ROBOT METHODS FOR HUMAN-ROBOT SPATIAL LANGUAGE INTERACTION

A Thesis

Presented to

the Faculty of the Graduate School

at the University of Missouri-Columbia

In Partial Fulfillment

of the Requirements for the Degree

Master of Science

by

ZHIYU HUO

Dr. Marjorie Skubic, Thesis Supervisor

MAY 2013

The undersigned, appointed by the dean of the Graduate School, have examined the thesis entitled

ROBOT METHODS FOR HUMAN-ROBOT SPATIAL LANGUAGE INTERACTION

presented by Zhiyu Huo,

candidate for the degree of master of science

and hereby certify that, in their opinion, it is worthy of acceptance.

Dr. Marjorie Skubic

Dr. James Keller

Dr. Yunxin Zhao

ACKNOWLEDGEMENTS

First and foremost, I would like to thank my adviser, Dr. Marjorie Skubic, for her valuable guidance and advice and for her vast reserve of patience and knowledge. I would like to also express my sincere thankfulness to my colleague Tatiana Alexenko who helped me with my research. Finally, I would like to acknowledge my family and friends for their encouragement and devotion.

ABSTRACT

This thesis investigates perception and human-robot interaction methods for a robot designed to perform a fetch task. Natural spatial language is studied to direct a mobile robot to navigate in an indoor environment, detect objects, and use them as reference landmarks in finding a target object. The perception focus is on Kinect-based furniture recognition which allows the robot to use furniture items as landmarks in the spatial language description. A two-step process is proposed to recognize furniture objects. Furniture samples are first classified using geometric features by a linguistic model; the second step uses color and texture for further discrimination into specific furniture items by a probability graphical model (PGM); both extrinsic and intrinsic confidence values are computed. Orientation is also captured to support intrinsic reference frames of furniture such as chairs and couches. A robot behavior model is proposed to improve recognition by changing the viewing perspective when the recognition confidence is low. Eight furniture items are used in experiments to test algorithms for furniture recognition, orientation detection and robot behavior. Human-robot interaction is further investigated through the translation of a processed fetch description into robot commands that execute the fetch task and use the furniture recognition when specified in the description. The approach utilizes natural language processing methods designed to tag and chunk raw descriptions (developed elsewhere). The processed spatial description is then used for translation into robot commands. A simulation experiment is presented to evaluate the method. The results show good performance of the perception and human-robot interaction algorithms.

Table of Contents

LIST OF FIGURES

LIST OF TABLES

Chapter 1 Introduction

1.1 Motivation

The methods presented in this thesis are used to investigate human-robot interaction using spatial language. The task context is the robot fetch task for an elderly user in a home-like setting. Thus, to support this task, a robot needs to have the capability of perception and natural interaction with a human user in an unstructured environment.

For perceptual capabilities, the robot needs to detect and recognize furniture pieces in an indoor environment. It also needs the capability of detecting the position and the orientation of a furniture item so that it can use the furniture items it has detected as reference landmarks when interacting with the user via spatial language. In this thesis, a fast and robust object recognition algorithm is proposed, specially designed for a robot to recognize indoor furniture items.

To interact with an elderly user, the robot should understand the spatial language description of a target object, given by the user. Natural language processing (NLP) methods developed elsewhere are used to tag and chunk the raw spatial descriptions. In this thesis, an approach is proposed to translate the processed (tagged and chunked) description into robot commands for executing the fetch task.

1.2 Primary Goal

The principle objectives of perception in this research include the following:

1) Planning a general scheme for a robot to extract furniture samples from the background scene.
2) Selecting proper features and classifiers to recognize furniture items.

3) Estimating the position and orientation of furniture items.
4) Developing and testing a method to detect them.
5) Developing and testing methods for robot behavior which allows the robot to improve furniture recognition performance by changing its position.

The principle objectives of human-robot interaction in this research include the following:

1) Building a NLP model to spatial language in an inroom environment.
2) Building a robot behavior model for an inroom robot object fetching task which is a practical problem in elder care.

1.3 Thesis Organization

Chapter 2 is a review of related work on thesis topics. The perceptual part includes some early robotics navigation solutions based on vision. Then a general literature review covers the subject of object recognition, which includes achievements using images and 3D data. Recent findings are also presented using RGB-Depth camera images to recognize objects. In addition, classifiers typically used for these problems are discussed. The human-robot interaction part includes work by colleagues of the thesis author in language tagging and chunking, which are utilized in the proposed methods. It also includes a discussion of recent work on spatial language and robot logic language for robot command representation.

Chapter 3 introduces the hardware and software used in this research and shows how they were used to work together.

Chapter 4 presents three objectives in perception. One is the development of an algorithm for furniture model building and classification. The second objective is the

definition and detection of the pose, especially as it pertains to the orientation of a furniture item. The third objective focuses on developing the behavior needed to reposition the robot to improve its furniture detection capabilities.

Chapter 5 proposes an algorithm to convert (tagged and chunked) human spatial language commands to robot navigation instructions and then robot control factors for a robot fetch task. It builds a bridge between natural spatial language command and robot control.

Chapter 6 shows the process and raw results of three perception experiments corresponding to the three objectives in chapter 4 and a robot simulation experiment which tested spatial language interpretation in chapter 5.

Chapter 7 discusses what factors affected the results and explains unexpected results.

Chapter 8 is the conclusion and perspective.

Chapter 2 Literature Review

This chapter reviews object recognition and vision navigation. It includes information from articles that discuss both appearance and depth information. The first section focuses on some early work on object detection in the robotics field. The second section focuses on appearance of objects which are always RGB or gray images. Both global and local approaches will be introduced in this part. The third section includes information on using three-dimensional data for object recognition. This last section introduces recent work on using data from Kinect for object detection and recognition, and ends with a discussion of classifiers used in object recognition. Section 2.6 discusses some previous work of robot spatial language understanding.

2.1 Early Works on Vision-Based Navigation in Robotics

Intelligent robots are required to have the capability of object recognition so that they can react quickly to the surrounding environment with appropriate behavior [5].

The object recognition task is realized by an optical camera. It is agreed that a camera is a vital sensor for a robot when it interacts with the outer world. The main function of such a perception module is to process the raw image and extract useful information. For robotics, object recognition not only means to determine the class of an object sample but also to use it for navigation. Prior research introduces complicated and robust recognition approaches based on pattern recognition, which enabled the robot to recognize some pre-modeled objects for localization and manipulation [3].

Usually the object recognition approach for a robot system designed for a specific purpose is not open, which means that the objects that need to be learned for recognition

are limited, and there are no other kinds of objects in the environment. A robotics task which has such a property is robot soccer.

Robotics soccer is a competition that uses a robot hardware platform and artificial intelligence software to play soccer games [3]. It is a good platform to test the technology of artificial intelligence and robotics. Because of the limitation of technology at that time, the soccer playing field and rules are much simpler than the rules pertaining to human soccer. Moreover, the standard of illumination, field size and the color and texture of the markers is strictly prescribed. A typical robot soccer match is shown in the following figure. The early work for robot soccer navigation was based on color.

Figure 1 Robot Soccer

The most important stages and approaches of vision-based perception for robot soccer are introduced here. From Figure 1, it can be seen that the elements in a game, which included robots and a field, had obvious color markers designed for recognition. Therefore, analyzing color information from a raw image has become a popular design method utilized by many robotics competition teams. Without losing generality, a perception approach has two steps [3]:

1) Low-level vision, which takes as input, a raw image and outputs a set of region candidates. These candidates always list color blobs with information on each color, area and position on the raw image.
2) High-level vision, which uses the regions to determine the 3D relative position of available objects.

Despite their differences, many of the proposed techniques can be viewed conceptually in terms of these two stages of processing.

In the low-level stage, a preprocessing of the image is performed, where image pixels are analyzed in order to extract useful information. This is generally the most computationally expensive task, and at each stage, the amount of information to be processed further is reduced. The first important step is color segmentation, which uses a color table to map pixels from raw image values to a class of symbolic colors considerably reducing the amount of information per pixel from 256 to the limited number of colors. The early robot developers always used manual calibration which was a time consuming task and prone to errors. Since lighting is always different on each testing field (even at different times of the day in the same place), robot competitions always establish illumination requirements in their rules [4][7]. Such rules led to teams developing automated vision calibration routines.

On the other hand, the high-level vision module performs a top-down image analysis, using features provided by low-level vision. The main objective is to find objects of interest and estimate their properties. In this stage contextual information and expectations of objects that might be in the image can be used. Usually starting from a list of region candidates of the appropriate color, binary rules are applied to discard false

perception. Physical features such as size and shape are used to quickly filter wrong candidates. Subsequently deeper analyses are performed [6].

Note that this review and further analysis is only restricted to regular cameras (e.g., single cameras or stereo cameras). For omnidirectional cameras or other kinds of sensor-based technology, the algorithm will be different.

2.2 Object Recognition by Image Data

Object recognition based on images is also of great interest to researchers outside the robotics field. In recent years, several novel methods for object recognition based on image data have been developed bringing significant changes to this field. Object recognition utilizes both global and local approaches [1].

2.2.1 Global Approach

The global approach uses global features which use information of the whole image as the sample; this means that all the pixels are regarded [1]. The main idea is to convert an image to a vector feature. The method for generating features includes not only simple statistical measures (e.g., mean values or histograms of features) but also more sophisticated dimension reduction techniques, i.e., subspace methods, such as principle component analysis (PCA) [12], independent component analysis (ICA) [13], or non-negative matrix factorization (NMF) [14].

Principal component analysis (PCA) is a mathematical procedure that uses an orthogonal transformation to convert a set of possibly correlated variables into a set of linearly uncorrelated variables called principal components [12]. The number of principal components is less than or equal to the number of original variables. In the conversion results, the first principal component has the largest possible variance. PCA is widely

used in image classification because it can convert a sample image to a vector which has a much smaller size thereby reducing computational complexity and saving time. The object of interest may then be easier to recognize.

Independent component analysis (ICA) [13] is a computational method for separating a multivariate input into additive subcomponents supposing the mutual statistical independence of the non-Gaussian source inputs. It is a special case of blind source separation. ICA can also convert natural image data to smaller size features, which reduces the computation in object recognition [1].

Global approaches are often used in areas where the image or patch of interest can be easily obtained as in character recognition and face recognition. Character recognition includes optical character recognition (OCR) and intelligent character recognition (ICR). OCR and ICR convert the image that contains text information to an ASCII code that can be stored in the computer [8]. OCR appears early in the character recognition literature and can recognize only machine print. By using a pattern-matching algorithm, OCR translates the shapes and patterns of machine-made characters into corresponding computer codes. Though most advanced systems are able to recognize multiple fonts, they can process only standard fonts such as Times New Roman and Arial. Once all characters in a given word are recognized, the word is compared against a vocabulary of potential answers for the final result [9]. ICR was proposed later with the development of pattern recognition. Compared with OCR, ICR converts more scrawled handwriting characters to their machine print (ASCII) equivalents [9]. The ability to recognize handwriting significantly broadens the range of applications that benefit from automated

ICR solutions, saving time and increasing accuracy to levels not attainable by OCR or human intervention.

Face recognition is another example of a global approach for recognition [10]. Kohonen's face recognition system [11] demonstrates that a simple neural net can perform face recognition for aligned and normalized face images. The type of network he employed computed a face description by approximating the eigenvectors of the face image's autocorrelation matrix; these eigenvectors are now known as Eigenfaces. Kohonen's system was not a practical success, however, because of the need for precise alignment and normalization. In following years many researchers tried face recognition schemes based on edges, inter-feature distances, and other neural net approaches. While several were successful on small databases of aligned images, none successfully addressed the more realistic problem of larger databases where the location and scale of the face is unknown.

2.2.2 Local Approach

Local Approaches do not use information from the whole image but extract interesting regions from the samples and generate features from those [1]. Ideally, the features are invariant to image scaling, translation, and rotation, and at least partially invariant to illumination changes. Local approaches usually work on finding interesting regions in the image, and they are of great importance in detecting these kinds of regions and finding ways to represent them and use their information for recognition [1]. Figure 2 shows how two images were matched by using local features.

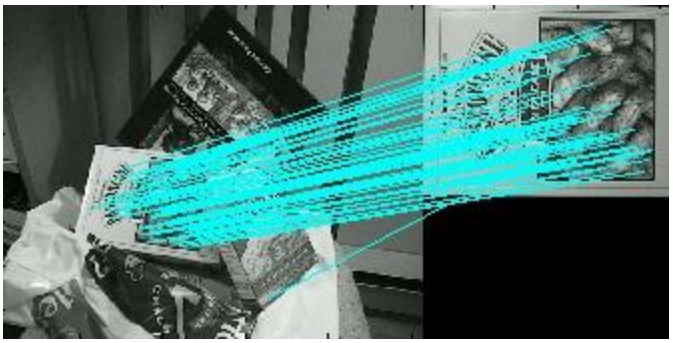


Figure 2 Local Approach Recognition Appliance –Recognizing the Target Book from an Image

An ideal interesting region detector provides additionally both shape (scale) and orientation of it. There are currently three kinds of popular detectors [1]: (1) corner based detectors; (2) region base detectors, (3) other approaches. Corner based detectors define an interest point or regions with a lot of image structure (e.g., edges), but they are not suited for uniform and smooth transitions regions [1]. Region based detectors regard local blobs of uniform brightness as salient regions. Therefore they are more suited for the letter [1]. Other approaches for example entropy based salient regions detection try to imitate the human's way of visual attention [1]. The following table lists the popular detectors with their properties.

Table 1 Common Properties of Detectors of the regions of interest [1]

Detector	Assigned Category	Invariance	Runtime	Repeat Ability	Number of Detections
Harris	Corner	None	Very Short	High	High
Hessian	Region	None	Very Short	High	High
Harris-Lap	Corner	Scale	Medium	High	Medium
Hessian-Lap	Region	Scale	Medium	High	Medium
DoG	Region	Scale	Short	High	Medium
Harris-Affine	Corner	Affine	Medium	High	Medium
Hessian-Affine	Region	Affine	Medium	High	Medium
MSER	Region	Projective	Short	High	Low
EBSR	Other	Scale	Very long	Low	Low
EBR	Corner	Affine	Very long	Medium	Medium
IBR	Region	Projective	Long	Medium	Low

In Roth and Winter's work [1], a feature descriptor has an invariance property in affine distortions, scale and rotation change, illumination change or compression artifacts. Its quality strongly depends on the power of the region detectors. For an instance, a very simple descriptor can be a pixel intensity vector in an interesting region which uses cross-correlation to compute similarity. A detector should detect descriptors accurately in location and shape, or it will lead to changing of the appearance of the descriptor. Therefore, robustness is also an important property of efficient region descriptors.

Table 2 Common Properties of Descriptors of the Region of Interest [1]

Descriptor	Assigned Category	Rotational Invariance	Dimensionality	Performance
SIFT	Distrib.	No	High	Good
PCA-SIFT	Distrib.	No	Low	Good
GLOH	Distrib.	No	High	Good
Spin images	Distrib.	Yes	Medium	Medium
Shape	Distrib.	No	Medium	Good
LBP	Distrib.	No	Very High	-
Differential Inv.	Filter	Yes	Low	Bad
Steerable Filters	Filter	Yes	Low	Medium
Complex Filters	Filter	Yes	Low	Bad
Cross Correlation	Other	No	Very High	Medium
Color Moments	Other	Yes	Low	-
Intensity Moments	Other	Yes	Low	-
Gradient Moments	Other	Yes	Low	Medium

The Local Approach uses the bag of words method [15] on matching for recognition and classification. The bag-of-words model was originally used in natural language processing (NLP) where the features were words. However, this model has been expanded to other topics like image processing. In image processing, the regions of interest are considered as "words" in a bag-of-word model.

2.3 Object Recognition by Three-Dimensional Point Data

Data that can be used for object recognition is not restricted to images. The stereo camera can collect depth information from an object sample—a practice which has been widely used in robotics. There are also other kinds of sensors, such as the laser scanner, which can get an accurate point cloud of an object sample. There are several existing techniques for feature extraction.

These methods represent and classify objects by different approaches. Depending on the type of model, there are two approaches to 3D point cloud recognition.

One of the models uses a mesh-based feature extraction [16], which uses the mesh data to represent the sample's geometry information. Diverse algorithms have been developed to build such a model. Mesh data are generated from the depth information of the sample. Then geometry features like corners or edges are extracted from the mesh data, after which recognition can be run based on those features.

The other kind of feature is a point-based feature. Similar to the mesh approach, geometry feature extraction can be done using point-based models. Point feature histograms (PFH) and fast point feature histograms (FPFH) [17] are another approach which uses the region of interest rather than geometry features. As point feature representations go, surface normal and curvature estimates are somewhat basic in their representations of the geometry around a specific point. Though extremely fast and easy to compute, they cannot capture too much detail, as they approximate the geometry of a point's k-neighborhood with only a few values. As a direct consequence, most scenes will contain many points with the same or very similar feature values, thus reducing their informative characteristics. PFH and FPFH are suitable to detect free-form shape objects and have high quality robustness.

2.4 Object Recognition by Using RGB-Depth Image

The research on object recognition using RGB-Depth image has generated a lot of interest in recent years, especially after Microsoft Kinect developed it for a daily life entertainment tool. Kinect is firstly used as a body sensor which can detect the human form and recognize a gesture so that people can use it to give commands to their entertainment devices. Microsoft and other research group have developed a series of

algorithms on human detection and action classification [32]. It has been widely used in Xbox games and other human detection fields.

Even Kinect, first designed for human detection, is also used as a recognition tool in other fields. A team from the University of Washington investigated a method for object recognition by using a Kinect sensor [18]. Their research focus was on how to select good features from depth frames. Motivated by local descriptors on images, in particular kernel descriptors, they developed a set of kernel features on depth images that model size, 3D shape, and depth edges in a single framework. The features used in their experiments include Size Kernel DES, Kernel PCA, Spin KDES, Gradient KDES and LBP-KDES. They used pyramid efficient match kernels as the classifier on their experiment. The experiments also tested the performance of different features. The classifiers used in the experiment are Linear SVM, kernel SVM, RF and their own classifier as reported in their results [18]. In their experiment they used a discriminate model for all the tests. No model for the objects was built. The dataset they used in the experiment are small-size daily use objects. In their experiment, there is no large scale object used; thus, it is not clear how this approach would perform on detecting large scale objects such as furniture.

2.5 Some Classifiers used in Object Recognition

When the feature set is determined, the next step for object recognition is classification. There are various classifiers that can be used in this step. Each of the classifiers has different property.

Naïve Bayes Classifier

The Bayes Network (BN) Classifier is based on a probability graphical model (PGM). It can derive a classification decision based on the given observation and BN structure. A naive Bayes classifier assumes that there is no relation between the features in a sample which means they are independent distributions [19]. The class that labels a sample is based on the probability it belongs to the class. Depending on the precise nature of the probability model, a naive Bayes classifier can be trained very efficiently by a supervised learning process. Even though it is not a new theory, Naïve Bayes classifier is still widely used in object recognition.

Decision Tree

A decision tree is used as a visualization and analytical decision making tool, where the expected values of competing alternatives are calculated. Decision tree learning, which uses a decision tree as a predictive model can map observations about an item to conclusions about the item's target value [20]. The advantage of using a decision tree is: (1) it is simple to understand and interpret; (2) it is a white box model which is easy to explain the result. The limitations are: (1) decision tree learners can create over-complex trees that do not generalize the data well. (2) There are concepts that are hard to learn because decision trees do not express them easily, such as XOR, parity or multiplexer problems.

Support Vector Machine

Support vector machines (SVMs, also known as support vector networks) are supervised learning models with associated learning algorithms that analyze data and recognize patterns used for classification and regression analysis [21]. A basic SVM

process takes a set of input data and predicts for each given input which of two possible classes forms the output making it a non-probabilistic binary linear classifier. Given a set of training examples, each marked as belonging to one of two categories, an SVM training algorithm builds a model that assigns new examples into one category or the other. An SVM model is a representation of the examples as points in space, mapped so that the examples of the separate categories are divided by a clear gap that is as wide as possible. New examples are then mapped into that same space and predicted to belong to a category based on which side of the gap they fall on.

K-Nearest Neighbor

The k-nearest neighbor algorithm (k-NN) is a method for classifying objects based on closest training examples in the feature space [22]. K-NN is a type of instance-based learning or lazy learning where the function is only approximated locally and all computation is deferred until classification. The k-nearest neighbor algorithm is amongst the simplest of all machine learning algorithms: An object is classified by a majority vote of its neighbors, with the object being assigned to the class most common amongst its k nearest neighbors (k is a positive integer, typically small).

Probabilistic Boosting-Tree

Boosting is a machine learning meta-algorithm for performing supervised learning. The Probabilistic Boosting-Tree is based on the question posed by Kearns [23] , i.e., can a set of weak learners create a single strong learner? A weak learner is defined to be a classifier which is only slightly correlated with the true classification (it can label examples better than random guessing). In contrast, a strong learner is a classifier that is arbitrarily well-correlated with the true classification.

2.6 Review on Robot understanding Spatial Language

Robot understanding of spatial language has been explored previously. Much of the work has focused on 2D navigation. For example, Gribble et al [24] proposed using the Spatial Semantic Hierarchy (SSH) [25] to represent and reason about space, using commands such as “go there” and “turn right”. The SSH can abstract an agent's spatial knowledge structure in a relatively independent way of the environment.

The work by Skubic et al [26] investigated the use of spatial relationships to establish a natural communication mechanism between people and robots especially for non-professional robot users. In their work they used the grid map to store and represent the surrounding environment of the robot and presented some algorithms to extract spatial information from the map and generate spatial language descriptions by using the histograms of forces [53]. Their experiments include the cases using a human, robot and objects as the reference. The result shows the possibility of robot interaction with human users by spatial language commands. There is a body of work on understanding 2D route instructions for guiding an agent or robot through an environment [27][28][29]. Tellex et al. also consider manipulative commands that move beyond the 2D ground plane, e.g., “*put the pallet on the truck*” [30]. In [30] Tellex and Kollar propose a model named Generalized Grounding Graphics (G^3) to take natural language commands as input and then output robot control commands. In their work, they build the structure of the grounding graphical model by using Spatial Description Clauses (SDCs) [27]. The model is trained on a command corpus and their corresponding grounding. They test the model in a forklift robot. The G^3 model dynamically generates a probability graphical model for

a natural language command by its semantic structure rather than other previous work which uses the likelihood method to find the grounding of a command.

This work has informed our project; however, much of it is focused on the more general natural language processing (NLP) problem and is limited in addressing the perceptual and cognitive challenges of our fetch task in a 3D environment.

Chapter 3 Research Platform

3.1 Robot Design

To fulfill the fetch task requirements, the thesis author and colleagues developed a mobile robot with the intelligence to navigate in an indoor environment and interact with a human. The robot is a differential drive robot with an RGB-Depth camera. The details of the robot components are listed below. A picture of the robot is shown in Figure 3.

Figure 3 Mobile Robot Used in this Project

1) Robot Base: A Pioneer 3-DX (P3DX) robot was used as the robot base component [31]. The P3DX robot debuted in the summer of 2003. The robot is a small lightweight two-wheeled, two-motor differential drive robot that is suitable for indoor laboratory research. The complete version of a P3DX robot includes a front sonar array and a rear sonar array, each with eight sensors, three lead acid storage batteries, optical wheel encoders, an ARCOS firmware microcontroller, and the Pioneer SDK mobile robotics software development package [31]. The

payload of the P3DX is 17Kg, which is enough to load a heavy upper structure for robotics research [31]. It has moving maximum speed of 1.2m/sec and 3 hours of power to sustain cruising from the battery. The P3DX also has I/O expansion capabilities that make it easy to connect with other controllers, sensors and even the actuators' load. Even though the P3DX is not an advanced and costly product; it is still the most suitable for this robotics research.

Figure 4 P3DX Robot Base

2) Tower Frame: The tower frame is made of light aluminum and holds a Kinect camera, an IBM laptop and a robot arm which is to be added in the future.
3) Kinect RGB-Depth Camera: The RGB-Depth camera is popular because it can provide high quality synchronized color and depth data [32]. The RGB camera can return a 640×480 three-channel image. The depth image is an IR image which has the same resolution as the RGB image but in gray scale (representing depth) with its value from 0 to 1023. Usually its effective detection range is from 0.5 meter to 8 meter which means it is appropriate for indoor use. The color and depth data from a Kinect camera will be used as the robot perception tool for furniture recognition, furniture pose determination and furniture searching tasks which will

be discussed in the following chapters. The Kinect camera rests on the top of the robot and is usually tilted between -31 degree and 30 degree.

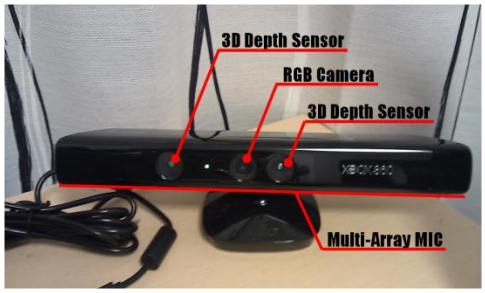


Figure 5 Kinect Camera Structure

4) SICK Laser Range Finder: The laser scanner on this robot is used for emergency obstacle avoidance so that it will not hit a person or other objects on the ground. An LMS200 laser range finder was used on this robot [33]. The laser range finder works in a mode to receive 180 laser signals within 1 degree intervals. This means that it can scan the front nearly 180 degrees without a dead zone. The laser range finder can have a standard 10% reflectivity for 30 m which is quiet enough for indoor detection. The minimum error for the laser distance is 1 mm.

Figure 6 SICK Laser Range Finder

5) Controller: The controller of the robot is an IBM laptop. It is used for running the robot's software which also guides the robot to interact with humans. It connects with other components through USB ports. The furniture recognition program is run on this computer.

3.2 Robotics Operating System (ROS)

The Robot Operating System (ROS) [34][35], which has been developed and maintained by Willow-Garage, is a software framework for the development of robotics systems. ROS provides libraries and tools to help software developers create robot applications. ROS is completely open source (BSD) and is free for anyone to use and change [34]. The software package contains hardware abstraction, device drivers, libraries, visualizers, message-passing, package management, and more. The image processing software that was used to realize the algorithm in this paper was written by using the Open Computer Vision Library (OpenCV), a package of ROS.

There are currently four versions of ROS in use [35]. They are ROS Box Turtle published in March 2010, ROS C Turtle published in August 2010, ROS Diamondback published in March 2011 and ROS Electric published in August 2011. The latest version Ubuntu 11.10 is the only operation system that perfectly supports ROS; this has been installed in the robot controller.

The work of this research is represented as several ROS packages and then deployed on the robotics platform.

Chapter 4 Methodology of Furniture Detection

This chapter consists of three parts--furniture recognition, furniture orientation detection and robot repositioning for recognition improvement. They work together to build a complete process that can collect enough information from a furniture sample for robot path planning. This information includes the category and the instance of a furniture sample and the pose of it. This information can help the robot on human-machine spatial language interaction by providing the class and the pose of reference objects.

Section 4.1 introduces how to get furniture samples and recognize them in an RGB-Depth image scene. Section 4.2 introduces the definition of pose to different kinds of furniture items and how to detect them. Section 4.3 discusses a robot behavior scheme designed to enable a robot to detect a furniture sample and improve recognition performance when its recognition confidence is low.

4.1 Furniture Items Recognition

Furniture items are considered good landmarks for indoor environment robot navigation [49][50][51]. With good recognition capability, a robot can localize itself relative to furniture landmarks in an environment with known furniture. For use in the fetch task in a home environment, the object recognition algorithm for a robot should have the following prerequisites:

(1) There are only a few samples needed for training.

(2) High recognition accuracy is needed.

(3) The recognition time should not be too long. A real-time recognition processing time should be less than 0.1s.

(4) The recognition should be able to tolerate some occlusion.

(5) For a strategy is needed for category recognition in the case that a specific instance has not been trained.

In this thesis the recognition algorithm is original, specific to this study, and it takes full advantage of the Microsoft Kinect Camera which is used as the main sensor of the robot. Both RGB information and depth information which are returned by the camera are used in recognition decisions. To better describe a furniture sample, a model is built based on the shape features. The information from this model is combined with discriminant classification features to determine the instance of a furniture sample.

In the human-robot interaction experiments, there are eight furniture pieces used in the indoor environment which need to be recognized, as shown in Figure 7.

Figure 7 Furniture items used

The recognition of furniture items has 4 steps:

1) Sample Extraction.

2) Main Plane Extraction.

3) Feature Generation.

4) Classification.

The result of the recognition can be either the category or the instance name of the furniture sample.

4.1.1 Sample Extraction

The first step is sample extraction which segments data in the RGB-Depth Image to retrieve a furniture sample. The RGB-Depth image is collected by an ROS package program. It allows the computer on the robot to communicate with the Kinect to input the RGB-Depth raw images. The frame received by the robot contains a three-channel 640 × 480 RGB image and a synchronous 640 × 480 gray depth image. The pixel value of the depth image can be converted to represent its distance from the camera. By distance information and the depth camera parameter, a 3D point cloud can be obtained from the raw image. By using the method in [37], the RGB image and the 3D point cloud data can be fused into a RGB-point cloud frame, which is a 3D color-scale scene. The samples are extracted from this RGB-point cloud scenario.

The procedure to obtain samples has 5 steps:

1) Obtain a RGB-point cloud scene and transform it from camera coordinates to robot coordinates. The robot coordinate frame is illustrated in Figure 5. The transformation matrix is:

$$\boldsymbol{P_r} = \begin{Bmatrix} \boldsymbol{x_r} \\ \boldsymbol{y_r} \\ \boldsymbol{z_r} \end{Bmatrix} = \begin{Bmatrix} \boldsymbol{1} & \boldsymbol{0} & \boldsymbol{0} \\ \boldsymbol{0} & \boldsymbol{\cos\theta} & \boldsymbol{\sin\theta} \\ \boldsymbol{0} & \boldsymbol{-\sin\theta} & \boldsymbol{\cos\theta} \end{Bmatrix} * \begin{Bmatrix} \boldsymbol{x_c} \\ \boldsymbol{y_c} \\ \boldsymbol{z_c} \end{Bmatrix} + \begin{Bmatrix} \boldsymbol{0} \\ \boldsymbol{0} \\ \boldsymbol{H} \end{Bmatrix} \quad \textbf{(1)}$$

- P_r and $x_r\ y_r\ z_r$ are the position of point in robot coordinate
- $x_c\ y_c\ z_c$ are the point position in camera reference
- θ is the tilt angle
- H is the camera height.

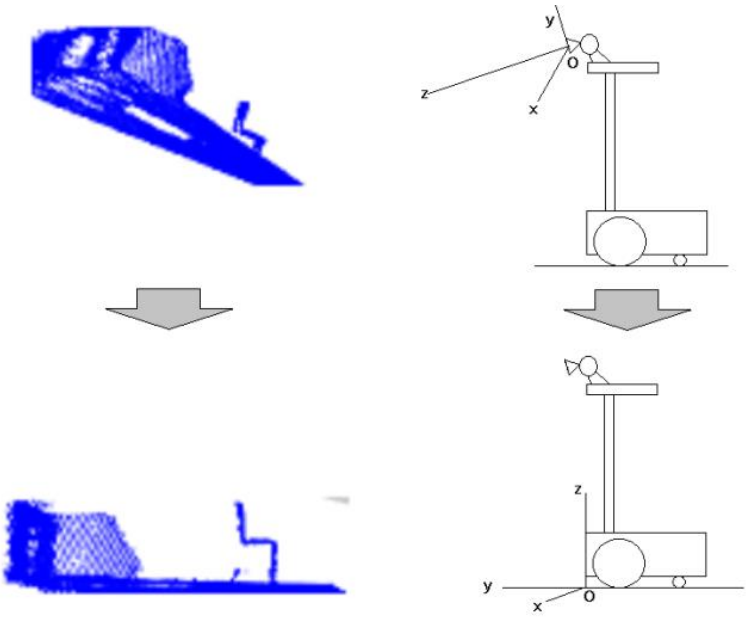


Figure 8 Changing the Coordinate to Robot Reference

2) Eliminate all the points where z-axis values are smaller than 0.1 m so that the points belonging to the ground are ignored leaving only the points that represents furniture items. (Figure 9)

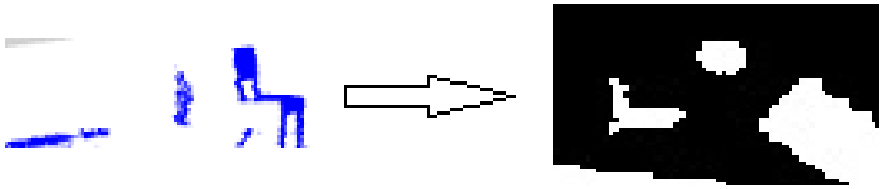


Figure 9 Generating the Sample Map

3) Plot a 2D grid map by projecting all the points to X-Y coordinate. The range of the coordinates is: $-3.0\text{M} \leq \text{X} \leq 3.0\text{M};\ 0\text{M} \leq \text{Y} \leq 4.0\text{M}$. The 2D grid map has a resolution of $0.1\text{M} \times 0.1\text{M}$. If a point in the point cloud falls into the range of a grid cell, the cell will be set as occupied. Unoccupied cells are labeled as background. Finally, complete step 3 by recording the index of the cell a point belongs to. (Figure 9 Right)

4) Find the connected components in the grid map image by using the method in [38] Label all the components with indices above zero (1, 2, 3…). The background cells which do not contain any points are labeled 0. Choose the points that belong to the same component to be a sample. Figure 10 shows the extraction process. The gray blob in the left figure represents the target object project to a 2-D map and the right one is the 3-D point belongs to that area.

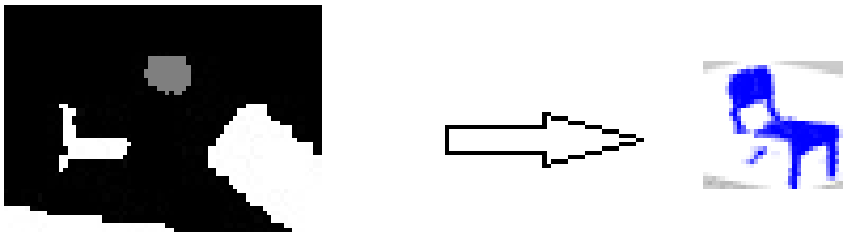


Figure 10 Target Extraction

5) Ignore the samples if the height of the highest point is larger than 2 meters or the component has fewer than 10 cells because it may be the part of a piece of wall or a piece of clutter on the ground. An example of an original image and the segmented sample is shown in the Figure 11.

Figure 11 An example of a raw image (left) and the segmented sample (right)

4.1.2 Plane Extraction

One feature that most furniture items have in common is that they have a relatively flat horizontal plane. For example, a chair has a fairly flat and level plane designed to comfortably fit its occupant and a dinner table has a flat and level plane designed to accommodate plates and glasses. Such a plane, which is designated as the "main plane" in this thesis, enables a furniture item to realize its designed function for daily use. The advantage of using a main plane to extract features is that even when small objects enter the main plane area, which may lead to a change in furniture shape, the plane can still be extracted and its shape can be used for classification. This improves the robustness of the classifier. The RANSAC method is used to extract the plane part from a furniture sample.

RANSAC

Random Sample Consensus (RANSAC) is an iterative method to estimate parameters of a mathematical model from a set of observed data, and it is capable of interpreting or smoothing data containing a significant percentage of gross errors [36]. Rather than using data to obtain an initial solution and then trying to eliminate the invalid data, RANSAC uses the initial data as little as possible and enlarges this set by testing the candidate points [36]. RANSAC is effective for model fitting, particularly when a significant percentage of data are outliers [36]. In a RANSAC algorithm, a minimal set is the smallest number of points required to uniquely define a given type of mathematical model (usually a geometric primitive). Then the resulting candidate shapes are tested against all points in the data to vote how many of the points are well approximated by the primitive (called the score of the shape). After a given number of trials, the shape which

approximates the most points is extracted and the algorithm continues on the remaining data.

The plane is another RGB-point cloud sample from which some features can be obtained. An obvious and common feature of a piece of furniture is that it has a large enough main plane to realize its designed function. For example, a bed should have a plane that allows people to lie down on it, a chair should have a plane that allows people to sit on it, and a desk should have a plane that allows people to use it as a level work surface. Therefore, by extracting the main plane and analyzing its property, different furniture items can be classified.

Figure 12 A Main Plane Extracted from a Chair

The procedure of using RANSAC to extract the main plane is shown in the following flow chart.

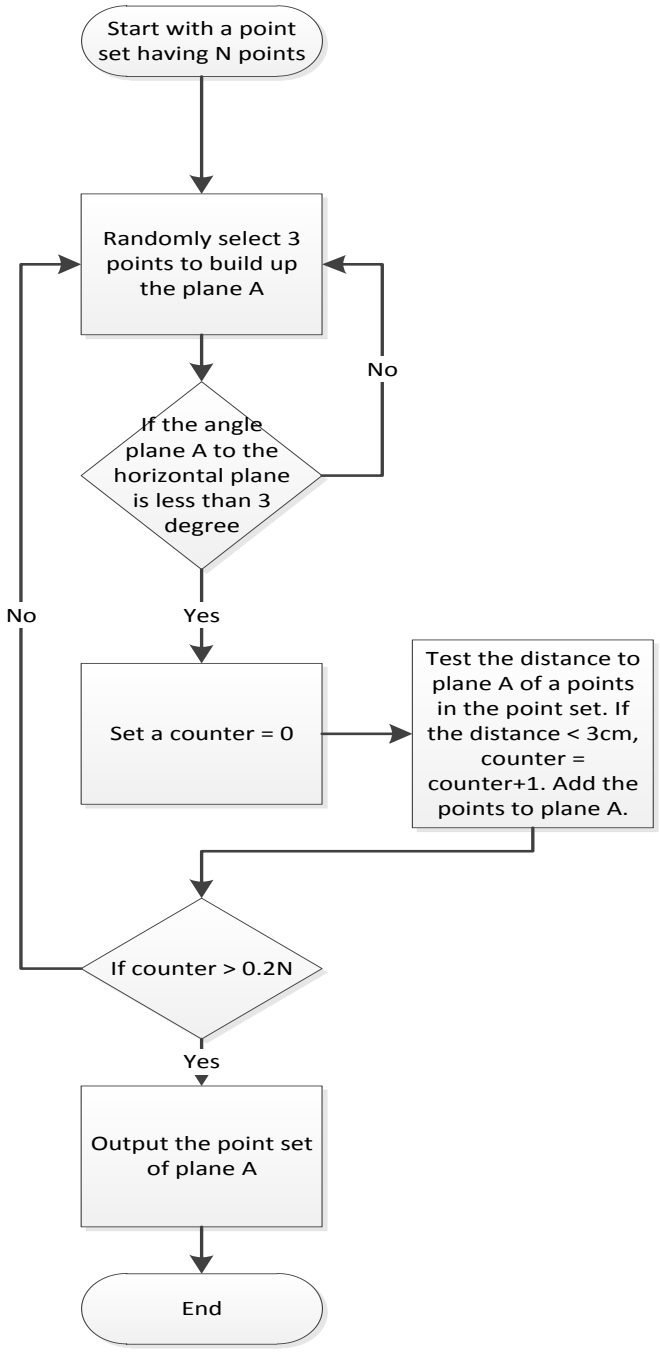


Figure 13 RANSAC Plane Extraction Procedure

The points of plane A is the result of main plane extraction. After the main plane is extracted, an image of the main plane that eliminates any other point not belonging to the plane is extracted from the RGB-depth scenario image. The pixels that do not belong to the plane are then set to be zero for all three channels. Figure 14 shows the result of plane

extraction for a table. Both the vertical planes and small object clutter on top are subtracted.

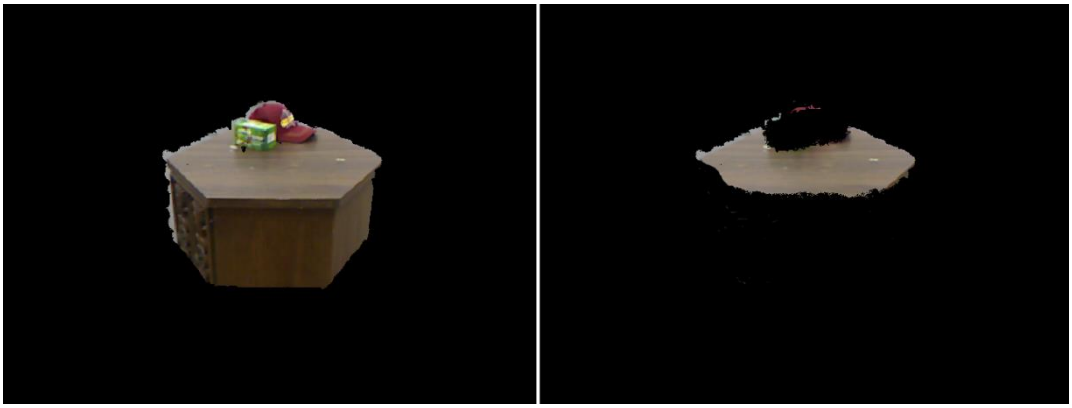


Figure 14 Plane Extracted from Cluttered Furniture

4.1.3 Features

A feature is an individual measurable heuristic property of a phenomenon being observed. Choosing discriminating and independent features is the key to any pattern recognition algorithm being successful in classification. In this thesis, a linguistic model was built to find the category of a furniture sample, and a probability graphical model (PGM) was built to determine the furniture sample's instance class. To take full advantage of the RGB-Depth information from the Microsoft Kinect, features were generated from both appearance and geometry information. The following features were used for classification.

Size

The size feature equals the number of cells in a 2D grid map. The map is the 2D projection of the furniture sample to the ground. Size is a geometry property of a piece of furniture which is typically different for different kinds of furniture. The size of a bed is larger than a chair or a small table. However, the size differs when the orientation of a furniture item with respect to the robot camera changes. Thus, the sample feature's size

based on camera orientation may be smaller than its actual size but cannot exceed its actual size.

Main Plane Height

The Plane height is represented by the mean of the highest and lowest height of a point in the plane point cloud. The main plane height is a good feature because each type of furniture often has a different height which rarely changes even when the furniture item is viewed from different angles. Height is a robust feature when defining and sorting furniture samples. For example, a dinner table is always much higher than the main plane of a chair so that these two types of furniture items can be easily separated by height.

Furniture Shape

The furniture items used in the experiment have two kinds of shape--table and chair shape. The value of this feature ranges from 0 to 1. The generation of this feature is shown below:

1) Ignore all the RGB information of the furniture sample and the points where height is lower than the highest point of the main plane.
2) Find the maximum distance for each angle (360 in total) of the furniture sample point cloud to the centroid of the furniture item.
3) Compute the membership angle as part of the chair back by using Equation (2

$$\boldsymbol{F_{side} = \sum_{i=1}^{360} Max(\frac{d_{\_ij}}{D_{\_i}})} \qquad \textbf{(2)}$$

4) Find the proportion of the angles that belongs to the chair side to confirm that the sample belongs to a chair. The membership is shown by Equation 3.

$$\boldsymbol{M = \frac{F_{side}}{360}} \qquad \textbf{(3)}$$

– *M* is the membership value of a furniture sample, which in this case belongs to the chair shaped furniture items. The larger the value *M* is, the more likelihood a sample belongs to chair.

Main Plane Texture

Roughness rate is used to help describe the texture type of the main plane. Each of texture has a different roughness rate which can be used as a feature. For example, a mono-color surface has a very low roughness while a strip or grid texture has a high roughness. The feature is a one-dimensional value that ranges from 0 to 1determined by the following steps:

1) Find the roughness rate. Use the image that contains the plane points only. Compute the rough rate of the plane by Equation 4-6:

$$\boldsymbol{D(P_i, P_j) = Norm(P_i, P_j)} \quad \textbf{(4)}$$

$$\boldsymbol{L_n = L(P_n) = \begin{cases} if\ MAX(D(P_n, P_k)) > 50 = 1 \\ else = 0 \end{cases}} \quad \textbf{(5)}$$

$$\boldsymbol{R = \frac{\sum_{i=1}^{N} L_n}{N}} \quad \textbf{(6)}$$

– N is the number of the points of the plane.

– P is the RGB vector of a pixel.

– R is the roughness rate wanted.

2) Find the texture type by using histograms. Define $H = \{H_1, \dots H_8\}$ which represent the histograms of eight directions from 0 to 315. $H_i = \frac{\sum_{j=1}^{N} h_{ij}}{N}$ in which h_{ij} means the *j*th point with the *i*th direction magnitude, and N is the number of points. If $H_i > 50$, the *i*th direction can be seen as it has a gradient which means the color in this direction has an obvious change; then:

$$\boldsymbol{P = \frac{N(H_i>50)}{8}} \qquad \textbf{(7)}$$

- P is the value that represents the texture style.
- The higher the P, the more the texture tends to become like a grid as it takes on a striped appearance.

RGB-Intensity of the Main Plane

The RGB-Intensity feature is a three-dimensional vector which is used to make the final decision when the furniture type is determined. This feature has normalized proportions of the red, green and blue color components of the plane image. The computation to generate this feature vector *F* from the plane image is,

$$\boldsymbol{F = \{R, G, I\}} \qquad \textbf{(8)}$$

$$\boldsymbol{R = \frac{\sum_{i=1}^{N} r_{pi}}{N}} \qquad \textbf{(9)}$$

$$\boldsymbol{G = \frac{\sum_{i=1}^{N} g_{pi}}{N}} \qquad \textbf{(10)}$$

$$\boldsymbol{I = \frac{\sum_{i=1}^{N} {r_{pi}+g_{pi}+b_{pi}}/{3}}{N}} \qquad \textbf{(11)}$$

- N is the number of all the pixels belonging to the furniture sample.
- The *r*, *g*, and *b* represent the red, green and blue values of the *i*th pixel of the sample value.
- *R* and *G* are the Red and Green component of the plane image.
- *I* is the intensity (grayscale of the plane image)

The features extracted from a sample reflect the properties of color, texture and geometry.

4.1.4 Furniture Model and Classifier Design

The classifier in this thesis has a hierarchical structure. The first layer is a fuzzy logic classifier which determines the category of a furniture sample [42][43]. The second layer

is a discriminant classifier which finds the instance name after the category is determined. The first part discusses how to use fuzzy logic rules which are generated by a series of training steps to determine the type of furniture. These types are not simply the labels of similar furniture items but can also describe some properties of the furniture samples. After the type is determined, a probability graphical model is used to classify a furniture sample in a pre-trained scale set by using the RGB-Intensity feature and texture feature.

Category Classification by Fuzzy Logic

Category Classification uses a linguistic model for furniture classification with features from depth information, which are size, main plane height and furniture shape as described in Section 4.1.3. The classifier is built on a fuzzy logic machine with three linguistic variables and their corresponding membership functions. The three linguistic variables and the corresponding values are defined by their depth features. These are size, plane height and furniture shape. To find the values and membership functions of the linguistic variables, the K-means clustering [54] method is used on each feature data set. By finding centroids of clusters for each feature, it can be concluded that different furniture categories have a different tendency for each of their feature values. From the training, each linguistic variable has two to three membership functions of different values that define particular ranges to describe the properties of a type. After running the training, the data of all the furniture items are put together to find the clustering tendency of the whole set. Through training, all the furniture samples can be grouped into five categories, which are *chair*, *small table, large table, couch an*d *bed*. The same category furniture items may vary in appearance but share the same geometry feature.

After training, the structure of the linguistic variables and their membership functions are shown in Figure 15:

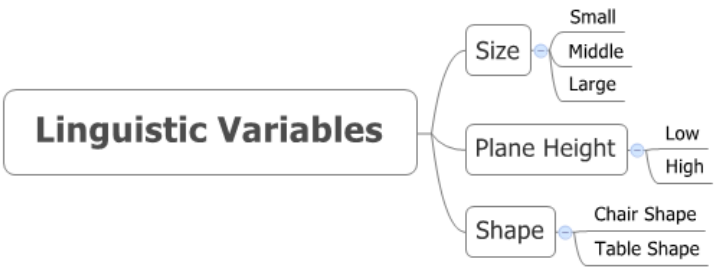


Figure 15 Linguistic Variables

The membership that a sample belongs to a category is defined by

$$\mu_{category_i} = \boldsymbol{mean}(\mu_{size_i} + \mu_{plane\ height_i} + \mu_{shape_i}) \quad \textbf{(12)}$$

- $\mu_{category_i}$ is the membership value of the *i* category.
- μ_{size_i}, $\mu_{plane\ height_i}$ and μ_{shape_i} are the corresponding membership values of the linguistic word in the rule that defines *i* category.

It uses a winner-take-all rule to determine the category decision result. The category with the highest membership can be selected as the result.

Furniture Instance Classification

For the samples used in this thesis, if there is more than one furniture item included in a category, the instance name of the sample will be found by using the RGB-Intensity and texture features.

For each category, a PGM is designed to discriminate each instance in each category set by using the RGB and texture features. Then, the instance name is determined. The *small table* and *chair* categories have more than one instance and their PGM for instance recognition is shown in Figure 16 and Figure 17.

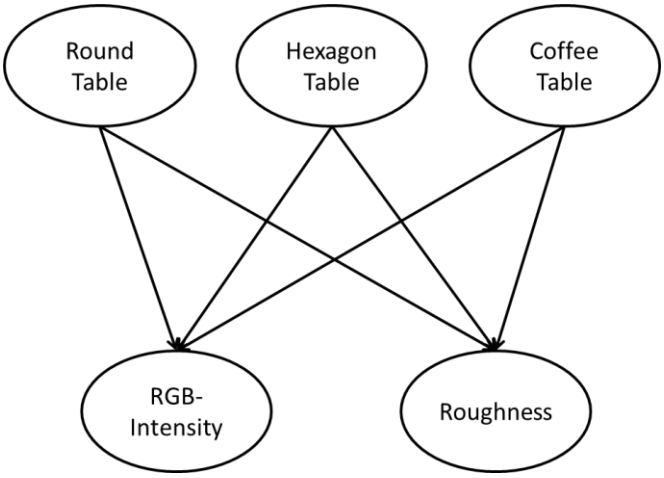


Figure 16 PGM of Small Table

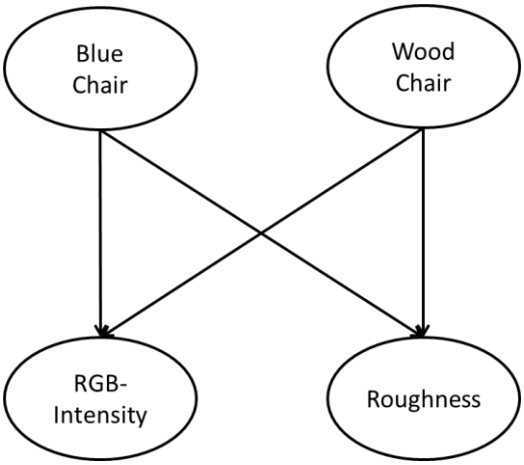


Figure 17 PGM of Chair

The two features use different models to compute their conditional probability. The RGB-Intensity features use a K-nearest-neighbor model (K-NN) [22] and the texture feature to uses a Gaussian model.

In both the small table and chair categories K-NN model, the *K* parameter is chosen to be five which is considered to be optimal from several trials. The probability that a sample belongs to an instance by K-NN is:

$$\boldsymbol{P(RGB-I|Instance_i)=\frac{n}{K}, K=5} \quad \textbf{(13)}$$

- *n* is the number of *ith* training instances that are the five closest samples to the testing sample.

The Gaussian model to compute the conditional probability of roughness in instances is:

$$\boldsymbol{P(Texture|Instance_i)=\frac{1}{\sqrt{2\pi}\sigma_i}e^{-\frac{1}{2}\left(\frac{r-u_i}{\sigma_i}\right)}} \quad (14)$$

- *r* is the roughness value of the testing sample
- u_i is the mean of the *ith* instance roughness training data and σ_i is the deviation of the *ith* instance roughness training data.

The probability a sample belongs to an instance by Bayes discriminant principle is:

$$\boldsymbol{P(Instance_i)\infty P(RGB-I|Instance_i)\times P(Texture|Instance_i)} \quad \textbf{(15)}$$

The result of *N* instances in a category is determined by Equation 16

(16:

$$\boldsymbol{i_{target}=arg\max_i\{P\left(sample\in instance(i=1)\right),\ldots,P\left(sample\in instance(i=N)\right)\}} \quad \textbf{(16)}$$

Confidence of Recognition

The confidence value measures the reliability of the recognition result. Usually, the confidence value of a recognition algorithm is related to the strength of a feature set in supporting a class. In this thesis, two factors are considered in determining the confidence. One is the intrinsic factor, and the other is the extrinsic factor.

The intrinsic factor is a number ranging from 0 to 1, which is the combination of the membership values of linguistic variables and confidence due to the RGB and texture features. It is defined as

$$C = Min(M, F) \quad (17)$$

- M is the membership that the furniture sample belongs to its category by Equation 12.
- F is the confidence from the instance recognition based on its corresponding RGB and texture features.

The extrinsic factor is generated from three aspects, which are distance, direction and completeness, which means that the extrinsic confidence value is based on the relative position between the furniture sample and the robot. Unlike other static recognition cases, a robot can move to change its position which may result in improved recognition confidence. Defining the proper relationship between relative position and confidence can help to guide the robot to a better position so that it can get a more reliable recognition result.

Equation 18-20 shows the method used to compute the extrinsic confidence parameters:

$$Con_{distance} = 1 - |d - 1.5| \quad (18)$$

$$Con_{direction} = 1 - \frac{|\theta - 270|}{90} \quad (19)$$

$$Con_{completeness} = \begin{cases} 0, if\ the\ sample\ is\ not\ large\ size\ and\ incomplete \\ 1, otherwise \end{cases} \quad (20)$$

So that:

$$Con_{extrinsic} = Mean(Con_{distance}, Con_{direction}, Con_{completeness}) \quad (21)$$

The extrinsic confidence of the recognition is defined as the Equation 21.

4.2 The Detection of the Pose of Furniture Items

The pose of a furniture item consists of 2 parts. One is furniture position, and the other is furniture orientation.

4.2.1 Position of Furniture Items

Empirically, to simplify the definition of an object position, it is typically computed as the geometry centroid of its projection on the ground. However, because the size of the furniture is relatively large, the pose relationship between parts of the furniture items may not be same, which means their relative position cannot be defined by a single direction. Hence, a furniture samples cannot be considered as a mass point. In this thesis, the position of a sample is represented by its corresponding connected region in the grid map. The distance between two objects (including the robot) is then defined as the distance between their nearest points from each other.

4.2.2 Orientation of Furniture Items

Human subject experiments have shown that users sometimes reference the intrinsic frame of some furniture items (e.g., couch, chair) [49][50][51]. Even without an intrinsic frame, references such as front and back may depend on the orientation of the furniture item (e.g., rectangular tables) [49][50][51]. Thus, it is important for a robot to precisely detect the orientation of a furniture item. It is very challenging to find the orientation of a piece of furniture when depending on appearance information only. Therefore, orientation is detected by the depth information (shape) as defined in Figure 18.

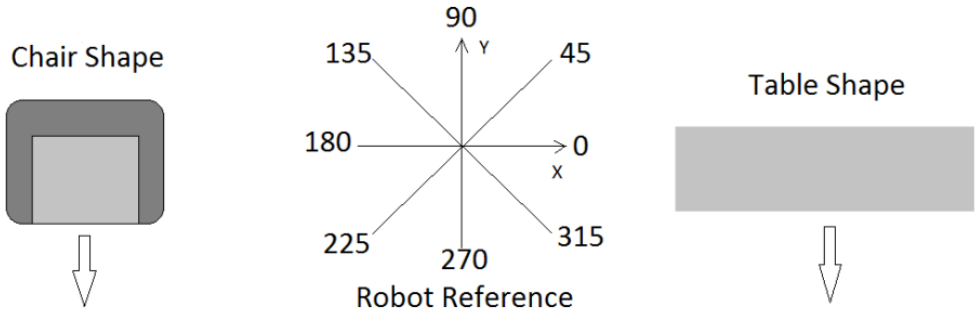


Figure 18 Furniture Orientation Coordinate

Figure 18 shows how all orientation values are based on the robot reference.). Figure 5 also demonstrates how the camera views furniture as facing its lens when the orientation is around 270^{o}, but when the orientation is around 90^{o}, the camera assumes that it is behind the furniture. However, the definition and detection of orientation are both different for chair shaped furniture and table shaped furniture.

Table Shape Furniture

The shapes of tables always have a symmetrical structure for both the long axis and short axis, which can be rectangular, oval and round. This section will talk about furniture pieces with these three shapes.

First the long axis is must be found and then the shorter one in the axis perpendicular to the long axis. The information that can be used for finding the long axis is the point cloud of the main plane. The reason that the other parts are not used for the task is that the parts under the plane may be hidden by the plane or the objects on the plane. To find the long axis, the following steps need to be followed.

1) Draw the map by projecting all the points in the point cloud to X-Y plane.
2) Set $P = \{p_1, \dots p_N\}$ sd the set of all the X-Y coordinate points in the first step.
3) Find the angle of each point by using $\theta_i = \arctan\left(\frac{x_{pi}}{y_{pi}}\right)$ to build set Θ. Set the number in Θ to be integers. Find the set of points P_M which has the largest distance to the centroid compared with other points that have the same angle. P_M is then the contour of the sample grid map.
4) Find the center $C(x_c, y_c)$ of the point cloud, set line $L(\theta)$: $x\cos\theta + y\sin\theta = \left\|x_c, y_c\right\|$ across the center. Change θ from 1 to 360 degree. For each $L(\theta)$, compute the distance values for all the P_M points to $L(\theta)$ and gathering them in

set $D(P_M, \theta)$. For each θ, find the average and standard deviation values of distances from D which are D_m and D_v.

5) Find the θ that minimize value in $D_v \times D_m$. This angle is on the long side. The angle perpendicular to the long side and facing the robot is chosen to be the orientation of the table shaped furniture item.

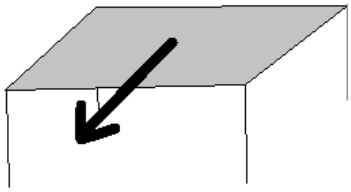


Figure 19 Table Shaped Furniture Item Orientation

Chair Shaped Furniture

The orientation of chair shaped furniture is defined as the angle that it faces to the robot camera reference. That is, the orientation is the direction the chair edge faces as shown in Figure 17.

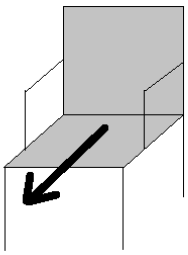


Figure 20 Chair Shaped Furniture Orientation

Chair shaped furniture has two main functional parts. One is the cushion which people sit on, and the other is the back of the chair. The key to finding the orientation is to accurately recognize the two parts and find the spatial relationship between them.

The procedure to find the chair back has four steps.

1) First, select all the points above the main plane as set S. Draw an image representing an area of 360 × 200 pixels. Project all the points to the image. Label all the cells where at least one point falls as shown in the figure below.

Figure 21 Upper Part Projection Image

2) Set the geometry centroid as the center from the first point of set S, Determine the X value as the angle of the points and the Y value as Equation 22

$$\boldsymbol{y = \frac{H - H_{min}}{H_{max} - H_{min}}} \qquad \textbf{(22)}$$

where

- H is the Z value of the sample point.
- H_{min} and H_{max} of the Z value in points set S.

3) Label each pixel in the image with the value from the Equation 23

$$\boldsymbol{p_{ij} = \frac{s_i + s_j}{2}} \qquad \textbf{(23)}$$

where

- *i* and *j* are the column and row index of a pixel,
- s_i and s_j are the portion shown as the labeled pixel in the corresponding row and column.

The pixels with $p_{ij} > 0.5$ are then labeled as part of the chair back (Figure 19).

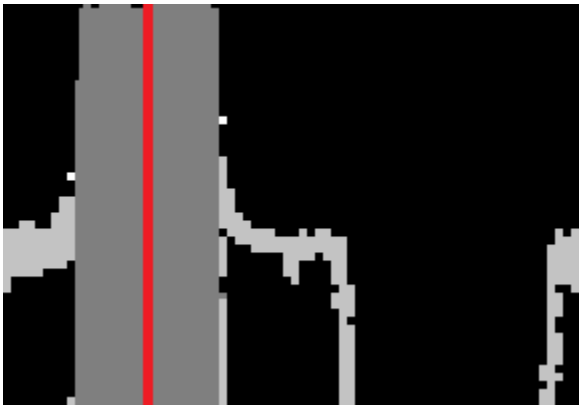


Figure 22 Back Part Extracted

4) Find the center of the back region. The angle of the back center is determined as the orientation of the furniture sample, e.g., the red line in the figure above.

4.3 The Robot Repositioning Behavior for Improved Furniture Recognition

When the robot is trying to classify a furniture sample in an experimental environment, it needs to have a good viewpoint of the furniture item. Because the training samples are limited and cannot allow a large variety of distances and orientations, it is necessary to give the robot the capability of moving to a proper pose where it can improve the recognition accuracy of furniture samples.

Therefore, a robot behavior is needed to navigate the robot to such a position which can give the robot as complete information on the furniture sample as possible.

This section describes how to drive a robot to find a target furniture item and improve the performance of furniture recognition by driving the robot to another place when the confidence of recognition is low. Based on the factors used to compute the confidence in Chapter 4, there are five reasons that lead to low confidence of recognition:

1) The distance between the furniture sample and robot camera is too large so that the sample image is not clear enough to present some details.
2) The furniture sample is not in the center of view.
3) The furniture sample is not completely in the Kinect view.
4) The furniture sample is completely in the Kinect view but some of its parts are occluded.
5) In addition, the result of the recognition experiments shows that orientation makes a difference in recognition performance.

Therefore, to improve recognition performance, the robot needs to move to a proper position where it can have the highest probability to recognize the furniture item with high confidence. This kind of position needs to fulfill the following two requirements.

1) The sample needs to be complete enough in the camera view which means that there are no depth points at the edge of the depth image except for large size furniture items.
2) For the chair shaped furniture samples, the robot needs to move to the front of the furniture sample (in furniture coordinates) as much as possible.

To determine the relative pose between the robot and a furniture sample, especially the direction relationship, the histograms of forces (HoF) method is used.

4.3.1 Histograms of Forces

The histograms of forces approach is used to compute the relative position between two objects. The objects used for this method can be either crisp or fuzzy. To measure the weight of spatial relation in an angle θ that *A to B*, assuming $\Delta_\theta(v)$ is a batch of vectors

of θ that has an interaction with A and B. The disjoint segment of the interaction that $\Delta_\theta(v)$ with A *and* $\Delta_\theta(v)$ with B is the weight of angle θ.

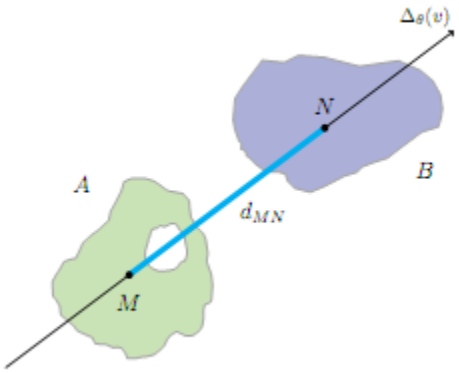
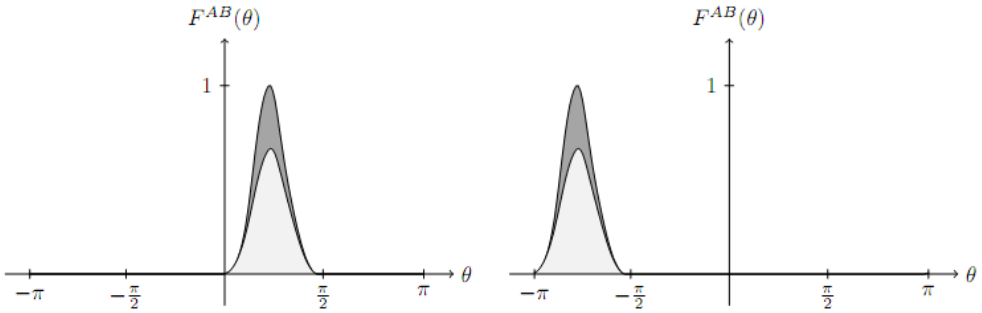


Figure 23 Two Objects [39]

Assuming a two objects A and B, for an angle θ, Hof shows the weight of how much "A is in direction θ of B". A typical histogram of forces is shown in the Figure 24.



(a) Histograms of forces for A wrt B (b) Histograms of forces for B wrt A

Figure 24 Histograms of Forces [40]

The histograms of constant (dark gray) and gravitational (light gray) forces for objects A and B are shown in Figure 23. By the histograms of forces the weight of each direction can be computed and then the direction of the robot to a furniture sample can be determined.

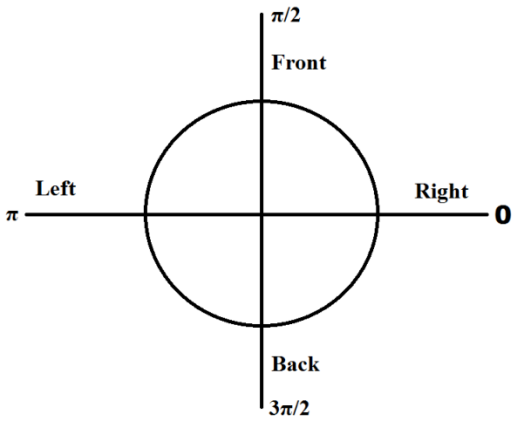


Figure 25 Directions from Histograms of Forces

Defining four main directions, which are front, left, back and right, as shown in Figure 25, 0 rad means 100 percent right, π/2 rad means 100 percent front, π rad means 100 percent left and 3π/2 means 100 percent back. To simplify the computation of direction, a Gaussian kernel is used for each angle to generate the weight of a direction. The equation to compute the weight of a direction is:

$$w_A = \frac{\int_0^{2\pi} G(x,A) * H_x \, dx}{\int_0^{2\pi} H_x \, dx} \qquad (24)$$

where

- H_x is the value of the histogram for x rad and
- $G(x, A)$ is the membership value that *x* belongs to direction *A*. In

$$G(x, A) = \frac{1}{\sqrt{2\pi} * \pi/4} e^{-\frac{1}{2}\left(\frac{D(x,A)}{\pi/4}\right)^2} \qquad (25)$$

- *D(x,A)* is the distance between x and the core angle of direction A.

By using histograms of forces, the relative position between the robot and furniture sample can be easily represented and quantified.

4.3.2 Robot Behavior

The behavior model of the robot when running this task is a three-tier structure [44][45] shown in Figure 26. The first tier is the perception tier which collects the sensor information including the target furniture sample. The second tier is the Intelligence tier. By using these recognition methods, the robot detects the class and pose of the samples and the corresponding confidences. These methods also enable the robot to plot the action scheme and determine the state of the machine. The third tier is the motor actions tier. In this tier, the scheme is converted to motor parameters to control robot movement.

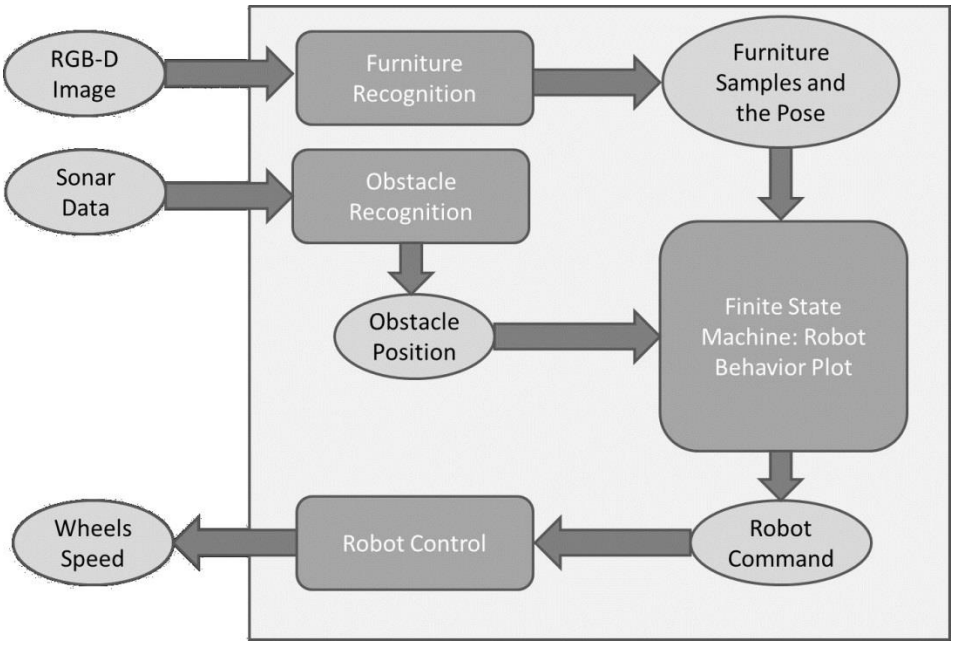


Figure 26 Robot Behavior Diagram

By using a finite state machine (FSM) [46], a strategy for robot behavior that can improve the recognition confidence is to navigate the robot to a place where higher extrinsic recognition confidence can be achieved.

A finite state machine is a mathematical model of computation used to design sequential logic circuits. The name represents an abstract machine that can be in one of a

finite number of states. The machine can only be in one state at a time. An FSM can change from one state to another when triggered by a conditional change which is called a transition. A particular FSM is defined by a list of its states and the triggering conditions for each transition. In the robotics domain, the FSM is used most often when building models of robotic behavior [45].

Figure 27 shows an FSM model. The robot is navigated by odometry when moving from the original pose to the target pose. To improve recognition performance, there is no need for the robot to move to a perfectly accurate position. When the robot detects the position of the furniture item as "front", the robot can stop and try to recognize the sample again. The FSM strategy's process steps are:

1) If the robot detects both low extrinsic confidence and low intrinsic confidence when performing recognition on a furniture sample, the recognition result is not reliable which means the robot needs to move to a better viewpoint to update the recognition result, thereby improving the reliability of the recognition.
2) Allow the robot to make a 90^{o} turn to the other side of the furniture sample.
3) Turn back to face to the furniture sample and repeat #2 again. If the robot is in "front" of the furniture sample, move an optimized distance from the sample (1.5 to 2 meters).
4) If the extrinsic confidence and intrinsic confidence increase which means the robot has moved to a good viewing place, navigate the robot closer until an ideal place is reached for detection.

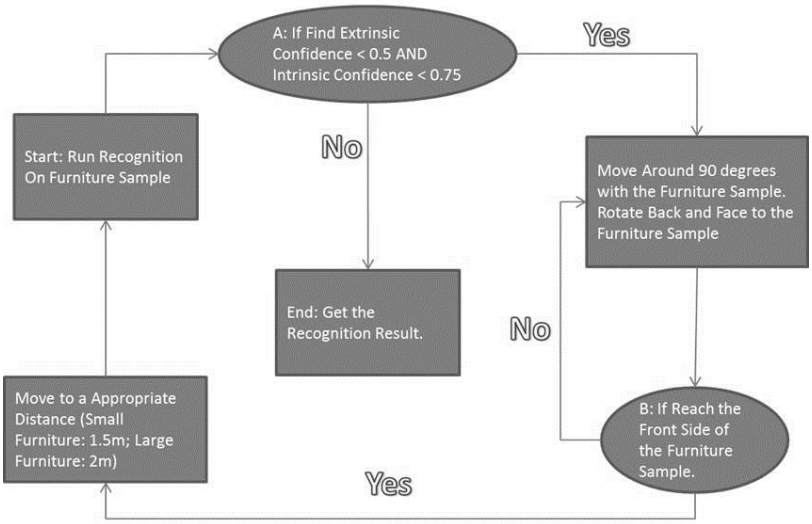


Figure 27 FSM for Robot Action

A typical scenario of these steps is shown in Figure 28. In this figure each image corresponds to one of the four steps in the steps of the FSM strategy for better navigation and positioning.

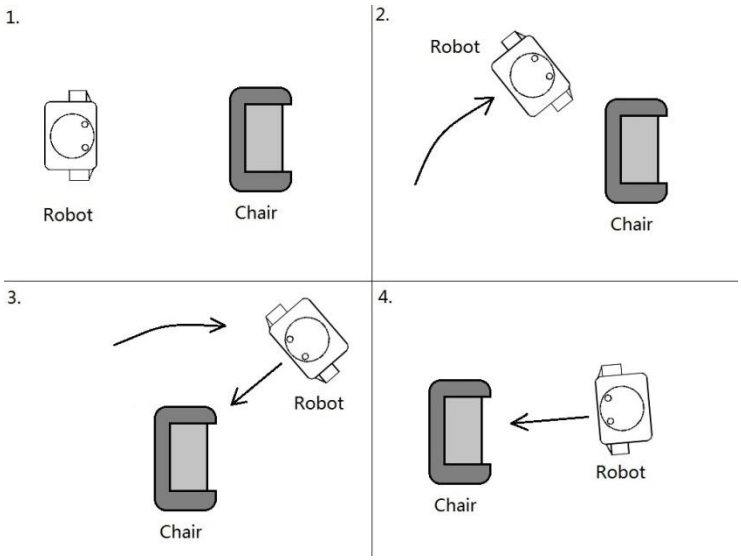


Figure 28 Robot Task Steps

Chapter 5 Translating Spatial Language into Robot Commands

In this research, the thesis author and colleagues designed a robot fetch task experiment to investigate the use of spatial language for human robot interaction. A set of human subject experiments was conducted in which spatial language descriptions were collected from younger and older adult subjects. The collected corpus was analyzed in [47] and templates were constructed to characterize the spatial language patterns for different test conditions. These templates provide a spatial language corpus that is used here to investigate the automatic translation of natural spatial language into robot commands. The mechanism includes two parts. The first step is to build a human-robot spatial language model which is used to tag words based on part of speech and then segment them into meaningful chunks; this step was accomplished by a colleague [48] but is briefly described here to show the output of this step. The second step uses the chunked description to generate robot navigation instructions which can be understood by a robot; the second step is part of the contribution of this thesis.

5.1 Semantic Chunks

The first step to interpret a human spatial language command is tagging and chunking into meaningful semantic chunks. It is done using a statistical natural language model. This work is done by my colleague. There are five steps in building a semantic chunk structure from a spatial description: (1) adding part-of-speech (POS) tags—(word, tag) tuples, (2) filtering out some "noise", (3) generating semantic chunks —(word, tag, chunk) triples and filtering out some "noise" is again possible if new chunk types are introduced, (5) building a tree structure.

Table 3 Chunk and POS Types

Chunk Type	Explanation
ORMTP	*Outside Room Target* Phrase
ORMRP	*Outside Room Reference* Phrase
FURTP	*Furniture Target* Phrase
FURRP	*Furniture Reference* Phrase
OBTP	*Object Target* Phrase
OBRP	*Object Reference* Phrase
IRMRP	*Inside Room Reference* Phrase
POS Type	**Explanation**
DT	*The* Word
NN	*Noun* Word
VBZ	*Verb* Word
IN	*In* Word
RM	*Room* Word
ON	*On* Word
DIR	*Direction* Word
TO	*To* Word
FUR	*Furniture* Word

There are seven kinds of high level chunks which can then be used to extract navigation instructions. They are shown in Table 3. Figure 29 shows an example of the interpretation and translation of the description "*the statue is in the room on the left on the table ahead to the left*" into a tree structure. The different kinds of chunks can be classified to eight types which work in different steps in the fetch task.

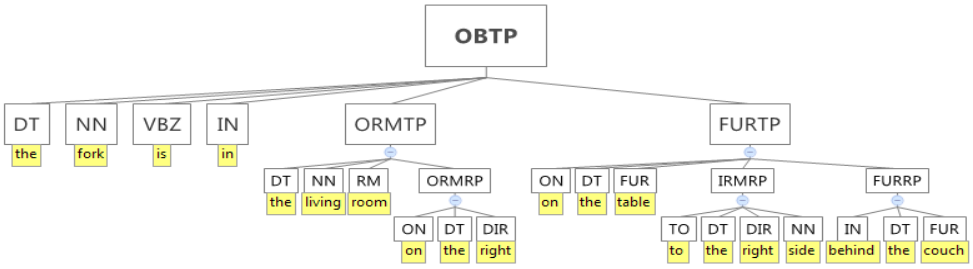


Figure 29.An example of a chunked spatial description. Chunk types are shown in Table 3

5.2 Interpreting Spatial Language

5.2.1 Modeling Spatial Relationships

When people communicate with each other about spatially oriented tasks, they typically choose relative spatial references rather than precise quantitative terms, e.g., *the eyeglasses are in the living room on the table in front of the couch* [52]. Although natural for people, it is not easy for a robot to follow such a description. Providing robots with the ability to understand and communicate with these spatial references has great potential for creating a more natural approach for human-robot interaction [49]. In previous work [26], the histogram of forces (HoF) [53] is used to model spatial relationships and, thus, provide a method for interpreting spatial language references in human-robot interaction. The HoF can quantize the spatial relationship between two crisp or fuzzy objects by providing weights of different directions [53]. By providing a quantitative model of these relationships, the HoF can be used to translate qualitative spatial relationships into robot instructions.

5.2.2 Modeling the Fetch Task

The environment of the fetch task investigated here is a two-room home with a hallway between the rooms, which is modeled after the physical lab space. The robot stands at the end of the hallway to wait for instruction before starting the task. To simplify the fetch task, the process is divided into three sub-tasks: (1) determine the target room and move to enter the room through the doorway, (2) move within the room to the place where the target object is located by following the spatial description, (3) search for the object around the goal location as specified in the spatial description. In the fetch task, the target objects are assumed to be on the surface of furniture items so that

the robot does not need to search inside the furniture. The robot uses its local perception for navigation in this task.

5.2.3 Reference-Direction-Target Model

Because the robot has no prior information about the furniture and object placement inside the room, it needs to use the information provided by the spatial language description. Therefore, a Reference-Direction-Target (RDT) model is proposed to translate the spatial description into navigation information that can be used directly as a navigation command for the robot. The RDT model includes three parts: Reference, Direction and Target. These components together comprise a RDT node. The three RDT components represent all types of navigation instructions a robot may need in an indoor environment.

Reference refers to the object or structure that is used in a relation. In the RDT model, the reference also provides a label that tells the robot what kind of behavior it should perform. The behavior can either be a basic action like spinning and moving forward or a complex action like searching or following a path. Several types of references are used in the fetch task, as described below.

NONE – No reference object is mentioned in the instruction, i.e., the robot action is not dependent on the objects around the robot. For example, "*turn right*" or "*go forward*". There is no target object for this reference type.

ROOM – The room is used as a reference for navigation, e.g., *"move halfway in*" or "*to the left of the room*". The Direction component determines which part of the room is the destination. Using a sense of direction, e.g. from a compass, and prior knowledge of the room structure, the robot can move to the target area and search for the target object.

It is assumed that the robot has a map of the environment structure, but it does not know where the furniture items are located within the rooms of the structure.

WALL – A wall is used as the reference, e.g., "*to the back wall*". The robot navigates close to a wall and may search for the target object.

ROBOT – The robot itself is used as the reference. The reference object does not directly appear in the description, but rather ego-centric references are used, e.g., "*to the left*" or "*in front of you*". These mean "to the left of the robot" or "in front of the robot" which uses the robot's local reference frame.

FURNITURE – A furniture item is used as the reference object. The reference frame that defines the direction differs for different types of furniture. These have been defined based on the results of spatial language experiments. For example, "in front of the couch" is typically defined using the intrinsic frame of the couch. The front side refers to the seating side of the couch independent of viewing angle.

Direction represents the position relationship between objects and tells the robot where it should move to search for the target. For the different references described above, the meaning of a direction is different. For NONE, the direction tells the robot the angle for motion. For other reference types, direction shows where the robot should move, relative to the specified reference. For different types of navigation instructions, the reference frame for direction may be defined differently [50]. The direction may not be defined by the intrinsic reference of the reference object. For example in the Couch reference shown in Figure 30, the directions are inversed to the couch's intrinsic reference because the current on is more likely to match human habit. The directions used in robot fetch commands include: front, left, right, back, central, side, and between. Table

4 shows the combinations of references and their corresponding directions and Figure 30 shows the pose relationship of them.

The Target component indicates the target furniture in the navigation instruction or can also be used as the reference of the target object. If there is not a target in a RDT node, the target is defaulted to be a table type furniture item. This is a natural assumption for the fetch task, as people usually put small objects on table-like furniture.

Table 4.References and Corresponding Directions

Reference	Category	Corresponding Direction
NONE	Dynamic	Front, Left, Right
ROOM	Dynamic	Left, Right, Back
WALL	Dynamic	Left, Right, Back, Side
ROBOT	Static	Front, Left, Right, Back
FURNITURE	Static	Front, Left, Right, Back, Between

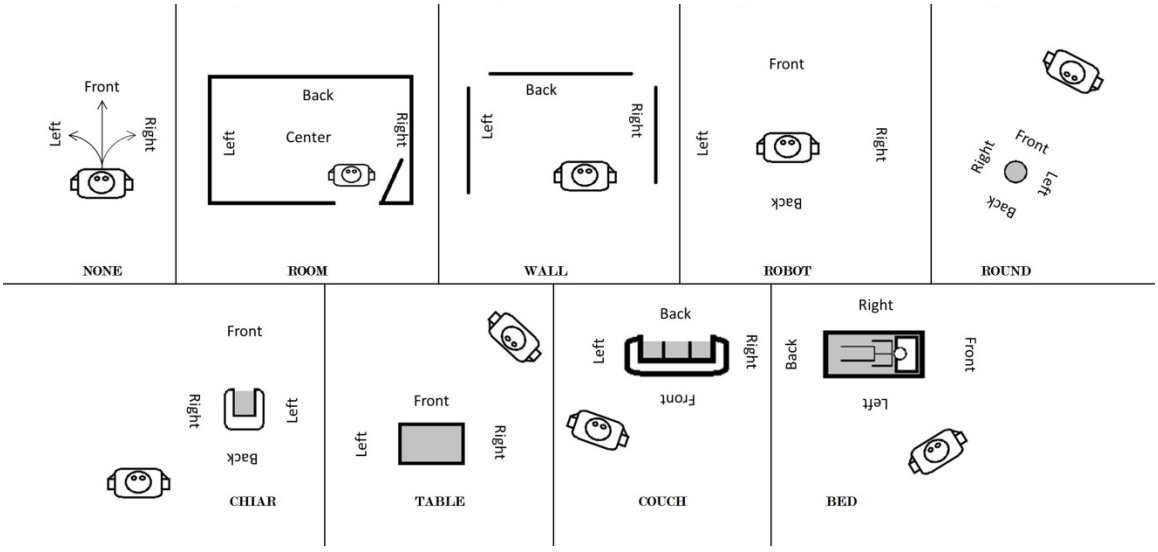


Figure 30 Reference Direction Map

5.2.4 Translating Chunks into Navigation Commands

For the fetch task, a dictionary of spatial language phrases is manually built for translating the words and phrases in the chunks to navigation commands that can be understood by the robot. The knowledge to build this dictionary is based on some human-

robot spatial language experiments [49][50][51]. From the 3 parts of the RDT model described above, the information also has three classes: (1) target room, (2) inside-room navigation command, and (3) target object. They can be extracted by searching the words, phrases and their corresponding tags in the chunks from the dictionary of spatial language phrases. In the fetch task, the target room is extracted directly from the ORMTP and ORMRP chunks, and target object is extracted from the OBTP chunk. FURRP chunks, FURTP chunks and IRMRP chunks provide navigation instructions within rooms.

The translation is a traversal process along the leaves of the parse tree. For the example shown in Figure 29, the parse tree is converted to a robot behavior model by 3 steps.

1) Preorder traverse the parse tree. List the phrases of the corresponding chunks sequentially. The phrases are: (1) OBTP: "*the fork is in*", (2) ORMTP: "*the living room*", (3) ORMRP: "*on the right*", (4) FURTP: "*on the table*", (5) IRMRP: "*to the right side*". (6) FURRP: "*behind the couch*".
2) Extract room information and target object information from ORMTP, ORMRP and OBTP using the dictionary. The room is *bedroom* and the target object is the *monitor*.
3) Generate navigation instructions by building the RDT nodes. The result is "robot-left-table". In a complex command, there may be more than one phrase that can be translated to a RDT node. Connect them sequentially to build a RDT chain (Figure 31).

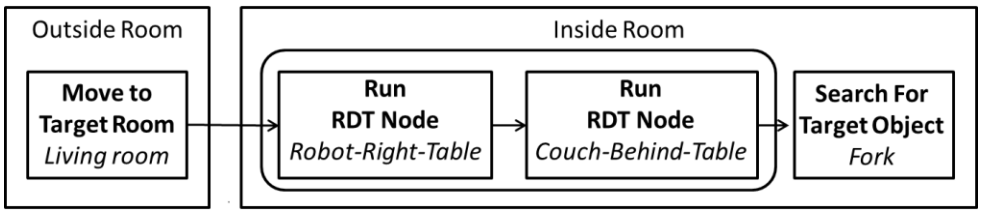


Figure 31 RDT Chain Model for the spatial description in Figure 1.

5.2.5 Robot Behavior Model

After translating the spatial descriptions into robot commands, the robot behavior model can be instantiated, and the robot is then ready to execute the command. The robot behavior model has a two-tier structure. The higher tier is a global model of the whole task which is the 3-subtask model. The lower tier is the robot actions as lead by the RDT nodes. The dynamic instructions and static instructions have different strategies which can be represented by state machines. The dynamic model is not as dependent on perception and recognition abilities but rather relies on sequential movement commands. However, the static command strategy requires the robot to search and recognize the reference and target items. The behavior model used for static commands is shown in Figure 32.

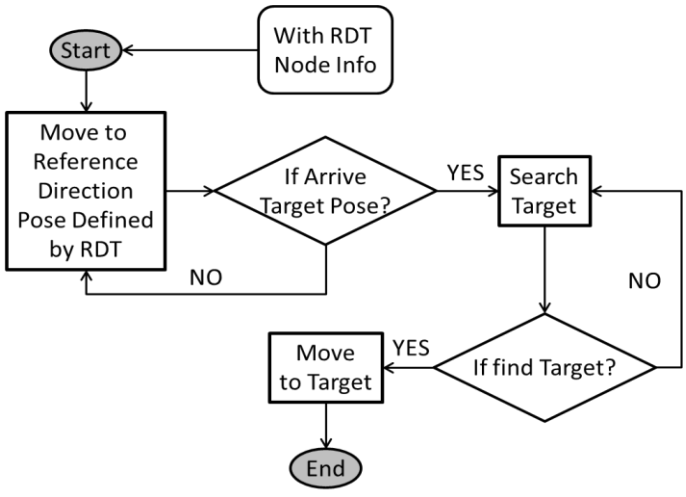


Figure 32 Robot Behavior Model in an RDT node

Chapter 6 Experiments and Results

This chapter discusses the experiments designed for testing the algorithms introduced in chapter 4 and chapter 5. In Section 6.1, 6.2 and 6.3, three experiments are run to verify the performance of the methods in chapter 4. The first one is a static experiment of furniture recognition performance in category and instance. The second one is detection of furniture pose including position and orientation. The third experiment is the behavior of the robot which includes navigation of the robot to a place that can improve recognition performance. Section 6.4 tests the performance of the robot interpretation by running a robot simulation experiment.

6.1 Furniture Recognition

To reduce the effect of any other factors that may disturb experimental results, a static experiment is run for furniture recognition. The robot acts like a stationary platform in this experiment. The dataset for the recognition experiment includes 12 kinds of furniture items which are different in size, color and shape. These furniture items are used to build up an indoor environment for the human-robot interaction experiments.

6.1.1 Database and Procedure

Building a Model of Furniture

As discussed in chapter 4, the furniture category model is built based on fuzzy logic. The method to define the membership functions of values for each linguistic variable used K-means clustering on the training data. Building this model consisted of two steps:

1) Use the data of each feature for K-mean clustering. Because the data is one-dimensional, it can also be seen as making histograms of data. Then, after finding

the centroid of each cluster, the membership functions are defined for the linguistic variables.

2) Find the differences in the different categories of furniture samples. By using clustering results and experience, five categories are determined--*small table, large table, chair, couch and bed*.

Table 5 and Table 6 show the linguistic variables and corresponding membership functions that are used as the model definition of each category of furniture. There are eight instances used for building this model. The values and membership functions of each linguistic variable are shown in the following table.

Table 5 Membership Functions for the Linguistic Rules

Size (s) (dm^2)	**Small**	$\begin{cases} if\ s < 25, f = 1 \\ if\ s \geq 25\ and\ s < 30, f = \dfrac{30 - s}{5} \\ else\ f = 0 \end{cases}$
	Middle	$\begin{cases} if\ s < 40, f = \dfrac{s}{40} \\ if\ s \geq 40\ and\ s < 70, f = 1 \\ if\ s \geq 70\ and\ s < 80, f = \dfrac{80 - s}{10} \\ else\ f = 0 \end{cases}$
	Large	$\begin{cases} if\ s < 110, f = \dfrac{s}{110} \\ if\ s \geq 110\ and\ s < 220, f = 1 \\ if\ s \geq 220\ and\ s < 230, f = \dfrac{230 - s}{10} \\ else\ f = 0 \end{cases}$
Plane Height (p) (Meter)	**Low**	$\begin{cases} if\ p < 0.55, f = 1 \\ if\ p \geq 0.55\ and\ p < 0.65, f = \dfrac{0.65 - p}{0.10} \\ else, f = 0 \end{cases}$
	High	$\begin{cases} if\ p \geq 0.65\ and\ p < 1.2, f = 1 \\ if\ p \geq 0.55\ and\ p < 0.65, f = \dfrac{p - 0.55}{0.10} \\ if\ p \geq 1.2\ and\ p < 1.3, f = \dfrac{1.3 - p}{0.1} \\ else, f = 0 \end{cases}$
Shape (c) (Chair Shape Likelihood Rate)	**Table**	$\begin{cases} if\ c < 0.25, f = 1 \\ if\ c \geq 0.25\ and\ c < 0.75, f = \dfrac{0.75 - c}{0.50} \\ else, f = 0 \end{cases}$
	Chair	$\begin{cases} if\ c \geq 0.50\ and\ c < 0.75, f = \dfrac{0.75 - c}{0.25} \\ if\ c \geq 0.75, f = 1 \\ else, f = 0 \end{cases}$

Figure 33 shows the values of each linguistic variable. The units used of each abscissa axis from top to bottom are dm^2, *meter* and non-unit.

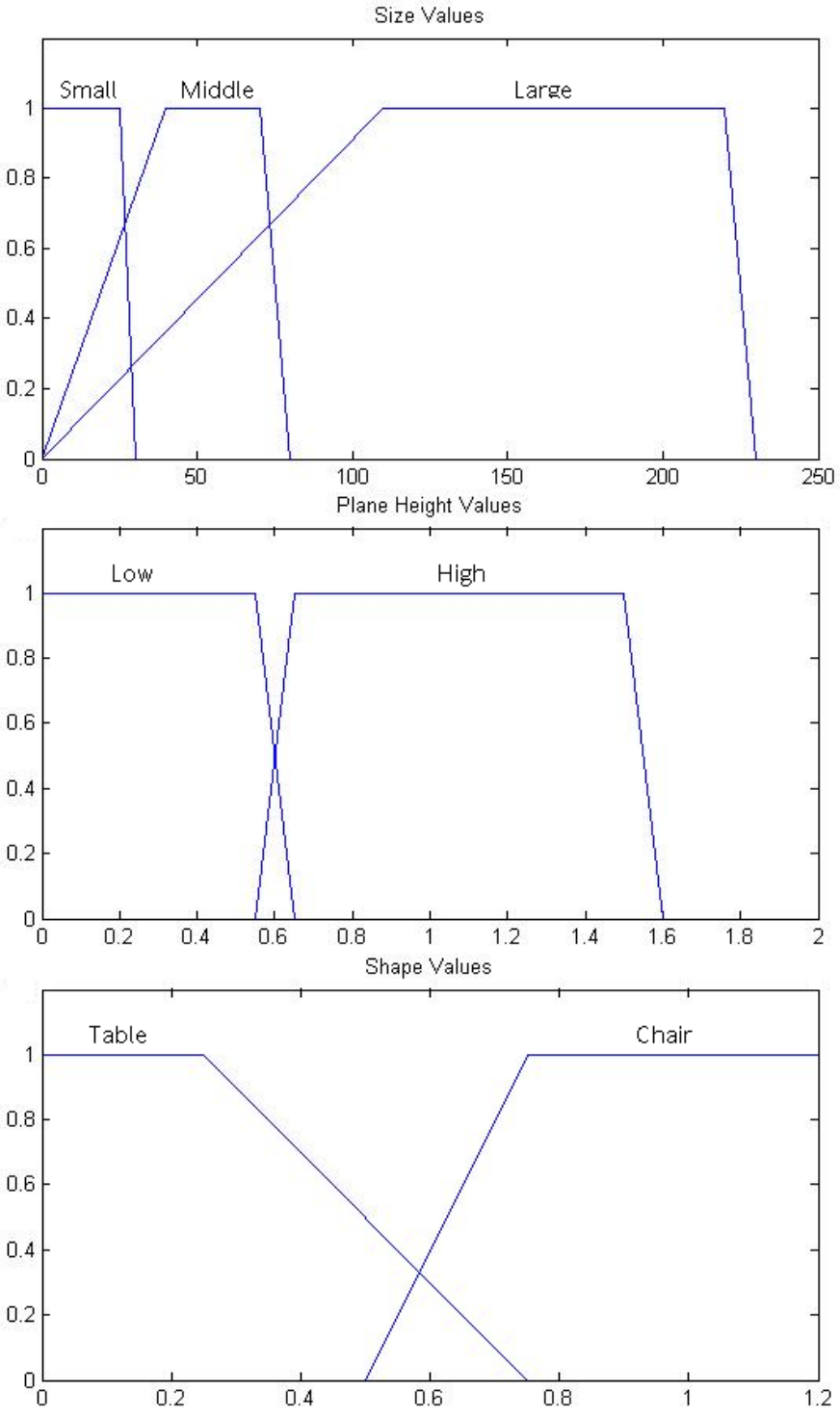


Figure 33 Linguistic Variable Values

The fuzzy rules that determine the category of a sample is shown in Table 4. There are five fuzzy logic rules that represent five categories of daily life furniture.

Table 6 Fuzzy Rules and Furniture types

Fuzzy Rules	**Category**	**Furniture instances included**
If Size is SMALL or MIDDLE and Plane Height is LOW and Shape is TABLE, THEN Category Name is SMALL TABLE.	1 – Small Table	Round Table Hexagon Table Coffee Table
If Size is SMALL or MIDDLE and Plane Height is LOW and Shape is CHAIR, THEN Category Name is CHAIR.	2 – Chair	Blue Chair Wood Chair
If Size is LARGE and Plane Height is HIGH and Shape is TABLE, THEN Category Name is Large Table.	3 - Large Table	Dinner Table
If Size is LARGE and Plane Height is LOW and Shape is CHAIR, THEN Category Name is COUCH.	4 - Couch	Couch
If Size is LARGE and Plane Height is LOW and Shape is Table, THEN Category Name is BED.	5 - Bed	Bed

Dataset

The dataset used for the recognition experiment includes 228 RGB-Depth images taken with the Kinect for eight furniture items. The numbers of samples for each instance are shown in Table 7.

Table 7 Dataset

Instance Name	Category	Total Number	Training Samples Number	Testing Samples Number
Round Table	Small Table	32	8	32
Blue Chair	Chair	24	8	24
Hexagon Table	Small Table	36	8	36
Wood Chair	Chair	24	8	24
Coffee Table	Small Table	32	8	32
Dinner Table	Large Table	32	8	32
Couch	Couch	24	8	24
Bed	Bed	24	8	24

The training images and testing images are taken from different distances and directions. These distances and directions cover all the positions where a sample could be positioned in the Kinect detection scale. For each instance, it took eight RGB-Depth images as training images. The distance of the robot from the furniture items was about 1.5 meters when doing the tests and the directions were 0^o, 45^o, 90^o, 135^o, 180^o, 225^o, 270^o and 315^o which were the same as the training. The distance of the sample from the camera was selected so that it could give the robot a complete view of the furniture samples with as much detail as possible. That is, the training samples were selected so that the furniture model could be as accurate as possible. The testing samples were selected from different distances and directions. They were collected from a distance from 1 m to 4 m which is the maximum distance realistically permitted by the Kinect camera. In this experiment, all the RGB-Depth images (including training samples) were selected as the testing samples.

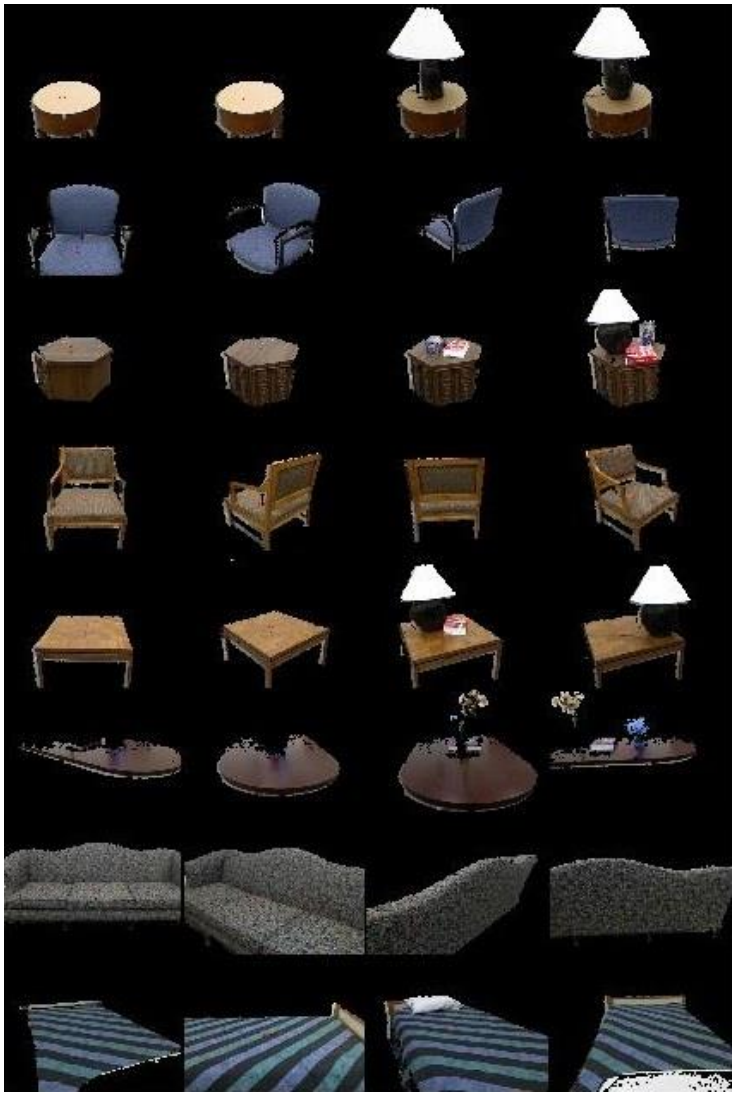


Figure 34 Test Samples

6.1.2 Result

Category Recognition Experiment

In this experiment, eight samples are used for training and all the samples (including training samples) were used for testing. The results of category recognition are shown in Table 8 and Table 9.

Table 8 Category Recognition Results

Category	Furniture items included	Data	Accuracy	Accuracy with clutter on top (only for table)
Small Table	Round Table	32	100%	100%
	Hexagon Table	36	66.7%	66.7%
	Coffee Table	32	100%	100%
Chair	Blue Chair	24	100%	N/A
	Wood Chair	24	100%	N/A
Large Table	Dinner Table	32	87.5%	75%
Couch	Couch	24	50%	N/A
Bed	Bed	24	62.5	N/A

Table 9 Category Recognition Confusion Matrix

Instance	Small Table	Chair	Large Table	Couch	Bed
Round Table (ST)	32				
Blue Chair (Chr)		24			
Hexagon Table (ST)	24	12	24		
Wood Chair (Chr)		24			
Coffee Table (ST)	32				
Dinner Table (LT)			28	4	
Couch (Cch)		12	12		
Bed (Bd)			9		15

Instance Recognition Experiment

The result of instance recognition is shown in Table 10, Table 11 and Table 12.

Table 10 Instance Recognition Results

Instance	Data	Accuracy	Accuracy with clutter on top (only for table)
Round Table	32	100%	100%
Blue Chair	24	100%	N/A
Hexagon Table	36	58.3%	50%
Wood Chair	24	91.6%	N/A
Coffee Table	32	90.6 %	87.5
Dinner Table	32	87.5%	75%
Couch	24	50%	N/A
Bed	24	62.5	N/A

Table 11 Confusion Matrix of Instance Recognition in Small Table

Ground Truth	Round Table	Hexagon Table	Coffee Table
Round Table	32		
Hexagon Table		21	3
Coffee Table		3	29

Table 12 Confusion Matrix of Instance Recognition in Chair

Ground Truth	Blue Chair	Wood Chair
Blue Chair	24	
Wood Chair	2	22

6.2 Furniture Orientation Detection

6.2.1 Procedure

A specially designed dataset that had 144 RGB-Depth images was used for the orientation detection experiment. It included six furniture instances and 24 samples for each of them. The 24 images included eight directions and three distances. Samples with clutter on top are not used in these tests. In the test, the directions of the furniture items

were assigned with 0^o, 45^o, 90^o, 135^o, 180^o, 225^o, 270^o, 315^o. The 3 directions were set as "near", "middle" and "far". The "near" distance was around 0.9m ~1.2m which meant that the robot was too close to the furniture sample. The middle range was around 1.8m~2.2m and the "far" was 3.2m~3.5m. The result of the furniture orientation test was shown by measuring the difference of the value to the ground truth. For Round Table and Hexagon Table their orientations are not computed because their round shape and orientation are not defined by its own coordinate. For the other table shape furniture items, it is assumed that they are facing toward the robot because they are symmetrical to the long axis so that they do not have orientation from 180 to 315 degree. Those positions are replaced by the data of 0 to 135 degree but with clutter on top.

6.2.2 Results

The results are shown from Table 13, Table 14 and Table 15, as the absolute difference between the ground truth orientation and the estimated orientation.

Table 13 Result of Furniture Orientation Experiment (Degree) (Near)

Instance	**0**	**45**	**90**	**135**	**180/0 clutter**	**225/45 clutter**	**270/90 clutter**	**315/135 clutter**
Blue Chair	7	11	76	9	9	3	0	3
Wood Chair	8	10	46	8	5	4	0	3
Coffee Table	3	4	12	6	2	5	7	9
Dinner Table	3	4	12	6	8	6	11	3
Couch	21	74	143	65	27	15	0	12
Bed	3	4	12	6	×	×	×	×

Table 14 Result of Furniture Orientation Experiment (Degree) (Middle)

Instance	**0**	**45**	**90**	**135**	**180/0 clutter**	**225/45 clutter**	**270/90 clutter**	**315/135 clutter**
Blue Chair	1	8	45	9	4	1	0	2
Wood Chair	5	6	33	5	4	2	0	1
Coffee Table	1	2	5	6	2	8	3	7
Dinner Table	2	5	6	4	9	4	6	11
Couch	28	65	112	55	62	18	0	13
Bed	2	1	7	4	×	×	×	×

Table 15 Result of Furniture Orientation Experiment (Degree) (Far)

Instance	0	45	90	135	180/0 clutter	225/45 clutter	270/90 clutter	315/135 clutter
Blue Chair	3	9	65	10	5	2	4	1
Wood Chair	7	18	24	11	15	11	3	4
Coffee Table	2	5	8	9	4	8	9	8
Dinner Table	5	11	9	7	6	5	3	7
Couch	30	59	132	47	72	29	6	19
Bed	6	2	8	7	×	×	×	×

6.3 Furniture Searching

6.3.1 Procedure

The robot experiment consisted of 24 trials for all eight kinds of furniture items. There are three kinds of furniture classified by shapes. Different starting states are set separately to them. For round shape furniture, there were two trials from different places where the confidence was low. The starting distance for running the trial was 3.5 m. The round shape tables does not have orientation so they are settled at the place where cannot be completely taken by camera. One is on the left side and the other is on the right side. The table shape furniture samples are placed at 3.5 m far with orientations were 0, 45 and 90 degree in the three trials when the robot start. The chair shape furniture samples are placed at 3.5 m far with orientations were 0, 90 and 180 degree in the three trials when start the robot. The maps below show the typical starting direction for each furniture piece. The condition that a trial is "*successful*" is that the robot moves to the optimal position in three minutes and gives the accurate instance recognition result with the extrinsic confidence > 0.5 and the intrinsic confidence > 0.75.

6.3.2 Result

The results of the trials are shown in Table 16, Table 17 and Table 18.

Table 16 Result of Robot Action Experiment (Round Shape Furniture)

Instance	**Place 1 (Incomplete View Left, 3.5M)**	**Place 2 (Incomplete View Right, 3.5M)**
Round Table	Y	Y
Hexagon Table	Y	Y

Table 17 Result of Robot Action Experiment (Chair Shape Furniture)

Instance	**Place 1 (3.5M, 0^o)**	**Place 2 (3M, 90^o)**	**Place 3 (3M, 180^o)**
Blue Chair	Y	Y	Y
Wood Chair	Y	Y	Y
Couch	Y	N	N

Table 18 Result of Robot Action Experiment (Table Shape Furniture)

Instance	**Place 1 (3.5M, 0^o)**	**Place 2 (3.5M, 45^o)**	**Place 3 (90^o)**
Coffee Table	Y	Y	×
Dinner Table	Y	Y	Y
Bed	Y	N	Y

Y means the robot successfully reached a position properly aligned with the furniture sample in the center view and reached the goal where extrinsic confidence > 0.5 and the intrinsic confidence > 0.75. N means the robot failed to finish the task and × means no experiment in this setting. The figure below shows some photos taken while running the experiment.

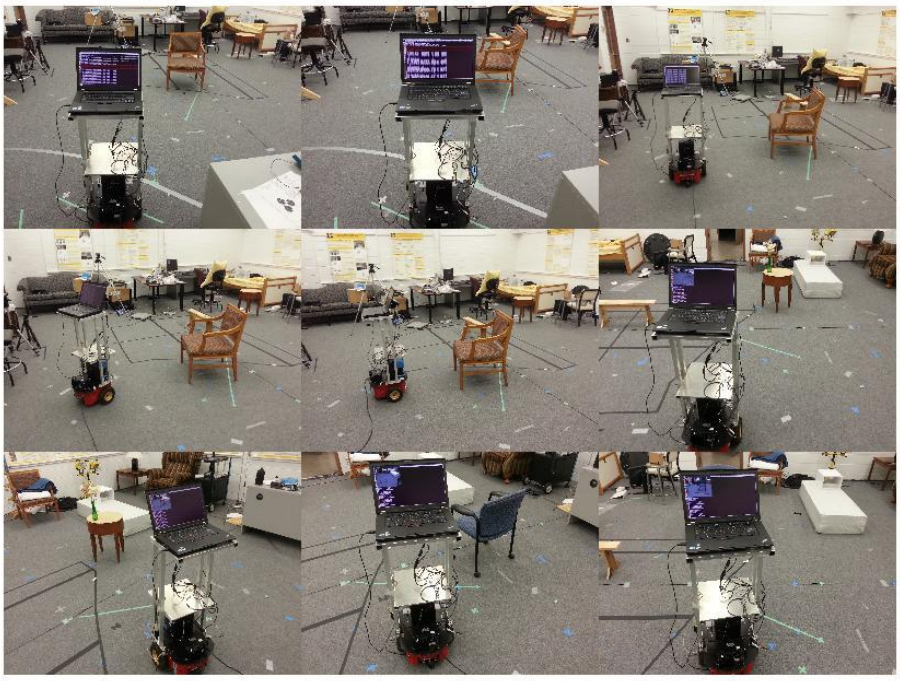


Figure 35 Furniture Searching Experiment

6.4 Robot Command Interpretation Experiment

The methods described in chapter 5 have been evaluated experimentally by executing robot spatial descriptions in a simulation environment and comparing the results to human performance (also in a simulation environment) using the same descriptions.

6.4.1 Simulation Environment and Experiment Design

Microsoft Robotics Studio is used for the simulation experiment environment. The virtual environment is a two-room home with a hallway between rooms, as shown in Figure 36. The robot starts at the back of the hallway. The robot used in this experiment is a differential drive Pioneer 3DX mobile robot with a Kinect mounted at a height of 1m. For the physical robot, RGB and depth images are used to recognize the furniture and small objects inside the room [50]. For the simulation experiment, the robot uses the

Kinect viewing cone and distance to determine when perception is likely to succeed. That is, if a furniture item or small object is in the viewing cone and at a close enough distance, the robot assumes that perception is successful. To simplify the problem, the viewing angle is not considered in the experiment. This method is used to approximate the robot's performance in a physical setting, which will be tested in future work. It also serves to test the spatial language methods independent of the perceptual challenges.

There are 6 scenarios in the experiment. Each has a unique target object, which are fork, glasses case, laptop, monitor, statue, and mug. In each scenario, the furniture positions are fixed while the object placement is different. Figure 36 shows the furniture and object locations in the scene. There are 149 template spatial language descriptions for the 6 robot fetch scenarios. The descriptions are converted to tree structures and translated to robot commands as described in section 5.1. In this experiment, the descriptions have been manually chunked so that they are reliable as ground truth for future NLP work.

For the human data, 48 undergraduates are asked to navigate through the virtual environment to arrive at a target specified in a spatial description. Each participant performed 12 trials, each with a template description; 576 trials were tested in total which were taken from the 149 unique spatial descriptions. Target objects were specified in the spatial descriptions, and subjects navigated until they reached the target location. For the robot, the same 149 descriptions were used; however, the target object was not included in the descriptions so that the robot had to determine the target based on the description structure and content. Each robot trial ended when the robot arrived at the position of the target furniture (as determined through the robot's reasoning processes) and turned its

viewing cone on the target furniture item, i.e., the furniture that held the small target object.

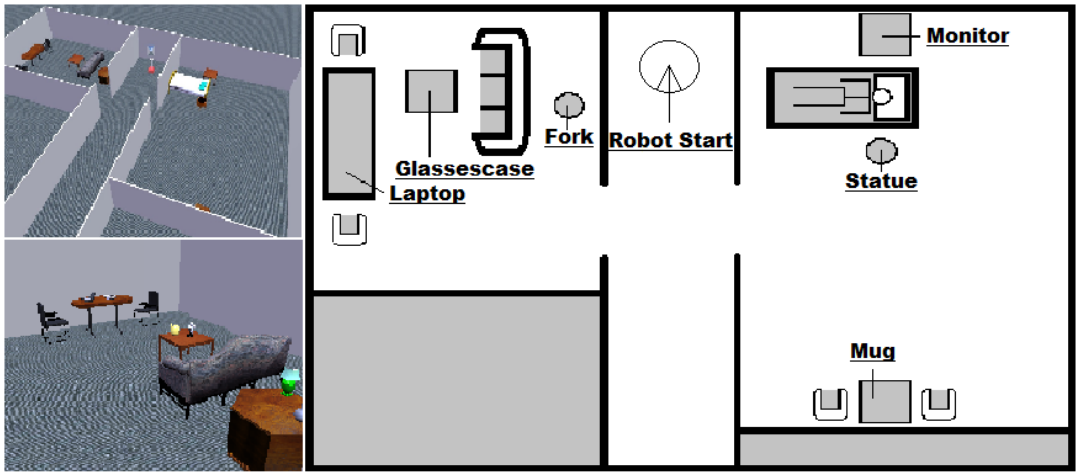


Figure 36 Simulation Experiment Environment

6.4.2 Result

In the experiment the robot state in each frame for each trial is recorded (Figure 37) and snapshots of the robot's sensor are taken at the end of the trials (Figure 38). To analyze the results of the robot experiment, several metrics are tested. Significant results were found for the following metrics: path length, percent spin time, percent stop time, and success rate. Path metrics are generated from the robot state record and compared to the human performance using the same metrics. The success rate is analyzed for the robot only, as all paths in the human subject data ended with the specified target object. To determine whether the trial was successful, it is checked whether the target object was in view in the sensor snapshot taken at the end of the trial.

The template description can be classified as "*How*" and "*Where*" types depending on the instructions. "*How*" descriptions were overwhelmingly dynamic, following a sequential, direction-like structure such as [Move] + [Direction] + [Move] + [Direction] + [Goal]. For example, "*Go forward, turn left, go straight, and you'll find the target.*

"*Where*" descriptions were more split, with a significant number of static descriptions, following a structure such as [Target] + [In] + [Room] + [Room Reference]. For example, "*The book was in the living room, against the back wall*."

The templates were also generated for different landmark conditions. The No-Landmark templates were unaltered. Goal-Landmark templates included a description of the table where the target object was located. For example, the Older-How-Robot description for the glasses case would read, "*Take a right through the door. Go forward and turn right and you'll find the glasses case on the table.*" Path-landmark templates included a description of a furniture item in the environment in addition to the table where the target was located. For example, a path-landmark description for the glasses case would read, "*Take a right through the door. Go forward and turn right and you'll find the glasses case on the table behind the couch.*"

Table 19 to Table 23 show the significant results of the experiment based on an items analysis using the 149 unique template descriptions. Mean values and standard deviations are included for each path metric. To better compare robot and human path metrics, only robot trials that were successful in determining the correct target are included in analysis. There are 123 successful robot trials out of the total 149 unique descriptions tested. The robot success rates are then analyzed for the how/where and different landmark test conditions. The overall success rate for the robot was 85%.

```
0.001678, 10.997830, 269.870789, 0.000000, 0.000000, 23245412, -1, -1
0.001679, 10.997820, 269.870789, 0.179898, 0.000509, 23245443, -1, -1
0.001672, 10.992640, 269.872803, 0.179900, 0.000499, 23245474, -1, -1
0.001666, 10.989810, 269.873199, 0.179901, 0.000497, 23245506, -1, -1
0.001654, 10.984500, 269.874298, 0.179901, 0.000493, 23245537, -1, -1
0.001642, 10.979010, 269.875305, 0.179902, 0.000488, 23245568, -1, -1
```

Figure 37 Robot State Log

Figure 38 Some Snapshots of Robot Local View When a trial finished

Table 19 Path Length for Human vs. Robots (Meter)

	Landmark	Mean	SD
Human	**Goal**	9.71	1.95
	None	9.82	2.26
	Path	9.22	2.08
	Total	9.54	2.10
Robot	**Goal**	8.66	2.30
	None	8.89	2.39
	Path	7.58	2.03
	Total	8.28	2.28

Table 20 Path Length for How vs. Where (Meter) for Robot Only

	Mean	SD
How	9.30	0.21
Where	8.42	0.28

Table 21 Percent Spin Time for Human vs. Robot (%)

	Landmark	Mean	SD
Human	**Goal**	17.82	6.78
	None	17.97	4.03
	Path	18.12	6.79
	Total	17.98	6.08
Robot	**Goal**	6.68	7.70
	None	6.20	7.54
	Path	28.24	25.69
	Total	15.31	20.36

Table 22 Percent Stop Time for Human vs. Robot (%)

	Type	Mean	SD
Human	**How**	10.25	4.95
	Where	7.43	4.41
Robot	**How**	0.22	0.42
	Where	1.15	7.14

Table 23 Successful Rate Result (%) for Robot Only

Types and Landmarks	**How vs. Where**		**Goal vs. Path vs. None**		
	How	**Where**	**Goal**	**Path**	**None**
Successful Rate	89.4	73.4	89.5	40.0	98.0

Chapter 7 Discussion on Results

This chapter discusses the results of the four experiments described in Chapter 6. The experiments in Chapter 6 showed the performance of the robot on furniture recognition, furniture orientation detection and robot furniture searching and ….

Experiment 1: Furniture Recognition

The first experiment showed the result of recognition by using the Kinect camera at different distances and in different orientations. From the results in chapter 6, it can be concluded that the recognition results were affected by both furniture category and pose.

The following four conclusions are made:

1) The chair shaped furniture items have nearly the same accuracy in recognition as table shaped furniture items.
2) The larger-sized furniture items were more difficult to recognize than smaller sized furniture items in either shape.
3) For the chair shaped furniture items, it was easier to make accurate decisions when they were facing the Kinect camera which means the orientation interval favored between 180^{o} and 360^{o}. Accurate navigation was almost impossible when the furniture sample (especially chair shape) had its back to the camera.
4) For the table shaped furniture items, it was easier to make accurate decisions when there was no clutter on the surface. However, clutter did not have a great effect.

Experiment 2: Furniture Orientation Detection

The results obtained in chapter 6 show the factors that affect performance of the orientation detection.

In Table 13, the following three conclusions were obtained:

1) For chair shaped furniture items, it is much easier for the system to make accurate decisions when the real orientation ranges from 180^{o} to 360^{o} (0^{o}).
2) The orientation of chair shaped furniture items was more difficult to detect than table shaped furniture items.
3) Clutter does not significantly affect the furniture orientation detection results of table shaped furniture items when using the method chosen for this study's experiments.

Experiment 3: Furniture Searching

This chapter discusses results obtained from the robot searching experiment. The following three conclusions describe the effect caused by the different factors in furniture detection.

1) The detection strategy robustly gave low confidence scores which triggered robot action when the robot was not in a good view point for recognition.
2) The recognition of chair shaped furniture items does not lend itself to high confidence when chairs are placed with their backs toward the robot Kinect camera, and this recognition hindrance made it difficult for the robot go to the right place when working with chair shaped furniture items.
3) Large size furniture item were not easily recognized when using the robot behavior.

Experiment 4: Robot Fetch Simulation

For experiment 4, several observations can be made from the experimental results. From the path length metric, it could be found that the robot has a shorter path than the human subjects in all command types and all landmark types. Thus, the approach allows the robot to achieve a more efficient path than the humans. It is also observed that the "Where" type command results in a shorter path length than the "How" type command across all robot and human trials.

Considering percent spin time, the robot takes less spin time in the Goal and None landmark cases than the humans but considerably more spin time than humans in the Path landmark cases. This demonstrates that giving the robot more information may not necessarily help.

The percent stop time results show that the robot spends much less stop time compared to the human trials in all command types and landmark cases, because the robot does not need to stop and hesitate on the next step.

When looking at the success rate results for the robot, "How" type commands have a higher success rate than "Where" type commands. Also, the commands with "Path" information show a much lower success rate when compared to other landmark cases. Several of the "Path" landmark cases were intentionally designed to include an ambiguous phrase, in an effort to observe how the human subjects would handle such situations. For example, the region "*in front of the couch*" might refer to the seating side of the couch if the couch's intrinsic frame is used, or it might refer to the opposite side depending on the robot position and a different reference frame being used. In many of these ambiguous cases, the robot assumed an intrinsic reference frame by default, and got

it wrong, because it was constrained from using any perceptual abilities to confirm the location as a person would. In spite of these ambiguities, the overall success rate was 85%, which indicates that performance is likely to improve if additional perceptual and reasoning capabilities are included.

Chapter 8 Conclusions and Perspectives

Several achievements were obtained in this project, which include the following:

1) Designed an intelligent robot that uses the Microsoft Kinect as a vision sensor for home-like scenarios.
2) Developed and tested a fast furniture recognition approach which uses both color and geometry information as features. This method had good performance even when there was clutter on top of furniture items.
3) Developed and tested a furniture orientation detection approach.
4) Developed and tested an approach that can improve the robot's recognition performance when recognition confidence is low.
5) Developed a framework to interpret natural spatial language command and tested it in a robot simulation fetch task.

The key difficulties that needed to be conquered in this project included the following:

1) First, it is challenging to calibrate the Kinect camera so that the RGB-Depth image can be properly converted to RGB-point cloud data.
2) Second, selecting appropriate features in recognition is difficult when furniture items are similar in shape or color although the Microsoft Kinect can capture color and depth information. Moreover, the features should not be affected by clutter on furniture items.
3) It is challenging to estimate the orientation of chair shaped furniture items. Originally, only the RGB part of the image was used for this task. However, that method was so unreliable that finally, only depth information was used for furniture orientation detection.

4) In the spatial language translation part, the language model needs a lot of data for training, and the training dataset must be manually edited.

In the results chapter, the experiments are presented in the order discussed in Chapter 4. The fuzzy logic system parameters were selected by computing statistics on all the training data. The best parameters were used in the test with several demonstrations. The strategy for the robot action to improve confidence and performance of recognition were built from recognition test data. It can be concluded that the detection performance is strongly dependent not only on intrinsic factors but also on extrinsic factors. It is verified that recognition quality can be improved by not only the recognition algorithm but also the reaction of the robot to the extrinsic environment. The spatial language parsing experiment shows the validity of the RDT model proposed in the thesis, which can be further developed later.

Although some progress has been made in these tasks, there is still room for improvement and some problems that need to be solved. The problems include:

1) The training process is too complex. It takes a lot of time to train a robot which means the adaptability of the robot to a new environment is weak. An online training algorithm is needed to reduce the work of training and improve the adaptability of the robot to the new environment.
2) Except for the plane, the other parts of the furniture sample, especially the part under the plane, are not used in recognition. Experiments focusing on other furniture parts led to inaccurate results, i.e., unreliable data from samples.

3) The orientation results of furniture samples were falsely returned by the system when results came from the same directions where the furniture samples were not well detected. The system need to be improved to deal with these situations.
4) In the robot furniture searching experiment, the robot sometimes failed in its task. A more elaborate robot behavior model is needed.
5) The spatial language model needs a larger corpus of human-robot spatial language commands for training and testing in more complex environments.

References

[1]. Roth, Peter M., and Martin Winter. "Survey of appearance-based methods for object recognition." Inst. for Computer Graphics and Vision, Graz University of Technology, Austria, Technical Report ICGTR0108 (ICG-TR-01/08) (2008).

[2]. DeSouza, Guilherme N., and Avinash C. Kak. "Vision for mobile robot navigation: A survey." *Pattern Analysis and Machine Intelligence, IEEE Transactions on* 24.2 (2002): 237-267.

[3]. Dodds, Ricardo, et al. "Benchmarks for robotic soccer vision." RoboCup 2011: Robot Soccer World Cup XV (2012): 427-439.

[4]. Robocup Rules, http://www.robocup.org

[5]. Se, Stephen, David Lowe, and Jim Little. "Mobile robot localization and mapping with uncertainty using scale-invariant visual landmarks." The international Journal of robotics Research 21.8 (2002): 735-758.

[6]. Röfer, Thomas. "Region-based segmentation with ambiguous color classes and 2-D motion compensation." RoboCup 2007: Robot Soccer World Cup XI(2008): 369-376.

[7]. Kitano, Hiroaki, et al. "RoboCup: A challenge problem for AI." AI magazine 18.1 (1997): 73.

[8]. Impedovo, S., L. Ottaviano, and S. Occhinegro. "Optical character recognition—a survey." International Journal of Pattern Recognition and Artificial Intelligence5.01n02 (1991): 1-24.

[9]. Due Trier, Øivind, Anil K. Jain, and Torfinn Taxt. "Feature extraction methods for character recognition-a survey." Pattern recognition 29.4 (1996): 641-662.

[10]. Heisele, Bernd, Purdy Ho, and Tomaso Poggio. "Face recognition with support vector machines: Global versus component-based approach." Computer Vision, 2001. ICCV 2001. Proceedings. Eighth IEEE International Conference on. Vol. 2. IEEE, 2001.

[11]. Kohonen, Teuvo. "Self-organization and associative memory." Self-Organization and Associative Memory, 100 figs. XV, 312 pages.. Springer-Verlag Berlin Heidelberg New York. Also Springer Series in Information Sciences, volume 8 1 (1988).

[12]. Jolliffe, Ian. “Principal component analysis”. John Wiley & Sons, Ltd, 2005.

[13]. Hyvarinen, Aapo, Juha Karhunen, and Erkki Oja. "Independent component analysis." STUDIES IN INFORMATICS AND CONTROL 11.2 (2002): 205-207.

[14]. Seung, D., and L. Lee. "Algorithms for non-negative matrix factorization." Advances in neural information processing systems 13 (2001): 556-562.

[15]. Mohamed Aly, Mario Munich and Pietro Perona,“Bag of Words for Large scale object recognition,” in computational vision lab Caltech, Pasadena, CA, USA.

[16]. Zaharescu, Andrei, et al. "Surface feature detection and description with applications to mesh matching." Computer Vision and Pattern Recognition, 2009. CVPR 2009. IEEE Conference on. IEEE, 2009.

[17]. Rusu, Radu Bogdan, Nico Blodow, and Michael Beetz. "Fast point feature histograms (fpfh) for 3d registration." Robotics and Automation, 2009. ICRA'09. IEEE International Conference on. IEEE, 2009.

[18]. Bo, Liefeng, Xiaofeng Ren, and Dieter Fox. "Depth kernel descriptors for object recognition." *Intelligent Robots and Systems (IROS), 2011 IEEE/RSJ International Conference on*. IEEE, 2011.

[19]. Friedman, Nir, Dan Geiger, and Moises Goldszmidt. "Bayesian network classifiers." Machine learning 29.2 (1997): 131-163.

[20]. Safavian, S. Rasoul, and David Landgrebe. "A survey of decision tree classifier methodology." Systems, Man and Cybernetics, IEEE Transactions on 21.3 (1991): 660-674.

[21]. Burges, Christopher JC. "A tutorial on support vector machines for pattern recognition." Data mining and knowledge discovery 2.2 (1998): 121-167.

[22]. Cover, Thomas, and Peter Hart. "Nearest neighbor pattern classification."Information Theory, IEEE Transactions on 13.1 (1967): 21-27.

[23]. Freund, Yoav, Robert Schapire, and N. Abe. "A short introduction to boosting."Journal-Japanese Society For Artificial Intelligence 14.771-780 (1999): 1612.

[24]. W. Gribble, R. Browning, M. Hewett, E. Remolina, and B. Kuipers, "Integrating Vision and Spatial Reasoning for Assistive Navigation", in Assistive Technology and Artificial Intelligence. Lecture Notes in Computer Science, V. Mittal, H. Yanco, J. Aronis and R. Simpson (Eds.), Springer-Verlag, Berlin, pp. 179-193, 1999.

[25]. B. Kuipers, "A Hierarchy of Qualitative Representations for Space," in Spatial Cognition. Lecture Notes in Artificial Intelligence 1404, C. Freksa, C. Habel, and K. Wender (Ed.), Berlin: Springer-Verlag, pp. 337-350, 1998.

[26]. Skubic, Marjorie, et al. "Spatial language for human-robot dialogs." Systems, Man, and Cybernetics, Part C: Applications and Reviews, IEEE Transactions on 34.2 (2004): 154-167.

[27]. T. Kollar, S. Tellex, D. Roy, and N. Roy, "Toward Understanding Natural Language Directions," Proc., 5th ACM/IEEE International Conference on Human-Robot Interaction, pp. 259, 2010.

[28]. M. MacMahon, B. Stankiewicz, and B. Kuipers, "Walk the Talk: Connecting Language, Knowledge, and Action," Route Instructions, pp 1475-1482, 2006.

[29]. A. Vogel and D. Jurafsky, "Learning to Follow Navigational Directions," Proc., 48th Annual Meeting of the Association for Computational Linguistics, pp. 806-814, 2010.

[30]. S. Tellex, T. Kollar, S. Dickerson, M. Walter, A. Banerjee, S. Teller and N. Roy, "Understanding Natural Language Commands for Robotic Navigation and Mobile Manipulation," Proc., Conf. on Artificial Intelligence (AAAI), 2011.

[31]. P3DX Robot Introduction, http://www.mobilerobots.com/researchrobots/pioneerp3dx.aspx

[32]. Leyvand, Tommer, et al. "Kinect identity: Technology and experience."Computer 44.4 (2011): 94-96.

[33]. LMS200 Laser Range Finder Instruction http://sicktoolbox.sourceforge.net/docs/sick-lms-technical-description.pdf

[34]. Quigley, Morgan, et al. "ROS: an open-source Robot Operating System." ICRA workshop on open source software. Vol. 3. No. 3.2. 2009.

[35]. ROS Wiki, http://ros.org

[36]. Choi, Sunglok, Taemin Kim, and Wonpil Yu. "Performance evaluation of RANSAC family." Proceedings of the British Machine Vision Conference. 2009.

[37]. Kinect Calibration, http://nicolas.burrus.name/index.php/Research/KinectCalibration

[38]. Dillencourt, Michael B., Hannan Samet, and Markku Tamminen. "A general approach to connected-component labeling for arbitrary image representations."*Journal of the ACM (JACM)* 39.2 (1992): 253-280.

[39]. Matsakis, Pascal, et al. "Linguistic description of relative positions in images."*Systems, Man, and Cybernetics, Part B: Cybernetics, IEEE Transactions on*31.4 (2001): 573-588.

[40]. Sledge, Isaac J., and James M. Keller. "Mapping natural language to imagery: Placing objects intelligently." Fuzzy Systems, 2009. FUZZ-IEEE 2009. IEEE International Conference on. IEEE, 2009.

[41]. Pratt M.J. and Wilson P.R., 1985, "Requirements for support of form features in a solid modeling system", CAM-I, R-85-ASPP-01.

[42]. Biacino, Loredana, and Giangiacomo Gerla. "Fuzzy logic, continuity and effectiveness." Archive for Mathematical Logic 41.7 (2002): 643-667.

[43]. Engelbrecht, Andries P. "Computational intelligence: an introduction". wiley, 2007.

[44]. Brooks, Rodney. "A robust layered control system for a mobile robot." Robotics and Automation, IEEE Journal of 2.1 (1986): 14-23.

[45]. Siegwart, Roland, and Illah R. Nourbakhsh. "Introduction to autonomous mobile robots". MIT press, 2004.

[46]. Lee, David, and Mihalis Yannakakis. "Principles and methods of testing finite state machines-a survey." Proceedings of the IEEE 84.8 (1996): 1090-1123.

[47]. Laura Carlson, Marjorie Skubic, Jared Miller, Zhiyu Huo, and Tatiana Alexenko, “Developing Human-Driven Robot Algorithms for the Comprehension of Spatial Descriptions in a Robot Fetch Task”, in preparation for resubmission to Topics in Cognitive Science, special issue on human-robot interaction.

[48]. Marjorie Skubic, Zhiyu Huo, Tatiana Alexenko, Laura Carlson, and Jared Miller, “Testing an Assistive Fetch Robot with Spatial Language from Older and Younger Adults” in preparation for resubmission to The 22nd IEEE International Symposium on Robot and Human Interactive Communication.

[49]. M. Skubic, Z. Huo, L. Carlson, X. Li, J. Miller, “Human-Driven Spatial Language for Human-Robot Interaction.” Workshops at the Twenty-Fifth AAAI Conference on Artificial Intelligence. 2011.

[50]. M. Skubic, T. Alexenko, Z. Huo, L. Carlson, J. Miller, “Investigating Spatial Language for Robot Fetch Commands.” Workshops at the Twenty-Sixth AAAI Conference on Artificial Intelligence. 2012.

[51]. M. Skubic, L. Carlson; X. Li, J. Miller, Z. Huo, “Spatial language experiments for a robot fetch task.” Human-Robot Interaction (HRI), 2012 7th ACM/IEEE International Conference on. IEEE, 2012.

[52]. L. A. Carlson, and P. L. Hill. “Formulating spatial descriptions across various dialogue contexts.” Spatial Language and Dialogue 1.9 (2009): 89-104.

[53]. Matsakis, Pascal, and L. Wendling. “A new way to represent the relative position between areal objects.” Pattern Analysis and Machine Intelligence, IEEE Transactions on 21.7 (1999): 634-643.

[54]. Jain, Anil K., and Richard C. Dubes. “*Algorithms for clustering data”*. Prentice-Hall, Inc., 1988.