the place grounding γ_2 and truck grounding γ_1 correspond to the relation "on."

Finally, in Figure 5-4(d), the command is "to the truck." The difference between this model and Figure 5-4(c) is that now the grounding γ_2 is a path for the robot that corresponds to the text λ_2^r, which is "to." At inference time, the system will search over state sequences of the robot relative to the object, "the truck."

EVENTs are more complex, because in general they may have one or two arguments. Figures 5-5 and 5-6 show the SDC trees and induced factor graphs for two similar commands: "Put the pallet on the truck" and "Go to the pallet on the truck." In the first case, "Put" is a two argument verb that takes an OBJECT and a PLACE. The model in Figure 5-5(b) connects the grounding γ_3 for "on the truck" directly to the factor for "Put." In the second case, "on the truck" modifies "the pallet." For this reason, the grounding γ_4 for "on the truck" is connected to "the pallet."

5.1.5 Features

To train the model the system extracts binary features s_k for each factor. These features enable the system to determine which aspects Γ correctly ground to a given SDC. There are two main types of features used in this model: semantic and geometric.

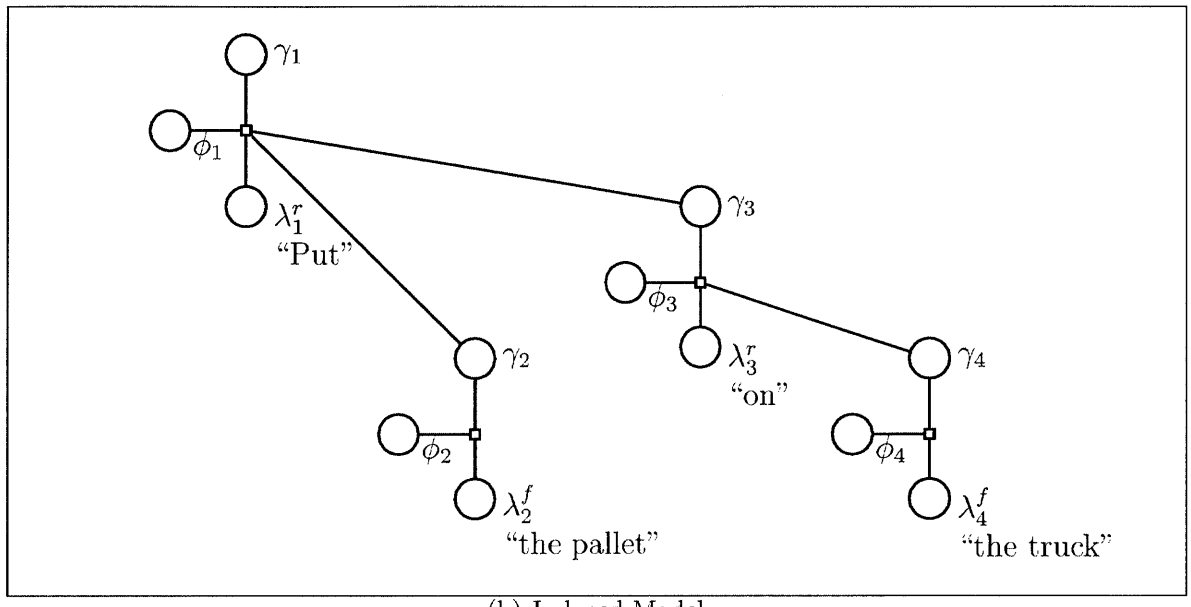
Geometric features generally enable the system to understand relations between paths, events, places, and objects. For a relation such as "on," a natural geometric feature is whether the figure grounding is supported by the landmark grounding. However, the feature $\mathit{supports}(\gamma_i^f, \gamma_i^l)$ alone is not enough to enable the model to learn that "on" corresponds to $\mathit{supports}(\gamma_i^f, \gamma_i^l)$. Instead we need a feature like $\mathit{supports}(\gamma_i^f, \gamma_i^l) \wedge$ "on" $\in \lambda_i^r$. More generally, we implemented a set of base features involving geometric relations between the γ_i. Then to compute features s_k we compute the Cartesian product of base features crossed with the presence of words in corresponding fields of the SDC. Although many features between geometric objects are continuous rather than binary valued, we discretize continuous features into uniform bins.

For OBJECTs and PLACEs, geometric features correspond to relations between two three-dimensional boxes in the world. All continuous features are first normalized so they are scale invariant, and then discretized to be a set of binary features. Examples include

- $\mathit{supports}(\gamma_i^f, \gamma_i^l)$. For "on" and "pick up."
- $\mathit{distance}(\gamma_i^f, \gamma_i^l)$. For "near" and "by."

$EVENT_1(r = \texttt{Put},$
$l = OBJ_2(f = \texttt{the pallet}),$
$l2 = PLACE_3(r = \texttt{on},$
$l = OBJ_4(f = \texttt{the truck})))$

(a) SDC tree



(b) Induced Model

$$p(\Phi|\Gamma, \text{SDCs}) = \frac{1}{Z} \times \Psi_1(\phi_1, \gamma_1, \gamma_2, \gamma_3, \lambda_1^r = \text{Put}) \times \Psi_2(\phi_2, \gamma_2, \lambda_2^f = \text{the pallet}) \times \Psi_3(\phi_3, \gamma_3, \gamma_4, \lambda_3^r = \text{on}) \times \Psi_4(\phi_4, \gamma_4, \lambda_4^f = \text{the truck})$$

(c) Factorization

Figure 5-5: In (a) is SDC tree for "Put the pallet on the truck." In (b) is the induced graphical model and factorization.

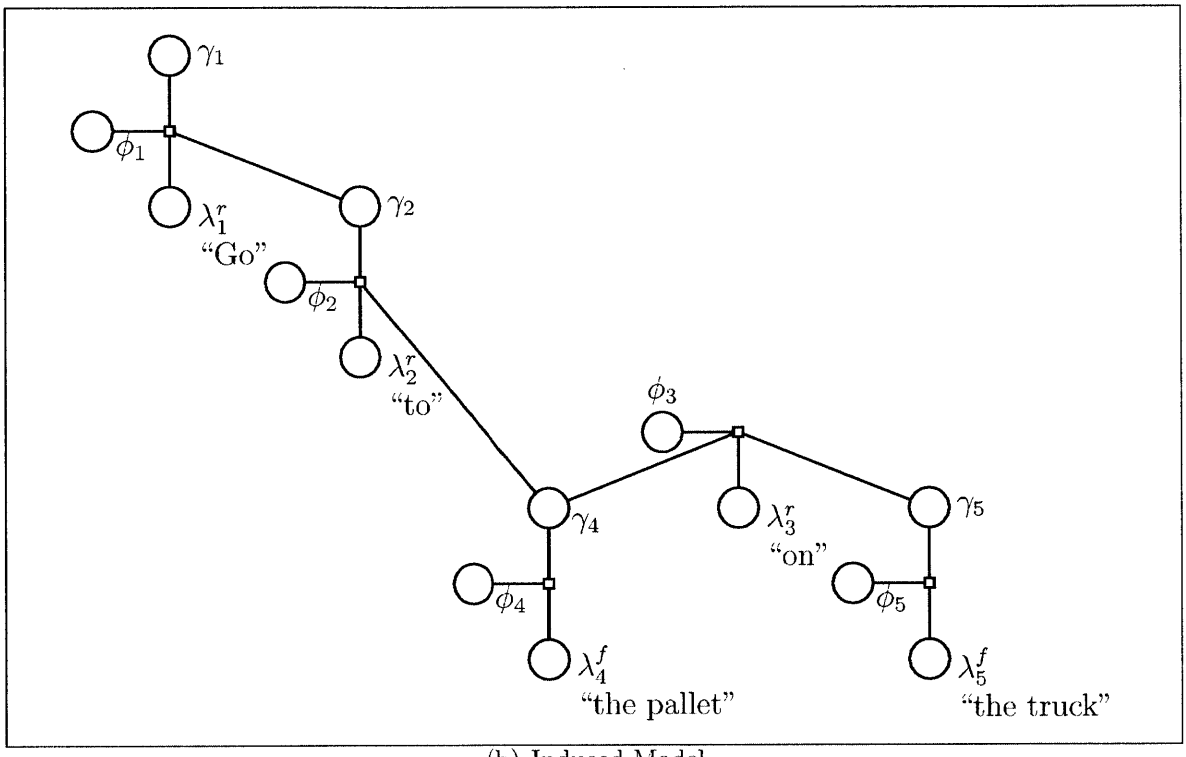
- $avs(\gamma_i^f, \gamma_i^l)$. For "in front of" and "to the left of." Attention Vector Sum (AVS) [Regier and Carlson, 2001], measures the degree to which relations like "in front of" or "to the left of" are true for particular groundings.

For commands such as "to the left," or "to the right," we compute a set of geometric features designed to capture various frames-of-reference in the environment. For example, we compute the AVS feature over all four cardinal directions and three frames of reference: the agent's starting orientation, the agent's ending orientation, and the agent's average orientation during the action sequence. This enables the system to learn the mapping from words onto a particular frame of reference.

For PATH and EVENT SDCs, groundings correspond to the location and trajectory of the robot and any objects it manipulates over time. Base features are

$$
\begin{aligned}
&EVENT_1(r = \texttt{Go} \\
&\quad l = PATH_2(r = \texttt{to}, \\
&\quad\quad l = OBJ_3(f = OBJ_4(f = \texttt{the pallet}), \\
&\quad\quad\quad r = \texttt{on}, \\
&\quad\quad\quad l = OBJ_5(f = \texttt{the truck}))))
\end{aligned}
$$

(a) SDC tree



(b) Induced Model

$$
\begin{aligned}
p(\Phi|\Gamma, \text{SDCs}) = &\frac{1}{Z} \times \Psi_1(\phi_1, \gamma_1, \gamma_2, \lambda_1^r = \text{Go}) \times \Psi_2(\phi_2, \gamma_2, \lambda_2^r = \text{to}) \\
&\times \Psi_3(\phi_3, \gamma_3, \gamma_4, \lambda_3^r = \text{on}) \times \Psi_4(\phi_4, \gamma_4, \lambda_4^f = \text{the pallet}) \\
&\times \Psi_5(\phi_5, \gamma_5, \lambda_4^r = \text{the truck})
\end{aligned}
$$

(c) Factorization

Figure 5-6: In (a) is the SDC tree for "Go to the pallet on the truck." In (b) is a different induced factor graph from Figure 5-5. The structural differences between the two models are highlighted in gray.

computed with respect to the entire motion trajectory of a three-dimensional object through space. Examples include:

- The displacement of a path toward or away from a ground object.
- The average distance of a path from a ground object.

Other geometric features are derived from those in Chapter 4.

Besides geometric features, the system must have semantic features that map noun phrases such as "the wheel skid" to a physical object in the world that might have a different label, such as "tire pallet." To address this issue we introduce a second class of base features that correspond to the likelihood that an unknown word actually denotes a known concept. The system computes word-word similarity using WordNet as well as from co-occurrence statistics obtained by downloading millions of images (and corresponding tags) from Flickr (as in Chapter 3). These base features correspond to $similar(\lambda_i^f, labels(\gamma_i^f))$, where "labels" gets the name of a place or object grounding.

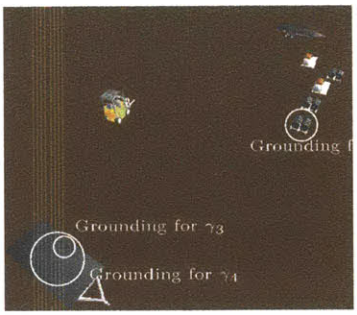
We select 49 base features for leaf OBJECT and PATH SDCs, 56 base features for internal OBJECT and PATH SDCs, 112 base features for EVENT SDCs and 47 base features for PATH SDCs for their relevance in understanding the semantics and geometry of the groundings. This translates to 147,274 binary features after the Cartesian product with words and discretization.

5.1.6 Inference

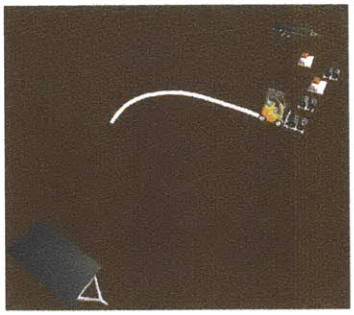
Given a command, we want to find the set of most probable groundings. During inference, we fix the values of Φ and the SDCs and search for groundings Γ that maximize the probability of a match, as in Equation 5.6. Because the space of potential groundings includes all permutations of object assignments, as well as every feasible sequence of actions the agent might perform, the search space becomes large as the number of objects and potential manipulations in the world increases. In order to make the inference tractable, we use a beam search with a fixed beam-width in order to bound the number of candidate groundings considered for any particular SDC. For OBJECT and PLACE groundings, the beam width is ten, for PATH and EVENT groundings the beam width over candidate state sequences is five and for transitions between sequential SDCs the beam width is two.

A second optimization is that we search in two passes: first the algorithm finds and computes the probability of candidate groundings for OBJECT and PLACE SDCs, and then uses those candidates to search the much larger space of robot action sequences, corresponding to EVENTs and PATHs. This optimization exploits the types and independence relations among SDCs to structure the search so that these candidates need to be computed only once, rather than for every possible EVENT. In the example in Figure 5-5(b), this optimization would cause the system to first search for groundings of "the pallet" and "the truck" and then searching for the grounding for the PLACE SDC, "on the truck," and finally searching for an EVENT grounding for "put."

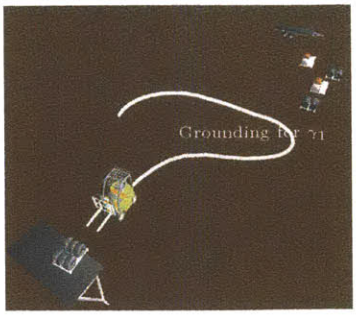
Once a full set of candidate OBJECT and PLACE groundings is obtained up to the beam width, the system searches over possible action sequences for the agent, scoring each sequence against the language in the EVENT and PATH SDCs of the command. In the example in Figure 5-5(b), the system would search over actions the robot can take with the pallet relative to the place that is "on the truck." After searching over potential action sequences, the system returns a set of object groundings and



(a) Object groundings



(b) Pick up the pallet



(c) Put it on the truck

Figure 5-7: A sequence of the actions that the forklift takes in response to the command, "Put the tire pallet on the truck." In (a) the search grounds objects and places in the world based on their initial positions. In (b) the forklift executes the first action, and picks up the pallet. In (c) the forklift puts the pallet on the trailer.

a sequence of actions for the agent to perform. Figure 5-7 shows the actions and groundings identified in response to the command "Put the tire pallet on the truck."

5.2 Evaluation

To train and evaluate the system, we collected a corpus of mobile manipulation commands paired with robot actions and environment state sequences. We use this corpus to train the G^3 model and also to evaluate end-to-end performance of the system at following realistic commands from untrained users.

5.2.1 Corpus

To quickly generate a large corpus of examples of language paired with robot plans, we posted videos of action sequences to Amazon's Mechanical Turk (AMT) and collected language associated with each video. The videos showed a simulated robotic forklift engaging in an action such as picking up a pallet or moving through the environment. Paired with each video, we had a complete log of the state of the environment and the robot's actions. Subjects were asked to type a natural language command that would cause an expert human forklift operator to carry out the action shown in the video. We collected commands from 45 subjects for twenty-two different videos showing the forklift executing an action in a simulated warehouse. Each subject interpreted each video only once, but we collected multiple commands (an average of 13) for each video.

Actions included moving objects from one location to another, picking up objects, and driving to specific locations. Subjects did not see any text describing the actions or objects in the video, leading to a wide variety of natural language commands including nonsensical ones such as "Load the forklift onto the trailer," and misspelled ones such as "tyre" (tire) or "tailor" (trailer). Example commands from the corpus

are shown in Table 5.1.

To train the system, each SDC must be associated with a grounded object in the world. We manually annotated SDCs in the corpus, and then annotated each OBJECT and PLACE SDC with an appropriate grounding. Each PATH and EVENT grounding was automatically associated with the action or agent path from the log associated with the original video. This approximation is faster to annotate but leads to problems for compound commands such as "Pick up the right skid of tires and place it parallel and a bit closer to the trailer," where each EVENT SDC refers to a different part of the state sequence.

The annotations above provided positive examples of grounded language. In order to train the model, we also need negative examples. We generated negative examples by associating a random grounding with each SDC. Although this heuristic works well for EVENTs and PATHs, ambiguous object SDCs such as "the pallet" or "the one on the right," are often associated with a different, but still correct object (in the context of that phrase alone). For these examples we re-annotated them as positive.

5.2.2 Cost Function Evaluation

Using the annotated data, we trained the model and evaluated its performance on a held-out test set. At this stage, we are assessing the model's performance at predicting the correspondence vector given access to SDCs and groundings. The test set pairs a disjoint set of scenarios from the training set in the same warehouse with language given by subjects from AMT.

SDC type	Precision	Recall	F-score	Accuracy
OBJECT	0.93	0.94	0.94	0.91
PLACE	0.70	0.70	0.70	0.70
PATH	0.86	0.75	0.80	0.81
EVENT	0.84	0.73	0.78	0.80
Overall	0.90	0.88	0.89	0.86

Table 5.2: Performance of the learned model at predicting the correspondence vector Φ.

Table 5.2 reports overall performance on this test set, with performance broken down by SDC type. The performance of the model on this corpus indicates that it robustly learns to predict when SDCs match groundings from the corpus. We evaluated how much training was required to achieve good performance on the test dataset and found that the test error asymptotes at around 1,000 (of 3,000) annotated SDCs.

For OBJECT SDCs, correctly-classified high-scoring examples in the dataset include "the tire pallet," "tires," "pallet," "pallette [*sic*]," "the truck," and "the trailer." Low-scoring examples included SDCs with incorrectly annotated groundings that the system actually got right. A second class of low-scoring examples were due to words that did not appear many times in the corpus.

For PLACE SDCs, the system often correctly classifies examples involving the relation "on," such as "on the trailer." However, the model often misclassifies PLACE SDCs that involve frame-of-reference. For example, "just to the right of the furthest skid of tires" requires the model to have features for "furthest" and the principal orientation of the "skid of tires" to reason about which location should be grounded to the language "to the right," or "between the pallets on the ground and the other trailer" requires reasoning about multiple objects and a PLACE SDC that has two arguments.

For EVENT SDCs, the model generally performs well on "pick up," "move," and "take" commands. The model correctly predicts commands such as "Lift pallet box," "Pick up the pallets of tires," and "Take the pallet of tires on the left side of the trailer." We incorrectly predict plans for commands like, "move back to your original spot," or "pull parallel to the skid next to it." "parallel" only appeared in the corpus twice, which was probably insufficient to learn a good model. "Move" probably had few good negative examples, since we did not have paths in which the forklift did not move in the training set to use as contrast.

5.2.3 End-to-end Evaluation

The fact that the model performs well at predicting the correspondence vector from annotated SDCs and groundings is promising, but does not necessarily translate to good end-to-end performance when inferring the groundings associated with a natural language command (as in Equation 5.1).

To evaluate end-to-end performance, the system inferred plans given only commands from the test set and a starting location for the robot. We segmented commands containing multiple top-level SDCs into separate clauses, and utilized the generated G^3 model to infer a plan and a set of groundings for each clause. Plans were then simulated on a realistic, high-fidelity robot simulator from which we created a video of the robot's actions. We uploaded these videos to AMT, where subjects viewed the video paired with a command and reported their agreement with the statement, "The forklift in the video is executing the above spoken command" on a five-point Likert scale. We report command-video pairs as correct if the subjects agreed or strongly agreed with the statement, and incorrect if they were neutral, disagreed or strongly disagreed. We collected five annotator judgments for each command-video pair.

To validate our evaluation strategy, we gave known correct and incorrect command-video pairs to subjects on AMT. In the first condition, subjects saw a command paired with the original video that a different subject watched when creating the command. In the second condition, the subject saw the command paired with random video that was not used to generate the original command. Table 5.3 depicts the percentage of command-video pairs deemed consistent for the three conditions. As expected, there is a large difference between commands paired with the original and randomly selected videos. Despite the diverse and challenging language in our corpus, new annotators agree that commands in the corpus are consistent with the original video. These results show that language in the corpus is understandable by a different annotator

and that some people are still bad at giving commands (e.g., 9% of the commands were un-followable).

Scenario	Precision
Commands paired with original video	$91\% \pm 1\%$
Commands paired with random video	$11\% \pm 2\%$

Table 5.3: The fraction of end-to-end commands considered correct by our annotators for known correct and incorrect videos. We show the 95% confidence intervals in parentheses.

We then evaluated our system by considering three different configurations. Serving as a baseline, the first consisted of ground truth SDCs and a random probability distribution, resulting in a constrained search over a random cost function. The second configuration involved ground truth SDCs and our learned distribution, and the third consisted of automatically extracted SDCs with our learned distribution.

Due to the overhead of the end-to-end evaluation, we consider results for the top 30 commands with the highest posterior probability. In order to evaluate the relevance of the probability assessment, we also evaluate the entire test set for ground truth SDCs and our learned distribution. Table 5.4 reports the performance of each configuration along with their 95% confidence intervals. The relatively high performance of the random cost function configuration relative to the random baseline for the corpus is due the fact that the robot is not acting completely randomly on account of the constrained search space. In all conditions, the system performs statistically significantly better than a random cost function.

The system performs noticeably better on the 30 most probable commands than on the entire test set. This result indicates the validity of our probability measure, suggesting that the system has some knowledge of when it is correct and incorrect. The system could use this information to decide when to ask for confirmation before acting.

Scenario	Precision
Annotated SDCs (top 30), learned cost	$63\% \pm 8\%$
Automatic SDCs (top 30), learned cost	$54\% \pm 8\%$
Annotated SDCs (all), learned cost	$47\% \pm 4\%$
Annotated SDCs (all), random cost	$28\% \pm 5\%$

Table 5.4: The fraction of commands considered correct by our annotators for different configurations of our system.

The system qualitatively produces compelling end-to-end performance. Even when the system gets a command wrong, it often gets parts of it right. For example, it might pick up the left tire pallet instead of the right one. Other factors include ambiguous or unusual language in the corpus commands, such as "remove the

goods" or "then swing to the right," that make the inference particularly challenging. Despite these limitations, however, our system successfully follows commands such as "Put the tire pallet on the truck," "Pick up the tire pallet" and "put down the tire pallet" and "go to the truck," using only data from the corpus to learn the G^3 model.

Although we conducted our evaluation with single SDCs, the framework supports multiple SDCs by performing beam search to find groundings for all components in both SDCs. Using this algorithm, the system successfully followed the commands listed in Figure 5.1. These commands are more challenging than single SDCs because the search space is larger, there are often dependencies between commands, and these commands often contain unresolved pronouns such as "it."

Our system is one step toward robust language understanding systems, but many challenges remain. One limitation of our approach is the need for annotated training data. Unsupervised or semi-supervised modeling frameworks in which the object groundings are latent variables have the potential to exploit much larger corpora without the expense of annotations. Another limitation is the size of the search space; more complicated application domains require deeper search and more sophisticated algorithms. Our model provides a starting point for building dialog systems, because it not only returns a plan corresponding to the command, but also groundings for each component in the command with confidence scores.

5.3 Conclusion

In this chapter we presented an approach to understand mobile manipulation commands. Building off of the work in Chapter 4, where the system extracted a flat semantic structure from the language, in this chapter a hierarchical semantic structure enables the system to factor the learning and inference and ground each linguistic component in an element of the environment (e.g., an object, place, path, or event). In addition, we have introduced the generalized grounding graph (G^3), which enables a system to automatically instantiate a probabilistic graphical model for grounding spatial language. The system automatically learns the meanings of complex manipulation verbs such as "put" and "take," from a corpus of natural language commands paired with correct robot actions as well as spatial relations such as "on" and "to." We demonstrate promising performance at following natural language commands.

Chapter 6

Conclusions

This thesis has focused on understanding unconstrained natural language commands, where a person gives an arbitrary natural language command to the robot and the robot infers and executes the corresponding plan.

In this thesis, we have built on the state of the art in two significant ways. First, we have provided a flexible semantic structure that is able to understand task-constrained spatial natural language commands. The representation is inspired both by the structure of spatial language [Denis et al., 1999] and by the need to keep a flexible representation that still has the ability to capture the breadth of the language in cases where the language is un-grammatical or challenging to parse. Further, we learn the mapping from the language onto this semantic structure.

Secondly, we have provided a set of models that enables a robot to learn the mapping from spatial language onto an unconstrained action space, which enables the system to reason about a wide variety of language. The models described in this thesis predict the state sequence of the robot (e.g., its path relative to other objects or places) given a natural language command, which means that an action specification is no longer in symbolic form, but now in terms of the language and states of the robot (e.g., "to" in "to the kitchen," means that the physical path of the robot ends near the kitchen). Thus, the system is able to learn the meanings of words in terms of spatial and semantic aspects of the environment, instead of abstracting actions into a fixed, discrete set.

In Chapter 4 and Chapter 5 we have taken a corpus-based approach to evaluate our system, enabling us to quantify its robustness.

6.1 Summary of Contributions

Chapter 3 has focused on simple commands of the form, "Find the computer cluster," enabling the system to infer the corresponding plan. We have learned a model of context from the captions of a large dataset of photos downloaded from Flickr and demonstrated both simulated and real-world experiments that use a small subset of detectable objects and scenes in order to robustly predict the location of landmark words. Further, we present a method that incorporates these predictions into the

object-search process, providing a method for choosing a robot plan that minimizes the expected distance to the goal object enabling the robot to find an object more quickly than a greedy approach.

In the Chapter 4 of this thesis, we take steps toward building a robust spatial language understanding system for three different domains: route directions, visual inspection, and indoor mobility. We take as input a natural language command such as "Go through the double doors and down the hallway" and extract a semantic structure called a Spatial Description Clause (SDC) from the language, and grounding each SDC in a partial or complete semantic map of the environment. By extracting a flat sequence of SDCs, the system was able to ground the language by using a probabilistic graphical model that is factored into three key components. The first component grounds novel noun phrases such as "the computers" in the perceptual frame of the robot by exploiting object co-occurrence statistics between unknown noun phrases and known perceptual features using the models from the Chapter 3 of the thesis. These statistics are learned from a large database of tagged images such as Flickr. Second, a spatial reasoning component judges how well spatial relations such as "past the computers" describe a plan. Third, a verb understanding component judges how well spatial verb phrases such as "follow", "meet", "avoid" and "turn right" describe a plan. Once trained, our model requires only a metric map of the environment together with the locations of detected objects in order to follow directions through it. This map can be given *a priori* or created on the fly as the robot explores the environment.

In Chapter 5 we presented an approach for automatically generating a probabilistic graphical model according to the structure of natural language navigation or mobile manipulation commands. Building off of the work in Chapter 4, where the system extracted a flat semantic structure from the language, in this chapter we use the hierarchical structure of a spatial language command to dynamically instantiated probabalistic graphical model that connects linguistic terms to grounded aspects of the environment including objects, places, paths, or events. The system automatically learns the meanings of complex manipulation verbs such as "put" and "take," from a corpus of natural language commands paired with correct robot actions as well as spatial relations such as "on" and "to." We demonstrate promising performance at following natural language commands.

6.2 Future Research Directions

6.2.1 Unsupervised Approaches

One of the challenges with the approach presented in this work is the requirement of having a large corpus of language paired with groundings available in order to learn the target concepts. We have gotten around some of these challenges by leveraging large online databases for learning the correlations between figure or landmark words and objects in the environment. Relaxing the annotation requirements of this data could allow future systems to leverage corpora orders of magnitude larger than the one

used in this thesis, which in turn could enable more robust and interactive systems for learning.

6.2.2 Semantic Mapping

In much of this work our system has assumed that the robot could detect seed objects and locations in a map of the environment. As such, current applications are limited to domains where a real-time object detector is available or those where a human has given a tour of the environment to the robot. However, there is a rich space for interaction between the perception and robotics communities in order to build a representation that contains additional semantics about the environment in an automatic and robust way.

6.2.3 Dialog

Our current system provides a foundational component of a conversational agent. Expanding the capabilities of our system that is able to engage in dialog will enable the system correct its plan or incorporate new information when it is uncertain.

6.2.4 Speech

Speech is one aspect of the problem that has not been addressed in this thesis. In order for a system to be robust, it must be able to translate from a speech command, to a textual representation into a plan for the robot. There is space for integrating uncertainty in the speech recognition into a model of grounding. Additional aspects of speech not dealt with in this work include disfluencies (e.g., "um" and "uh") and corrections to a plan such as, "go down the hallway... I mean, go through the doors." There has been some work by Dzifcak et al. [2009] and Cantrell et al. [2010], but using an approach that is focused more on the linguistics than on the grounding aspects of the problem.

6.2.5 Parsing

Our current system has a separate parsing and grounding step. In particular, we take language, parse it to a semantic structure and then ground each element of the semantic structure in the environment. However, the environment can be used to inform the parser and the parser can be used to inform the grounding. For example, this could enable the system to resolve ambiguous parses such as those involving ambiguous prepositional phrase attachment. Combining the parsing and grounding processes in this way could enable for more robust language grounding.

6.2.6 Additional Linguistic Structure

Some parts of the language were not currently handled by our decomposition into SDCs, such as events where the robot must recognize a time or a situation that is

happening and conditionally execute the corresponding actions such as, "At noon, come to my office and lead the visitor to the kitchen." In addition, there are actions that must be repeated until another event occurs, such as "Make sure no one touches my project while I'm at group meeting." Extending the system handle language like this requires the ability to recognize events and react accordingly.

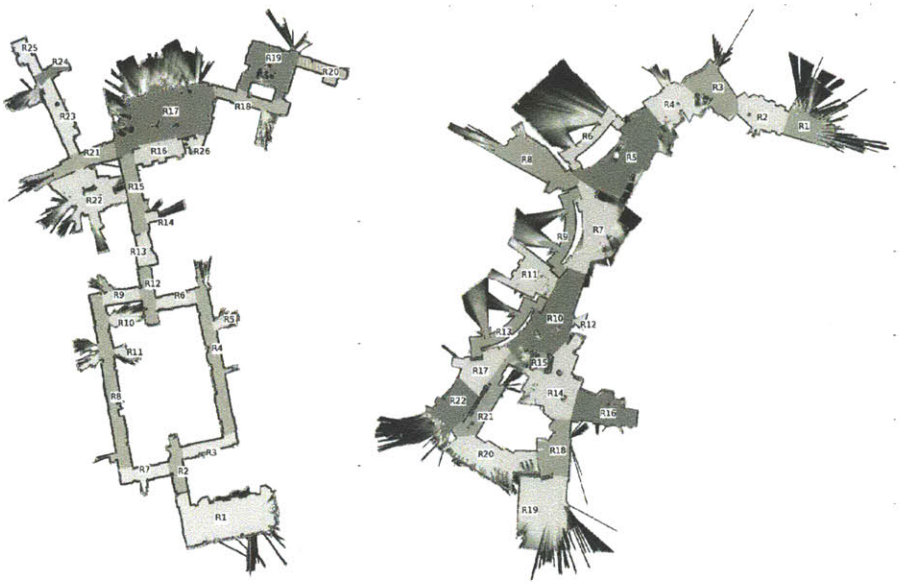
Appendix A

The Structure of Spatial Language

As a part of this thesis, we have collected corpora from four different domains, each of which focuses on spatial language. The focus on spatial language is primarily because it offers the promise that all of the elements could be grounded in the environment. Further, the ability to understand spatial language is important for any system that uses natural language in mobile robotics, since giving and following commands is natural for humans and there is a natural correctness metric. For example, if the task is to command a robot to reach a destination or to inspect a certain object, we can evaluate whether the robot reached the correct final destination. The availability of a corpus and a concrete correctness metric enable an offline component-based evaluation of our system, which makes it easier to determine its robustness and enables us to quickly test new models. In this chapter we analyze corpora of spatial language that we have collected in four different domains: route directions, inspection, indoor mobility and mobile manipulation.

A.1 Corpora

In this thesis we have collected corpora around four different domains: route directions, indoor inspection, indoor mobility and mobile manipulation. There was a natural progression between the various corpora. Route directions captured language that involved landmarks ("the doors") and relations (e.g. "to the doors"). Indoor inspection enabled us to expand the set of commands to those that involved three dimensional structure (e.g. "go up" or "go down"). The indoor mobility dataset enabled us to expand to verbs that people would like robots to execute in natural environments (e.g. "follow", "meet" or "bring") and manipulation required us to expand the set of commands to those that involved moving objects around the environment ("pick up the tire pallet"). This section gives some examples from the various corpora and discusses how these corpora were collected. Examples are shown from each of the corpora in Table A.1.



(a) Stata, Floor 8 (b) Stata, Floor 1

Figure A-1: Two environments where we have collected route directions. For the inspection domain, we collected a separate corpus in Stata Floor 1.

A.1.1 Route Directions

The first corpus of language consists of directions where subjects were asked write down directions from one location in an environment to another location, as if they were directing another person. Language was acquired on two different floors of the same building: Floor 1 is an atrium with a coffee shop and classrooms, while Floor 8 is an office building with a computer lab and offices (Figure A-1). A total of 30 subjects were asked to write directions between 20 different starting and ending locations, for a total of 300 directions. Experimenters did not refer to any of the areas by name, instead using codes labeled on a map as in Figure A-1. In order to reduce or eliminate various biases subjects were from the general population of MIT, between the ages of 18 and 30 years old, were proficient in English, were unfamiliar with the test environment, and were approximately of equal genders (47% female and 53% male subjects). An example set of directions from the corpus is shown in Table A.1.

A.1.2 Indoor Inspection

In order to explore the domain of flying robots that can perform visual inspection, we asked subjects to describe how they would command a micro-air vehicle (MAV), as seen in Figure 1-1(b), in order to inspect various objects in the environment. We collected a corpus on Floor 1 from Figure A-1 because of its high ceilings and resemblance to an indoor mall, which are too high for most ladders to safely access, and require specialized lifting equipment for even simple inspections and maintenance.

Route Directions
-With your back to the windows, walk straight through the door near the elevators. Continue to walk straight, going through one door until you come to an intersection just past a white board. Turn left, turn right, and enter the second door on your right (sign says "Administrative Assistant.")
-In the lounge area, orient yourself so that the spiral staircase is on your left and you can see the elevators toward the glass windows. go out the double doors to the right of the glass windows. walk down the hallway in front of you until you reach the end; you must turn right or left. turn left and walk until you are forced to turn right. walk until (you will pass double doors and a grey door with keycard that you must walk through) reach a white bulletin board. stop
-Go through the double doors past the elevators on your left. Then through the gray door down the corridor, take the left through blue doors. Turn the corner, and first on left.
Visual Inspection
-Go forward until you are able to make a left. Then move ahead until you reach the opposite wall, then make a right. Go straight, past one staircase and to the next staircase. At the top of this staircase you will find a fire alarm on your left at approximately 7ft up.
-go forward towards the square concrete column. make a left and head past room 124 towards the other side of the building. make a right when possible and head towards the staircase on the right side, which is next to the circular table and chairs climb to the top of the staircase and stop before the double doors. Look to your left.
-Go straight until the hallway narrows. Take a left and proceed through the open space. After you go under the stairs and past the large column, turn right. Go past the square column, turn right, go up the stairs, and stop. The fire alarm is now about 8 feet above the ground, to your left.
Indoor Mobility
-Wait by the staircase next to 391, and bring Susan to my room when she comes in.
-Please go to the question mark and wait for Nick. When he gets here, bring him back here.
-Meet the two people at the elevators and bring them to 32-331
Manipulation
-Lift the tire pallet.
-Back up a bit, then swing left and pick up the tire pallet with the box pallet to its left.
-Pick up the pallet of tires on the right and wait.

Table A.1: For the route direction, inspection, and the manipulation corpora, all commands are for the same destination/video, showing the challenge of understanding general language. Comparing to Figure A-4, complex verbs can be seen for the mobility/manipulation domains and simple for the route directions/inspection domain.

In this study, subjects were familiarized with the environment and given the vehicle's starting pose for seven different inspection tasks. Each subject was asked to write down instructions for a human pilot to fly a MAV to the desired object and take video of that object, which resulted in forty-nine natural language commands over seven different destinations and seven different subjects. Subjects were all engineering undergraduates unfamiliar with the system. Table A.1 shows an example set of directions from this corpus. Objects to be inspected were difficult to see from the ground and included a wireless router mounted high on the wall, a hanging sculpture, and an elevated window.

When compared to natural language instructions given in a corpus of walking directions, we noticed many more metrical specifications such as "turn between 90 and 135 degrees." and instances of "fly up." The former may be because of the task: instructing a flying vehicle, or it may be because subjects knew they were instructing a robot rather than a person, and were modifying their language.

A.1.3 Indoor Mobility

Expanding beyond route directions and indoor inspection, we wanted to collect an open-ended corpus that represented tasks that people would like a robot to execute in an office-type environment. Subjects were asked to imagine a robot that can recognize people and objects, but not manipulate them. We collected 262 commands from 12 subjects, an average of 22 commands per subject. Most subjects were robotics researchers. When looking at the resulting commands in Table A.1, we can see that many of the commands in the corpus include verbs relating to people's motion (or *verbs of motion*), such as "meet" and "follow." This is different from the previous domains, which mostly contained verbs that were variants of "go", "turn left" and "turn right."

However, the corpus that was collected contained complex language that was not strictly spatial. This led us to collect a more focused corpus specifically around the spatial motion verbs: "bring," "meet," "avoid," "follow," and "go." For each verb, we created ten natural language commands that used each verb, along with ten compound commands that involved at least two different verbs, such as "Follow the person to the kitchen. Then move toward the bathroom. Next go down the hall to the lounge."

A.1.4 Mobile Manipulation

The final domain that is explored in this thesis is that of mobile manipulation. This expands on the language in the previous domains because it includes the need to ground aspects of the environment that involve the motion of other objects. To quickly generate a large corpus of example manipulation commands paired with robot plans, we generated random videos of a simulated robotic forklift (e.g. Figure 1-1(c)). These videos were uploaded to Amazon's Mechanical Turk (AMT), and subjects were asked to type a natural language command that they would give to an expert human forklift operator to command them to carry out the action shown in the video. Videos

included a simulated forklift picking up a pallet or moving through the environment. Paired with each video, we had a complete log of the state of the environment and the robot's actions.

We collected commands from 45 subjects for twenty-two different videos showing the forklift executing an action in a simulated warehouse. Subjects did not see any text describing the actions or objects in the video, leading to a wide variety of natural language commands including nonsensical ones such as "Load the forklift onto the trailer," and misspelled ones such as "tyre" (tire) or "tailor," (trailer). Example commands from the corpus are shown in Figure A.1.

A.1.5 The Ability of Humans to Follow Commands

We have analyzed the ability of humans to follow the commands given by humans in the route directions and the manipulation domains. In the route direction study, subjects were asked to follow a different set of directions that another subject had created, in order to understand if there were certain people who were better or worse at giving point-to-point directions. The experimenter would read the directions to the subject and follow the subject as the subject followed directions. When the subject became lost, they were asked to report this and the trial was concluded. For the manipulation domain, we gave subjects the original video paired with the language that a subject on Amazon's Mechanical Turk was given and asked them if the two matched.

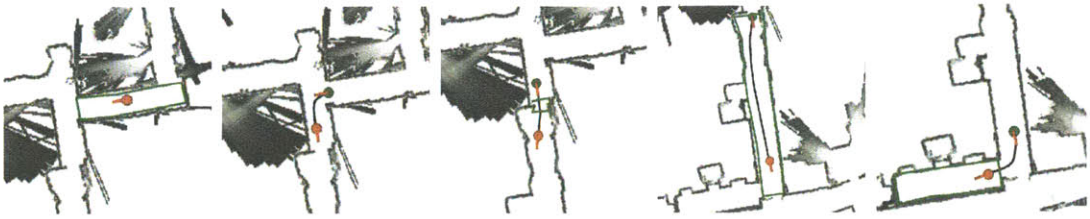
In Table A.2, we can see the percentage of time another person are able to follow the directions in each of the corpora. In addition to the fact that between 10% and 15% were found to be difficult to follow, we also found that there is high variance in the quality of route instructions: some were able to give followable commands 100% of the time and others were only able to do it 30% of the time.

	Dir. Floor 8	Dir. Floor 1	Manipulation
Best direction giver	100%	100%	-
Worst direction giver	30%	20%	-
All directions	85%	86%	91%

Table A.2: The percentage of directions in our corpus that were successfully followed to the end by another person. *Dir. Floor 8* and *Dir. Floor 1* are the corpus of 300 route direction commands collected on the 8th and 1st floors of our environment. *Manipulation* are the commands from the Section A.1.4.

A.2 The Structure of Spatial Language

Spatial language in all of these domains breaks down into clauses that are sequential and where each clause contains figure, verb, spatial relation, and landmarks. Clauses tend to correspond to a compositional action that the robot should execute (e.g.



(a) v:starting sr:in l:this hall

(b) v:turn left

(c) v:walk sr:through l:this metal door

(d) v:walk down sr:straight l:the hall

(e) v:turn right sr:into l:the hall in front of the elevators

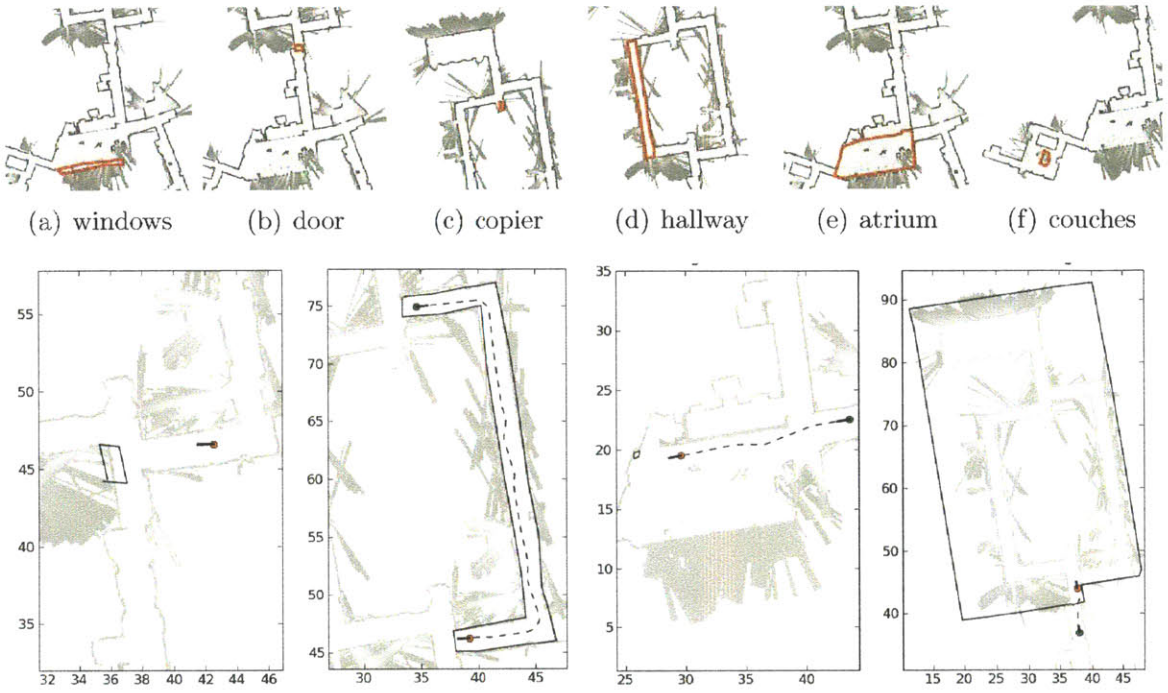
Figure A-2: An example from our corpus: each subfigure is a clause in the spatial language paired with the grounding in the environment (e.g. a path and landmark). The start location of a robot path is shown in green, the end location is shown in red and the corresponding landmark is a green polygon (when it exists).

Figure A-2). Each clause tends to be is sequential. In route directions and inspection, sequential clauses refer to the sequential regions in the environment on the way to the final destination (e.g. Table A.3). For the indoor mobility and manipulation domains sequential means that there is a time ordering: the first clause should be executed before the later ones.

Grounding Statistics	
Overlap	92.3%
Strictly overlap	38.5%
Contained	5%
Reordered	0.4%

Table A.3: Statistics about groundings from the route directions corpus. Strictly overlap means that robot path groundings share more than a common endpoint and reordered means that although two clauses were sequential in the language, they were not sequential relative to the ground-truth path. 99.6% of the clauses were sequential.

Each of the clauses in the natural language commands can decomposed into one of four components: a figure, a verb, a spatial relation and a landmark. For example, a command such as "go to the doors" would have a figure that is implicitly "(you)", a verb "go," a spatial relation "to," and landmark "the doors." Landmarks are objects or places that a person is meant to see along the path to the destination region (as seen in Figure A-3), along with the largest and smallest landmarks. Spatial relations such as "past" and "through" describe how an agent plan should appear relative to these landmarks (as in Figure A-3(g-j). Finally, imperative verbs such as "go," "put," "turn right" or "turn left" tell a person what to do . For the route directions domain, these verbs were generally variants of "go straight," "turn right" or "turn left" (see Figure A-4). Statistics about the partial paths can be seen in Table A.3 and Figure A-3. In Figure A-4, we can see the most frequent words of various types



(a) windows (b) door (c) copier (d) hallway (e) atrium (f) couches

(g) shortest path (h) longest path (i) smallest landmark (j) biggest landmark

Figure A-3: In (a-f) are a set of landmarks from our corpus. In (g-k) are the largest and smallest landmarks and path groundings that people used in the route directions corpus. Each corresponded to a clause in the command. In (a) the corresponding clause was *turn towards the large cement-colored double doors*, in (b) *follow the hall*, in (c) *walk towards the M&M doll* and in (d) *enter building 36*. The solid line corresponds to the landmark and the dotted line corresponds to the path of the robot.

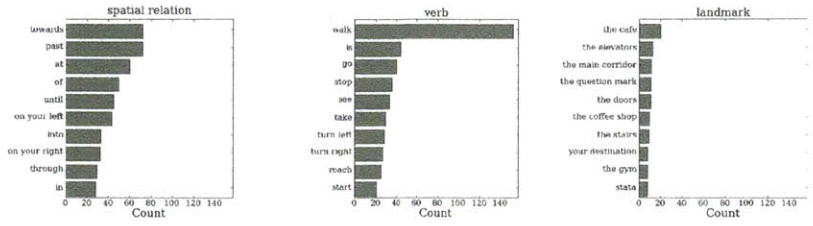
in the corpus.

	mean	std	min	max
landmark area	42.94m^2	±102.28m^2	0m	1964m^2
path length	5.53m	±5.57m	0m	40.83m

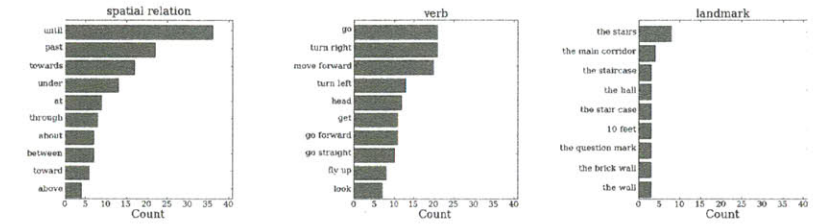
Table A.4: Statistics from the groundings from the route directions corpus.

A.3 Conclusions

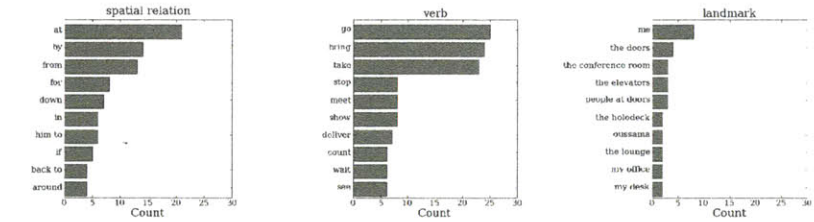
In this chapter we have given examples of spatial language in four different domains and provided some statistics about the structure of spatial language in our corpora.



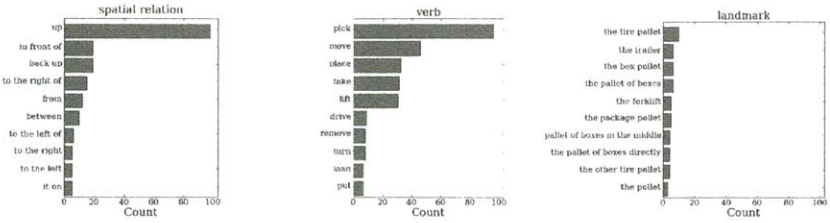
(a) Route Directions



(b) Inspection



(c) Indoor Mobility



(d) Manipulation

Figure A-4: Above are the histograms of the most frequent words that appear in the fields of the SDC for each domain. In (a) is route directions, in (b) is a histogram for inspection and (c) for indoor mobility.

Bibliography

Dana H. Ballard and Christopher M. Brown. *Computer Vision*. 1982. ISBN 978-0131653160.

Irving Biederman. Recognition-by-components: a theory of human image understanding. *Psychological Review*, 94(2):115–147, April 1987. ISSN 0033-295X.

H. Blum. A transformation for extracting new descriptors of shape. *Models for the Perception of Speech and Visual Form*, pages 362–380, 1967.

S. R. K. Branavan, Harr Chen, Luke S. Zettlemoyer, and Regina Barzilay. Reinforcement learning for mapping instructions to actions. In *Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP*, volume 1 of *ACL '09*, pages 82–90, Stroudsburg, PA, USA, 2009. Association for Computational Linguistics. ISBN 978-1-932432-45-9.

Emma Brunskill, Thomas Kollar, and Nicholas Roy. Topological mapping using spectral clustering and classification. In *International Conference on Intelligent Robots and Systems*, IROS '07, pages 3491–3496, October 2007.

Guido Bugmann, Ewan Klein, Stanislao Lauria, and Theocharis Kyriacou. Corpus-Based robotics: A route instruction example. *Proceedings of Intelligent Autonomous Systems*, pages 96—103, 2004.

Rehj Cantrell, Matthias Scheutz, Paul Schermerhorn, and Xuan Wu. Robust spoken instruction understanding for hri. In *Proceedings of the 5th ACM/IEEE international conference on human-robot interaction*, HRI '10, pages 275–282, New York, NY, USA, 2010. ACM. ISBN 978-1-4244-4893-7.

Philip Cohen, David McGee, and Josh Clow. The efficiency of multimodal interaction for a map-based task. In *Proceedings of the sixth conference on applied natural language processing*, ANLC '00, pages 331–338, Stroudsburg, PA, USA, 2000. Association for Computational Linguistics.

Philip R. Cohen and Sharon L. Oviatt. The role of voice input for human-machine communication. *Proceedings of the National Academy of Sciences of the United States of America*, 92(22):9921, 1995.

Emily M. Craparo. Natural language processing for unmanned aerial vehicle guidance interfaces. Master's thesis, Massachusetts Institute of Technology, June 2004.

Michel Denis, Francesca Pazzaglia, Cesare Cornoldi, and Laura Bertolo. Spatial discourse and navigation: an analysis of route directions in the city of venice. *Applied Cognitive Psychology*, 13(2):145–174, 1999. ISSN 1099-0720.

Juraj Dzifcak, Matthias Scheutz, Chitta Baral, and Paul Schermerhorn. What to do and how to do it: translating natural language directives into temporal and dynamic logic representation for goal management and action execution. In *Proceedings of the 2009 IEEE international conference on Robotics and Automation*, ICRA'09, pages 3768–3773, Piscataway, NJ, USA, 2009. IEEE Press. ISBN 978-1-4244-2788-8.

Pablo Espinace, Thomas Kollar, Alvaro Soto, and Nicholas Roy. Indoor scene recognition through object detection. In *Proceedigns of the IEEE International Conference on Robotics and Automation*, ICRA'10, pages 1406–1413, May 2010.

Mark Everingham, Luc Gool, Christopher K. Williams, John Winn, and Andrew Zisserman. The pascal visual object classes (VOC) challenge. *International Journal of Computer Vision*, 88:303–338, June 2010. ISSN 0920-5691.

Pedro Felzenszwalb, David Mcallester, and Deva Ramanan. A discriminatively trained, multiscale, deformable part model. In *IEEE International Conference on Computer Vision and Pattern Recognition*, CVPR '08, pages 1–8, June 2008.

Ruifang Ge and Raymond J. Mooney. A statistical semantic parser that integrates syntax and semantics. In *Proceedings of the Ninth Conference on Computational Natural Language Learning*, CONLL '05, pages 9–16, Stroudsburg, PA, USA, 2005. Association for Computational Linguistics.

Rafael Gonzalez and Richard Woods. *Digital Image Processing.* Addison-Wesley Publishing Company, New York, NY, 1992.

Stevan Harnad. The symbol grounding problem. *Physica D: Nonlinear Phenomena*, 42(1-3):335 – 346, 1990. ISSN 0167-2789.

Sachi Hemachandra, Thomas Kollar, Nicholas Roy, and Seth Teller. Following and interpreting narrated guided tours. In *Proceedings of the IEEE International Conference on Robotics and Automation*, ICRA '11, Shanghai, China, 2011.

Kai-yuh Hsiao, Nikolaos Mavridis, and Deb Roy. Coupling perception and simulation: Steps towards conversational robotics. In *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems*, volume 1 of *IROS '03*, pages 928–933. IEEE, Oct. 2003. ISBN 0780378601.

Ray S. Jackendoff. *Semantics and Cognition*, pages 161–187. MIT Press, 1983. ISBN 0262620014.

Thomas Kollar and Nicholas Roy. Utilizing object-object and object-scene context when planning to find things. In *Proceedings of the IEEE international conference on Robotics and Automation*, ICRA'09, pages 4116–4121, Piscataway, NJ, USA, 2009. IEEE Press. ISBN 978-1-4244-2788-8.

Hadas Kress-Gazit, Georgios E. Fainekos, and George J. Pappas. Translating structured english to robot controllers. *Advanced Robotics*, 22(12):1343–1359, 2008. ISSN 0169-1864.

Geert-Jan M. Kruijff, Hendrik Zender, Patric Jensfelt, and Henrik I. Christensen. Situated dialogue and spatial organization: What, where... and why. *International Journal of Advanced Robotic Systems*, 4(1):125–138, March 2007.

Frank R. Kschischang, Brendan J. Frey, and Hans-Andrea Loeliger. Factor graphs and the sum-product algorithm. *IEEE Transactions on Information Theory*, 47(2): 498 –519, Feb 2001. ISSN 0018-9448.

Taku Kudo. CRF++: Yet another CRF toolkit. http://crfpp.sourceforge.net, 2009.

John D. Lafferty, Andrew McCallum, and Fernando C. N. Pereira. Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In *Proceedings of the Eighteenth International Conference on Machine Learning*, ICML '01, pages 282–289, San Francisco, CA, USA, 2001. Morgan Kaufmann Publishers. ISBN 1-55860-778-1.

Barbara Landau and Ray Jackendoff. Whence and whither in spatial language and spatial cognition? *Behavioral and Brain Sciences*, 16(02):255–265, 1993.

Michael Levit and Deb Roy. Interpretation of spatial language in a map navigation task. *IEEE Transactions on Systems, Man, and Cybernetics, Part B*, 37(3):667–679, June 2007. ISSN 1083-4419.

J. Lighthill. Artificial intelligence: A general survey. In *Artificial Intelligence: a paper symposium*, pages 1–21, 1973.

Matthew MacMahon, Brian Stankiewicz, and Benjamin Kuipers. Walk the talk: connecting language, knowledge, and action in route instructions. In *Proceedings of the 21st national conference on artificial intelligence*, volume 2 of *AAAI '06*, pages 1475–1482. AAAI Press, 2006. ISBN 978-1-57735-281-5.

Matthew Tierney MacMahon. *Following Natural Language Route Instructions.* PhD thesis, University of Texas at Austin, Department of Electrical & Computer Engineering, August 2007.

Marie-Catherine De Marneffe, Bill Maccartney, and Christopher D. Manning. Generating typed dependency parses from phrase structure parses. In *Proceedings of LREC*, pages 449–454, Genoa, Italy, 2006.

Davide Marocco, Angelo Cangelosi, Kerstin Fischer, and Tony Belpaeme. Grounding action words in the sensorimotor interaction with the world: experiments with a simulated icub humanoid robot. *Frontiers in Neurorobotics*, 4(0), 2010. ISSN 1662-5218.

Gale L. Martin. The utility of speech input in user-computer interfaces. *International Journal of Man-Machine Studies*, 30:355–375, April 1989. ISSN 0020-7373.

Cynthia Matuszek, Dieter Fox, and Karl Koscher. Following directions using statistical machine translation. In *Proceedings of the 5th ACM/IEEE international conference on Human-robot interaction*, HRI '10, pages 251–258, New York, NY, USA, 2010. ACM. ISBN 978-1-4244-4893-7.

David Meger, Per-Erik Forssén, Kevin Lai, Scott Helmer, Sancho McCann, Tristram Southey, Matthew Baumann, James J. Little, and David G. Lowe. Curious george: an attentive semantic robot. *Robotics and Autonomous Systems*, 56:503–511, June 2008. ISSN 0921-8890.

George A. Miller, Richard Beckwith, Christiane Fellbaum, Derek Gross, and Katherine J. Miller. Introduction to wordnet: an on-line lexical database. *International Journal of Lexicography*, 3(4):235–244, 1990.

Joseph Modayil and Benjamin Kuipers. Autonomous development of a grounded object ontology by a learning robot. In *Proceedings of the 22nd national conference on Artificial intelligence*, volume 2, pages 1095–1101. AAAI Press, 2007. ISBN 978-1-57735-323-2.

Raymond J. Mooney. Learning to connect language and perception. In *Proceedings of the 23rd national conference on Artificial intelligence*, volume 3 of *AAAI '08*, pages 1598–1601. AAAI Press, 2008. ISBN 978-1-57735-368-3.

Rolf Müller, Thomas Röfer, Axel Lankenau, Alexandra Musto, Klaus Stein, and Andreas Eisenkolb. Coarse qualitative descriptions in robot navigation. In *Spatial Cognition II, Integrating Abstract Theories, Empirical Studies, Formal Methods, and Practical Applications*, pages 265–276, London, UK, 2000. Springer-Verlag. ISBN 3-540-67584-1.

Robert B. Ochsman and Alphonse Chapanis. The effects of 10 communication modes on the behavior of teams during co-operative problem-solving. *International Journal of Man-Machine Studies*, 6(5):579–619, 1974. ISSN 0020-7373.

Ingmar Posner, Peter Corke, and Paul Newman. Using text-spotting to query the world. In *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems*, IROS '10, pages 3181 –3186, Oct. 2010.

Terrance Philip Regier. *The Acquisition of Lexical Semantics for Spatial Terms: A Connectionist Model of Perceptual Categorization.* PhD thesis, University of California at Berkeley, 1992.

Terry Regier and Laura A Carlson. Grounding spatial language in perception: An empirical and computational investigation. *Journal of Experimental Psychology. General*, 130(2):273–298, June 2001.

Deb Roy. Grounding words in perception and action: computational insights. *Trends in Cognitive Sciences*, 9(8):389–396, 2005. ISSN 1364-6613.

Deb Roy and Ehud Reiter. Connecting language to the world. *Artificial Intelligence*, 167:1–12, September 2005. ISSN 0004-3702.

Deb Roy, Kai-Yuh Hsiao, and Nikolaos Mavridis. Conversational robots: building blocks for grounding word meaning. In *Proceedings of the HLT-NAACL 2003 workshop on Learning word meaning from non-linguistic data*, volume 6 of *HLT-NAACL-LWM '04*, pages 70–77, Stroudsburg, PA, USA, 2003. Association for Computational Linguistics.

Alexander I. Rudnicky. Mode preference in a simple data-retrieval task. In *Proceedings of the workshop on Human Language Technology*, HLT '93, pages 364–369, Stroudsburg, PA, USA, 1993. Association for Computational Linguistics. ISBN 1-55860-324-7.

Nobuyuki Shimizu and Andrew Haas. Learning to follow navigational route instructions. In *Proceedings of the 21st international jont conference on artifical intelligence*, IJCAI '09, pages 1488–1493, San Francisco, CA, USA, 2009. Morgan Kaufmann Publishers Inc.

Marjorie Skubic, Dennis Perzanowski, Sam Blisard, Alan Schultz, William Adams, Magda Bugajska, and Derek Brock. Spatial language for human-robot dialogs. *IEEE Transactions on Systems, Man, and Cybernetics, Part C: Applications and Reviews*, 34(2):154–167, may 2004. ISSN 1094-6977.

Dan I. Slobin. Two ways to travel: Verbs of motion in English and Spanish. *Grammatical constructions: Their form and meaning*, pages 195–219, 1996.

Yuuya Sugita and Jun Tani. Learning semantic combinatoriality from the interaction between linguistic and behavioral processes. *Adaptive Behavior - Animals, Animats, Software Agents, Robots, Adaptive Systems*, 13:33–52, March 2005. ISSN 1059-7123.

Leonard Talmy. The fundamental system of spatial schemas in language. In Beate Hamp, editor, *From Perception to Meaning: Image Schemas in Cognitive Linguistics*, pages 199–234. Mouton de Gruyter, 2005.

Stefanie Tellex and Deb Roy. Grounding spatial prepositions for video search. In *Proceedings of the 2009 international conference on Multimodal interfaces*, ICMI-MLMI '09, pages 253–260, New York, NY, USA, 2009. ACM. ISBN 978-1-60558-772-1.

Sebastian Thrun, Wolfram Burgard, and Dieter Fox. *Probabilistic robotics.* MIT Press, 2008.

Barbara Tversky and Paul U. Lee. How space structures language. In *Spatial Cognition, An Interdisciplinary Approach to Representing and Processing Spatial Knowledge*, pages 157–176, London, UK, 1998. Springer-Verlag. ISBN 3-540-64603-5.

Eric J. Vanetti and Gary L. Allen. Communicating environmental knowledge: The impact of verbal and spatial abilities on the production and comprehension of route directions. *Environment and Behavior*, 20(6):667–682, 1988.

Andew J. Viterbi. Error bounds for convolutional codes and an asymptotically optimum decoding algorithm. *IEEE Transactions on Information Theory*, 13(2): 260–269, Apr. 1967. ISSN 0018-9448.

Adam Vogel and Dan Jurafsky. Learning to follow navigational directions. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*, ACL '10, pages 806–814, Stroudsburg, PA, USA, 2010. Association for Computational Linguistics.

Yuan Wei, Emma Brunskill, Thomas Kollar, and Nicholas Roy. Where to go: interpreting natural directions using global inference. In *Proceedings of the 2009 IEEE international conference on robotics and automation*, ICRA'09, pages 3761–3767, Piscataway, NJ, USA, 2009. IEEE Press. ISBN 978-1-4244-2788-8.

Yorick A. Wilks. Natural language understanding systems within the AI paradigm: a survey and some comparisons. Technical report, Stanford, CA, USA, 1974.

Terry Winograd. *Procedures as a representation for data in a computer program for understanding natural language.* PhD thesis, Massachusetts Institute of Technology, 1970.

Yuk Wah Wong and Raymond J. Mooney. Learning for semantic parsing with statistical machine translation. In *Proceedings of the main conference on Human Language Technology Conference of the North American Chapter of the Association of Computational Linguistics*, HLT-NAACL '06, pages 439–446, Stroudsburg, PA, USA, 2006. Association for Computational Linguistics.

T.Y. Zhang and C.Y. Suen. A fast parallel algorithm for thinning digital patterns. *Communications of the ACM*, 27(3):236–239, 1984. ISSN 0001-0782.