3D SCENE DESCRIPTION AND CONSTRUCTION
USING SPATIAL REFERENCING LANGUAGE

A Dissertation
presented to
the Faculty of the Graduate School
at the University of Missouri-Columbia

In Partial Fulfillment
of the Requirements for the Degree
Doctor of Philosophy

by
Samuel Blisard
Dr. Marjorie Skubic, Dissertation Supervisor
DECEMBER 2010
The undersigned, appointed by the dean of the Graduate School, have examined the dissertation entitled

3D SCENE DESCRIPTION AND CONSTRUCTION
USING SPATIAL REFERENCING LANGUAGE

presented by Samuel Blisard,
a candidate for the degree of doctor of philosophy,
and hereby certify that, in their opinion, it is worthy of acceptance.

________________________________________
Professor Marjorie Skubic

________________________________________
Professor James M. Keller

________________________________________
Professor Dominic Ho

________________________________________
Associate Professor Zhihai He

________________________________________
Professor Chi-Ren Shyu
Dedication

This work is dedicated to my fellow Hillbillies of the Ozark Mountains.
Acknowledgements

First and foremost, I have to thank my incredibly patient and loving wife Amy. It has been a long road, but ever since she’s been a part of my life, things have been a lot easier to bear. My family, Janice, Rex, Shasta, Lillie, and Gracie Blisard, have always been there and I probably would have quit a long time ago without their constant support. Finally, my Grandparents, Helen and Joe Bill Blisard and the late Arthur and Juanita Gaddy-I hope I’ve done you proud.

I would be remiss if I didn’t thank my dear friends Ozy, Anna, Inux, Indiana, and Bruno Sjahputera. They opened their home to Annie and I during the final weeks of finishing this work and have been incredible friends to me ever since I began my graduate career in 2001. I can think of no people I’d rather have with me while stomping around the woods of the Ozarks than them.

In the category of never allowing my schooling to interfere with my education, I have to thank the staff of the Chapman Academy. John Leveron, Rich Greiner, GySgt Craig Johnston, and Jeff Harper ran an incredible school and I am forever richer for having had the chance to attend during my studies here at Mizzou. If there’s one regret I have about graduate school, it is that I didn’t discover Chapman sooner.

Dr. Shirley Huffman-If you had not given me that D in Honors Calc II, I probably never would have gone down this path. Some people might think that I am crazy, but that’s the grade I’m proudest to have earned (and there’s no joke
about it—it was earned with her help at 6:00 AM every morning Monday through Friday in MSU’s Mathematics Department).

Don Arni and Tom Best of the Glasgow School District reminded me of my roots in the hills of rural southern Missouri. Besides helping out the kids on the NSF GK-12 fellowship, Tom taught me how to weld and operate the tools in a machine shop which is something I had always wanted to learn. Don reminded me of how much I owe to Ted Tarkow and the ’93 Missouri Scholars Academy. It might have been only 3 weeks, but it changed my outlook forever.

My buddies out in the greater DC area also have contributed greatly to my sanity and in no particular order: Paul and Lisa Branum, Christine Groff, Karen and Jeff Campione, Judy McGinley, Dan and Suzanne Stein, Brent and Latonia Page, Stacy and Brandon McBride, and Wesley and Kerry Wilson.

This work would have been impossible had it not been for the support of my bosses at the Navy Center for Applied Research in Artificial Intelligence (NCARAI) at the Naval Research Laboratory, Washington DC: Dennis Perzanowski, Ph.D. and Alan Schultz.

And last, my incredible advisor here at Mizzou, Dr. Marjorie Skubic. She above all others has been instrumental in improving this work and assisting me in the final days of finishing it off. She has given me an incredible jumping off point into the real world and for that I’m incredibly thankful.
Contents

Chapter

1 Introduction 1
  1.1 Motivation . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 1
  1.2 Investigations . . . . . . . . . . . . . . . . . . . . . . . . . . . 1
  1.3 Contributions . . . . . . . . . . . . . . . . . . . . . . . . . . . 2
  1.4 Utility . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 3

2 Related Work and Background 5
  2.1 Spatial Language for Human-Robot Interaction . . . . . . . . . . 5
  2.2 Computational Linguistics . . . . . . . . . . . . . . . . . . . . . 12
  2.3 Linguistic Descriptions of Relative Positions . . . . . . . . . . . 16
  2.4 Histogram of Forces . . . . . . . . . . . . . . . . . . . . . . . . 18

3 AVS Comparison 23
  3.1 The Attentional Vector Sum . . . . . . . . . . . . . . . . . . . . . 23
  3.2 Comparison between HoF and AVS . . . . . . . . . . . . . . . . . 26

4 Extending Spatial Referencing Language to 3D 34
  4.1 Overview . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 34
  4.2 SIFT Cloud Model Generation . . . . . . . . . . . . . . . . . . . . 35
  4.3 Recognition and Placement of the Object Models and Planar Pro-
      jection . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 39
4.3.1 Projections and Convex Hulls .................................. 39

4.4 Generation of a Description of the Spatial Relationship Between
Two Objects ......................................................... 43

4.4.1 Generating Linguistic Descriptions in 3D ................. 43

4.4.2 NEAR ......................................................... 47

4.4.3 ABOVE and BELOW ....................................... 48

4.4.4 ON TOP .................................................. 48

4.4.5 INSIDE .................................................. 48

4.5 Test Cases .................................................... 49

4.6 Discussion .................................................... 54

5 3D Construction Methods ............................................. 59

5.1 Methodology and Background .................................. 59

5.2 Architectural Example ........................................ 60

5.3 Simplifying Assumptions ...................................... 64

5.4 Representation ................................................. 65

5.5 Parser ........................................................ 66

5.6 Spatial Rendering System .................................... 70

5.7 Evaluation Metrics ............................................ 73

6 3D Construction Test Buildings .................................... 75

6.1 Synthetic Building ............................................. 75

6.2 Barre’s Old Labor Hall [2] ................................... 90

6.3 The Egyptian Theater [3] .................................... 96

6.4 The Oregon Central School [4] ............................... 98

6.5 Coghlan Castle [5] ........................................... 117

6.6 Discussion .................................................... 125
7 3D Descriptions from Inside the Constructed Model 129

8 Discussion 132
  8.1 Strengths .................................. 132
  8.2 Limitations .................................. 133

9 Conclusions 136
  9.1 Summary .................................. 136
  9.2 Future Work .................................. 137

Bibliography 140

Appendix

A Appendix A: SRLib 146
  A.1 Background .................................. 146
  A.2 Compilation .................................. 146
  A.3 Use ........................................ 147
  A.4 Improvements .................................. 147

B Appendix B: SpatialArch Manual 149
  B.1 Preliminaries .................................. 149
  B.2 Compilation .................................. 149
  B.3 Program Flow .................................. 149
  B.4 Controls ...................................... 150
  B.5 Concluding thoughts .................................. 150

C Appendix C: Ground Truth and Linguistic Descriptions of the Buildings 152
  C.1 Ground Truth Tables .................................. 152
C.2 Qualitative Linguistic Descriptions .......................... 154
  C.2.1 Barre’s Old Labor Hall .................................. 154
  C.2.2 Egyptian Theater ......................................... 160
  C.2.3 Oregon Central School .................................... 162
  C.2.4 Coghlan Castle ............................................. 165

Vita 171
Tables

Table

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>6.1</td>
<td></td>
<td>75</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>155</td>
<td>156</td>
<td>157</td>
<td>158</td>
<td>159</td>
</tr>
</tbody>
</table>
Figures

Figure

1.1 3D Model Generation System ......................................... 4

2.1 Histogram of Angles versus the Histogram of Forces .......... 19
2.2 The Histogram of Constant Forces between the Robot and Object
   5 in Figure 2.3 .............................................................. 20
2.3 Evidence Grid Map Corresponding to Figure 2.2 ............... 21

3.1 AVS Model ................................................................. 25
3.2 Grazing line for an L shaped object ................................. 26
3.3 Grazing line for a triangle ............................................. 27
3.4 Confidence Region ........................................................ 28
3.5 AVS Regions for a small box. ......................................... 29
3.6 AVS Problems with a Triangle ....................................... 30
3.7 AVS Problems with an L-Shaped Object ............................ 31
3.8 Histogram of Forces with a Triangle ............................... 32
3.9 Histogram of Forces with an L-Shaped Object ................. 33

4.1 Keypoint cloud generation ............................................ 37
4.2 Matched Keypoints in 2D ............................................. 44
4.3 Keypoint Clouds for Extensions ................................... 45
4.4 Fuzzy Rules .............................................................. 47
<table>
<thead>
<tr>
<th>Figure</th>
<th>Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>6.38</td>
<td>Coghlan Castle second floor plan</td>
<td>122</td>
</tr>
<tr>
<td>6.39</td>
<td>Coghlan Castle first floor rendering (green) vs. Ground Truth (yellow)</td>
<td>123</td>
</tr>
<tr>
<td>6.40</td>
<td>Coghlan Castle second floor rendering (green) vs. Ground Truth (yellow)</td>
<td>124</td>
</tr>
<tr>
<td>6.41</td>
<td>Tree for Coghlan Castle</td>
<td>125</td>
</tr>
<tr>
<td>7.1</td>
<td>Example of a position in a walkthrough</td>
<td>130</td>
</tr>
</tbody>
</table>
Chapter 1

Introduction

1.1 Motivation

Humans excel at communicating to one another about the objects and spaces that inhabit our lives. For example, communicating about rooms in a house or a lost pair of keys can be quickly and easily discussed among humans; however, we are sorely disappointed when attempting to have a similar conversation with a Roomba. Machines excel at performing billions of calculations per second or performing efficient database lookups, but when faced with an open environment and natural language, they fall woefully short of our expectations. This work addresses some key challenges of this problem by investigating 3D Spatial Referencing Language (SRL) using a 2D modeling tool called the Histogram of Forces (HoF). The HoF provides a quantitative measure of qualitative spatial relationships (for example, a is to the left of b). I consider the generation of 3D spatial descriptions using vision based models of an environment and also consider the construction of a 3D model through the interpretation of spatial language.

1.2 Investigations

The first problem encountered when designing a system that is to model human expectations is to show evidence that it will mimic human performance. In this work, I compare the HoF-based spatial referencing tool to Regier and
Carlson’s [1] Attention Vector Sum (AVS), which was developed directly from human subject data using a curve fitting method and has been shown to accurately model human spatial behavior. Limitations of the AVS are also discussed.

The second investigation is an collaborative effort in which 3D spatial language is generated from stereo vision images. The system uses a SIFT keypoint cloud model of an object, stereo vision for recognition and placement, and a spatial reasoning component to describe the locations of objects in the environment using spatial referencing language. This system models the relationships above, below, left, right, front, behind, inside, on top of, near, and far.

The third investigation is the use of SRL to construct 3D buildings from an architectural description language that I developed. The input to this system is natural language; the output is a 3D model in OpenGL which has the capability to qualitatively describe the positions of the rooms with respect to each other. Two metrics were developed to compare the 3D constructed model to ground truth floor plans of buildings. Test cases include both synthetic and real buildings taken from the National Park Service Register of Historic Buildings. A diagram of this system is shown in Fig. 1.1.

1.3 Contributions

Compared the HoF spatial modeling tool to the AVS model;

Extended the 2D HoF modeling tool to support 3D linguistic descriptions using stereoscopic vision data and 3D models;

Developed new algorithms for NEAR, ON TOP, INSIDE, and UNDERNEATH;

Proposed an architectural description language;
Developed an approach to extract spatial knowledge from architectural descriptions;

Developed a method to render the 3D architectural descriptions;

Proposed both quantitative and qualitative metrics to measure the performance of the 3D construction;

Demonstrated initial steps for supporting a walkthrough of a 3D environment which generates 3D spatial descriptions.

1.4 Utility

This work advances Human-Robot Interaction (HRI) and remote Human-Human Interaction (HHI) in several ways. The investigation comparing the HoF and AVS models indirectly shows that the HoF method captures human spatial referencing expectations and at the same time overcomes several problems inherent with the AVS model. The 3D spatial language work enables more efficient and natural communication with a robot or remote agent. Finally, by showing how one can construct a virtual 3D environment from spatial descriptions and then generate spatial descriptions as a part of a walkthrough of the model, I have demonstrated a new application for 3D spatial referencing language that could be used for either HRI or HHI.
Figure 1.1: 3D Model Generation System
Chapter 2

Related Work and Background

2.1 Spatial Language for Human-Robot Interaction

Spatial referencing language is often used in human-to-human communication, e.g., when providing instructions on where to go or where to look. A parent might tell a child to “sit on the stool behind the counter.” Or, she might say, “look for my car in front of the school.” This type of qualitative referencing dramatically contrasts the native language of robots, based on control parameters such as joint angles or motor velocities. Even a translation into Cartesian coordinate frames is awkward for a person communicating with a robot that shares his workspace. We would not consider it natural to tell a person to go to a location or pick up an object at a numerical Cartesian coordinate. For natural interfaces with robots, we should have an alternative.

Previous work has shown how spatial referencing language can be used in the context of human-robot interaction. Skubic et al. [6] describe a system which generates linguistic descriptions from sonar range data for use in mobile robots. This illustrated the use of language from the robot to a human user. Later, the work was extended to add the ability to issue commands from a human user to a robot and the approach was demonstrated in a dialog between the user and robot, using spatial referencing language [7]. As part of the work, simple algorithms were introduced to model the primary directions of LEFT, RIGHT,
FRONT, and BEHIND (LRFB), as referenced by an environment landmark, in this case, a segmented object from an occupancy grid (OG) map. This work was furthered developed in [8]. Human subject experiments showed that in most cases the developed algorithms worked well; however, objects with intrinsic axial information were poorly modelled. As a result, the LRFB algorithms were refined to reflect this.

There is quite a large corpus of work on the linguistics of spatial language; however, spatial language and robotics is a relatively new field of research. One of the areas for employing spatial language and linguistics with robotics is in making robotic wheelchairs. Müller et al. [9] created a control strategy for giving directions to a semi-autonomous wheelchair using relatively coarse qualitative route descriptions such as “turn left,” “follow corridor,” etc. Gribble et al. [10] also investigated the use of Kuiper’s Spatial Semantic Hierarchy (SSH)[11] to reason about space. The SSH has five levels: metrical, topological, causal, control, and sensorimotor. They discuss creating a user interface using three levels, the topological (“go there”), causal (“go straight”, “turn left”), and control (“stop”); however they did not actually implement the system.

Zelek proposed the use of a lexical template in order to tie together spatial references and robot commands [12] for use in 2D robot navigation. In his templates, commands are given in the form of a verb, destination, direction, and speed with the spatial reference tied to the destination reference object. In his work, the reference objects were walls and doors that were identified utilizing two laser range finders, each mounted on a pan-tilt head. He utilized a potential field technique to assign goal regions which would allow the robot to stop whenever it made it to the region.

Stopp et al. [13] developed a two-arm mobile robot designed for assembly tasks that utilized relative spatial references. The relative spatial references
(e.g. FRONT, LEFT) are used with the robot’s geometric world model for the identification of objects. In the world model, a user selects an object by using a relational expression, such as “the spacer on the right.” The actual computation of the spatial relations use the center of gravity and a bounding box to approximate an object. Hasegawa et. al [14] have developed a system which uses similar references in order to navigate a small humanoid robot through an obstacle course. Petrosino and Gold [15] have investigated the relationships “close, near, and far” in a similar domain (i.e. robot navigation); however, their system does not take into account those relationships between objects other than the robot.

Another technique to analyze the environment around the robot is to use the relative position of line segments around the robot. Due to the nature of the sensors commonly employed on robots, we generally only have a collection of points at a relative distance from the robot for collision and obstacle detection (e.g. sonar and laser range finders). Frommberger[16] notes that in interior environments, one commonly has to contend with walls, hallways, and doors, all of which can be modeled with line segments. The spatial locations of these line segments built up from range data and used to train the robot for goal-oriented behavior. These line segments are encoded into tiles and are input to a Markov model to facilitate navigation through a simple office-like environment. The advantage to his system is that it learns quickly and the paths which the robot follows tend to be smoother than with traditional wall-following behaviors. It is also worthy to note that this not only uses the spatial information of the current locations of the tilings, but also temporal information as the robot moves through the environment.

Spatio-Temporal systems are thought to be part of the way that humans are able to navigate through the world while following directions or by reading a map. Krishna and Kalra[17] use a SOM as input to a resilient backpropagation neural network in order to learn spatio-temporal sequences for robot navigation.
and simple behaviors. The input to their SOM was the readings from the laser range finder and they were able to have their robot learn behaviors like wall-following and paths through a cluttered environment through the use of a fuzzy rule base. Another system which works in a similar spatio-temporal fashion for robotic arm motion and manipulation of objects has been developed by Sugiura et. al[18].

There has also been some thought as to using biologically inspired models for learning spatial relationships. Schulz et al.[19] used a neural network to evolve a system to evolve a language to communicate between robots about the space surrounding them. The core of their algorithm is RatSLAM[20], which is a model of the hippocampus of the rat. Using this system, they were able to train a SOM as input to the neural net and have it output 3-syllable words describing what the robots were seeing through their camera systems. The robots were able to evolve a language and then use this language to navigate through a simple office environment.

An interesting problem when dealing with SRL and robots in particular is that humans tend to assume, for lack of a better phrase, that robots are either incredibly stupid or incredibly smart when it comes to their spatial reasoning capabilities (most preferring to err on the “stupid” side). Tenbrink et al.[21] performed a series of experiments looking at how humans communicate with a robot which could understand basic spatial tasks and was directed by the human to go to certain goal locations in the environment. Their primary goal in these experiments was to look at the kinds of language people used when communicating with the robot in a collaborative task. They found that humans tend to always take the perspective of the robot regardless of perceived difficulty as opposed to switching under cognitive loads as suggested by Tversky et al.[22].

Perzanowski et al.[23] conducted a similar experiment to Tenbrink’s using
a Wizard of Oz experiment where humans used a touch screen displaying video from the robot and speech to guide the robot through a maze to find a target card labeled with the word “FOO.” They found that when given these modalities, people once again assumed that the robot could only understand basic commands like “turn right,” “turn left fifteen degrees,” “drive forward one meter,” etc. Additionally, once users found the minimal set of commands to make the robot move through the environment, they stopped testing the system and relied on just the commands which worked. About half of the users in their study essentially used verbal joysticking in order to navigate the robot instead of higher level commands (e.g. “move until the box is to your left”). Moratz et al. [24] showed a similar behavior in that about half of their participants used goal-based directions while the other half used direction-based directives when interacting with the robot. The studies of Tenbrink et al. Moratz et al., and Perzanowski et al. show that the common assumption about the capabilities of the robot are more in line with industrial robots than ones like R2-D2.

Another way that SRL comes into play is through the use of perspective and the relative placements of objects in the environment. Kennedy et al. [25] used previous work by Trafton et al. [26] on perspective taking to build StealthBot. StealthBot uses a cognitive model written in ACT-R to try and hide from a patrolling guard. In order to perform this task, it uses the spatial location and movement of the target object (in this case, the patrolling guard) and obstacles in the environment which it knows are large enough to hide behind. By modeling the space and movement, it is able to alter its route in order to remain hidden from the patrolling human. A similar system has also been investigated by Liu et. al. [27].

SRL is also tied to the proximity of objects. Regier and Carlson [1] note this in their development of the AVS in that the further away an object is, the more
likely it will be described in terms of the primary direction to it (i.e. “ABOVE,” “RIGHT,” etc.). As objects move closer, they note that humans will make use of a secondary or compound description to describe where the object is in space relative to another object (e.g. “The ball is to the above-right of the stapler” at closer distances versus “The ball is above the stapler” at longer distances). These proximity effects are also shown in Kelleher et al.[28] work in which they pay particular attention to modeling the proximity effects of the choice of reference objects.

In work closely related to this dissertation, Moratz et al.[29] investigated spatial relations for controlling a mobile robot. In their experiment, test subjects directed a robot to a specified location in a field of goal objects (small blocks that were easily represented as points) situated between the human and the robot. The test subjects then used a computer to type directions in natural language to the robot. The interesting result is that about half of the subjects used the goal objects to direct the robot, whereas the other half decomposed the control actions into simple path segments (e.g. “drive a bit forward”, “come ahead to the right”). The authors theorize that the reason for this discrepancy is that the test subjects did not realize the full capability of the robot to use the goal objects as spatial references. Another finding of interest is that the operators, when using the decomposed directions, used the robot’s perspective instead of their own frame of reference. One might think this is natural, since humans may underestimate the ability of the robot. Tversky et al.[22] observed a similar result in human-human communication in experiments they performed when the listener had a significantly higher cognitive load than the speaker. Possibly, the test subjects in the Moratz experiment used the same type of language expressions they might have used for a person and may have subconsciously viewed a human subject as requiring a higher cognitive load.
For the work described here, I utilize Matsakis’ Histogram of Forces (HoF) [30] [31] to model spatial relationships. In Section 2.3, there is a discussion on the HoF and how it is used in the context of spatial referencing for robots. The features captured in the HoF are somewhat similar to previous work by Regier’s Attention Vector Sum (AVS)[1]. The AVS method builds a set of weighted vectors from each point in the referent object to the trajector point which is the center of attention from the referent object. This set of vectors is summed and was shown to correlate well with human responses. However, the HoF method is more general as it considers a set of vectors from each point in the referent to each point in the trajector and works with any shape or size of objects. Additionally, the HoF is invariant to scale and rotation, which is an attractive feature and does not need to have its parameters changed to fit different scenarios.

Some approaches have been proposed previously for computing a point that is BETWEEN two objects. Quinlan [32] gives an overview of tactics for generating the closest point BETWEEN two objects, whereas Cheng [33] suggests using the centroids between two objects (where the objects in question came from images). However, both of these approaches have drawbacks that keep them from generalizing to be robust in all cases. These problems arise from configurations of objects which have odd shapes or configurations which can lead to the placement of a BETWEEN point being in a non-intuitive position.

Another problem with SRL is that by its nature, it is not precise and can require multiple referent objects in order to pick out a distinct target object or action. Often, humans will use color or some other distinguishing characteristic of an object in order to determine where in space an object needs to be placed. Brick and Scheutz[34] have created a system called The Robotic Incremental Semantic Engine (RISE) which uses an incremental approach in order to try and eliminate ambiguity in SRL. Their system starts out with minimal information presented
to the user (or to the robot) and builds the proper location through subsequent utterances. For example, consider the task of placing a block on top of another block. In this example, one would say “which one?” as it is ambiguous as to which block the human or robot is referring to. This would be clarified by an utterance of “the one to the right” or “place the red block on top of the green block.”

2.2 Computational Linguistics

Computational Linguistics is generally defined to be the scientific study of language from a computational perspective. There are two areas that need to be investigated, the first being the cognitive side that deals with how humans acquire, produce, and understand language, and the second is how humans understand the relationship between linguistic utterances and the world. Perhaps the most common way that we structure language is through the use of grammar to determine what is well-formed (it follows the rules of the grammar) or ill-formed (it violates rules of the grammar) [35]

However, even grammars do not work out all that well. Edward Sapir has a famous quote: “All grammars leak,”[36] which tells us that even if an utterance is not well-formed in the sense of grammar, it can still be understood by the people who are communicating. There are two camps in Computational Linguistics that try to understand this discrepancy, the rationalist and empiricist camps.

For a rationalist (such as Noam Chomsky), the argument is that humans have some form of innate linguistic capability that has been given to us by nature. One possible reason for this is an idea known as the poverty of the stimulus proposed by Chomsky [37]. The key to this concept is that given the incredible complexity of language that is heard during a child’s early years and their limited mental facilities, there must be some hard-wired capabilities in their minds that allow for the learning of language. This idea is part of the reason that the fields of
machine learning and AI have seen a preponderance of learning where large sets or frames of data are loaded into a system along with complex rules in order to achieve learning [35].

Empiricists assume that the brain does have some cognitive abilities; however, they are more general than the rationalist approach. They feel that a child’s brain has basic facilities for association, pattern recognition, and generalization which can be applied to the rich environment of stimuli in which they find themselves [35].

The system developed at NRL which processes natural language called NAUTILUS [38], utilizes predicates to process a grammar, which would make it lean towards the empiricist camp. However, the thought that there are some high-level spatial ideas being processed would tend to push it more to the rationalist camp as this is a more high-level structure that is innate. As Manning states, rationalism and empiricism aren’t mutually exclusive; rather, they are various shades of the same idea [35].

One of the best pieces for an overview of spatial language and linguistics is Landau and Jackendoff’s work [39], in which they investigate the differences in encoding places and objects, with the key difference being in how they are encoded in the brain and conveyed to other people. This is important as this work needs a firm base in how it is that people effectively communicate spatial locations between each other.

One area unique to this particular facet of HRI is that different cultures tend to have a great variety of differences in the way spatial language is used. It has been well established that language structures space [40], and that structuring can have effects on the cognitive processes involved when using spatial language to communicate. In chapter two of Regier’s book The Human Semantic Potential [41], several of these problems are addressed. In this work, he states that spatial
relations are often expressed by closed-class forms, which is a relatively small set of linguistic forms that adds members only rarely (e.g. spatial prepositions and verbal prefixes). There are also open-class forms which capture a skeletal conceptual structure. Consider enter, which is an open-class form as it conveys a wide range of possible enterings. In contrast, into, which is a preposition, is a closed-class form. However, the structure of these open-class and closed-class forms can vary radically between cultures.

As an example of this cross-linguistic variation that is brought up in Regier’s book [41], let us first look at Guugu Yimithirr. In this Australian language, there is heavy reliance on using north, south, east and west to describe the spatial locations of an object. For example, an English speaker might say “Bob is standing in front of the tree,” whereas the Guugu Yimithirr speaker would say “Bob is due north of the tree.” Speakers of Guugu Yimithirr must constantly stay aware of the four cardinal directions in order to adequately describe the locations of objects in the world. This view of the world through cardinal directions can even be different depending on whether one was raised in the city or the countryside as evidenced in the Indian language Tamil. For Tamil speakers, the rural dialect uses the idea of cardinal directions, whereas urban speakers use relative directions that mirror the way that English speakers communicate spatial locations.

The cross-cultural aspect of spatial language is not just limited to cardinal directions. Some languages tend to map metaphorical statements to objects. In the Mexican language Mixtec, in order to say that a stone is under the table, the Mixtec speaker would say “The stone is located at the table’s belly”[41]. In this example, the table is considered to be a quadruped, so the belly is understood to be under it. A tree, however, is considered to be bipedal and therefore a bird at the top of the tree is said to be on the tree’s head.

We can’t even be sure that there exist semantic universals. In [41], Regier
discusses Gropnik’s work which shows that children and adults have radically different views of even simple spatial terms (e.g. down, in, and on). Gropnik observes that children tend to view these terms as ways of moving objects whereas adults would view them as locations. [41].

People use spatial language in a variety of ways and do not restrict the language to primary directions. Linguistic hedges (such as a little, mostly, etc.) may be used to refine a referenced region. Zimmer’s experiments demonstrated a surprising variety of responses [42]. In his experiments, he used a reference object (RO) and a to-be-located object (LO). These were dots on a computer screen and the test subjects were to say what was the reference between the first dot (RO) and the second dot (LO). The phrases that the subjects were to use to classify the spatial relation with respect to the LO were above, left and above, left, left and below, below, right and below, right, and right and above. In his results, he finds that there is quite a bit of variance when the subjects were classifying objects that were intermediate between one of these eight primary directions.

Even when dealing with well-defined spatial problems in our own language, certain spatial localities are easier to conceptualize than others. Grabowski’s [43] work with the concept of Origo showed that human subjects tend to take on the perspective of the addressee by imagining themselves in the other person’s location to observe the environment. For this reason, in FRONT of an object can be computed with little cognitive load, behind with slightly more, and LEFT/RIGHT with more difficulty than FRONT/BEHIND. This higher cognitive load for humans could be a cause for the difficulty that humans have when taking on a different spatial perspective.

Finally, we need to look at the problems inherent with allocentric versus egocentric reference frames. An allocentric reference frame references points within a framework with metrics external to the objects located within it (e.g. using com-
pass directions). An egocentric frame is one in which the orientation and position are related to the perspective of the object that is currently under consideration. In terms of language, this is similar to the difference between Guugu Yimithirr which is allocentric, and English which tends to be egocentric. Klatzky gives a nice summary of the various aspects of egocentric versus allocentric reference frames in the opening chapter of [44].

2.3 Linguistic Descriptions of Relative Positions

Keller et. al [45], used a system of fuzzy rules to produce linguistic spatial descriptions of the images in terms of RIGHT, ABOVE, LEFT, and BELOW. Bloch[46] also used a system of fuzzy rules in order to describe the relative positions between objects presented to her system. Krishnapuram et al. [47] viewed the objects as fuzzy regions and used possibility distributions over the $\alpha$ cuts to calculate spatial relations. As a precursor to the HoF, Miyajima and Ralescu [48] used the Histogram of Angles to perform spatial reasoning. Keller and Wang[49] also used the Histogram of Angles as an input into a neural network to classify whether an object was ABOVE, BELOW, LEFT, and RIGHT of another object.

In 1999, Matsakis and Wendling [30] introduced the HoF, which generalizes the histogram of angles method. In their method, the key difference is that sections of the object are considered rather than their constituent points as shown in Figure 2.1. Two types of force histograms have been used for spatial language generation, namely the histograms of Constant Forces, which is similar to the Histogram of Angles in its final result, and the histogram of Gravitational Forces, which weights the portions of objects that are closer more heavily. Extending Keller’s work [45], features are extracted from the $F_{0}^{AB}$ (the constant forces histogram for objects A and B) and together with $F_{2}^{AB}$ (the gravitational forces histogram), are used as input into a system of fuzzy rules which is used to generate spatial references [7].
For any direction $\theta$ in which forces are computed, different values can be extracted from the analysis of each histogram. For example, according to $F_{r}^{AB}$, the degree of truth that the proposition “A is in the direction of B” is $a_{r}(\theta)$. This value is a fuzzy number; hence it is in the range $[0, 1]$ with 0 meaning that the proposition is completely false and 1 that the proposition is completely true. Moreover, according to $F_{r}^{AB}$, the maximum degree of truth that can be attached to the proposition from another source of information is $b_{r}(\theta) \in [a_{r}(\theta), 1]$. The direction $\theta$ for which $a_{r}(\theta)$ is maximum is called the main direction. The main direction is a key feature for my algorithms as it gives the direction towards which most of the object under consideration lies.

For the use of the spatial reasoning component, we utilize $F_{0}^{AB}$ and $F_{2}^{AB}$ to calculate an “opinion” about where the object B is in the direction of $\theta$ from A in the four primary directions (ABOVE, BELOW, LEFT, and RIGHT) (note that for the robotics work, ABOVE is replaced with BEHIND and BELOW with FRONT). The results from this process are fed to a series of fuzzy rules which generate the linguistic descriptions in the primary direction (“The object A is mostly to the right of me”) along with a secondary linguistic hedge (“but somewhat forward”, etc.) and a qualifier as to how well the conditions are satisfied. Finally, there is also an assessment of how well the primary and secondary directions describe the relationship; if the assessment is not good enough, then we look for the state of “surrounds.” Full details of how this works can be found in [31]. For more details on the the HoF, the reader is referred to Sjahputera [50].

After the linguistic descriptions were developed, additional features were added to aid in the spatial language for HRI. Skubic et al. [6] extended the original linguistics to add support for “surrounds”, descriptions of groupings, and qualitative distance (e.g. “The object is close”). With this work, the algorithms have moved further towards allowing the use of strictly qualitative descriptions
for describing most situations in the robot's environment.

This was integrated into NRL's WAX system to support a multi-modal dialog using spatial referencing language. Environmental information is taken from segmented Evidence Grid (EG) maps, which are generated with laser and sonar sensor data. The ability to name objects in the environment was added through the use of dialog to make the interaction more natural. In addition, support for further spatial language queues has been included (e.g. “What is the object in front of you?”) [51].

2.4 Histogram of Forces

Initially, the HoF algorithm was developed by Matsakis and Wendling [30] for modeling the spatial relationships between 2D objects in images. The benefit of using their approach is that the HoF is based in solid theory and readily lends itself to being used in fuzzy spatial relations such as “to the right of,” “behind,” etc. [31]. We also have the benefit of low-computational handling of heading changes which makes for easy switching between allocentric and egocentric views of the current scene/EG map.

The relative position of a 2D object A with regard to another object B is represented by a function $F_{AB}$ from $\mathbb{R}$ into $R_+$. For any direction $\theta$, the value $F_{AB}^{\theta}$ is called the histogram of forces associated with (A,B). The object A is called the argument, and the object B is called the referent. If the elementary forces are in an inverse ratio $d^r$ where $d$ represents the distance between the points under consideration, then $F$ is denoted by $F_\theta$.

The values of $r$ are used to generate two different histograms, $F_0$ and $F_2$ which are the histograms of constant and gravitational forces, respectively. For my work, I concentrate on the histogram of constant forces as it provides a global view of the relationship between the two objects under consideration. The histogram
of gravitational forces weights pieces of the objects which are closer together more heavily, which can cause the *main direction* to be skewed towards those pieces which are closer. In initial work, the EG maps gave a full view of the environment; I felt that the $F_0$ histogram would yield more reliable results. This concept has been continued in the extensions presented in this work.

![Figure 2.1: Histogram of Angles (a) utilizing the constituent points versus the Histogram of Forces (b) utilizing longitudinal sections (from [30])](image)

The HoF uses longitudinal slices of the objects which are then integrated to determine how much of the object is in a particular slice. In Figure 2.1(b), we can see how it is that these slices are created in a general sense. In practice, parallel lines rotate through an angle $\theta$ and all slices are considered. This process yields the HoF, an example of which is shown in Figure 2.2 which corresponds to the map in Figure 2.3.

For computing points to the LEFT, RIGHT, FRONT, and BEHIND of the robot, the referent B is the robot. For consideration of between, this is not the case and we look at both of the objects being the argument and the referent in turn. For the generation of confidence regions around objects, we switch the argument and referent and look at the whether the robot calculated the LEFT, RIGHT, FRONT, and BEHIND points to be in a region which corresponds to those areas around the objects.
Figure 2.2: The Histogram of Constant Forces between the Robot and Object 5 in Figure 2.3
Figure 2.3: Evidence Grid Map Corresponding to Figure 2.2, the robot is the small circle inside the larger, and the heading is indicated by the line segment from the center of the larger circle to its circumference
The F-histogram associated with this (A,B) notation is represented by a
discrete number of values for $\theta$ from $[0, 360]$ where $\theta \in Z$. As the HoF can work
with either raster or vector data (using the points along the contour of the object),
the values that are shipped to it are the contour points, which comprise the outside
boundary of the objects, assimilated into polygons and handled as vector data.
The computation of $F^{AB}$ is a complexity of $O(n\log(n))$ where $n$ denotes the total
number of vertices [30]. More details on how the vector data is handled can be
found in Skubic et. al[6].

In recent work, Matsakis et. al [52] has investigated Allen relations in order
to solve some of the problems associated with overlapping regions. The current
system used for this work can, at best, state that an object is surrounded when it
is overlapping with another object. Using Allen relations, a multitude of different
relationships can be modelled from touching to objects being interwoven. These
relationships developed will compliment my work well in the future.
Chapter 3

AVS Comparison

3.1 The Attentional Vector Sum

Regier and Carlson’s AVS [1] was developed as a computational model of how people use spatial language. In their experiments, participants are shown a range of target points around an object and were asked to rate how well the point shown satisfies the spatial preposition ABOVE on a scale of 0-9. Using this corpus of data, a curve fitting algorithm is used to extract parameters for the AVS model. Regier and Carlson were able to demonstrate that the AVS matches human responses. However, parameters may need to be generated for any new referent object with a different size or shape. In order to show that the HoF method correlates well with human subject data in describing the areas around an object, I implemented the AVS and investigated the spatial referenced areas around the objects.

As one might expect, the AVS, at its core, is a simple sum of vectors which are weighted according to how much they line up with the primary direction under consideration (which work for their investigations of the direction ABOVE). The AVS model is defined by the following equations:

\[
AVS: \text{above} = g(\sum_i a_i \vec{c}_i) \ast \text{height}
\]  

(3.1)

where \(a_i\) is a function which defines the width of the attentional beam by the
formula \( a_i = \exp[-(\text{Euclidean distance from the trajector to point } i \text{ of the landmark})/(\lambda \sigma)] \) and \( \vec{c}_i \) is a unit vector in the direction of interest. The function \( g(x) \) is defined by:

\[
g(x) = [(\text{slope} \times x) + y_{\text{intercept}}]
\]

which is just the standard equation for a line. The height function \( \text{height} \) is defined by:

\[
\text{height} = \frac{\frac{1}{1+\exp[\text{highgain} \times (\text{hightop} - y)]} + \frac{1}{1+\exp[\text{1} \times (\text{lowtop} - y)]}}{2}
\]

The variables \( \text{slope}, \text{y}_{\text{intercept}}, \lambda, \sigma \), are parameters for AVS that are used to fit the AVS to the data being modeled. The parameters \( \lambda \) and \( \sigma \) are used to tune the width of the attentional beam. The variables \( \text{slope} \) and \( \text{y}_{\text{intercept}} \) are used to vary the angular contribution to the total sum (so that points which create a large angle from the primary direction are weighted less). The parameter \( \text{highgain} \) controls the height function’s weighting (to cause drop off of the value as the object moves below the grazing or highest point of the object in the case of ABOVE). The variables \( \text{hightop} \) and \( \text{lowtop} \) are simply the top and bottom of the object and limit the effects of the \( \text{height} \) function. The output of the AVS is scaled to give a qualitative number which represents how much a given trajector point is in the primary direction from the landmark (ABOVE, BELOW, LEFT, or RIGHT). Essentially, this is a vector sum from all points in the object to a single trajector point. This process is shown in Fig 3.1 from Regier and Carlson[1], Fig. 5.

This method has some advantages, namely that it can accurately describe human subject data. It is also able to demonstrate the grazing line effect, which happens with objects that do not have a flat top (e.g. triangles and L-shaped objects). The grazing line influences the scores that a point is rated higher for
Figure 3.1: The attentional vector-sum model. Panel a, illustrates the attentional field, focused on the landmark (LM), near the trajector (TR). Different parts of the landmark receive different amounts of attention. Panel b illustrates the vectors rooted at each point of the landmark, pointing toward the trajector. Panel c illustrates the attentionally weighted vectors. Panel d illustrates the direction of the attentionally weighted vector sum in the direction of the trajector from the landmark. Panel e illustrates the orientation of the attentionally weighted vector sum, relative to vertical upright. (from [1])
“above” than it should be as it is below the top of the object. This is one of the effects for which AVS is famous for being able to handle. As shown in Fig. 3.2, note the values inside of the L and how the grazing line affects the acceptability of the particular point being ABOVE the L. This effect is present in Fig. 3.3.

Figure 3.2: A figure illustrating the grazing line effect on an L-shaped object from [1]. The numbers are rankings on a scale of 0-9 as to the acceptability that the point in question satisfies the phrase “this point is ABOVE the object.” figure from Regier and Carlson[1]

The language which is generated by the SRL software relies on a base of fuzzy rules to interpret features from the HoF in order to generate the linguistic descriptions. In previous work [8], the idea of a confidence region was used to look at areas around objects which have low, medium, and high levels of confidence of being in one of the four primary directions. This work has been expanded to make a comparison with the AVS.

3.2 Comparison between HoF and AVS

We compute Confidence Regions (CR) based on a mapping of the quality of the linguistic descriptions around our objects. By using a small virtual object which can move to each point in the area under consideration and capturing the
Figure 3.3: A figure illustrating the grazing line effect on a triangle from [1]. The numbers are rankings on a scale of 0-9 as to the acceptability that the point in question satisfies the phrase “this point is ABOVE the object.” (from [1])
confidence value for the linguistic description, this value can be used to create high, medium, and low regions of confidence around the objects in the OG map world, an example of which is shown in Fig. 3.4. The regions with high confidence are shown in red, medium confidence is green, and low confidence is yellow. Qualitatively, the higher the value, the stronger the evidence that the robot is to the \{left, right, front, behind\} of the object in question.

Figure 3.4: HoF Confidence Regions are generated for object 1 for the direction ABOVE. Red is high confidence, green is medium confidence, and yellow is low confidence.

The values $C_r$ used to generate the confidence scales in Figure 3.4 are as follows:

$C_r \geq 0.92 = \text{high}$

$0.8 \leq C_r < 0.92 = \text{medium}$

$C_r < 0.8 = \text{low}$

The results for these regions around the objects are consistent with results from Bloch [46], namely that as one moves farther away from the object, the area which satisfies the spatial preposition ABOVE grows in size and spreads away
from the object. This is also consistent with results of Regier and Carlson [1]. As shown in Fig. 3.9, the AVS has a similar pattern as the HoF CR. The comparisons provide further validation that the HoF method produces results consistent with human expectations.

![Figure 3.5: AVS Regions for a small box at the bottom of the figure (represented by blue circles) for the spatial reference of ABOVE. Red corresponds to high confidence, green to medium confidence, and yellow to low confidence.](image)

Since we have established that simple objects compare well, let us consider the grazing line effect. As shown in Fig. 3.6, the grazing line effect is shown as the AVS tends to bias slightly in the direction of the slant of the triangle. A similar feature is noted in Fig. 3.7. In all cases, the maximum value occurs where there is a balance between the vectors contributing to the figure. As this is a vector sum, this in and of itself is not very exciting. However, there is a serious drawback which is the lower acceptability on both sides of the object. While it is biased, this bias does not match up as well with the human scores noted in Figs. 3.3 and 3.2 due to how the scores are calculated in a row-scan fashion as opposed to a global score. This is a serious drawback, which is overlooked by Regier and Carlson as
when they generate figures showing the AVS for ABOVE and object, they only consider a single row in the figure which shows a clear rising and falling, with the maximum point being the point which most greatly satisfies the proposition that the trajector is ABOVE the landmark. This allows for a shifting and ignoring that whole areas have equal weight of being ABOVE when they should not be considered as such. While it may model human responses well, it ignores effects which would cause serious problems when using it for a robotics application.

Figure 3.6: AVS regions around a triangle. The green circles denote the maximum value in the row. The lines and polyhedra denote different confidence regions (high above triangle, medium between the black lines and the yellow polygons).

In comparison, consider how the Histogram of Forces deals with these two objects. Note that the grazing line effect is clearly shown and that HoF does not make the mistake of giving points which are clearly not above in the objects being labelled as such. Additionally, this is an intentional crippling of the SRL software in that only one direction is being considered. Normally, these intermediate regions would be labelled as being to the left or right of the object with a linguistic hedge if necessary. When all primary directions are considered, this yellow region
Figure 3.7: AVS regions around an L-shaped object. The green circles denote the maximum value in the row.
would be overpowered by the higher confidence that the point under consideration was to the LEFT or RIGHT of the triangle.

![Figure 3.8: HoF Confidence Regions around the triangle.](image)

The HoF method offers several benefits over AVS, especially in the robotics domain. One advantage is that there is no need to do any curve-fitting to compute parameters, so the HoF works well when presented with a new or novel object. It is also able to deal with grazing line effects and more closely match human subject data than the AVS in these situations. Additionally, the ability to work with generalized objects is a major advantage for use in mobile robots, as sensor readings are uncertain by nature and we do not know \textit{a priori} the shape of all objects in the environment. While we do not match the AVS exactly, we do match the general trend for areas further away from objects which is important for mobile robots as they move around the environment. For use with mobile robots, the HoF model is more flexible than the AVS.
Figure 3.9: HoF Confidence Regions around an L-shaped object.
Chapter 4

Extending Spatial Referencing Language to 3D

4.1 Overview

In this chapter I present work which demonstrates the extension of 2D SRL to 3D. Previous work in SRL for linguistic descriptions has centered around its use in the domains of HRI, 2D scene descriptions, and 2D geospatial information; however, the approach can be used for other domains as well. Here, I show an approach using SRL for scene description based on input from a computer vision system. This is joint work which has not been published yet, so its availability is limited. For completeness, the entire process is presented in this work as its necessary to for understanding my contribution to the system. In this work, Erik E. Stone created the SIFT keypoint cloud models of objects to be recognized and Robert H. Luke III did the stereo vision and SIFT keypoint recognition and placement.

The following components are included:

1. SIFT cloud model generation; (Stone)

2. Recognition and placement of the models and planar projections; (Luke and Blisard)

3. Generation of a description of the spatial relationship between the two objects. (Blisard)
The system works as follows: first, an object is modeled by taking stereo pictures from multiple angles and computing the SIFT keypoints for this series of photos. These 3D points become a cloud of points which are then used to match and place the recognized objects into the environment of the robot. These keypoint clouds, after being placed in the environment, are then projected onto the viewing plane and the horizontal plane. The convex hulls for these two planes are then calculated and the two views are fed through a system of fuzzy rules to determine which description best fits the scene presented to the robot.

The initial version of the software used sonar or laser range data represented as an Occupancy Grid (OG) map and could only produce descriptions that objects were to the LEFT, FRONT, RIGHT, and BEHIND the robot. It could also tell if the robot was partially surrounded or completely surrounded. The new language system can generate these descriptions, as well as the descriptions ABOVE, ON TOP, BELOW, and INSIDE and works with both range sensor data and stereo vision systems.

4.2 SIFT Cloud Model Generation

Each object is represented by a 3D SIFT keypoint cloud model acquired during a priori training, which is stored as a database of SIFT keypoints. Each keypoint has a 128 dimensional feature vector which describes it, a direction from which it was viewed (which is determined based on the image it came from), and a three-dimensional position. The SIFT keypoint cloud models are generated from a set of twenty four images taken around 360 degrees of an object using a single web camera. The camera is kept stationary, while the object is rotated on a lazy susan. The images are taken against a flat, bare wall to help reduce the number of keypoints generated from the background. After the twenty-four images of the object have been captured, SIFT keypoints are extracted from all the images and
used to build the keypoint cloud model.

In order to construct these images, the first step is to take the twenty-four images of the object are manually examined and the coordinates of a bounding box, depicted in Fig. 4.1, which contains the entire object in all the images, are determined. The bounding box is used to reduce the chance of false matches and to reduce the processing time.

Once the coordinates of this bounding box have been determined, the following process is repeated for every keypoint, $K$, in every image, $I$, in the set of twenty-four images to determine whether it should be included in the three-dimensional model:

Step 1: Determine that keypoint $K$ lies inside the predefined bounding box and has not been marked “discarded” (how keypoints get marked “discarded” is mentioned in later steps). If either of these conditions is not satisfied, skip keypoint $K$, otherwise, if both conditions are met, move on to Step 2.

Step 2: Look for a matching SIFT keypoint in image $I - 1$ based on $K$’s 128 dimensional SIFT feature vector. If no match is found, skip keypoint $K$ and move on to the next keypoint. If a matching keypoint is found in image $I - 1$, look to see if the matching keypoint in image $I - 1$ has been marked “discarded.” If so, abandon keypoint $K$, and move on to the next keypoint. If not, determine if the matching keypoint lies in the allowed search area of image $I - 1$. This search area is determined by $K$’s position in image $I$, the maximum allowed vertical movement between images, $\Delta y$, the minimum required horizontal movement between images, $\Delta x$, and the bounding box. (The purpose of the maximum allowed vertical movement between images is to help reduce false matches, and the purpose of requiring a minimum horizontal movement is to help throw out background points which lie inside the bounding box and are matched between images. Currently values of 4 are used for both $\Delta y$ and $\Delta x$, which were determined through
Figure 4.1: Diagram depicting a SIFT keypoint, K, in image, I, along with the predetermined bounding box, which should encompass the entire object, overlaid on all the images. In addition, the restricted search areas determined by K's position in image I, the required minimum horizontal movement, $\Delta x$, and the maximum vertical movement, $\Delta y$, are overlaid in images $I - 1$ and $I + 1$. Figure courtesy of Erik Stone.
experimentation.) If the matching keypoint in image I-1 does lie in this search area, then a match is said to have been found in image I - 1. This same process is then repeated to determine if there is a match in image I + 1. If a match is found in image I - 1 and I + 1, then mark the three keypoints as “discarded” (so they will not be used to add another keypoint to the model) and proceed to the next step. If a match is not found in both image I - 1 and I + 1, abandon keypoint K, and move on to the next keypoint.

Step 3: Look for matches to keypoint K in all images that are within plus or minus 60 degrees of image I (excluding images I - 1 and I + 1), based on K’s 128 dimensional feature vector. Mark any matching keypoints as “discarded.” The purpose of this step is to try to prevent the same keypoint from being added to the three-dimensional model more than once.

Step 4: Look for matches to keypoint K, based on K’s SIFT feature vector, in all images that are not within plus or minus 60 degrees of image I. Mark any matching keypoints as “discarded” (if they are not already). If any matches are found, then keypoint K is not unique on the object and will not be suitable for matching. As a result, abandon keypoint K, and move on to the next keypoint.

Step 5: Using the known image coordinates of keypoint K in image I and the matching keypoints in image I - 1 and I + 1, along with the specifications of the camera, calculate a position in 3D space for keypoint K as described in section IVasdf.

Step 6: Add a new keypoint to the three-dimensional model with the SIFT feature vector of keypoint K, the view angle of image I, and the 3D position calculated in Step 5.

At the conclusion of this process, the SIFT keypoint cloud model has been created for an object based off of the 24 images.
4.3 Recognition and Placement of the Object Models and Planar Projection

4.3.1 Projections and Convex Hulls

The first problem that must be solved is how to place the SIFT keypoint cloud model which has been recognized by the cameras into the environment. The placement of objects in the 3D space is performed by the following five steps:

(1) Build the SIFT keypoints for a pair of stereo images and match the keypoints;

(2) Determine the 3D position of matched keypoints by triangulating the direction vectors from each camera to each matched keypoint, which results in point clouds of keypoints in 3-space;

(3) Match the recognized keypoints to the keypoint database of known objects;

(4) Transformation of the known keypoint locations to those in the current scene;

(5) Keypoints for all known objects in the current scene are projected onto the horizontal and vertical planes and their convex hulls are found.

The details of these steps are given as follows.

Step 1: SIFT Stereo Matching

Stereo cameras take a pair of images of a given scene from which a set of keypoints is then built for each image. This can result in several thousand keypoints per image. Each keypoint in the left camera is matched against all keypoints in the right camera according to Lowe’s matching procedure[53]. This matching is
performed by determining the Euclidean distance between two keypoint descriptors in 128 dimensional space. For a given keypoint in the left camera, the two smallest descriptor distances to keypoints in the right image are saved. If the smallest distance divided by the second smallest distance is less than a threshold value, usually 0.6, then the keypoints are considered to be a match.

Step 2: Keypoint Triangulation

Each matched keypoint must then be placed into the 3-dimensional scene. This action is similar to computing the locations of keypoints when building the database. Because the cameras’ locations and orientations are known, when a match between the two images is found, one can determine the view vector from each camera to a keypoint in 3-space. Where these two vectors intersect is the location of the matched keypoint. Continuing this procedure for all matching keypoints results in several clouds of keypoints in 3D space.

Step 3: Object Matching

Next, the known keypoints from the point clouds are determined. The same matching procedure used for keypoints between the stereo cameras is used for matching these keypoints to keypoints of known objects. This results in several sets of keypoints related to objects in the known database.

Step 4: Object Transformation

Each set of keypoints resulting from the previous step has a structure similar to the matched keypoints in the known object database. The next step is to determine the transformation required to move the keypoints in the object database to the matched keypoints in the current scene. When this is accomplished, all keypoints of the known object can be moved into the scene.

Unfortunately, many of the keypoints matched to a known object are incorrect. This is often because numerous areas of an object have a similar pattern. From the training phase, it is known from which camera angle each keypoint orig-
inated. Therefore, for the current scene, the viewing direction of the cameras can be determined by averaging the camera angles saved in each keypoint. As a result of this, all keypoints more than 45 degrees from the current viewing direction are discarded.

To further resolve the matched keypoints, the distance between all matched keypoints in the scene are determined. This is also computed for all matched keypoints in the database. This results in two distance matrices. This is an $N^2$ operation; however, there are usually less than 100 keypoints per object, this is not too computationally expensive. Dividing the distances in each matrix on a per element basis results in a relative matrix. Each column in the relative matrix represents the similarity in the distances from that keypoint to all other keypoints for both the current scene and the object database. Ideally, these relative values would all be 1, but for this application the keypoint is thrown out unless the median of any column is between 0.5 and 2.0.

By this stage, one can be reasonably confident that the remaining keypoints are accurate. The transformation of the matched keypoints between the object database and the current scene must now be computed. This transformation matrix is computed using a least squares method.

Due to imprecision in the cameras, the objects can appear larger or smaller than they actually are. The mean of matched keypoints in the current scene is computed and the distance between that mean and all keypoints in that cloud are summed. The same is done for matched keypoints in the object database. The keypoints in the object database are then scaled to match the keypoints in the current scene.

Again using the means of the keypoint clouds, a difference is computed to translate the known object to the current scene. Now that the point clouds are centered in the same location, the rotation matrix must be computed. The method
described by Umeyama[54] is used in the application. In this process, a matrix $A$ is created from the $n$ remaining matched keypoints in the known database; $A$ is therefore an $n \times 3$ matrix. A second matrix $B$ is computed in a similar fashion using the remaining points in the current scene. A new matrix $H$ is computed as $H = A \ast B^T$. A singular value decomposition of $H$ results in three separate matrices $U$, $S$ and $V$. Matrix $S$ handles the scaling of $A$ onto $B$, but because the keypoints have already been scaled, $S$ is set to the identity matrix. If it is the case that $\det(U) \ast \det(V) < 0$, the resulting transformation matrix will be a reflection instead of a rotation. This is corrected by setting $S_{33}$ to negative one. The rotation matrix can then be set to $R = V S U^T$. Now knowing the appropriate scaling, translation and rotation, all points in the database for the matched object can be transformed into the scene.

Step 5: Convex Hulls

After all matched objects have been transformed into the scene; they must be projected onto the vertical and horizontal planes. Multiple projection schemes were experimented, such as eigenvectors, but they all seemed to end in a similar result. It was decided that the easiest scheme was also the most effective. For the horizontal plane, the $x$ and $y$ values are used and the $z$ values are set to zero. For the vertical view the $x$ and $z$ values are used and the $y$ values are set to zero.

Using the projected keypoints, a convex hull is computed for both the vertical and horizontal planes for each object. The hulls are computed by Matlab’s convex hull routine, which is an implementation of the Qhull algorithm. This algorithm begins with the bottom most point and sweeps a line counter-clockwise starting at zero degrees. Each time a point is intersected, the current location is translated to that point, and it is added to the list of convex hull points. The algorithm terminates when the original point is reached.
4.4 Generation of a Description of the Spatial Relationship Between Two Objects

In previous work by Luke et al.[55], it has been shown how SIFT keypoints can be used to generate linguistic descriptions of objects presented to a robot’s vision system. A drawback to this system was that only the convex hulls of the recognized keypoints were used to generate the linguistic descriptions. While this method worked for many cases, if an object was elongated or obscured, then the proper linguistic description would not be generated. Such a case is shown in Fig. 2, which shows the SIFT clouds for a can and a large box from several angles. The red points are the only ones that are recognized in the image, as they are on the side of the object facing the camera system. If just the face points are used, then the description is “The box is to the right of the can.” However, if one uses the full convex hull generated by placing the full SIFT cloud into the environment, then the description changes to “The box is to the right of the can and extends to the rear.” The convex hulls that correspond to the horizontal projection are shown in Fig. 4.3.

By using this method, the entire object can be considered when computing the linguistic descriptions instead of just the recognized keypoints.

4.4.1 Generating Linguistic Descriptions in 3D

Once the convex hulls have been projected onto the vertical and horizontal planes, they are used as inputs into a Spatial Referencing (SR) system [7], [8]. The extended SR version has the addition of triangular fuzzy membership functions for the two relations of “NEAR” and “SOMEWHAT NEAR”, which are used in conjunction with a modified dictionary to generate the linguistic relationships of ABOVE, BELOW, NEAR, ON TOP, and INSIDE. In previous work[56, 6, 57,
Figure 4.2: Point cloud models of the can and a box placed in the environment viewed from above. Using only the matching keypoints in red leads to the description “The box is to the right of the can.” Image courtesy of Robert H. Luke III.
Figure 4.3: The convex hulls for the Box (red) and the can (blue) in the horizontal plane. The linguistic description generated is “The box is to the right of the can and extends to the rear”
58], features from the HoF were used to describe the locations of objects and to calculate points which could be described by the spatial prepositions FRONT, BEHIND, LEFT, RIGHT, and BETWEEN objects. However, these calculations and descriptions were done for a 2D occupancy grid map of the environment. In order to extend the previous work, the first problem was modifying the SR system to generate spatial linguistic descriptions in both the vertical plane and the horizontal plane. In addition, the extension takes a new look at the SURROUND descriptions, which previously had been used for the robot to inform its user that it was either SURROUNDED or PARTIALLY SURROUNDED by an object[6].

The second part is the use of two simple fuzzy membership functions shown in Fig. 4.4 which use the distance between two convex hulls’ bounding boxes to determine whether the objects are NEAR or SOMEWHAT NEAR.

The process for generating the horizontal, vertical, and near descriptions is as follows:

1. Use the HoF to generate features which are used to generate the linguistic description between the two convex hulls in the horizontal plane;

2. Use the HoF to generate features which are used to generate the linguistic description between the two convex hulls in the vertical plane;

3. Use the distance between the objects as input into the fuzzy membership functions for NEAR and SOMEWHAT NEAR;

4. Using the information from the fuzzy membership functions, the linguistic descriptions, and the SURROUNDSD description, generate the final linguistic descriptions for the two planes and any special relationships between the two objects (i.e. NEAR, SOMEWHAT NEAR, ABOVE, ON TOP, BELOW, and INSIDE).
4.4.2 NEAR

The simple fuzzy membership functions shown in Fig. 4.4 are used to evaluate how close the two objects are to each other. The input to the fuzzy rules is the distance between the two objects as measured from the closest horizontal edges of their bounding boxes. The reason for using these edges (e.g. the top edge of the bounding box of a table and the bottom edge of a book’s bounding box which is resting on top of the table in the vertical plane projection) is to be able to tell whether the objects are next to each other, or one object is inside of the other object. In the case of an object inside another object, its closest distance will be 0 if only the closest distance is used. The output of the NEAR and SOMEWHAT NEAR membership functions is a fuzzy number in the range of [0, 1].

Figure 4.4: Fuzzy membership functions for NEAR and SOMEWHAT NEAR.
4.4.3 ABOVE and BELOW

In the case of ABOVE and BELOW, the linguistic descriptions generated in the horizontal planes will be that the objects are SURROUNDED or PARTIALLY SURROUNDED; however, they are not surrounded in the vertical plane. If the object is BELOW, no further checking is done, as the English preposition UNDER does not imply that contact is made with another object. However, if the object is ABOVE, the NEAR relationship is used to test to see if the object is in contact with the other object, which would mean that the object is ON TOP of the other object, rather than ABOVE.

4.4.4 ON TOP

If the linguistic description generated is that one of the objects is ABOVE the other one, then the values for NEAR and SOMEWHAT NEAR are checked. If the object is either SOMEWHAT NEAR, or both rules return with a value of 0, then the relationship is that one object is ABOVE the other object. However, if the objects satisfy the fuzzy rule:

If one object is ABOVE the other object and the objects are NEAR and (SURROUNDED or PARTIALLY SURROUNDED) horizontally, then the object is ON TOP of the other object.

then the linguistic description “the object A is ABOVE the object B” is changed to “the object A is ON TOP of the object B.”

4.4.5 INSIDE

If the linguistic descriptions for the horizontal and vertical planes are both SURROUNDED or PARTIALLY SURROUNDED and it is not the case that one object is ON TOP of the other object, then the description is generated to indicate
that one object is INSIDE the other object.

By using these simple rules, the linguistic descriptions that describe the placement of objects has been enhanced by using the richer input given by objects recognized by the robot’s vision system. Test cases are presented in the next section.

### 4.5 Test Cases

In the experiments which follow, images of the scenes presented are shown along with the keypoint clouds placed in the environment. The linguistic descriptions associated with the convex hulls in the horizontal and vertical planes are also shown. The first test is on a scene shown previously in Fig. 4.3 and in more detail in Fig. 4.6. For this scene, the horizontal description is “The can is to the left of the truck box,” and for the vertical plane “The can is to the left of the truck box.” These two descriptions are what one would expect. The fuzzy description of their proximity is that they are near. The interesting point is that in the horizontal description, the language has changed to demonstrate that the truck box extends to the rear of the can. With the previous system, this was impossible.

For the next scene, a candy box has been placed on top of the can, as shown in Fig. 4.8. For this scene, the horizontal descriptions are “The truck box is to the right of the can,” “The truck box is to the right of the candy box,” and “The candy box is surrounded by the can.” For the vertical description, the relationship between the can and the candy box to the truck box are the same as in the horizontal case; however, the relationship for the candy box and the can is “the candy box is on top of the can,” and “The can is under the candy box.” Finally, all three objects are considered to be close. This demonstrates the spatial prepositions ON TOP and UNDER.

For the next experiment shown in Fig. 4.10, two cans are placed on top
Figure 4.5: Input image for Fig. 4.6
Figure 4.6: Fig. 4.5 shows the scene presented; shown here are the SIFT keypoint cloud models. The horizontal descriptions generated:

“The can is to the left of the truck box,”

“The truck box is to the right of the can but extends to the rear.”

The vertical descriptions:

“The can is to the left of the truck box,”

“The Truck Box is to the right of the can.”

The near description is:

“The can is near the truck box.” Images courtesy of Luke.
Figure 4.7: Input image for Fig. 4.8. Courtesy of Luke
Figure 4.8: Fig. 4.7 shows the scene presented; shown here are the SIFT keypoint cloud models. The horizontal descriptions generated:
“The fish can is to the left of the truck box,”
“The truck box is to the right of the fish can but extends to the front,”
“The candy box is to the left of the truck box,”
“The truck box is to the right of the candy box but extends to the front,”
“The candy box is surrounded,”
“The fish can is surrounded.”
The vertical descriptions:
“The can is to the left of the truck box,”
“The Truck Box is to the right of the fish can,”
“The truck box is to the right of the candy box,”
“The candy box is on top of the fish can,”
“The fish can is under the candy box,”
The near description:
“The can and the candy box are near,”
“The can and the truck box are near,”
“The candy box and the truck box are near.”
of the truck box to test the algorithm. Due to calibration problems with the cameras, the object models were not placed in the precise vertical position; hence, the overlapping vertical convex hulls. The system is able to overcome this by looking at the distance between the overlapping hulls. By setting this number to a reasonable level of confidence that the object is near (in this case, 0.9), we are able to overcome this problem. The linguistic descriptions for the horizontal projections in Fig. 4.10 are that the cans are surrounded in the horizontal plane. In the vertical plane, both cans in relationship to the truck box generate the descriptions “The can is on top of the truck box,” and “The truck box is under the can.” Finally, the cans are both near the truck box and “somewhat near” to each other.

The final test is shown in Fig. 4.11, which shows two scenes of the can being placed deeper inside the truck box. In the first scene, the can is protruding enough that the relationship is “the can is on top of the truck box.” By using a higher cut-off for the confidence value of the can being NEAR, the description changes to “The can is inside of the truck box.” As the can sinks into the box, the original value for the cut-off yields that the can is inside of the box. In addition to this problem of needing to “tweak” the values, when the can is considered to be inside of the truck box, the truck box is still considered to be below the can. A better possible description would be that the truck box contains the can.

### 4.6 Discussion

In this chapter, I have shown how Lowe’s SIFT algorithm can be used to create 3D SIFT keypoint clouds, how the cloud models can be positioned into the environment, and how 3D linguistic descriptions can be generated. We have also demonstrated that the system is robust to noise caused by inexpensive cameras, and that it is able to be tuned for different scenarios for INSIDE or ON TOP. The
Figure 4.9: Input image for Fig. 4.10. Image courtesy of Luke
Figure 4.10: Fig. 4.9 shows the scene presented; shown here are the SIFT keypoint cloud models. The horizontal descriptions generated:

“The can baseball can is to the left of the fish can,”
“The fish can is to the right of baseball can,”
“The fish can is surrounded,”
“The baseball can is surrounded.”

The vertical descriptions:

“The baseball can is to the left of the fish can,”
“The fish can is to the right of the baseball can,”
“The truck box is below the baseball can,”
“The truck box is below the fish can,”
“The baseball can is on top of the truck box,”
“The fish can is on top of the truck box.”

The near description:

“The baseball can is somewhat near the fish can,”
“The baseball can is near the truck box,”
“The fish can is near the truck box.”

Images courtesy of Luke
Figure 4.11: The image above corresponds to the SIFT cloud below the input image. Descriptions are generated for the can as it goes inside of the truck box. For the left side, the horizontal description generated is:
“The can is surrounded,”
The vertical descriptions:
“The truck box is under the can,”
“The can is on top of the truck box.”
On the right side, the difference is the description for the can has become “The can is inside of the truck box.” This behavior is caused by the nearness rule and for the left images, the near description is “The can is near the truck box,” whereas in the scene presented on the right side, the description has changed to “The can is somewhat near the truck box.” The ABOVE description is not tripped as the truck box and the can are overlapping in their SIFT cloud models in the vertical projection. Images courtesy of Luke
phrase that an object is “on top of” another object implies that the two objects are touching, so ideally one would want them to be very close. However, there could be situations where there would be some overlap. Consider the situation of balls on top of a pool table. In the vertical projection, the balls would almost certainly be considered to be inside of the pool table; however, most people would say, “the balls are on top of the pool table.” There is also the problem that when the can is INSIDE the truck box, the truck box is considered to be BELOW the can. The solution to this problem is to write a rule that if object 1 is considered to be inside of object 2, then object 2 contains object 1. Using the information at hand, this should be an easy addition to the algorithms and will further enhance spatial language for scene description.
Chapter 5

3D Construction Methods

5.1 Methodology and Background

In this chapter, I investigate the suitability of using natural spatial referencing language to generate 3D models of buildings. These descriptions are inspired by the way one would discuss the architecture of a building and the placement of rooms and salient objects and their relationship to each other. Starting with quantitative descriptions of the building and placement of rooms, I will then move into qualitative descriptions of the placement of rooms and objects in the environment from both exocentric and egocentric perspectives. Ultimately, this will allow for a quick and intuitive way for a human collaborator to see if the machine system (or possibly another human) understands the environment as he or she understands it.

This capability to rapidly build schematics could be considered similar to Barbu et. al [59] systems to learn game behavior and construct buildings from Lincoln Logs. In their systems, one robot is an observer and another is watching to determine how to recreate the building that the first robot has created. Through repeated observations, the knowledge of how to create a building is shown to have been transferred to the observing robot as it looks at a finished building and is able to reason as to how to recreate it. Here, I am also modeling structures; however, natural language descriptions are used to generate a 3D space, ultimately for use
in ensuring that a robot and its human collaborators have a shared understanding of how the language has structured space.

As a part of this research, I developed a library called SRLib to enable an interface to previously developed algorithms for spatial referencing language. Essentially this is a wrapper class that encapsulates functions for calculating points to the LEFT, RIGHT, FRONT, BEHIND, BELOW, and BETWEEN objects in the environment in addition to the techniques outlined in Chapter 4. It also has the capability of using 6 different dictionary files to generate egocentric or exocentric spatial referencing language that can generate language for robot use, or between objects in the horizontal and vertical planes. The API for this library is described in Appendix A.

The 3D construction system shown in Fig. 5.1 is driven by input in the form of a simplified architectural description of the building under consideration. This description is first parsed into its constituent parts of speech through the use of the Stanford Parser[60] which tags the words to indicate which part of speech they comprise. A perl script is used to generate a tree structure that contains the knowledge gleaned from the parsed output of the description. This tree is used to generate an XML file which transfers the data into the spatial language processing and rendering system (SpatialArch) which uses this information in conjunction with the HoF algorithms to create 3D environments and render a full model of the building. In experiments, the model is then verified against the ground truth given by floor plans provided for the original architectural descriptions both quantitative and qualitative metrics.

5.2 Architectural Example

To start with, Fig. 5.2 shows the front view of Barre’s Old Labor Hall in Barre, Vt. This building will be revisited in Chapter 6; it is included here as an
Figure 5.1: The Architectural System Flowchart
illustration, as it has the richest architectural description of the buildings tested. Consider this description:

The first floor plan of the Labor Hall is made up of two areas, the front rooms and entrance composing one area and the meeting hall to the rear. All of the windows on the first and second floors of the building are 1/1 sash windows with flat stock trim. The ceilings in the entrance and two rooms are nine feet high. The stair hall which measures 10’x18’, is accessed from the outside through double leaf, five panel doors. This space contains a stairway leading upstairs along the southwest wall and a doorway accessing the northeast room. The stair hall flows into the southwest room through a large opening in the wall. A southeast door opens into the meeting hall and a door behind the stairs leads to the basement.

The northeast room is 19’x25’ and retains much of its original fabric. The main architectural elements in this room include its maple flooring, pressed tin ceiling, bead board dado, windows, and plaster walls. There are three windows on the northwest wall and two on its northeast wall all having flat stock trim. A modern door has been added to the southeast wall to allow access to the meeting hall.

The southwest room is 17’x25’ and has gone through some change over time. Modern wall paneling (craftwall) and ceiling materials are in the process of being removed to reveal the original plaster surfaces. There are three windows on the northwest wall and three on the southwest wall. A large opening approximately eight feet wide has been created in the southeast wall of this room that enters into the meeting hall. The remains of a small, 4’x4’ restroom are located in the northeast corner of the room.

The meeting hall area occupies the southeast portion of the structure and measures 48’x83’ with 13’ ceilings. A 9’x14’ room has been constructed at some point in time in the north corner of this space. On the southwest elevation two
large freight doors open out to a loading platform. Much of the meeting hall’s original fabric remains having been encapsulated by a drop ceiling and false walls in some locations. The main architectural elements of this space include the engaged decorative posts and beams, the maple flooring, windows and bead board dado and plaster walls and ceilings. The engaged posts and beams are nonstructural and run at twelve foot intervals along the length of the meeting hall. The dado wraps the lower portion of the post with the remaining area of the post covered in plaster. At ceiling height the post meets a boxed beam constructed out of beaded board and flat stock. These beams span the width of the hall. On some of the posts are intact remains of the decorative scheme they were painted in. This pattern consists of a blue-gray on the dado, brown on the chair rail, and a type of grained finish on the plaster part of the post. Just above the dado and below the beam on the post are stencils applied in gold finish. On a few areas of the wall surfaces are remnants of earlier finishes including stenciled ornament, and painted patterns. (from [2])

Figure 5.2: Photo of Barre’s Old Labor Hall, Barre, VT (from Flanders[61])
One challenge in processing this description is that the spatial referencing information is hard to find. There are many extraneous pieces of information in the description that do not satisfy the central theme of this work, which is the construction of a 3D model of the spaces inside this building. It is therefore necessary to pare down the language in order to focus on the spatial language for this project. In this case, the language I am interested in includes the room dimensions and where they are located spatially inside of the building. Extracting this information is outside the scope of this work as it is a linguistics problem. Instead, a simplified description will be used, as follows:

The Barre Old Labor Hall faces south and is 50 x 108 and is 26 feet tall. It has two floors. The stair hall is 10 x 18 and is in the center of the front wall. The eastern room is 19 x 25 and is to the east of the stair hall. The western room is 17 x 25 and is to the west of the stair hall. The meeting hall is 48 x 83 and is north of the western room. The ninebyfourteen room is 9 x 14 and is in the south-west corner of the meeting hall. The upper east room is 28 x 25 and is in the south-east corner of the second floor. The upper west room is 18 x 13 and is in the south-west corner of the second floor. The northern room is 15 x 12 and is to the north of the upper west room. (Note: all units are in feet)

Note that the facing information is incorrect and these unique names do not appear in the original description. This is due to simplifying assumptions that are presented in the next section.

5.3 Simplifying Assumptions

In order to focus on interpreting the 3D spatial referencing language, I have made some simplifying assumptions. The first simplification is that all buildings can be constructed exocentrically regardless of the orientation that the description is presented. The reason for this is that all buildings when described using
cardinal directions have an external frame, which if rotated, does nothing to the actual structure and relationships between elements in the building. In fact, if one considers the 4 primary and 4 intermediary directions, we essentially have a modulo 8 system. With this in mind, it is trivial to determine the facing of the building and find an offset in order to translate all of the exocentric descriptions into one that always faces south. This simplifies the problem as it means that there is only one orientation that needs to be considered when constructing a building. After construction, using the same offset, the building can be rotated into the correct orientation by simply reversing the process.

A second assumption is that the information will initially come in a certain order. This is to establish boundaries on the floor plan that are then used to build up the following floors. While some modern architectures employ overhangs from the building’s base footprint, most buildings are defined by their first floor in terms of the exterior shape and dimensionality. Of course, there are exceptions, some of which will be addressed in the experimental results.

Finally, only certain relationships are recognized. The descriptions allowed are exocentric (north, south, east, west, their intermediaries, and center), egocentric (left, right, front, behind, above, below), and relative to other rooms. In addition, range information (i.e. how far away a room is located) can be used to specify where a room is located in reference to another landmark.

5.4 Representation

The representation used in this work is an n-ary tree as shown in Fig. 5.3. This was selected as it is flexible, computable and can be constructed incrementally.

As shown in Fig. 5.3, a building is composed of a root node name Building. This node contains information about the overall building dimensions. Under-
neath the Building node lie n-Floor nodes, which contain information regarding the dimensions of the floors. Underneath the Floor nodes, n-Room and/or n-Portal nodes are located. These nodes are used to represent the location and dimension of rooms and portals (e.g. doors, windows, stairs, etc.). For this work, I am concentrating on just the placement of rooms; however, initial support has been made for portals in the form of an elevator.

The use of this tree structure is beneficial in that XML files are tree-like in their structure, which means that by performing a tree traversal of the building representation, one can easily create a well-formed XML document containing the information which makes this data interoperable with a multitude of other systems.

5.5 Parser

The parser used in this work is the Stanford Parser[60]. Its purpose is to tag parts of speech to determine where in the sentence structure the information is located. While it would certainly be possible to not make use of the parser at all, it lends a great deal of flexibility in that it can label whether information belongs to the subject of the sentence or the direct object (or other pieces of grammar).
This is extremely useful as it allows for extraneous words to be ignored. Using the BOLH example, the output from the parser renders the following output for the description given in section 5.2 (as it is passed through the perl-based tree-building system):

item->The/DT<-
item->Barre_Socialist_Labor_Party_Hall/NNP<-
*Found Barre_Socialist_Labor_Party_Hall/NNP
item->faces/VBZ<-
***** got facestring -> north-west xlate to south offset -3/<----*****
item->north-west/NNS<-
The index is: 3
item->and/CC<-
item->is/VBZ<-
item->50/CD<-
item->x/JJ<-
The index is: 7
item->108/CD<-
item->and/CC<-
item->is/VBZ<-
item->20/CD<-
***got -> Barre_Socialist_Labor_Party_Hall, 0, 0, 0, 50, 108, 20<-
****

The tags generated correspond to a tree structure similar to diagramming a sentence into its constituent grammatical parts. This first sentence sets up the tree structure with its root building node. The parts of speech that are necessary for this program are nouns, dimensions, distances, cardinal directions, and spatial prepositions. These are tagged with NNS or NNP (noun phrases, singular or
plural), CD (numbers), NN or JJ (adjective), and IN (preposition), respectively. Also, it is shown in the output that the system has set the orientation the offset (the building is originally facing northwest) and will translate any following descriptions based on the cardinal direction offset. At the end, the building’s overall dimensions are loaded and the tree construction will begin with the root Building node.

The parser and the perl-based system work in tandem; the parser feeds one sentence at a time into the tree-construction system that then extracts the information and places it logically in the tree’s structure. Some a priori knowledge about the HoF is assumed, namely that when dealing with rectangular objects, the main direction of greatest truth is perpendicular to the center of the object under question. This is from previous results in exploring Confidence Regions (CR) (briefly touched upon in Chapter 3) in Blisard[8]. Knowing the topology of the Confidence Region space, the perl-based tree system uses the directions of greatest truth to place objects in the virtual environment with their associated dimensional information. In addition, if there is displacement information included, then this is used to offset the placement.

This calculation using the direction of greatest confidence from the objects is only possible as the rooms/objects that are being considered are regular-shaped (e.g. non-convex rectangular polygons). I have shown in previous work [8] that the area which will yield the highest values of confidence for x being in the direction of y is found on the perpendiculars describing the primary directions emanating from the centroids of this class of objects. This enables the placement of objects, which is later checked during the qualitative assessment, to be placed in areas with the highest degree of confidence that they satisfy their spatial reference. It is also possible to place objects off this main direction as shown in Fig. 5.4 by keeping the object’s bounding box within a set of rays defined by the diagonal of the box.
Figure 5.4: Possible placements for an object to the right/east of the elevator from the perspective of an observer in the bottom of the screen (north/forward is to the top of the image)

Again, this only works for rectangular shaped objects; non-standard shapes will require a more elaborate method of defining the CR involving the eigenvectors of the covariance matrix for the convex hull vertices of the object and has not been investigated in the context of architectural descriptions.

When objects under consideration refer to previous tree entries, a search must be made through the tree to find the object under consideration. This tree search is facilitated by assigning to each entry a unique identifier in the form of FFx, RRx, or PPx (for Floors, Rooms, and Portals, with a unique number, respectively). This ensures uniqueness in the tree’s nodes. The search is a standard in-order traversal of the leaves.

Once the parsing/processing has completed, the perl-based system performs an in-order traversal and writes out an XML file with the placements of the building entities. At this point, the parsing/knowledge extraction phase is complete and it is time to render the model using the XML file.

There is also some inherent knowledge in the system. The default units are
feet and the default height for a story is 12 feet; however this is a subjective number (anywhere from 8-16 feet is considered acceptable). If no height information for the building is provided, the default value of 12 is used; otherwise, the height and number of stories is used to infer the story height. Default knowledge is also present for elevators (modern ones conformant to the Americans with Disabilities Act) and door sizes. It is also possible to describe a room as small, medium, or large that sets a percentage of the floor plan to determine their sizes. A small room is 5% of the total floor space, a medium room is 15%, and large is 30%. These numbers are arbitrary and would require a human subject experiment to make a determination of what makes humans classify a room as a certain size.

5.6 Spatial Rendering System

Once the XML file has been created, it is up to the spatial rendering system to read the file and render the building. The XML file is processed, and a representation is built using voxels 1 foot cubed in size. The reason for this level of fidelity is that most descriptions encountered are measured to the nearest foot.

As an example, consider Fig. 5.5 with the description and the following XML file:

```xml
<building name = "Test_Building" x = "0" y = "0" z = "0" width = "30" height = "20" depth = "15">
  <floor name ="FF1" x = "0" y = "0" z = "0" width = "30" height = "10" depth = "15"> 
    <room name ="RR1 "kitchen" x = "20" y = "0" z = "5" width = "10" height = "10" depth = "0" />
    <portal name ="PP1 front door" x = "15" y = "0" z = "15" width = "4" height = "8" depth = "1" />
  </floor>
</building>
```
This yields the tree shown in Fig. 5.5.

This rendering shown in Fig. 5.6 is simple enough that multiple views are not necessary in order to see that it is a correct rendering. As the orientation is set, the phrase “northeast corner of the first floor” sufficiently describes the location of the kitchen. This model is a full 3D model, and as such, it is possible to view the model from many different views. Finally, this is a right-handed coordinate system with the origin at the northwest corner of the building, positive x is to the right, positive y is up, and negative z is toward the reader.

The spatial rendering system can also perform spatial referencing in 3D as was shown in Chapter 4. To accomplish this, the projections onto the horizontal and vertical planes are performed as before, within this self-contained system. The steps for this process are as follows:

1. Calculate the convex hulls of the argument and referent objects;
Figure 5.6: The Test_Building faces south and is 30 x 15 and is 20 feet tall. It has 2 floors. The kitchen is 10 x 10 and is in the northeast corner of the first floor.
(2) Project convex hulls onto the horizontal and vertical planes;

(3) Generate SRL as shown in Chapter 4.

Finally, the system has the ability to place a virtual robot into the environment and have it describe the salient rooms around it based on its current perspective as it moves through the rendered environment. This capability and experiments are described in Chapter 7.

5.7 Evaluation Metrics

In this section, metrics are presented as a means of comparing the constructed 3D model to the actual building (the ground truth). I propose metrics to evaluate the buildings based on both quantitative and qualitative characteristics. For the quantitative evaluation, I will compare the surface area covered in the horizontal plane with the known ground truth as an average of the overlapping areas of the rooms. This average is calculated by the formula:

\[
\frac{1}{n} \sum_{i} \left(1 - \frac{|GT_i - R_i|}{GT_i}\right) \times 100
\]

where \(GT_i\) is the area of the ground truth, \(R_i\) is the rendered area shared with \(GT_i\) and \(i\) ranges over the \(n\) rooms under consideration. This yields a percentage of the overlap over all the rooms of the building that measures how closely in a quantitative manner the rendering matches the ground truth. While this is nowhere near as methodical as Pesaresi and Bedediktsson’s Differential Morphological Profile[62] for image segmentation as used by Shyu et. al in GeoIRIS[63], it should capture the aspect of whether or not the rooms are placed correctly with regard to the ground truth.

In order to evaluate the qualitative aspect, I pick a salient room and use the HoF to generate linguistic descriptions based on the rendering and the floor plan and compare them to see how well they match. The description is considered a full match if they generate the same linguistic description, 0.5 of a match if the
direction is correct but a linguistic hedge is used in the rendering, and 0 if the linguistic descriptions do not match. Once again, these scores are used to create a percentage match of the entire set of descriptions provided by the ground truth.

Ideally, these two measures should be in complete agreement if the rendering matches the ground truth. However, the qualitative metric may generate a higher score, as the placement of the rooms may be correct from a qualitative perspective, but their dimensions or other assumptions may cause them to not overlap at all. Consider the case of a description of a bathroom that is down a short hall to the right of an elevator. Most people would describe its location as to the right of the elevator when in fact it is to the back-right of the elevator. In this case, both of the metrics will yield low scores for the placement of this room. This configuration is shown in Fig. 5.7.

Figure 5.7: An example where both qualitative and quantitative measures would fail. Most people would say the bathrooms are to the left or right of the elevator, but they are actually to the back-right and back-left of the elevator. The circle with an arrow denotes viewing perspective.
Chapter 6

3D Construction Test Buildings

To explore the 3D construction approach, a series of test cases is presented in this chapter. There are a total of 5 buildings, the first being a synthetic one created for testing and the remaining 4 examples for which there is real-world ground truth gathered from the National Park Service historical building registry.

6.1 Synthetic Building

This is a test of the capabilities of the system. Descriptions will be added incrementally to illustrate how different types of descriptions are analyzed with the resultant spatial rendering. Then I will investigate how well the 3D spatial description generated from the constructed 3D models agree with the description generated from the ground truth.

Description 1:

Table 6.1: Test Buildings

<table>
<thead>
<tr>
<th>Building</th>
<th>Feature</th>
</tr>
</thead>
<tbody>
<tr>
<td>Synthetic Building</td>
<td>Demonstration of functionality</td>
</tr>
<tr>
<td>Barre’s Old Labor Hall</td>
<td>Rich architectural description</td>
</tr>
<tr>
<td>Egyptian Theater</td>
<td>Balcony and Mezzanine</td>
</tr>
<tr>
<td>Oregon Central School</td>
<td>Slightly non-rectangular</td>
</tr>
<tr>
<td>Coghlan Castle</td>
<td>Multiple small rooms and a turret</td>
</tr>
</tbody>
</table>
The Synthetic_Building faces south and is 40 x 40 and is 20 feet tall. It has 2 floors.

Parser Output:

The DT Synthetic_Building/NNP faces/VBZ south/JJ and/CC is/VBZ 40/CD x/JJ 40/CD and/CC is/VBZ 20/CD feet/NNS tall/JJ. It/PRP has/VBZ 2/CD floors/NNS.

The information from this is extracted through the tree builder script. It first pulls out the name and the base floor dimensions and the overall height by looking for the appropriate tags. The number of floors is used as a divisor to the overall height, which sets a global variable to the height of a story. If no information about height were provided, then it would have defaulted to use the “standard” story height of 12 feet. As the parse is read in, at the end of each sentence the current information is pushed into the tree. At the end of the first sentence, only the <building> node is created. When the second sentence is processed, the two floor nodes are added using the building’s outer dimensions. Since this is the end of the sentence, the internal tree is traversed to write the following XML file:

Tree in XML format:

<?xml version = "1.0" encoding = "UTF-8" standalone="no" ?>

<building name = "Synthetic_Building" x = "0" y = "0" z = "0"
width = "40" depth = "40" height = "20" faces = "south">

    <floor name = "FF1" x = "0" y = "0" z = "0" width = "40"
depth = "40" height = "10">

    </floor>

    <floor name = "FF2" x = "0" y = "0" z = "0" width = "40"
depth = "40" height = "10">

    </floor>

</building>
Figure 6.1: The beginning Synthetic Building tree

Which results in the following rendering:

Description 2: “The rec_room is 10 x 10 and is in the center of the first floor,”

This description adds new information to be added to the internal tree. First, the parser tags the name, dimensions, and placement which is then processed to build a preliminary string describing this room. The reference of “center of the first floor” tells the program to search the tree to find the dimensions of the first floor to place the room in the center. As this is the last line, the tree is again traversed and the following XML file is generated:

Tree in XML format:

```xml
<?xml version = “1.0” encoding = “UTF-8” standalone=”no” ?>
<building name = “First_Example” x = “0” y = “0” z = “0” width = “40” depth = “40” height = “20” faces = “south”>
  <floor name = “FF1” x = “0” y = “0” z = “0” width = “40” depth = “40” height = “10”>
    <room place = “RR1 rec_room” x = “14” y = “0” z = “14” width = “10” depth = “10” height = “10” />
  </floor>
</building>
```
Figure 6.2: The Empty Synthetic Building
Figure 6.3: Synthetic Building Tree with first room node added

```
<floor name="FF2" x="0" y="0" z="0" width="40"
depth="40" height="10" >
</floor> </building>
```

This renders into the following two views:

The next description adds references with respect to the rec_room.

**Description 3:**

The broom_closet is 2 x 2 and is 2 feet to the south of the rec_room. The shoe_closet is 3 x 3 and is 4 feet to the north of the rec_room. The gun_closet is 4 x 4 and is 6 feet to the west of the rec_room. The sundry_closet is 5 x 5 and is 9 feet to the east of the rec_room.

For these examples, the tree is searched for the dimensions of the referent objects (in this case, the rec_room) and its dimensions are used to determine the direction and distance needed to satisfy the description. Since these are exocentric descriptions, this placement is easily performed and is shown in Fig. 6.7.

Egocentric descriptions are also supported by exploiting previous work with HoF Confidence Regions that was revisited in Chapter 3. By using the directions
Figure 6.4: Synthetic Building with first room added from the front
Figure 6.5: Synthetic Building with first room added from the top
Figure 6.6: Tree corresponding to Fig 6.7
Figure 6.7: The Synthetic Building with rooms added NSEW of the \textit{rec\_room}
which have the highest confidence (namely, in the middle of an object going perpendicular to the object), I can place rooms around a referent using egocentric language. Consider the following description:

*Description 4:*

*The Synthetic Building faces south and is 40 x 40 and is 20 feet tall. It has 2 floors. The rec_room is 10 x 10 and is in the center of the first floor. The broom_closet is 2 x 2 and is in front of the rec_room. The shoe_closet is 3 x 3 and is 2 feet behind the rec_room. The gun_closet is 4 x 4 and is 4 feet to the left of the rec_room. The sundry_closet is 5 x 5 and is 7 feet to the right of the rec_room.*

This description is parsed as before, and in the case where there is no distance information given (broom closet), the room is placed next to the referent object as shown in Figure 6.8. The tree structure is identical to the tree shown in Fig. 6.6.

Another aspect that needs to be considered is to be able to handle the case of objects above and below another object. Two views of the Synthetic Building are shown in Figs. 6.9, after the following two sentences have been added to the description:

*Description 5:*

*The second_floor_bathroom is 6 x 6 and is above the sundry_closet. The basement is 7 x 7 and is below the gun_closet.*

Above and below are handled in a similar fashion as the previous relationships, i.e. search the tree for the referent object and use its known dimensions as a basis for the placement of the room.

This placement is performed in a similar manner as before, namely going in the direction of greatest confidence in above and below. The tree structure is the same as Fig. 6.6, except the basement is a child node of FF1 and the second_floor_bathroom is a child node of FF2. The placement is slightly off as I
Figure 6.8: The Synthetic Building with rooms added egocentrically
Figure 6.9: Synthetic Building showing rooms added ABOVE and BELOW from the front.
am using the starting coordinates for the referent object translated either above or below the current floor to render the new room. Table ?? shows the contents of the nodes.

Next, I consider whether the 3D spatial descriptions generated from the 3D constructions will correspond to the description generated from the ground truth building. Consider the following description:

*Description 6:*

*The Synthetic_Building faces south and is 40 x 40 and is 20 feet tall. It has 2 floors. The rec_room is 10 x 10 and is in the center of the first floor. The broom_closet is 2 x 2 and is in front of the rec_room. The shoe_closet is 3 x 3 and is 2 feet behind the rec_room. The gun_closet is 4 x 4 and is 4 feet to the left of the rec_room. The sundry_closet is 5 x 5 and is 7 feet to the right of the rec_room. The second_floor_bathroom is 6 x 6 and is above the sundry_closet. The basement is 7 x 7 and is below the gun_closet.*

Which was used to create Figs. 6.9 and ?? For test purposes, one referent room was selected; other rooms are compared to the referent room. First, 3D convex hulls are computed using the qhull algorithm. The second step is to project the vertex voxels of the convex hull onto the horizontal plane in the same manner as was previously done for the SIFT keypoint clouds models in Chapter 4. These convex hulls are then used as input to the language generation component.

First, we will consider the linguistic descriptions from an egocentric perspective and see how they compare to the exocentric description used to build this collection of rooms (italics is the rendered building using egocentric descriptions, bold is using exocentric descriptions). These two are similar due to the orientation of the building.

->*The RR1 broom_closet is in front of the rec_room*

*(the description is satisfactory)*
Table 6.2: Synthetic Building Node Contents

<table>
<thead>
<tr>
<th>Node</th>
<th>Contents</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Building</strong></td>
<td>name=”Synthetic_Building”  x=”0” y=”0” z=”0” width=”40” depth=”40” height=”20” faces=“south”</td>
</tr>
<tr>
<td>FF1</td>
<td>name=“FF1”  x=”0” y=”0” z=”0” width=”40” height=”10”</td>
</tr>
<tr>
<td>FF2</td>
<td>name=“FF2”  x=”0” y=”0” z=”0” width=”40” height=”10”</td>
</tr>
<tr>
<td>RR1</td>
<td>place=“rec_room”  x=”14” y=”0” z=”14” width=“10” depth= “10” height= “10”</td>
</tr>
<tr>
<td>RR2</td>
<td>place=“broom_closet”  x=”18” y=”0” z=”25” width=“2” depth= “2” height= “10”</td>
</tr>
<tr>
<td>RR3</td>
<td>place=“shoe_closet”  x=”17.5” y=”0” z=”8” width=“3” depth= “3” height= “10”</td>
</tr>
<tr>
<td>RR4</td>
<td>place=“gun_closet”  x=”5” y=”0” z=”17” width=“4” depth= “4” height= “10”</td>
</tr>
<tr>
<td>RR5</td>
<td>place=“sundry_closet”  x=”32” y=”0” z=”16.5” width=“5” depth= “5” height= “10”</td>
</tr>
<tr>
<td>RR6</td>
<td>place=“second_floor_bathroom” x=”32” y=”10” z=“16” width=“6” depth= “6” height= “10”</td>
</tr>
<tr>
<td>RR7</td>
<td>place=“basement”  x=”5” y=”-10” z=”17” width=“7” depth= “7” height= “10”</td>
</tr>
</tbody>
</table>
The RR1 broom_closet is south of the rec_room
(the description is satisfactory)
The RR2 shoe_closet is behind the rec_room
(the description is satisfactory)
The RR2 shoe_closet is north of the rec_room
(the description is satisfactory)
The RR3 gun_closet is to the left of the rec_room
The RR3 gun_closet is to west of the rec_room
The RR4 sundry_closet is to the right of the rec_room
(the description is satisfactory)
The RR4 sundry_closet is east of the rec_room
(the description is satisfactory)
The RR5 second_floor_bathroom is to the right of the rec_room
(the description is satisfactory)
The RR5 second_floor_bathroom is to the east of the rec_room
(the description is satisfactory)
The RR5 second_floor_bathroom is above right of the rec_room
(the description is satisfactory)
The RR6 basement is to the left of the rec_room
but extends forward relative to the rec_room
(the description is satisfactory)
The RR6 basement is to the west of the rec_room
but extends south relative to the rec_room
(the description is satisfactory)
The RR6 basement is to the below left of the rec_room
(the description is satisfactory)
These descriptions match quite well with what one would expect. In a similar fashion, one can use global descriptions that generate language virtually identical to the descriptions here. This is not surprising due to standard frame being used and the orientation of the descriptions being given. Note that all of the linguistic descriptions for the following four buildings can be found in Appendix C.

6.2 Barre’s Old Labor Hall [2]

The first real building I am investigating is Barre’s Old Labor Hall found in Barre, VT. This historical building was the initial inspiration for this work as the interior architectural description is very rich with information as to the layout of the building and had an accompanying floor plan to provide ground truth. The full description was previously shown in the first section. The simplified architectural description language is shown below:

The Barre_Old_Labor_Hall faces south and is 50 x 108 and is 26 feet tall. It has two floors. The stair_hall is 10 x 18 and is in the center of the front wall. The eastern_room is 19 x 25 and is to the east of the stair_hall. The western_room is 17 x 25 and is to the west of the stair_hall. The meeting_hall is 48 x 83 and is north of the western_room. The ninebyfourteen_room is 9 x 14 and is in the south-west corner of the meeting_hall. The upper_east_room is 28 x 25 and is in the south-east corner of the second floor. The upper_west_room is 18 x 13 and is in the south-west corner of the second floor. The northern_room is 15 x 12 and is to the north of the upper_west_room.

This 3D construction is shown in Figs. 6.10-6.12.

As can be seen on the first floor, there is a 4’ section that is misplaced, which causes the matching for the first floor to drop to 92%. The second floor is assumed to be correct as it matches the description; however, no floor plan could
Figure 6.10: Barre’s Old Labor Hall (BOLH) from the front.
Figure 6.11: BOLH First Floor
Figure 6.12: BOLH Second Floor
Figure 6.13: BOLH’s first floor plan [2]
Figure 6.14: BOLH rendering (green) vs. Ground Truth (yellow)
be found for the second floor. The qualitative descriptions of the building relative
to the stair_hall match for all but one half-match, which has yields a score of 90%.
The tree for this building is shown in Fig. 6.15.

6.3 The Egyptian Theater [3]

The Egyptian Theater was built in 1922 in Coos Bay, OR originally as a playhouse which was converted in 1925 to a movie theater. This rendering is interesting as it necessitates creating floors that do not span the entire footprint of a building (the Mezzanine and Balcony). The architectural description is primarily concerned with the history and interior decorations and is limited in terms of the layout of the building. This perhaps is because there is much inherent knowledge about a playhouse/movie theater that can be deduced from simply stating that there are balconies and Mezzanines (e.g. a balcony will face the stage and not be above it). The architectural description language for this building is as follows:

The Egyptian theater faces south and is 75 x 145 and is 45 feet tall. It has 3 floors. The box_office is 10 x 10 and is in the center of the southern
Table 6.3: BOLH Node Contents

<table>
<thead>
<tr>
<th>Node</th>
<th>Contents</th>
</tr>
</thead>
<tbody>
<tr>
<td>Building</td>
<td>name=&quot;Barre_Old_Labor_Hall&quot;  x=&quot;0&quot;  y=&quot;0&quot;  z=&quot;0&quot;  width=&quot;50&quot;  depth=&quot;108&quot;  height=&quot;26&quot;  faces=&quot;south&quot;</td>
</tr>
<tr>
<td>FF1</td>
<td>name=&quot;FF1&quot;  x=&quot;0&quot;  y=&quot;0&quot;  z=&quot;0&quot;  width=&quot;50&quot;  depth=&quot;108&quot;  height=&quot;10&quot;</td>
</tr>
<tr>
<td>FF2</td>
<td>name=&quot;FF2&quot;  x=&quot;0&quot;  y=&quot;13&quot;  z=&quot;0&quot;  width=&quot;50&quot;  depth=&quot;108&quot;  height=&quot;10&quot;</td>
</tr>
<tr>
<td>RR1</td>
<td>place=&quot;stair_hall&quot;  x=&quot;18&quot;  y=&quot;0&quot;  z=&quot;90&quot;  width=&quot;10&quot;  depth=&quot;18&quot;  height=&quot;13&quot;</td>
</tr>
<tr>
<td>RR2</td>
<td>place=&quot;east_room&quot;  x=&quot;28&quot;  y=&quot;0&quot;  z=&quot;83&quot;  width=&quot;19&quot;  depth=&quot;25&quot;  height=&quot;13&quot;</td>
</tr>
<tr>
<td>RR3</td>
<td>place=&quot;west_room&quot;  x=&quot;0&quot;  y=&quot;0&quot;  z=&quot;83&quot;  width=&quot;17&quot;  depth=&quot;25&quot;  height=&quot;13&quot;</td>
</tr>
<tr>
<td>RR4</td>
<td>place=&quot;north_corner&quot;  x=&quot;0&quot;  y=&quot;0&quot;  z=&quot;69&quot;  width=&quot;9&quot;  depth=&quot;14&quot;  height=&quot;13&quot;</td>
</tr>
<tr>
<td>RR5</td>
<td>place=&quot;meeting_hall&quot;  x=&quot;0&quot;  y=&quot;0&quot;  z=&quot;0&quot;  width=&quot;48&quot;  depth=&quot;83&quot;  height=&quot;13&quot;</td>
</tr>
<tr>
<td>RR6</td>
<td>place=&quot;upper_east_room&quot;  x=&quot;22&quot;  y=&quot;13&quot;  z=&quot;83&quot;  width=&quot;28&quot;  depth=&quot;25&quot;  height=&quot;13&quot;</td>
</tr>
<tr>
<td>RR7</td>
<td>place=&quot;upper_west_room&quot;  x=&quot;0&quot;  y=&quot;13&quot;  z=&quot;95&quot;  width=&quot;18&quot;  depth=&quot;13&quot;  height=&quot;13&quot;</td>
</tr>
<tr>
<td>RR8</td>
<td>place=&quot;upper_northwest_room&quot;  x=&quot;0&quot;  y=&quot;13&quot;  z=&quot;83&quot;  width=&quot;15&quot;  depth=&quot;12&quot;  height=&quot;13&quot;</td>
</tr>
</tbody>
</table>
wall. The concession stand is 25 x 10 and is 25 feet to the north of the box office. The stage is 60 x 40 and is in the center of the northern wall. The mezzanine is 75 x 25. The restrooms are 40 x 15 and in the center of the southern wall of the mezzanine. There is a 10 x 10 office to the west of the restrooms. The balcony is 75 x 70. The projection room is 50 x 20 and is near the center of the southern wall.

Note that this description uses exterior features (the walls) to place rooms. It also uses a concept of near that was discussed in Chapter 4. The renderings for this description and the ground truth are shown in Fig. 6.8:1-8 (note that the external walls are not drawn so as to not obscure the structure).

For the Egyptian Theater, the first floor has a 90% match in terms of overlapping space. The Mezzanine also fares well at 90%. The positioning of the projection room on the balcony is slightly off and too large, which leads to an 80% match of the overlapping area (this is also due to the curved feature not being matched). When comparing the linguistic descriptions generated, qualitatively the buildings match from the perspective of the box office and the concession stand, except for the projection room which adds the linguistic hedge of “shifted to the front” when it should be above the concession stand. Overall, quantitatively the match is 64% and qualitatively 100%. The tree for the Egyptian Theater is shown in Figure 6.25.

6.4 The Oregon Central School [4]

The Oregon Central School was built in 1909-1910 in Milton-Freewater, OR; a rural community of less than 6,000 inhabitants. It has a somewhat unique design in that it is made of two slightly different sized blocks where one is slightly wider and the other is slightly longer. This description makes some use of quantitative
Table 6.4: Egyptian Theater Node Contents

<table>
<thead>
<tr>
<th>Node</th>
<th>Contents</th>
</tr>
</thead>
<tbody>
<tr>
<td>Building</td>
<td>name=&quot;Egyptian_Theater&quot; x=&quot;0&quot; y=&quot;0&quot; z=&quot;0&quot; width=&quot;75&quot; depth=&quot;145&quot; height=&quot;45&quot; faces=&quot;south&quot;</td>
</tr>
<tr>
<td>FF1</td>
<td>name=&quot;FF1&quot; x=&quot;0&quot; y=&quot;0&quot; z=&quot;0&quot; width=&quot;75&quot; depth=&quot;145&quot; height=&quot;15&quot;</td>
</tr>
<tr>
<td>FF2</td>
<td>name=&quot;mezzanine&quot; x=&quot;0&quot; y=&quot;15&quot; z=&quot;120&quot; width=&quot;75&quot; depth=&quot;25&quot; height=&quot;10&quot;</td>
</tr>
<tr>
<td>FF3</td>
<td>name=&quot;balcony&quot; x=&quot;0&quot; y=&quot;30&quot; z=&quot;75&quot; width=&quot;75&quot; depth=&quot;70&quot; height=&quot;15&quot;</td>
</tr>
<tr>
<td>RR1</td>
<td>place=&quot;box_office&quot; x=&quot;31.5&quot; y=&quot;0&quot; z=&quot;135&quot; width=&quot;10&quot; depth=&quot;10&quot; height=&quot;15&quot;</td>
</tr>
<tr>
<td>RR2</td>
<td>place=&quot;concession_stand&quot; x=&quot;23.5&quot; y=&quot;0&quot; z=&quot;99&quot; width=&quot;25&quot; depth=&quot;10&quot; height=&quot;15&quot;</td>
</tr>
<tr>
<td>RR3</td>
<td>place=&quot;stage&quot; x=&quot;6.5&quot; y=&quot;0&quot; z=&quot;0&quot; width=&quot;60&quot; depth=&quot;40&quot; height=&quot;15&quot;</td>
</tr>
<tr>
<td>RR4</td>
<td>place=&quot;restrooms&quot; x=&quot;16.5&quot; y=&quot;15&quot; z=&quot;130&quot; width=&quot;40&quot; depth=&quot;15&quot; height=&quot;15&quot;</td>
</tr>
<tr>
<td>RR5</td>
<td>place=&quot;projection_room&quot; x=&quot;11.5&quot; y=&quot;30&quot; z=&quot;125&quot; width=&quot;50&quot; depth=&quot;20&quot; height=&quot;15&quot;</td>
</tr>
</tbody>
</table>
Figure 6.17: Egyptian Theater overhead view
Figure 6.18: First Floor of the Egyptian Theater[3]
Figure 6.19: Mezzanine Level of the Egyptian Theater[3]

Figure 6.20: Balcony Level of the Egyptian Theater[3]
Figure 6.21: Egyptian Theater first floor plan
Figure 6.22: Egyptian Theater Mezzanine floor plan
Figure 6.23: Egyptian Theater Balcony floor plan
Figure 6.24: Egyptian Theater rendering (green) vs. Ground Truth (yellow)
Figure 6.25: Tree for the Egyptian Theater
descriptions, so these are used (instead of making guesses based on the floor plan). The architectural description language for this building is:

*The OR Central School faces sourn and is 69 x 95 and is 26 feet tall. It has 2 floors. The sw corner classroom is 25 x 31 and is in the south-west corner of the first floor. The se corner classroom is 25 x 31 and is in the south-east corner of the first floor. The nw corner classroom is 25 x 31 and is in the north-west corner of the first floor. The ne corner classroom is 25 x 31 and is in the north-east corner of the first floor. The upper sw classroom is 25 x 31 and is in the south-west corner of the second floor. The upper nw classroom is 25 x 31 and is in the north-west corner of the second floor. The library is 25 x 95 and is along the eastern-wall on the second floor.*

As stated in the description, all of the classrooms are the same size. I selected this building to show how the corners of the building can be used to place rooms. Of course, not just floors can be used as a reference; any named component could be used to place the rooms (or subrooms) of a building using these intermediary cardinal directions. It also makes use of the eastern-wall as a reference as to where to place the library. The renderings and ground truth are shown in Fig. 6.10 1-7.

The renderings map qualitatively for the rooms that have been rendered quite well, with both the ground truth and 3D construction’s linguistic descriptions matching for the rooms that are depicted giving a qualitative score of 100%. However, from a qualitative standpoint, due to the rectangular shape of the floor plan, the classrooms on the western side only overlap 70%, giving a combined score of 83% for all the rooms rendered. However, considering the missed rooms lowers this average down to 60%. Adding those rooms in will help; however, without the ability to render this T-shape, their placement will not be exact and do not help to improve the score. The tree is shown in Fig. 6.33.
Figure 6.26: Oregon School front view

Table 6.5: Oregon School Node Contents

<table>
<thead>
<tr>
<th>Node</th>
<th>Contents</th>
</tr>
</thead>
<tbody>
<tr>
<td>Building</td>
<td>name=&quot;OR_Central_School&quot; x=&quot;0&quot; y=&quot;0&quot; z=&quot;0&quot; width=&quot;69&quot; depth=&quot;95&quot; height=&quot;26&quot; faces=&quot;south&quot;</td>
</tr>
<tr>
<td>FF1</td>
<td>name=&quot;FF1&quot; x=&quot;0&quot; y=&quot;0&quot; z=&quot;0&quot; width=&quot;69&quot; depth=&quot;95&quot; height=&quot;13&quot;</td>
</tr>
<tr>
<td>FF2</td>
<td>name=&quot;FF2&quot; x=&quot;0&quot; y=&quot;13&quot; z=&quot;0&quot; width=&quot;69&quot; depth=&quot;95&quot; height=&quot;13&quot;</td>
</tr>
<tr>
<td>RR1</td>
<td>place=&quot;sw_corner_classroom&quot; x=&quot;0&quot; y=&quot;0&quot; z=&quot;64&quot; width=&quot;25&quot; depth=&quot;31&quot; height=&quot;13&quot;</td>
</tr>
<tr>
<td>RR2</td>
<td>place=&quot;se_corner_classroom&quot; x=&quot;44&quot; y=&quot;0&quot; z=&quot;64&quot; width=&quot;25&quot; depth=&quot;31&quot; height=&quot;13&quot;</td>
</tr>
<tr>
<td>RR3</td>
<td>place=&quot;nw_corner_classroom&quot; x=&quot;0&quot; y=&quot;0&quot; z=&quot;0&quot; width=&quot;25&quot; depth=&quot;31&quot; height=&quot;13&quot;</td>
</tr>
<tr>
<td>RR4</td>
<td>place=&quot;ne_corner_classroom&quot; x=&quot;44&quot; y=&quot;0&quot; z=&quot;0&quot; width=&quot;25&quot; depth=&quot;31&quot; height=&quot;13&quot;</td>
</tr>
<tr>
<td>RR5</td>
<td>place=&quot;upper_sw_corner_classroom&quot; x=&quot;0&quot; y=&quot;13&quot; z=&quot;64&quot; width=&quot;25&quot; depth=&quot;31&quot; height=&quot;13&quot;</td>
</tr>
<tr>
<td>RR6</td>
<td>place=&quot;upper_nw_corner_classroom&quot; x=&quot;0&quot; y=&quot;13&quot; z=&quot;0&quot; width=&quot;25&quot; depth=&quot;31&quot; height=&quot;13&quot;</td>
</tr>
<tr>
<td>RR7</td>
<td>place=&quot;library&quot; x=&quot;44&quot; y=&quot;13&quot; z=&quot;0&quot; width=&quot;25&quot; depth=&quot;95&quot; height=&quot;13&quot;</td>
</tr>
</tbody>
</table>
Figure 6.27: First floor of the Oregon School
Figure 6.28: Second Floor of the Oregon School
Figure 6.29: Oregon School first floor plan [4]
Figure 6.30: Oregon School second floor plan[4]
Figure 6.31: Oregon School first floor rendering (green) vs. Ground Truth (yellow)
Figure 6.32: Oregon School first floor rendering (green) vs. Ground Truth (yellow)
Figure 6.33: Tree for the Oregon School
6.5 Coghlan Castle [5]

Coghlan Castle is a 2-story residence built between 1906-1909 in Rolla, ND. The only qualitative description available is the overall footprint of 40 x 50 feet with a 12 feet in diameter two-story turret. The residence is in disrepair, so the conservators have made guesses as to the uses of the rooms (which leads to multiple bedrooms). This building will also show the shortcomings of using just the architectural description as the building is L-shaped and not rectangular. The architectural description language for this building is:

\textit{Coghlan_Castle faces south and is 40 x 50 and is 24 feet tall. It has 2
floors. There is a two story turret outside of the south-west corner of the
first floor. The study is 15 x 20 and is in the south-east corner of the
first floor. The living\_room is 15 x 20 and is in the south-west corner of
the first floor. The dining\_room is 20 x 10 and is north of the living\_room.
The bedroom is 20 x 10 and is north of the study. The kitchen is 20x20 and
is north of the bedroom. The pantry is 5 x 15 and is in the north-east corner
of the first floor. The entrance\_hall is 10 x 20 and is right of the
living\_room. The upper\_sw\_bedroom is 15 x 15 and is in the southwest corner
of the second floor. The upper\_se\_bedroom is 15 x 15 and is 10 feet to the
east of the upper\_sw\_bedroom. The upper\_nw\_bedroom is 15x15 and is 3 feet
to the north of the upper\_sw\_bedroom. The upper\_ne\_bedroom is 15 x 15 and
is 3 feet to the north of the of the upper\_se\_bedroom. The bathroom is 5 x 5 and
is to the left of the upper\_ne\_bedroom. The master\_bedroom is 10 x 8 and is
in the north-east corner of of the second floor.}

The constructions for this building are shown in Figs. 6.34-6.36.

The first big issue is the L-shaped section of the building which is not captured in the rendering. This immediately causes the quantitative score to drop
Figure 6.34: Coghlan Castle front view
Figure 6.35: First floor of Coghlan Castle
Figure 6.36: Second Floor of Coghlan Castle
Figure 6.37: Coghlan Castle first floor plan [5].
Figure 6.38: Coghlan Castle second floor plan[5].
Figure 6.39: Coghlan Castle first floor rendering (green) vs. Ground Truth (yellow)
Figure 6.40: Coghlan Castle second floor rendering (green) vs. Ground Truth (yellow)
to 75% if it was considered in the metric. In addition, the rooms are not the correct size and they are slightly off, which will hurt the score and gives 79%. The qualitative descriptions are also reduced as four of the thirteen descriptions have a hedge in the ground truth that do not appear in the descriptions generated from the rendered model, dropping the qualitative score down to 85%. Still, for using an imprecise data set, this is encouraging as to the robustness of using natural language to describe the placement of rooms. The tree for Coghlan Castle is shown in Fig. 6.41.

6.6 Discussion

In this chapter, I’ve shown that the architectural description language and spatial rendering system I have developed is capable of constructing a set of buildings. For buildings such as office buildings, and classroom buildings which have a rectangular footprint that is subdivided into smaller rooms, it works quite well. For buildings that are non-rectangular or have curving structures, it does not do
<table>
<thead>
<tr>
<th>Node</th>
<th>Contents</th>
</tr>
</thead>
<tbody>
<tr>
<td>Building</td>
<td>name=&quot;Coghlan_Castle&quot; x=&quot;0&quot; y=&quot;0&quot; z=&quot;0&quot; width=&quot;40&quot; depth=&quot;50&quot; height=&quot;24&quot; faces=&quot;south&quot;</td>
</tr>
<tr>
<td>FF1</td>
<td>name=&quot;FF1&quot; x=&quot;0&quot; y=&quot;0&quot; z=&quot;0&quot; width=&quot;40&quot; depth=&quot;50&quot; height=&quot;12&quot;</td>
</tr>
<tr>
<td>FF2</td>
<td>name=&quot;FF2&quot; x=&quot;0&quot; y=&quot;12&quot; z=&quot;0&quot; width=&quot;40&quot; depth=&quot;50&quot; height=&quot;12&quot;</td>
</tr>
<tr>
<td>PP1</td>
<td>name=&quot;turret&quot; x=&quot;-3&quot; y=&quot;0&quot; z=&quot;47&quot; width=&quot;6&quot; depth=&quot;6&quot; height=&quot;30&quot;</td>
</tr>
<tr>
<td>RR1</td>
<td>place=&quot;study&quot; x=&quot;25&quot; y=&quot;0&quot; z=&quot;30&quot; width=&quot;15&quot; depth=&quot;20&quot; height=&quot;12&quot;</td>
</tr>
<tr>
<td>RR2</td>
<td>place=&quot;living_room&quot; x=&quot;0&quot; y=&quot;0&quot; z=&quot;30&quot; width=&quot;15&quot; depth=&quot;20&quot; height=&quot;12&quot;</td>
</tr>
<tr>
<td>RR3</td>
<td>place=&quot;dining_room&quot; x=&quot;0&quot; y=&quot;0&quot; z=&quot;19&quot; width=&quot;20&quot; depth=&quot;20&quot; height=&quot;12&quot;</td>
</tr>
<tr>
<td>RR4</td>
<td>place=&quot;bedroom&quot; x=&quot;22.5&quot; y=&quot;0&quot; z=&quot;19&quot; width=&quot;20&quot; depth=&quot;10&quot; height=&quot;12&quot;</td>
</tr>
<tr>
<td>RR5</td>
<td>place=&quot;kitchen&quot; x=&quot;22&quot; y=&quot;0&quot; z=&quot;0&quot; width=&quot;20&quot; depth=&quot;20&quot; height=&quot;12&quot;</td>
</tr>
<tr>
<td>RR6</td>
<td>place=&quot;pantry&quot; x=&quot;35&quot; y=&quot;0&quot; z=&quot;0&quot; width=&quot;5&quot; depth=&quot;15&quot; height=&quot;12&quot;</td>
</tr>
<tr>
<td>RR7</td>
<td>place=&quot;entrance_hall&quot; x=&quot;16&quot; y=&quot;0&quot; z=&quot;30&quot; width=&quot;10&quot; depth=&quot;20&quot; height=&quot;12&quot;</td>
</tr>
<tr>
<td>RR8</td>
<td>place=&quot;master_bedroom&quot; x=&quot;30&quot; y=&quot;12&quot; z=&quot;0&quot; width=&quot;10&quot; depth=&quot;8&quot; height=&quot;12&quot;</td>
</tr>
<tr>
<td>RR9</td>
<td>place=&quot;upper_sw_bedroom&quot; x=&quot;0&quot; y=&quot;12&quot; z=&quot;35.5&quot; width=&quot;15&quot; depth=&quot;15&quot; height=&quot;12&quot;</td>
</tr>
<tr>
<td>RR10</td>
<td>place=&quot;upper_se_bedroom&quot; x=&quot;25&quot; y=&quot;12&quot; z=&quot;35&quot; width=&quot;15&quot; depth=&quot;15&quot; height=&quot;12&quot;</td>
</tr>
<tr>
<td>RR11</td>
<td>place=&quot;upper_nw_bedroom&quot; x=&quot;0&quot; y=&quot;12&quot; z=&quot;20&quot; width=&quot;15&quot; depth=&quot;15&quot; height=&quot;12&quot;</td>
</tr>
<tr>
<td>RR12</td>
<td>place=&quot;upper_ne_bedroom&quot; x=&quot;25&quot; y=&quot;12&quot; z=&quot;20&quot; width=&quot;15&quot; depth=&quot;15&quot; height=&quot;12&quot;</td>
</tr>
<tr>
<td>RR13</td>
<td>place=&quot;bathroom&quot; x=&quot;20&quot; y=&quot;12&quot; z=&quot;20&quot; width=&quot;5&quot; depth=&quot;5&quot; height=&quot;12&quot;</td>
</tr>
<tr>
<td>Building</td>
<td>Feature</td>
</tr>
<tr>
<td>-----------------------</td>
<td>---------------------------------</td>
</tr>
<tr>
<td>Synthetic Building</td>
<td>Demonstration of functionality</td>
</tr>
<tr>
<td>Barre’s Old Labor Hall</td>
<td>Rich architectural description</td>
</tr>
<tr>
<td>Egyptian Theater</td>
<td>Balcony and Mezzanine</td>
</tr>
<tr>
<td>Oregon Central School</td>
<td>Slightly non-rectangular</td>
</tr>
<tr>
<td>Coghlan Castle</td>
<td>Multiple small rooms and a turret</td>
</tr>
</tbody>
</table>
as well; however, it still does render a qualitative score that would allow a person to navigate through the building. Using imprecise language to describe a building means that some uncertainty will creep in; however it is still a useful model in a qualitative sense. The results for these experiments are shown in Table 2.

In order to address these shortcomings, there are a few strategies that could be investigated. The first is to add support for curved structures by either using rhomboid voxels or increasing the fidelity of the rendering through the use of much smaller voxels. Another method would be to add common “odd” shapes, such as L or T shaped, as recognized descriptions in order to increase the match between the 3D construction and ground truth.

Even with these shortcomings, this is a good prototypical system for the construction and understanding of SRL in three dimensions. This shows a proof of concept which could be extended to support other shapes and more modern styles (e.g. the Opera House in Sydney, Australia).

Finally, this work provides a starting point for future human subject experiments. As models can be generated and spatial references extracted, it would be interesting to consider how this could be used in the future to compare with human performance. The next chapter considers this in more detail.
3D Descriptions from Inside the Constructed Model

As shown in Chapter 6, it is possible to use the proposed architectural description language to construct 3D models and to verify the models against ground truth. Another use for these 3D models involves the concept of a walkthrough of the environment using a virtual agent. This capability is useful as it allows an investigation into what a person would experience if placed in the same environment. This is also a critical step if these 3D SRL algorithms are to be validated through the use of a human subject experiment.

In this chapter, I demonstrate the preliminary walkthrough capability of the system with an example description generated at a point between the concession stand and the box office in the Egyptian Theater model. In this case, it is the teal cube shape with its perspective denoted by the arrow as shown in Fig. 7.1. This virtual agent is composed of voxels and has descriptions generated in the same manner as described in previous chapters. Consider Fig. 7.1 which shows a virtual agent in the Egyptian Theater.

The descriptions generated are:

The concession stand is behind me;

The restroom is above and extends to the left of me;

The Projection Room is above and extends to the left of me;
Figure 7.1: Example of a virtual agent’s position in a walkthrough on the first floor of the Egyptian Theater
The box office is to the front-left of me.

Descriptions generated in this manner could be compared to the descriptions given by a human to determine how well the 3D SRL generation matches human expectations. By performing this future experiment, more evidence will be collected to reinforce that the algorithms developed in this work match human expectations.
Chapter 8

Discussion

The core of this Dissertation revolves around the theme of investigating how 3D spatial referencing language can be used to make interactions with machine systems more intuitive and natural. The first investigation shows how algorithms originally designed for scene description and robot dialog in 2D could be augmented to work with 3D models. By adding the capability to describe the spatial location of objects using stereoscopic vision, the domain of the HoF-based algorithms has been enhanced and extended to the generation of 3D SRL and the construction of 3D models using architectural description language.

8.1 Strengths

The greatest strength of the approach is its flexibility in regards to the kinds of spatial information it can process. The HoF tools were first used for simple 2D images and maps and they are now capable of modeling 3D spatial relationships based on stereoscopic vision, 2D images, robot maps, or voxel models. It has been said that there is a need for a 3D version of the HoF; however, I would argue that it is sufficient to use the existing algorithms as the 2D HoF tools mirror 2D spatial referencing phrases used in the English language. By extending these planar tools to handle 3D tasks in a similar planar fashion, I believe that results consistent with human language are provided.
The architectural description language, while simple, is also a strength. It allows for novice users to quickly become familiar with the kinds of language that can be used to describe an environment. By utilizing a small subset, it is easier to isolate these kinds of passages in complex descriptions with extraneous information.

While the 3D constructions might be crude by today’s standards, they capture the essence of what defines a room inside of a structure. While these are technically rooms, there is nothing restricting the creation of a large empty floor and treating the rooms as objects to investigate other problems. As the room voxels carry with them the name of the room they are associated with, they could be manipulated easily as a group or individually. They currently represent 1-foot cubes, but they could be shrunk to represent centimeters or millimeters should one want greater fidelity at the cost of more overhead in dealing with the large number of voxels required to create a full environment.

The system also has the initial capability to generate a walkthrough of environments described by architectural description language as described by a virtual agent on the inside of the model. Such descriptions may provide a basis of communicating with a robot or another person from a remote location. This capability will also allow for future research involving human subject experiments on the validity of the proposed SRL tools developed.

8.2 Limitations

The biggest challenge in using natural language to model space is that often times it is imprecise and ambiguous. If one is going to accurately describe a building, measurements must be taken and a full survey should be performed. However, this is rarely the case when humans discuss architecture. After reviewing hundreds of records from the National Park Service and the General Services Administra-
tion, it is the exception rather than the rule that there will be good quantitative
descriptions of a building. Therefore, one must use a priori knowledge that, for
humans, is common but rare in machines. Knowing that a mezzanine is an inter-
mediary balcony naturally places restrictions on its size and location. For a system
to be flexible and robust, this type of domain knowledge must be incorporated.

There is also the problem that the current system can currently only handle
rectangular shaped buildings. Odd shapes and curved structures do not perform
well; however these could be modeled in future extensions. The modeling of
these structures will necessitate higher fidelity in the voxels in order to accurately
portray them in a virtual environment.

It would also be beneficial to enable a full modeling with constraints on the
virtual environment. This would enable a more fluid placement of rooms to fit
with known information. The HoF could guide how to place an object to the right
of another, but only if constraints on its placement and an ability to not collide
with other already placed object is included. For example, the modeling approach
could be used to resolve ambiguities in a particular phrase as additional language
descriptions are processed. This would allow for the use of ternary placements,
e.g. “The conference room is to the right of the elevator and below the atrium.”

Another limitation is that the merging of the various planes used for the
descriptions is not automatic. Currently, only the horizontal plane at the origin
and a plane parallel to the viewing plane are used in the convex hull projections.
This is not ideal as it is tied to the location of the viewing plane and does not
allow an easy way to switch perspective. Another drawback is that non-convex
objects are not considered with this method by calculating the convex hull.

In terms of software development and moving this work into a production
system, recent advancements with the boost C++ library have made it extremely
appealing for this work. The programming for this project takes place using a
perl script (for its superior text handling capabilities), tinyXML (for dealing with the XML files), and a C++ driver program to perform the OpenGL renderings and integrate the SRLib subroutines. It would be a great benefit to rewrite these programs using boost as it should lead to a more unified code base and one which is much easier to understand. Finally, the convex hulls are destroyed as soon as they are used as they are stored in a temporary file that is overwritten. It would be beneficial from an efficiency standpoint to save the convex hulls in RAM.
Chapter 9

Conclusions

9.1 Summary

In this work, I have shown how it is possible to use a 2D spatial modeling tool and expand it to generate 3D linguistic descriptions using images from simple webcams. The webcams are used to create a very inexpensive stereo vision system which can recognize objects in the world using SIFT keypoint models. Being able to reason about where objects are relative to each other is a very interesting capability. As webcams increase their resolution, this system will be able to expand to more and more devices making use of stereo vision.

I have also shown how it is possible to use architectural descriptions to build up models in a 3D virtual world that can be used to facilitate better understanding between humans and robotic systems. These virtual environments correspond well to ground truth. Even when they differ somewhat, by describing them using qualitative language, they still retain much of the relationships between rooms in the building. I have also shown that the comparison of the ground truth versus the rendered system shows that, in a qualitative sense, this approach is able to convey spatial information that is tolerant to mistakes in placement. This system of 3D spatial referencing language can provide a starting point for future investigations.

Finally, I have demonstrated that the basis for algorithms developed in this work and by others utilizing the HoF have a sound basis in terms of human spatial
modeling. The comparison of the HoF based algorithms for SRL with Regier and Carlson’s AVS model[1] shows that this spatial referencing tool compares favorably to human subject data. In summary, the list below shows the contributions that I have made:

- Compared the HoF spatial modeling tool to the AVS model;
- Extended the 2D HoF modeling tool to support 3D linguistic descriptions using stereoscopic vision data and 3D models;
- Developed new algorithms for NEAR, ON TOP, INSIDE, and UNDERNEATH;
- Proposed an architectural description language;
- Developed an approach to extract spatial knowledge from architectural descriptions;
- Developed a method to render the 3D architectural descriptions;
- Proposed both quantitative and qualitative metrics to measure the performance of the 3D construction;
- Demonstrated initial steps for supporting a walkthrough of a 3D environment which generates 3D spatial descriptions.

9.2 Future Work

There are interesting options for continuing this work. In particular, the generation of 3D virtual environments through natural language has many potentially useful purposes for both human robot interaction and human to remote human interaction. From allowing naïve users to quickly grasp a robot’s plan of
action to allowing better interaction between warfighter and unmanned systems, this idea of using a visual model to establish greater trust in the system is intriguing. The first step beyond this is to have the ability to model arbitrarily shaped buildings. This will solve the problems shown in Coghlan Castle and the Oregon Central School. A second extension would be to support online processing which would display information as it comes in, as opposed to having to process the entire passage first. Finally, it would be useful to add the capability of using linguistic hedges to modify the models after they have been built, for example, being able to shrink or grow a room and then move it a few feet so that both the human and machine system have the same understanding of reality.

One of the core areas that must be addressed whenever performing HRI research is the human aspect. In order to verify that the 3D SRL developed conforms to human expectations, a human subject experiment will need to be performed. Using this to construct a virtual environment from an architectural language description, a human subject experiment could be conducted in which participants go through a building and describe where objects are around them are located at predetermined points and compare those results to the 3D SRL generated.

Finally, the ability to perform walkthroughs of a 3D environment is quite exciting as it allows the use of any data that can be quantified as 3D points. This capability has applications for both human-human and human-machine interactions. This would be of particular use in situations where there is limited bandwidth for communication as being able to exploit a priori knowledge about the placements of objects, one could build a functional model of the environment quickly. This model could also be used to augment automated tour guide information by dynamically generating the spatial references to objects in the environment on demand. One could potentially use it to describe the placement of a tumor in
relationship to the hippocampus of the brain in an MRI scan. Educational software could use it to describe where various frog organs are located with regards to each other when performing virtual dissections. It can allow for an autonomous agent to explain what it was doing as it performed a mission and where it took action. Since we live and interact all the time in 3-space, the possibilities for applications of this system is only limited by one’s imagination.
Bibliography


Appendix A

Appendix A: SRLib

A.1 Background

SRLib is a wrapper/library to the HoF code developed for this and other works and can be found on kronos.cirl.missouri.edu in the /home/shared directory as SRLib.tar.bz2. It is written in C++ and makes use of the STL (albeit poorly). In current discussions with Ozy Sjahputera (as of 12/10/2010), some major updates using Dr. Matsakis’ new HoF code will need to be made in the future. Should one use this code, be advised it would probably be better to use whatever Ozy and I come up with in the near future.

The code builds into a shared library under a GNU/Linux system. You will need to place the library in your LD_LIBRARY_PATH in order to use it. Once that’s there, you should be able to access the API through the use of the srlib.h header file.

A.2 Compilation

After untarring the compressed file, take a look at the “shared_lib_build.sh” script to make sure that no tweaks need to be made for the current system you are building under. It has been tested under the latest Gentoo and Ubuntu distributions; however, your mileage may vary. Once the shared library has been built, add it to your /usr/lib folder and update the linker appropriately. Generally, this
is done by invoking ldconfig as root, so one may need a system administrator’s help on your particular system.

A.3 Use

Once the library is installed, link to it by adding “-lsrlib” to the link line for your C++ programs under linux and including the “srlib.h” header file. In theory, that should be all that is necessary to make use of the API. An example of the wrapper’s intended use is shown in the SpatialArch main.cpp file and in the driver.cpp file included here.

The wrapper contains quite a few functions for access; however the process that most will find useful is the following:

(1) Create an “Object” using the prototype provided;

(2) Run SRLingDesc or SRLingDescH (for looking along a particular heading) with the appropriate dictionary file to generate the linguistic descriptions;

(3) Run SRFindABLR with two objects to find the front, back, left, and right points (returned in a vector) of the referent object.

There are 6 options for dictionary files, one of which needs to be passed every time a call to SRLingDesc is made. These files support egocentric with self, horizontal, vertical, and exocentric, horizontal, and vertical. Play around with them and see what happens.

A.4 Improvements

Several potential improvements were listed in the main text of my Dissertation. Currently, the biggest one is replacing the histogram code with the new and
improved code from Dr. Matsakis. Check with Ozy or myself (sam.blisard@nrl.navy.mil) for the status of improvements to this code base.
Appendix B

Appendix B: SpatialArch Manual

B.1 Preliminaries

There are a couple of pieces of software which will need to be installed, and in particular places in order to run this software. First, install the stanford-parser.tar file to “/usr/local/share”. Second, download Tree::Nary from http://www.cpan.org, install, and modify it according to the Tree::Nary file included (run diff between them to figure out where to make the changes in the “find” tree function-by-default it looks for an exact match but it needs to do substring searches). Finally, install SpatialArch in your /NetBeansProjects folder. This software tarball can be found on kronos in /home/shared as SpatialArch.tar.bz2

B.2 Compilation

It should build out of the box as a NetBeans project. Note that in the tree.pl and main.cpp, there exist a “prefix” which needs to be set to your current home directory; otherwise things won’t be found.

B.3 Program Flow

The first step is to set up the foo.txt file in /usr/local/share/stanford-parser/stanford-parser-date/ with the architectural description language you want to have generated. Several examples are included in this directory, which can be
copied into foo.txt for comparison’s sake. Once SpatialArch is running, after setting up the environment it executes the /NetBeansProjects/SpatialArch/tree2.pl script which handles invoking the parser and extracting knowledge/building the arch.xml file which is used as input into SpatialArch. After the xml tree is passed in, it is processed by the use of the tinyxml library referenced in spatialarch.h. SpatialArch also makes calls to SRLib with the generate_descriptions function call.

The generate_descriptions call does several things. The first is that it sets up the referent object and generates the convex hull of the object specified in the function’s argument. It does this through the use of a temporary file which is constantly being overwritten (but the information is stored in Object1). It then performs a similar task to all objects which are not the object in order to generate their convex hull, perform a projection, and generate the linguistic descriptions. At the very end, “Object” is created using SRLib and the final calls are made to SRLib to perform the actual description generation.

B.4 Controls

Several keys have been bound to manipulate the view of the program. EDSF move the camera forward, back, left, and right, respectively. The r key moves the camera up (+y) and the c key moves down (-y). The IJKL keys are responsible for rotations around the origin. By using these keys, one can see the building from any angle. Several debugging keycodes are also defined in the keyboard section, but they are fairly straight-forward.

B.5 Concluding thoughts

To the future grad student looking at this code, I hope that the helpful comments I’ve made in the code are helpful. This is research and NOT PRO-
DUCTION code, so it’s going to be buggy as all get out since it’s held together with spit and bailing wire.

The big things I’d do are:

(1) Convert the mess that is tree2.pl into C++ boost;

(2) Use boost’s geometry library to perform the convex hulls and projections in a truly programmatic way.

(3) Use Ozy’s new spatial library as it will be far easier than SRLib.
Appendix C

Appendix C: Ground Truth and Linguistic Descriptions of the Buildings

C.1 Ground Truth Tables
These are the tables which contain the node information for ground truth versus the rendered model. The ground truth is denoted by the prefix GTx, where x is a number. pd stands for Primary Direction and ranges from 0-15 (16 equally spaced headings on a compass, 0 is north).
C.2 Qualitative Linguistic Descriptions

C.2.1 Barre’s Old Labor Hall

->-
-

RR1 stair_hall <- pd = 0

->-
-

GT1 stair_hall <- pd = 7

(no desc. due to bug)

->The RR2 east_room is loosely behind-right of the RR3 west_room

(the description is rather satisfactory)
<- pd = 10

->The GT2 east_room is loosely behind-right of the GT3 west_room

(the description is rather satisfactory)
<- pd = 10
Table C.1: Barre Old Labor Hall Ground Truth

<table>
<thead>
<tr>
<th>Node</th>
<th>Contents</th>
</tr>
</thead>
<tbody>
<tr>
<td>Building</td>
<td>name=&quot;Barre_Old_Labor_Hall&quot; x=&quot;0&quot; y=&quot;0&quot; z=&quot;0&quot; width=&quot;50&quot; depth=&quot;108&quot; height=&quot;26&quot; faces=&quot;south&quot;</td>
</tr>
<tr>
<td>FF1</td>
<td>name=&quot;FF1&quot; x=&quot;0&quot; y=&quot;0&quot; z=&quot;0&quot; width=&quot;50&quot; depth=&quot;108&quot; height=&quot;10&quot;</td>
</tr>
<tr>
<td>FF2</td>
<td>name=&quot;FF2&quot; x=&quot;0&quot; y=&quot;13&quot; z=&quot;0&quot; width=&quot;50&quot; depth=&quot;108&quot; height=&quot;10&quot;</td>
</tr>
<tr>
<td>RR1</td>
<td>place=&quot;stair_hall&quot; x=&quot;18&quot; y=&quot;0&quot; z=&quot;90&quot; width=&quot;10&quot; depth=&quot;18&quot; height=&quot;13&quot;</td>
</tr>
<tr>
<td>GT1</td>
<td>place=&quot;GT1_stair_hall&quot; x=&quot;20&quot; y=&quot;0&quot; z=&quot;92&quot; width=&quot;10&quot; depth=&quot;19&quot; height=&quot;13&quot;</td>
</tr>
<tr>
<td>RR2</td>
<td>place=&quot;east_room&quot; x=&quot;28&quot; y=&quot;0&quot; z=&quot;83&quot; width=&quot;19&quot; depth=&quot;25&quot; height=&quot;13&quot;</td>
</tr>
<tr>
<td>GT2</td>
<td>place=&quot;GT2_east_room&quot; x=&quot;30&quot; y=&quot;0&quot; z=&quot;86&quot; width=&quot;18&quot; depth=&quot;24&quot; height=&quot;13&quot;</td>
</tr>
<tr>
<td>RR3</td>
<td>place=&quot;west_room&quot; x=&quot;0&quot; y=&quot;0&quot; z=&quot;83&quot; width=&quot;17&quot; depth=&quot;25&quot; height=&quot;13&quot;</td>
</tr>
<tr>
<td>GT3</td>
<td>place=&quot;GT3_west_room&quot; x=&quot;0&quot; y=&quot;0&quot; z=&quot;86&quot; width=&quot;21&quot; depth=&quot;25&quot; height=&quot;13&quot;</td>
</tr>
<tr>
<td>RR4</td>
<td>place=&quot;north_corner&quot; x=&quot;0&quot; y=&quot;0&quot; z=&quot;69&quot; width=&quot;9&quot; depth=&quot;14&quot; height=&quot;13&quot;</td>
</tr>
<tr>
<td>GT4</td>
<td>place=&quot;GT4_north_corner&quot; x=&quot;1&quot; y=&quot;0&quot; z=&quot;71&quot; width=&quot;9&quot; depth=&quot;14&quot; height=&quot;13&quot;</td>
</tr>
<tr>
<td>RR5</td>
<td>place=&quot;meeting_hall&quot; x=&quot;0&quot; y=&quot;0&quot; z=&quot;0&quot; width=&quot;48&quot; depth=&quot;83&quot; height=&quot;13&quot;</td>
</tr>
<tr>
<td>GT5</td>
<td>place=&quot;GT5_meeting_hall&quot; x=&quot;0&quot; y=&quot;0&quot; z=&quot;0&quot; width=&quot;48&quot; depth=&quot;83&quot; height=&quot;13&quot;</td>
</tr>
<tr>
<td>RR6</td>
<td>place=&quot;upper_east_room&quot; x=&quot;22&quot; y=&quot;13&quot; z=&quot;83&quot; width=&quot;28&quot; depth=&quot;25&quot; height=&quot;13&quot;</td>
</tr>
<tr>
<td>RR7</td>
<td>place=&quot;upper_west_room&quot; x=&quot;0&quot; y=&quot;13&quot; z=&quot;95&quot; width=&quot;18&quot; depth=&quot;13&quot; height=&quot;13&quot;</td>
</tr>
<tr>
<td>RR8</td>
<td>place=&quot;upper_northwest_room&quot; x=&quot;0&quot; y=&quot;13&quot; z=&quot;83&quot; width=&quot;15&quot; depth=&quot;12&quot; height=&quot;13&quot;</td>
</tr>
</tbody>
</table>
### Table C.2: Egyptian Theater Ground Truth

<table>
<thead>
<tr>
<th>Node</th>
<th>Contents</th>
</tr>
</thead>
<tbody>
<tr>
<td>Building</td>
<td>name=&quot;Egyptian_Theater&quot; x=&quot;0&quot; y=&quot;0&quot; z=&quot;0&quot; width=&quot;75&quot; depth=&quot;145&quot; height=&quot;45&quot; faces=&quot;south&quot;</td>
</tr>
<tr>
<td>FF1</td>
<td>name=&quot;FF1&quot; x=&quot;0&quot; y=&quot;0&quot; z=&quot;0&quot; width=&quot;75&quot; depth=&quot;145&quot; height=&quot;15&quot;</td>
</tr>
<tr>
<td>FF2</td>
<td>name=&quot;mezzanine&quot; x=&quot;0&quot; y=&quot;15&quot; z=&quot;120&quot; width=&quot;75&quot; depth=&quot;25&quot; height=&quot;10&quot;</td>
</tr>
<tr>
<td>FF3</td>
<td>name=&quot;balcony&quot; x=&quot;0&quot; y=&quot;30&quot; z=&quot;75&quot; width=&quot;75&quot; depth=&quot;70&quot; height=&quot;15&quot;</td>
</tr>
<tr>
<td>RR1</td>
<td>place=&quot;box_office&quot; x=&quot;31.5&quot; y=&quot;0&quot; z=&quot;135&quot; width=&quot;10&quot; depth=&quot;10&quot; height=&quot;15&quot;</td>
</tr>
<tr>
<td>GT1</td>
<td>place=&quot;GT1 box_office&quot; x=&quot;33&quot; y=&quot;0&quot; z=&quot;140&quot; width=&quot;8&quot; depth=&quot;6&quot; height=&quot;15&quot;</td>
</tr>
<tr>
<td>RR2</td>
<td>place=&quot;concession_stand&quot; x=&quot;23.5&quot; y=&quot;0&quot; z=&quot;99&quot; width=&quot;25&quot; depth=&quot;10&quot; height=&quot;15&quot;</td>
</tr>
<tr>
<td>GT2</td>
<td>place=&quot;GT2 concession_stand&quot; x=&quot;24&quot; y=&quot;0&quot; z=&quot;116&quot; width=&quot;26&quot; depth=&quot;9&quot; height=&quot;15&quot;</td>
</tr>
<tr>
<td>RR3</td>
<td>place=&quot;stage&quot; x=&quot;6.5&quot; y=&quot;0&quot; z=&quot;0&quot; width=&quot;60&quot; depth=&quot;40&quot; height=&quot;15&quot;</td>
</tr>
<tr>
<td>GT3</td>
<td>place=&quot;GT3 stage&quot; x=&quot;7&quot; y=&quot;0&quot; z=&quot;0&quot; width=&quot;61&quot; depth=&quot;37&quot; height=&quot;15&quot;</td>
</tr>
<tr>
<td>RR4</td>
<td>place=&quot;restrooms&quot; x=&quot;16.5&quot; y=&quot;15&quot; z=&quot;130&quot; width=&quot;40&quot; depth=&quot;15&quot; height=&quot;15&quot;</td>
</tr>
<tr>
<td>GT4</td>
<td>place=&quot;GT4 restrooms&quot; x=&quot;24&quot; y=&quot;15&quot; z=&quot;116&quot; width=&quot;35&quot; depth=&quot;20&quot; height=&quot;15&quot;</td>
</tr>
<tr>
<td>RR5</td>
<td>place=&quot;projection_room&quot; x=&quot;11.5&quot; y=&quot;30&quot; z=&quot;125&quot; width=&quot;50&quot; depth=&quot;20&quot; height=&quot;15&quot;</td>
</tr>
<tr>
<td>GT5</td>
<td>place=&quot;GT5 projection_room&quot; x=&quot;17&quot; y=&quot;30&quot; z=&quot;124&quot; width=&quot;40&quot; depth=&quot;17&quot; height=&quot;15&quot;</td>
</tr>
</tbody>
</table>
Table C.3: Oregon School Ground Truth

<table>
<thead>
<tr>
<th>Node</th>
<th>Contents</th>
</tr>
</thead>
<tbody>
<tr>
<td>Building</td>
<td>name=&quot;OR_Central_School&quot; x=&quot;0&quot; y=&quot;0&quot; z=&quot;0&quot; width=&quot;69&quot; depth=&quot;95&quot; height=&quot;26&quot; faces=&quot;south&quot;</td>
</tr>
<tr>
<td>FF1</td>
<td>name=&quot;FF1&quot; x=&quot;0&quot; y=&quot;0&quot; z=&quot;0&quot; width=&quot;69&quot; depth=&quot;95&quot; height=&quot;13&quot;</td>
</tr>
<tr>
<td>FF2</td>
<td>name=&quot;FF2&quot; x=&quot;0&quot; y=&quot;13&quot; z=&quot;0&quot; width=&quot;69&quot; depth=&quot;95&quot; height=&quot;13&quot;</td>
</tr>
<tr>
<td>RR1</td>
<td>place=&quot;sw_corner_classroom&quot; x=&quot;0&quot; y=&quot;0&quot; z=&quot;64&quot; width=&quot;25&quot; depth=&quot;31&quot; height=&quot;13&quot;</td>
</tr>
<tr>
<td>GT1</td>
<td>place=&quot;GT1 sw_corner_classroom&quot; x=&quot;1&quot; y=&quot;0&quot; z=&quot;68&quot; width=&quot;25&quot; depth=&quot;27&quot; height=&quot;13&quot;</td>
</tr>
<tr>
<td>RR2</td>
<td>place=&quot;se_corner_classroom&quot; x=&quot;44&quot; y=&quot;0&quot; z=&quot;64&quot; width=&quot;25&quot; depth=&quot;31&quot; height=&quot;13&quot;</td>
</tr>
<tr>
<td>GT2</td>
<td>place=&quot;GT2 se_corner_classroom&quot; x=&quot;38&quot; y=&quot;0&quot; z=&quot;68&quot; width=&quot;25&quot; depth=&quot;27&quot; height=&quot;13&quot;</td>
</tr>
<tr>
<td>RR3</td>
<td>place=&quot;nw_corner_classroom&quot; x=&quot;0&quot; y=&quot;0&quot; z=&quot;0&quot; width=&quot;25&quot; depth=&quot;31&quot; height=&quot;13&quot;</td>
</tr>
<tr>
<td>GT3</td>
<td>place=&quot;GT3 nw_corner_classroom&quot; x=&quot;1&quot; y=&quot;0&quot; z=&quot;7&quot; width=&quot;25&quot; depth=&quot;27&quot; height=&quot;13&quot;</td>
</tr>
<tr>
<td>RR4</td>
<td>place=&quot;ne_corner_classroom&quot; x=&quot;44&quot; y=&quot;0&quot; z=&quot;0&quot; width=&quot;25&quot; depth=&quot;31&quot; height=&quot;13&quot;</td>
</tr>
<tr>
<td>GT4</td>
<td>place=&quot;GT4 ne_corner_classroom&quot; x=&quot;38&quot; y=&quot;0&quot; z=&quot;0&quot; width=&quot;24&quot; depth=&quot;34&quot; height=&quot;13&quot;</td>
</tr>
<tr>
<td>RR5</td>
<td>place=&quot;upper_sw_classroom&quot; x=&quot;0&quot; y=&quot;0&quot; z=&quot;64&quot; width=&quot;25&quot; depth=&quot;31&quot; height=&quot;13&quot;</td>
</tr>
<tr>
<td>GT5</td>
<td>place=&quot;GT5 upper_sw_classroom&quot; x=&quot;0&quot; y=&quot;13&quot; z=&quot;67&quot; width=&quot;25&quot; depth=&quot;26&quot; height=&quot;13&quot;</td>
</tr>
<tr>
<td>RR6</td>
<td>place=&quot;upper_nw_classroom&quot; x=&quot;0&quot; y=&quot;0&quot; z=&quot;0&quot; width=&quot;25&quot; depth=&quot;31&quot; height=&quot;13&quot;</td>
</tr>
<tr>
<td>GT6</td>
<td>place=&quot;GT6 upper_nw_classroom&quot; x=&quot;0&quot; y=&quot;13&quot; z=&quot;6&quot; width=&quot;25&quot; depth=&quot;29&quot; height=&quot;13&quot;</td>
</tr>
<tr>
<td>RR7</td>
<td>place=&quot;library&quot; x=&quot;44&quot; y=&quot;13&quot; z=&quot;0&quot; width=&quot;25&quot; depth=&quot;95&quot; height=&quot;13&quot;</td>
</tr>
<tr>
<td>GT7</td>
<td>place=&quot;GT7 library&quot; x=&quot;38&quot; y=&quot;13&quot; z=&quot;0&quot; width=&quot;25&quot; depth=&quot;101&quot; height=&quot;13&quot;</td>
</tr>
<tr>
<td>Node</td>
<td>Contents</td>
</tr>
<tr>
<td>------</td>
<td>----------------------------------------------</td>
</tr>
<tr>
<td>Building</td>
<td>name=&quot;Coghlan_Castle&quot; x=&quot;0&quot; y=&quot;0&quot; z=&quot;0&quot; width=&quot;40&quot; depth=&quot;50&quot; height=&quot;24&quot; faces=&quot;south&quot;</td>
</tr>
<tr>
<td>FF1</td>
<td>name=&quot;FF1&quot; x=&quot;0&quot; y=&quot;0&quot; z=&quot;0&quot; width=&quot;40&quot; depth=&quot;50&quot; height=&quot;12&quot;</td>
</tr>
<tr>
<td>FF2</td>
<td>name=&quot;FF2&quot; x=&quot;0&quot; y=&quot;12&quot; z=&quot;0&quot; width=&quot;40&quot; depth=&quot;50&quot; height=&quot;12&quot;</td>
</tr>
<tr>
<td>PP1</td>
<td>name=&quot;turret&quot; x=&quot;-3&quot; y=&quot;0&quot; z=&quot;47&quot; width=&quot;6&quot; depth=&quot;6&quot; height=&quot;30&quot;</td>
</tr>
<tr>
<td>RR1</td>
<td>place=&quot;study&quot; x=&quot;25&quot; y=&quot;0&quot; z=&quot;30&quot; width=&quot;15&quot; depth=&quot;20&quot; height=&quot;12&quot;</td>
</tr>
<tr>
<td>GT1</td>
<td>place=&quot;GT1 study&quot; x=&quot;25&quot; y=&quot;0&quot; z=&quot;32&quot; width=&quot;12&quot; depth=&quot;17&quot; height=&quot;12&quot;</td>
</tr>
<tr>
<td>RR2</td>
<td>place=&quot;living_room&quot; x=&quot;0&quot; y=&quot;0&quot; z=&quot;30&quot; width=&quot;15&quot; depth=&quot;20&quot; height=&quot;12&quot;</td>
</tr>
<tr>
<td>GT2</td>
<td>place=&quot;GT2 living_room&quot; x=&quot;0&quot; y=&quot;0&quot; z=&quot;32&quot; width=&quot;15&quot; depth=&quot;20&quot; height=&quot;12&quot;</td>
</tr>
<tr>
<td>RR3</td>
<td>place=&quot;dining_room&quot; x=&quot;0&quot; y=&quot;0&quot; z=&quot;19&quot; width=&quot;20&quot; depth=&quot;10&quot; height=&quot;12&quot;</td>
</tr>
<tr>
<td>GT3</td>
<td>place=&quot;GT3 dining_room&quot; x=&quot;0&quot; y=&quot;0&quot; z=&quot;19&quot; width=&quot;21&quot; depth=&quot;13&quot; height=&quot;12&quot;</td>
</tr>
<tr>
<td>RR4</td>
<td>place=&quot;bedroom&quot; x=&quot;22.5&quot; y=&quot;0&quot; z=&quot;19&quot; width=&quot;20&quot; depth=&quot;10&quot; height=&quot;12&quot;</td>
</tr>
<tr>
<td>GT4</td>
<td>place=&quot;GT4 bedroom&quot; x=&quot;25&quot; y=&quot;0&quot; z=&quot;19&quot; width=&quot;13&quot; depth=&quot;12&quot; height=&quot;12&quot;</td>
</tr>
<tr>
<td>RR5</td>
<td>place=&quot;kitchen&quot; x=&quot;22&quot; y=&quot;0&quot; z=&quot;0&quot; width=&quot;20&quot; depth=&quot;20&quot; height=&quot;12&quot;</td>
</tr>
<tr>
<td>GT5</td>
<td>place=&quot;GT5 kitchen&quot; x=&quot;17&quot; y=&quot;0&quot; z=&quot;0&quot; width=&quot;21&quot; depth=&quot;15&quot; height=&quot;12&quot;</td>
</tr>
<tr>
<td>RR6</td>
<td>place=&quot;pantry&quot; x=&quot;35&quot; y=&quot;0&quot; z=&quot;0&quot; width=&quot;5&quot; depth=&quot;15&quot; height=&quot;12&quot;</td>
</tr>
<tr>
<td>GT6</td>
<td>place=&quot;GT6 pantry&quot; x=&quot;32&quot; y=&quot;0&quot; z=&quot;0&quot; width=&quot;5&quot; depth=&quot;13&quot; height=&quot;12&quot;</td>
</tr>
<tr>
<td>RR7</td>
<td>place=&quot;entrance_hall&quot; x=&quot;16&quot; y=&quot;0&quot; z=&quot;30&quot; width=&quot;10&quot; depth=&quot;20&quot; height=&quot;12&quot;</td>
</tr>
<tr>
<td>GT7</td>
<td>place=&quot;GT7 entrance_hall&quot; x=&quot;16&quot; y=&quot;0&quot; z=&quot;32&quot; width=&quot;10&quot; depth=&quot;17&quot; height=&quot;12&quot;</td>
</tr>
</tbody>
</table>
Table C.5: Coghlan Castle Ground Truth Second Floor

<table>
<thead>
<tr>
<th>Node</th>
<th>Contents</th>
</tr>
</thead>
<tbody>
<tr>
<td>RR8</td>
<td>place=&quot;master_bedroom&quot; x=&quot;30&quot; y=&quot;12&quot; z=&quot;0&quot; width=&quot;10&quot; depth=&quot;8&quot; height=&quot;12&quot;</td>
</tr>
<tr>
<td>GT8</td>
<td>place=&quot;GT8 master_bedroom&quot; x=&quot;15&quot; y=&quot;12&quot; z=&quot;0&quot; width=&quot;14&quot; depth=&quot;11&quot; height=&quot;12&quot;</td>
</tr>
<tr>
<td>RR9</td>
<td>place=&quot;upper_sw_bedroom&quot; x=&quot;0&quot; y=&quot;12&quot; z=&quot;35.5&quot; width=&quot;15&quot; depth=&quot;15&quot; height=&quot;12&quot;</td>
</tr>
<tr>
<td>GT9</td>
<td>place=&quot;GT9 upper_sw_bedroom&quot; x=&quot;0&quot; y=&quot;12&quot; z=&quot;37&quot; width=&quot;15&quot; depth=&quot;15&quot; height=&quot;12&quot;</td>
</tr>
<tr>
<td>RR10</td>
<td>place=&quot;upper_se_bedroom&quot; x=&quot;25&quot; y=&quot;12&quot; z=&quot;35&quot; width=&quot;15&quot; depth=&quot;15&quot; height=&quot;12&quot;</td>
</tr>
<tr>
<td>GT10</td>
<td>place=&quot;GT10 upper_se_bedroom&quot; x=&quot;23&quot; y=&quot;12&quot; z=&quot;37&quot; width=&quot;15&quot; depth=&quot;15&quot; height=&quot;12&quot;</td>
</tr>
<tr>
<td>RR11</td>
<td>place=&quot;upper_nw_bedroom&quot; x=&quot;0&quot; y=&quot;12&quot; z=&quot;20&quot; width=&quot;15&quot; depth=&quot;15&quot; height=&quot;12&quot;</td>
</tr>
<tr>
<td>RR12</td>
<td>place=&quot;upper_ne_bedroom&quot; x=&quot;25&quot; y=&quot;12&quot; z=&quot;20&quot; width=&quot;15&quot; depth=&quot;15&quot; height=&quot;12&quot;</td>
</tr>
<tr>
<td>GT12</td>
<td>place=&quot;GT12 upper_ne_bedroom&quot; x=&quot;23&quot; y=&quot;12&quot; z=&quot;20&quot; width=&quot;15&quot; depth=&quot;15&quot; height=&quot;12&quot;</td>
</tr>
<tr>
<td>RR13</td>
<td>place=&quot;bathroom&quot; x=&quot;20&quot; y=&quot;12&quot; z=&quot;20&quot; width=&quot;5&quot; depth=&quot;5&quot; height=&quot;12&quot;</td>
</tr>
<tr>
<td>GT13</td>
<td>place=&quot;GT13 bathroom&quot; x=&quot;15&quot; y=&quot;12&quot; z=&quot;20&quot; width=&quot;8&quot; depth=&quot;9&quot; height=&quot;12&quot;</td>
</tr>
</tbody>
</table>
-> The RR4 north_corner is mostly to the left of the RR3 west_room but somewhat to the rear
(the description is satisfactory)
<- pd = 5

-> The GT4 north_corner is loosely behind-left of the GT3 west_room
(the description is satisfactory)
<- pd = 6
match(0.5)

-> The RR5 meeting_hall is behind the RR3 west_room but extends to the left.
(the description is satisfactory)
<- pd = 7

-> The GT5 meeting_hall is behind the GT3 west_room but extends to the left.
(the description is satisfactory)
<- pd = 7

match = 4.5/5 = 90%

C.2.2 Egyptian Theater

-> The RR1 box_office is in front of the RR2 concess
(the description is satisfactory)
<-

-> The GT1 box_office is in front of the GT2 concess

(the description is satisfactory)
<-

-> The RR3 stage is behind the RR2 concess

(the description is satisfactory)
<-

-> The GT3 stage is behind the GT2 concess

(the description is satisfactory)
<-

-> The RR4 restrooms/ is in front of the RR2 concess

(the description is satisfactory)
<-

-> The GT4 restrooms/ is in front of the GT2 concess
(the description is satisfactory)
<- pd = 1

-> The RR5 projection_room is in front of the RR2 concess

(the description is satisfactory)
<- pd = 1

-> The GT5 projection_room is in front of the GT2 concess

(the description is satisfactory)
<- pd = 1

match = 100%

C.2.3 Oregon Central School

-> The RR2 se_corner_classroom is almost to the right of the RR1 sw_corner_classroom but extends to the rear relative to
(the description is satisfactory)
<- pd = 11

-> The GT2 se_corner_classroom is loosely to the right of the GT1 sw_corner_classroom and extends to the rear relative to
(the description is rather satisfactory)
<- pd = 11
The RR3 nw_corner_classroom is behind the RR1 sw_corner_classroom but extends to the left relative to
(the description is satisfactory)
<- pd = 7

The GT3 nw_corner_classroom is behind the GT1 sw_corner_classroom but extends to the left relative to
(the description is satisfactory)
<- pd = 7

The RR4 ne_corner_classroom is mostly behind the RR1 sw_corner_classroom but extends to the right relative to
(the description is satisfactory)
<- pd = 9

The GT4 ne_corner_classroom is behind the GT1 sw_corner_classroom but extends to the right relative to
(the description is satisfactory)
<- pd = 9

RR5 upper_sw_classroom <- pd = 9
Arg ling desc ->-
-
The GT5 upper_sw_classroom is behind-left of the GT1 sw_corner_classroom

(the description is satisfactory)

<- pd = 6
Arg ling desc ->The GT5 upper_sw_classroom is behind-left of the GT1 sw_corner_classroom

(the description is satisfactory)

(note: this is due to a bug overlapping in the horizontal plane, match = 0)

The RR6 upper_nw_classroom is behind the RR1 sw_corner_classroom but extends to the left relative to

(the description is satisfactory)

<- pd = 7

The GT6 upper_nw_classroom is behind the GT1 sw_corner_classroom but extends to the left relative to

(the description is satisfactory)

<- pd = 7

The RR7 library is behind-right of the RR1 sw_corner_classroom

(the description is satisfactory)
The GT7 library is behind-right of the GT1 sw_corner_classroom

(the description is rather satisfactory)

match = 6/7 - 85.7% w/bug, 100% w/o.

C.2.4 Coghlan Castle

The RR1 study is mostly to the right of the RR2 living_room but extends to the rear.

(the description is satisfactory)

The GT1 study is loosely behind-right of the GT2 living_room

(the description is rather satisfactory)

The RR3 dining_room is behind the RR2 living_room but extends to the left.

(the description is satisfactory)

The GT3 dining_room is behind the GT2 living_room
but extends to the left.
(the description is satisfactory)
<- pd = 7
(match 1)

-> The RR4 bedroom is mostly behind the RR2 living_room
but extends to the right.
(the description is satisfactory)
<- pd = 9
(match 1)

-> The GT4 bedroom is mostly behind the GT2 living_room
but extends to the right.
(the description is satisfactory)
<- pd = 9
(match 1)

-> The RR5 kitchen is behind the RR2 living_room
but extends to the right.
(the description is satisfactory)
<- pd = 9
(match 1)

-> The GT5 kitchen is behind the GT2 living_room
but extends to the right.
(the description is satisfactory)
<- pd = 9
(match 1)
-> The RR6 pantry is mostly behind the RR2 living_room
but extends to the right.
(the description is satisfactory)
<- pd = 9

-> The GT6 pantry is mostly behind the GT2 living_room
but extends to the right.
(the description is satisfactory)
<- pd = 9
(match 1)

-> The RR7 entrance_hall is almost to the right of the RR2 living_room
but extends to the rear.
(the description is satisfactory)
<- pd = 11

-> The GT7 entrance_hall is almost to the right of the GT2 living_room
but extends to the rear.
(the description is satisfactory)
<- pd = 11
(match 1)

-> The RR8 master_bedroom is mostly behind the RR2 living_room
but extends to the right relative to
(the description is satisfactory)
<- pd = 9
The GT8 master_bedroom is behind the GT2 living_room

(the description is satisfactory)
<- pd = 8
(match 0.5)

The RR9 upper_sw_bedroom is to the right-front of the RR2 living_room

(the description is satisfactory)
<- pd = 14

The GT9 upper_sw_bedroom is to the right-front of the GT2 living_room

(the description is satisfactory)
<- pd = 14
(match 1)

The RR10 upper_se_bedroom is to the right of the RR2 living_room but extends forward relative to
(the description is satisfactory)
<- pd = 13

The GT10 upper_se_bedroom is to the right of the GT2 living_room but extends to the rear relative to
(the description is satisfactory)
<- pd = 11
(match 0.5)
-->The RR11 upper_nw_bedroom is behind the RR2 living_room but extends to the left.

(the description is satisfactory)
<- pd = 7

-->The GT11 upper_nw_bedroom is behind the GT2 living_room but extends to the left.

(the description is satisfactory)
<- pd = 7

(match 1)

-->The RR12 upper_ne_bedroom is loosely behind-right of the RR2 living_room

(the description is satisfactory)
<- pd = 10

-->The GT12 upper_ne_bedroom is behind the GT2 living_room but extends to the right.

(the description is satisfactory)
<- pd = 9

(match 0.5)

-->The RR13 bathroom is behind the RR2 living_room but extends to the right.

(the description is satisfactory)
<- pd = 9
The GT13 bathroom is behind the GT2 living_room but extends to the right.

(the description is satisfactory)

<- pd = 9

(match 1)

Score: 4 * 0.5 + 9*1 = 11/13 = 84.6% = 85%
VITA

Sam Blisard grew up in the hills of Southern Missouri in a town called Gainesville. While there he excelled academically at Gainesville High School, attended the 1993 session of the Missouri Scholars Academy, and graduated Valedictorian of his class in 1995. He then attended Missouri State University where he earned B.S. degrees in Computer Science and Mathematics. He began his graduate studies in 2001 at the University of Missouri where he earned an M.S. in Computer Science in 2004 and his Ph.D. in Electrical and Computer Engineering in 2010. He is also an NRA certified pistol instructor, scuba diver, all around outdoorsman, and a native born Hillbilly of the Ozark Mountains. He currently lives in Annandale, VA with his wife Amy and a blue heeler named Annie.