UNIVERSIDAD VERACRUZANA

Artificial Intelligence Research Center

An Approach Based on the Extraction of Geometrical Components for Household Furniture Detection by an Autonomous Mobile Robot

T H E S I S

To obtain the degree
PhD in Artificial Intelligence

Presents
MIA Oscar Alonso Ramírez

Supervisor:
Dr. Antonio Marín Hernández

Xalapa-Enriquez, Ver. December 2017

Acknowledgements

I would like to express my gratitude:

To my supervisor, Dr Antonio Marin for all his lessons and support during all these years.

To Dr Homero Rios, Dr Ericka Rechi and Dr Michel Devy for their advices, which improved this work.

To my friends, to those I made during these years and those who were from the beginning, thank you for sharing this journey.

To my family for always encourage me to keep going.

To CONACYT for the scholarship granted to me for graduate studies.

Contents

Chapter 1

Introduction

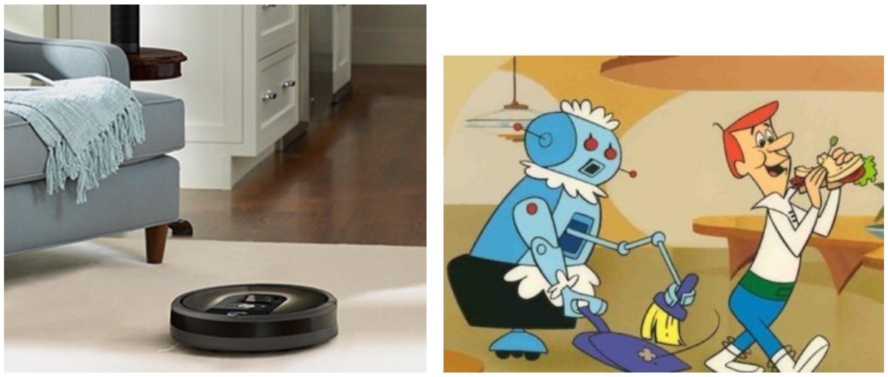
1.1 Intelligent Robotics

For many years, humans have been trying to create an intelligent machine capable of performing tasks the same way or even better than humans. First attempts can be found in the 18th century, where some mechanic devices in the shape of humans or animals could perform certain tasks and movements. [32] However, it is until the second half of the 20th century where we can find a great development of "robotics" as a scientific domain and the emergence of the firsts intelligent robots.

On one hand, robotics as a interdisciplinary branch of engineering, deals with the design, construction, operation and application of robots. On the other, artificial intelligence is a branch of computer science, whose main goal can be described as the development of the computers ability to perform different task emulating the human intelligence. Combining robotics and artificial intelligence create the possibility of having truly autonomous intelligent robots. More specific, robotics can create a machine capable of navigate and manipulate our environment and artificial intelligence can provide the understanding of environment.

In the recent years, the advance in robotics has been accelerated. Many robots have been created in different laboratories around the world, each one with its advantages and disadvantages. However it is hard for these robots to go from the laboratories to our homes. Since the laboratories environments are very controlled, it is hard for the robots to adapt to uncontrolled human environments. The few robots that are nowadays in our homes are programmed to perform very specific tasks.

Nowadays, the vacuum cleaner robot “Roomba” is the major example of a robot that has successfully enter to our homes (Figure 1.1(a)). But this robot does not localize within the environment, it only react to what its proximity sensors detect; moreover it can not interact with humans and much less anticipate its movements. So, we are still pretty far from having a truly service robot, like Rosie the robot from the Jetsons (Figure 1.1(b)).



(a) (b)

Figure 1.1: Example of service robots, in a) the vacuum robot Roomba and in b) a fictional robot, Rosie from the cartoon “The Jetsons”, a great example of a service robot.

The main reason why it is so difficult for robots to adapt to real human environments, is because they are very dynamic. This situation requires that robots perform several tasks at the same time, for example, localize himself, avoid collisions, detect objects, interact with humans and maybe even anticipate the humans movements. All of this tasks are basic to humans, but represent a challenge for robots.

Robots nowadays do not have a full understanding of human environments, which may seem a little contradictory since there are computers being really good at playing chess or go, and there are even robots exploring Mars. But, the fact is that the number of variables present in human environments is very big. Humans can process them almost unconsciously, but for a robot, that number of variables makes those tasks very challenging. This is known as the Moravec's paradox, which say "it is comparatively easy to make computers exhibit adult level performance on intelligence tests or playing checkers, and difficult or impossible to give them the skills of a one-year-old when it comes to perception and mobility."

For many research groups around the world, the tasks of robot's localization and navigation can be considered solved, however, only under certain circumstances.

1.2 Context

Nowadays there are several algorithms and methods to solve different tasks required by autonomous service robots. For example, there are algorithms that a robot can use to navigate and localize itself in a given environment, although these are not perfect, they provide very good results in most of the humans environments. These algorithms are mostly based on laser or sonar sensors, and most of them rely on the geometry of the environment.

There are other algorithms like path planning or the detection of certain objects that improve human-robot interaction, however they are not enough if we wish to have a robot as an assistant in our homes.

In order to improve current services provided and their quality, we must incorporate new methods and algorithms to extract more useful information about the environment. One of these tasks would be the detection of objects present in the environment. However we believe that specially household furniture will provide more information about the environment and then the robot could reasoning about it.

For a robot to detect all the pieces of furniture in a home or in an office environment, is still an open problem. This one is actually been tackled by many researchers and there are some results where they can detect some pieces of furniture but there are always certain constraints. Most of this works detect the furniture through cameras on-board the robot, analyzing either a 2D or 3D images. These detection algorithms normally focus on the shape of the object, on characteristics extracted at certain points or the object appearance.

We believe that it is important to identify household furniture in the 3D space, and not only as obstacles, but as a part of the environment. Because, if the robot knows what kind of furniture are present, this would give more information to perform its tasks. For example, if the robot has some uncertainty about where it is in the environment, the presence of a bed can suggest that the robot is in a bedroom, or the presence of the couch says that it is in the living room.

3D detection algorithms can be computationally very expensive, so when the detection is performed by a robot it is very important to find a proper way to represent the objects and to extract its characteristics. A good representation will lead to a fast processing and a good classification.

RGB-D cameras have become popular in the later years, they are not the most accurate however they are an inexpensive sensor to obtain 3D data. The color information has also been used in object detection, but it is a characteristic that can be easily affected by change in the light conditions. For that reason and considering that some furniture can change their color over time (for example when a bed changes its bedspread), it was decided not to use the color information at this point for the detection of furniture.

1.3 Hypothesis and Objectives

This work examines the current approaches to furniture detection and propose a variant for furniture a detection that works with noisy or incomplete data at linear time.

The main goal of this work is the detection of household furniture allowing in the future to improve the understanding of the environment and the quality of services provided.

A secondary goal will be the creation of an structure which permits to determine the characteristics of horizontal planes, as robot requires to know that is not the same an horizontal surface for a dinning table than the horizontal plane of a bed

1.3.1 Hypothesis

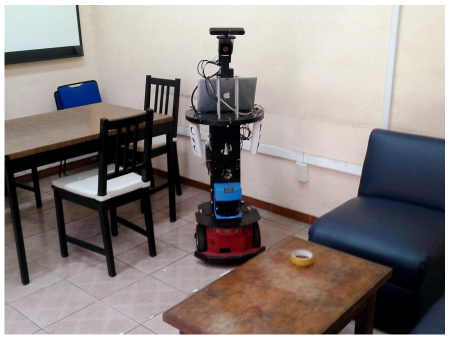
It is possible to represent the furniture within a house-like environment as a combination geometrical components and to use that representation to detect the furniture on a scene.

1.3.2 General objective

Identify the furniture present at a home-like environment to provide useful information that a robot can use to solve effectively future tasks

1.3.3 Specific objectives

- To create or recover 3D models of common household furniture for comparison and evaluation.
- To develop algorithms to extract the geometrical components from furniture in scenes.
- To determine measures of the geometrical components to characterize them
- To create a graphical representation for every piece of furniture
- To propose similarity measures to compare the models graph and the scene graph


(a)

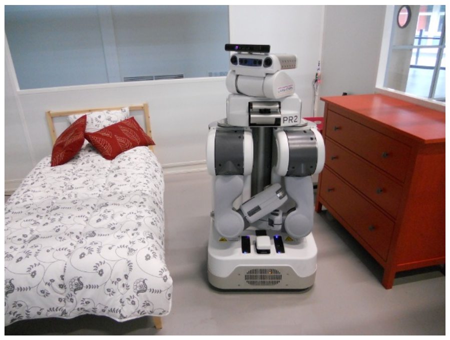

(b)

Figure 1.2: Robots in a) The robot UVerto at the Artificial Intelligence Research Center, Universidad Veracruzana, and in b) the robot Max, a PR2 robot at the LAAS-CNRS.

1.4 Considerations

Our furniture detection will be focused on those types of furniture that can be moved by a human (or a robot) being for do cleaning or to give another aspect to the room. Given that fixed furniture like closets or bookcases can be incorporated to the map of the environment while doing SLAM, these have not being considered along this work. On other hand, it is important to detect the furniture regularly; because it is not enough for the robot to memorize where they are and then just assume they will remain there.

This furniture detection will require a learning phase where the robot will learn a specific piece of furniture, the learning of a generic model for all the possibilities of a type of furniture (for example all the variations for a chair) is out of our scope.

Our propose uses and consider only an RGB-D camera, mounted on the head of a robot. These types of cameras provide a 3D image and have become very popular because of their low price, which make them available for practicably any robotic platform. No other sensor will be used for the detection, in order to avoid an increase of the amount of data to process.

The figure 1.2 shows two different robotic platforms on which this approach was developed and tested.

The robot Uverto (Figure 1.2(a)) is a robot, assembled at the Research Center for Artificial Intelligence at the Universidad Veracruzana. It is a two wheeled differential robot, equipped with an RGB-D camera mounted over a Pan/Tilt unit, a laser range finder and sonars.

Max, a PR2 robot (Fig 1.2(b)), belongs to the Laboratory for Analysis and Architecture of Systems (Laboratoire d'Analyse et d'Architectures des Systèmes, LAAS). The PR2 robot is a four wheeled omnidirectional robot, he has two arms with grippers, lasers, sonars, two stereo camera and a Kinect camera mounted on its head.

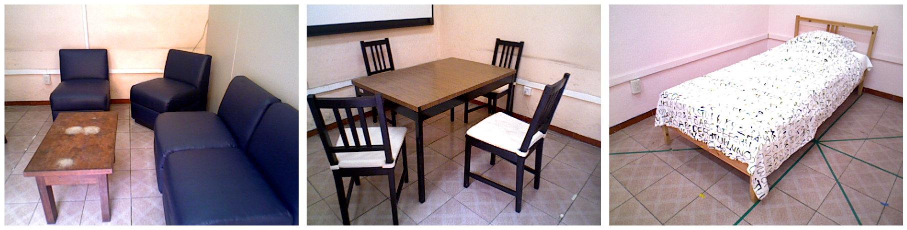
Both robots are working with ROS (Robot Operating System) which let us to capture the point clouds from the RGB-D camera and also provides all the transformations between the different reference frames from the robot. Having this transformations allows to transform the point cloud from a reference frame on the camera to one in the base of the robot or in the map, eliminating any inclination the camera may have.

The Figure 1.3 shows some images from the two environments used for experimentation. Both of them represent a house-like environment representing a living room, a dinning room and a bedroom.

1.5 Proposal

Based on an analysis of the furniture it was determined that they can be represented as a combination of geometrical components and they all have one main horizontal plane linked to their functionality. Therefore, finding these horizontal planes on a scene, can help us to detect the furniture. The figure 1.4 highlights the horizontal planes on a house-like environment.

We have analyze the common techniques for plane segmentation on 3D images and we determinate that we needed a different approach to detect the planes being more flexible with the mathematical model but more consistent with the human perception of a plane surface. So, among our specific goals


(a) Environment at the CIIA

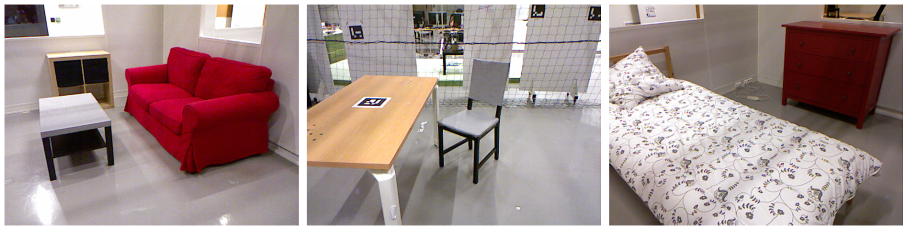
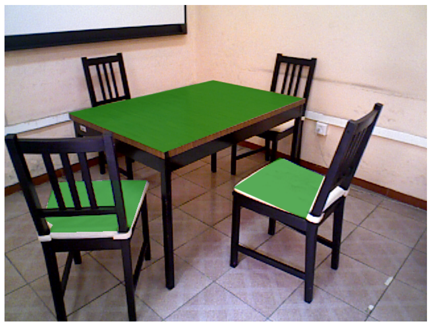

(b) Environment at LAAS

Figure 1.3: Environments


(a)

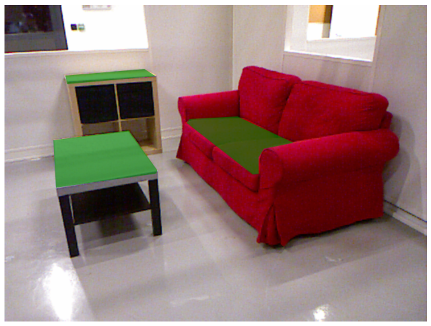

(b)

Figure 1.4: Horizontal planes present on the environments

there was the creation of a new plane detection strategy.

One of the problems detected when a common technique for plane detection was performed on a scene from our environment, was that a plane could be detected across different objects. For example, if the plane from a table was detected and its points extracted, not only it extracted the points from the table surface, it also extracted some points from the walls that satisfy the same plane equation.

A second problem was that some furniture like the couch or the bed have slightly curved surfaces, for a human this is obviated, but not for the common plane detectors. Often, this detector would find just small portions of the planes or planes with some tilting that does not correspond to the full surface of the furniture.

This problems were solved by using an approach based on the distribution of the points. Analysing the distributions of the points our approach can find the horizontal and vertical planes that conform the furniture.

Since the main goal for this work is the detection of different types of furniture, we needed to have the furniture models, so the robot can learn their characteristics and later compare them with the objects in the scenes.

Some of the most common 3D model formats were analysed and it was decided to use a point cloud representation for the furniture models. These models were created by us from multiple views of the pieces of furniture taken with the robot's camera.

Finally we propose to represent the pieces of furniture as a graph, where the nodes correspond to the different geometrical components and its characteristics and the edges represent the adjacency between them. The detection of the pieces of furniture is based on a probabilistic approach, proposed to measure the similarity of this graphs.

The next chapter will discuss the related work and explain how it relates to our propose. Chapter 3 will give an overview to the data and algorithms required for this approach. Chapter 4 and 5 will detail our proposal; the extraction and comparison of geometric component in Chapter 4 and the creation and comparison of the graph representation on Chapter 5. Finally, the evaluations are presented in Chapter 6 and the conclusions on Chapter 7.

Chapter 2

Related Work

Nowadays there are many works for object detection, depending on the application they can focus on a specific a set of objects or they can focus on a more general approach and try to detect a variety of objects. Is very common that these works are tested over object datasets (either working with 2D images [13] or with point clouds [25]). Some of the most popular datasets have everyday common objects like: balls, hats, some fruits and others, one example is the dataset presented in [25]. But there are also some databases with specific types of objects, for example, the one presented in [26]. Particularly the detection of furniture is not as common, but there are some works that have attacked this problem using different techniques.

This chapter will review some techniques used for object detection, first it will cover feature points and planes in common objects, then there will be an overview of some techniques for furniture detection and how these techniques relate with furniture detection. Since it will be necessary for our approach to built a model of the objects, this chapter will also explore different techniques for model creation.

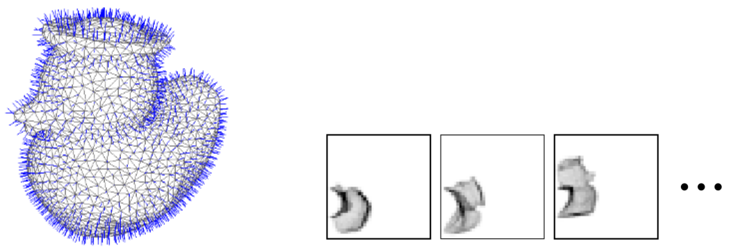

(a) (b)

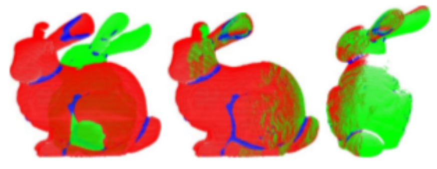
Figure 2.1: Example of a surface representation, in a) a polygonal surface mesh with surface normals and in b) spin-images. Figure taken from [22]

2.1 Point features

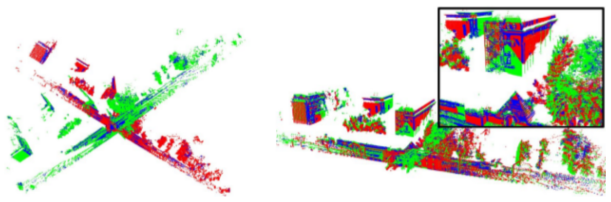
Many works in robotics has been done over 3D features and characteristics of them. Among the most popular features based on the object's texture information are Spin-image [23], SIFT [28] and SURF [6]. On the other hand, Fast Point Feature Histogram (FPFH) [41] and Viewpoint Feature Histogram (VFH) [43] are some geometry based shape descriptors.

The Spin-image is a shape descriptor used to match surfaces. On this case, the surface of the object must be described as a mesh with 3D points and surface normals. This is complemented with a descriptive image that encode properties of the surface attached to each surface point (Figure 2.1). In [22], the use of spin images results in an efficient multiple object recognition.

The Fast Point Feature Histograms (FPFH) is a multi-dimensional feature that captures local geometry information between the angles of a point and its neighbors. This feature is an improvement (in terms of computational time) of other feature histograms descriptors used for geometrical surface descriptions. In [41] this feature was proved successful to register point clouds. In figure 2.2 two examples are shown. In figure 2.2(a) the result after the registration of two partial views of a bunny model from the Stanford 3D Scanning Repository [26] is shown. In figure 2.2(b) a result is shown from a registration of a large point cloud from a real-world outdoor scene.



(a)



(b)

Figure 2.2: Example of results of using FPFH for registration. In 2.2(a), two partial views of the bunny model and the result obtained. In 2.2(b), on the left, two overlapping point clouds (red and green) and on the right the alignment result; the blue points correspond to persistent FPFH points. Figures taken from [41]

This feature and some others were tested on some common datasets in [2]. The comparison is made between features for point clouds available in the PCL library (this is a library that contains several algorithms for point clouds, a more detailed explanation will be given in the next chapter). It was concluded that features should be extracted only on a set of keypoints, since they can be computationally very expensive. A large number of keypoints improve the recognition results but increases the computational time.

In order to compare the different features, a subset of 3D point clouds from the dataset in [25] were used. This dataset contains 51 different objects categories, however for these comparisons only 10 of them were selected.

Choi et al. in [11] propose a pose estimation algorithm for grasping objects, that includes the boundary of the objects and the depth discontinuities. They also state a common problem for point features performances; the majority of these approaches, are tested on objects with sufficient curvature changes, but when they are faced with other type of objects, (for example industrial and home objects which are mostly planar) they are not discriminative enough.

Point Features is not the only option. Other works have proposed alternatives. For example Bo et al. in [9] proposed a set of kernel features. These features, based on depth images, capture different aspects of the objects like size, shape and depth edges. This features proved to be useful for object recognition.

Another alternative to point features is pair point features. One of the most popular works on point pair feature is the one presented in [14] by Drost et al. This algorithm is used for matching 3D model with 3D point cloud scenes. They propose the creation of a global model descriptor based on oriented point pair features. This model consists of all model point pair features, which will be stored on a hash table, representing a mapping from the point pair feature space to the model, where similar features on the model are grouped together. At the recognition stage, for each feature the hash table is queried and a fast voting, similar to the generalized Hough Transform, is performed to determinate the match.

Some improvements over the algorithm from Drost have been proposed. One on them is presented in [8], where they mention some drawbacks like a

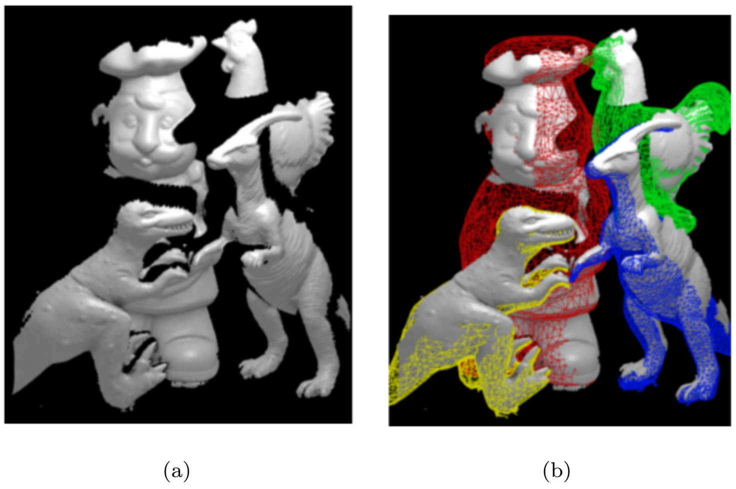
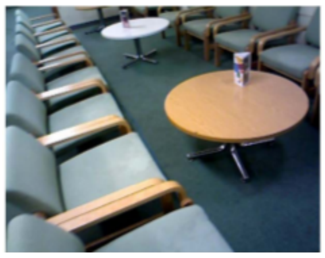


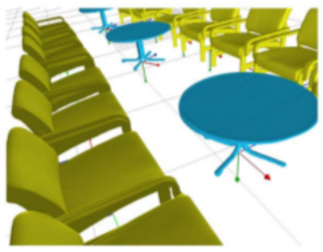
Figure 2.3: Recognition results from the methods presented in [14]. In 2.3(a) a scene from a dataset and in 2.3(b) the results, where it can be observed all objects were found. Figures taken from [14]

relatively high dimensionality of the search space, and sensibility to outliers and low density scenarios. In this article is proposed an improve by performing a previous segmentation of the objects and a weighted voting scheme.

The combination of different types of information can be useful to detect objects that can be difficult to detect using texture based features. For example, in [48] color information is combined with shape information in order to recognize objects. Another example of combining color information can be found in [3] where they characterize points from a point cloud based on color, its 3D position and normals. Color information can be interpreted as a probabilistic image used for object detection as in [20] or it can be combined with different sensors as in [30] where they fusion 3D depth data with color and also temperature information from a thermal camera. In [38] color information is used to create a superpixel segmentation (with the algorithm presented in [1]) and characterize them along with geometric features. However color can be a not very reliable feature. In [10], advantages and disadvantages of different color spaces are presented.

In [44] is presented a new 3D SLAM paradigm which instead of working


(a)


(b)

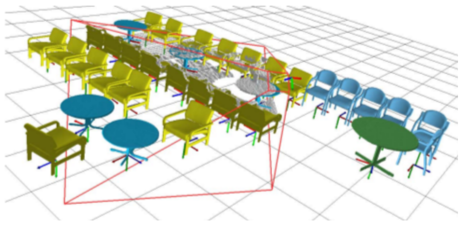
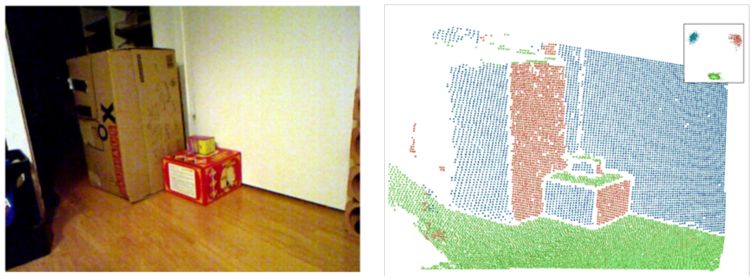

(c)

Figure 2.4: An example of a mapped scene with slam++. In 2.4(a) a view from the scene, in 2.4(b) the rendered objects and in 2.4(c) the mapped scene at the object level. Figures taken from [44]

with low-level primitives (like points or lines), they work "object oriented". Even when the work from [14] was presented for free form objects, this work uses those point pair features for the detection of furniture, probing to be very successful when objects occupy most of the camera's field of view, but poor for distant objects or partly occluded. On this paper, is performed a dense surface reconstruction, taking advantage of repeated known objects for compress representation, in this case a chair and a table. An example of a mapped scene with this approach is shown in figure 2.4.

2.2 Planes extraction

Our robot is expected to work within a home-like environment which, as many other human-made environment consist mainly of planes. As stated in [45] "it is a reasonable step to represent the surrounding 3D scene by a collection of planar patches extracted from the 3D point cloud".



(a) (b)

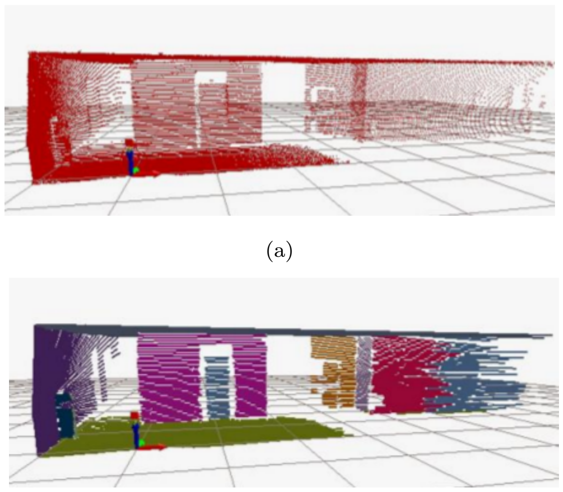
Figure 2.5: Example of plane segmentation, in 2.5(a) a rgb image corresponding to the input point cloud and in 2.5(b) the segmented planes. Figures taken from [21]

In [45] are mentioned three main algorithms for plane extraction on 3D which are, Expectation Maximization (EM), Region Growing (RG), and RANdom SAmple Consensus (RANSAC), perhaps being RANSAC the most popular approach. However this algorithms do not perform well on all circumstances, they perform well on convex scenes but may fail in cluttered or noisy scenes [45].

The FPFH is used to identify planes and other basic surfaces types in [5], giving good results in particular in close-up with low noise.

One alternative for plane detection was presented in [21] where they segment planes in household environments using an RGB-D camera. In figure 2.5 is shown the result for segmenting points on the point cloud with similar surface normal. They are also able to detect obstacles, graspable objects and they can segment and classify all the planes in the scene. Using local surface normals they cluster all the points with a similar normal and then they merge similar planes. Taking advantage of the organized structure of the point clouds allows them to detect obstacles and segment the objects and to correct measurement errors and to reconstruct the original geometry in far ranges.

In [46] another approach for plane segmentation is presented. They emphasize the importance of planar surfaces as landmarks and present a plane extraction for SLAM based on an iterative implementation of RANSAC. They combine information from a 2D laser scan and 3D information from an RGB-D


(a)

(b)

Figure 2.6: Results from the plane segmentation from [4], in 2.6(a) the input point cloud of an indoor scene and in 2.6(b) the segmented planes. Figures taken from [4]

camera.

For An et al. [4] is essential to describe the environment concisely with geometric features, particularly planes, to optimize the processing of the 3D data. They present an incremental plane extraction based on line segments with data from a laser scanner. An example of their results are shown in figure 2.6.

This work presents the importance of geometrical components have to describe the environment in which the robot is present.

2.3 Furniture detection

Using semantic information, robots can improve some of their tasks, specially those involving reasoning about the environment or the object within. For this reason some works have incorporated semantic maps either based on coarse features [35] or based on objects [31]. The work in [17] enable the robot to reason

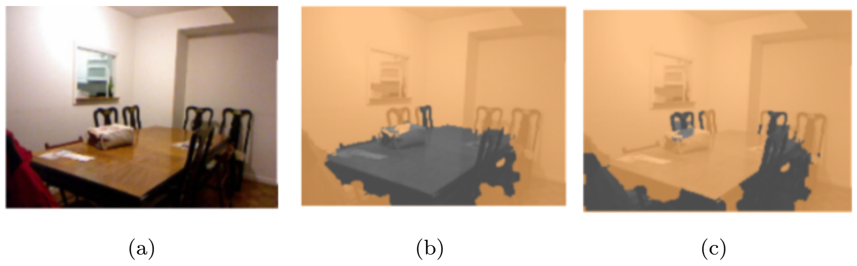


Figure 2.7: Explample of the segmentation presented in [38]. In 2.7(a) an rgb image, in 2.7(b) and 2.7(c) the results for the segmentation of the table and the chair respectively. Figures taken from [38]

with the spatial and semantic information, including the type of rooms and the objects present in them however a furniture detection was not performed. Other works focus on small objects and not pieces of furniture to build their semantic representation of the scene. In [27] the furniture is considered as obstacles in the environment and in [31] they are considered as large horizontal planes where they can find objects. In [24] some pieces of furniture are incorporated (they are detected using Iterative Closest Point) but two cameras are required, the robot's and another located on the environment.

According to Varvaoukas et al. in [47], "robots need to acquire capabilities for recognizing and estimating different pieces of furniture and different types of rooms, so that they achieve fluid collaboration and communication with humans".

The work in [38] presents a segmentation and recognition in RGB-D images. By exploiting the depth channel they perform a binary segmentation of the object from the background. A result is shown on figure 2.7. Another segmentation of furniture is presented in [20] using the color information as a probability value.

Other works perform furniture detection on point clouds, exploiting the full 3D information available.

In [47] is presented a system for room and furniture recognition combining information from self-captured models and internet-derived models. The recognition is based on some features calculated on binary images from different views of the objects.

In [49], Wu et al. represent a geometric 3D shape as a probability distribution of binary variables on a 3D voxel grid, using a Convolutional Deep Belief Network. The shape representation they propose can be used for object recognition and for shape completion. They construct their own database including some pieces of furniture like chairs, couch and tables. They depend on large-scale 3D CAD models for the construction of the database.

Qi et al. [37] present a 3D object classification on data obtained from CAD model, on different types of objects including pieces of furniture. Based on previous approaches where they detect the furniture using 3D shape models and Convolutional Neural Networks (CNNs) they determinate that the existing 3D CNN approaches are unable to fully exploit the 3D representations.

2.4 Furniture and planes

Using plane segmentation to detect furniture, Rusu et al. in [42], create semantic 3D object maps for a kitchen environment. They create an hybrid map comprised of two parts, one is a triangulated surface map (mainly used for navigation and manipulation tasks) and the other part is a static semantic map which contain the objects in the scene and also the walls, floor and ceiling. Figure 2.8 shows an example of the surface reconstruction. The semantic maps used in this work an expanded in [36] by integrating information about appearance and articulation of the objects.

The objects they incorporate to the map are mainly pieces of furniture from a kitchen environment, they select furniture that have certain functionality and that are fixed to the environment like cupboards, tables and shelves. To detect these objects instead of learning a single global model, they use geometrical techniques to split the scene and extract horizontal and vertical planes. These planes are later classified to identify the pieces of furniture, by processing separately, they claim they simplify the features needed and in general improve the classification.

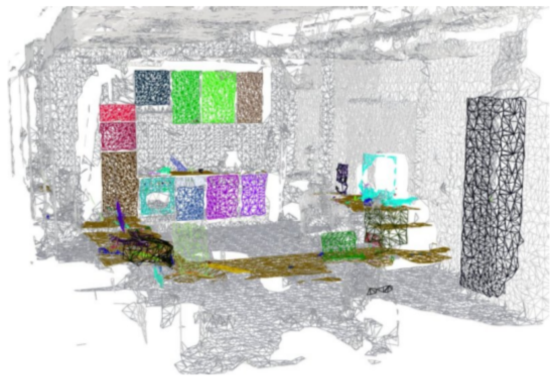


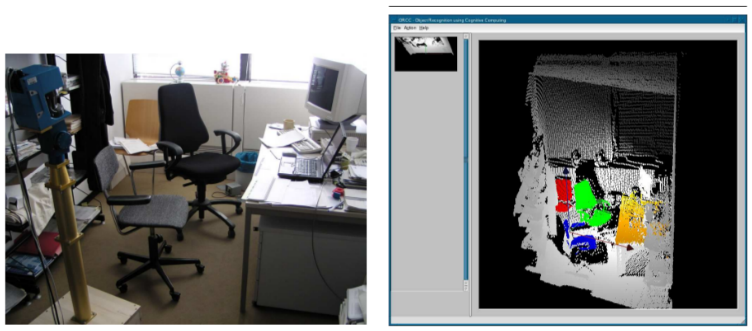
Figure 2.8: Surface reconstruction with the furniture candidates. Figure taken from [42]

The plane segmentation is archived with a hierarchical scheme with multiple level of detail. The point cloud is decomposed in an octree scheme and planar areas are searched with a RANSAC variant. When a plane is detected at any level, its equation is refined with points from a higher octree level.

Another work for furniture detection based on planes is present by Wunstel and Moratz in [50]. This approach uses a laser range to detect pieces of furniture within an office environment (chairs and tables). Based on the geometry of the objects they segment the object in three different layers, all the components and their relation between them are stored in a graph. An example of the results obtained is shown in figure 2.9

The importance of the furniture's planar surface is also exploited in [29] where a topological map of a house-like environment is created highlighting the furniture's planes as areas of interest.

Gunther et al. [18] presents a semantic mapping technique of an indoor environment, using RGB-D images. The furniture models are represented as a semantic model in an OWL-DL ontology. They propose a three step method; first, a triangle mesh is created from a point cloud, second they perform a planar region classification and then the furniture detection, and finally they adjust the furniture pose using ICP [7]. Their result is a mesh representation of the environment enriched by CAD objects models corresponding to the detected



(a) (b)

Figure 2.9: Object detection on an office environment. In 2.9(a) an rgb image of the scene and in 2.9(b) the rendered range data with the detected furniture colored. Figures taken from [50]

pieces of furniture. In average, it took around 9 seconds for them to process a single point cloud.

2.5 Models creation

As it has been mentioned, it is important to have models of furniture from which extract their components and to compare the data viewed on the scene. Because we can not be sure to find the models for our furniture onlilne ans since the most popular models may not be fit for our approach, it is necessary to create our own models.

Ruhnke et al. [39] present an algorithm to create 3D model of furniture from point clouds with partial views of the objects. They use an iterative matching procedure based on interest points extracted with a Harris corners detector over range images created from the point clouds. This approach allow them to recursively merge partial point clouds representing partial views of the objects even if the images come from different environments. They highlight the importance for the robot to work with partial views because in most of the applications the full 3D map of the environment with its objects completely visible is not available.

An online system for large and fine scale volumetric reconstruction is presented in [34]. They use a simple spatial hashing scheme that allow them real-time access and updates. Even when they operations are designed to be efficient for parallel graphics hardware, they can be computationally very expensive for a robot.

In [33] is presented an approach for a robot to learn models from the Web. They state that extracting this information from the internet would help the robots to learn and recognize similar objects in new environments. However at this point, labelled 3D data is still not very popular in the web and not all the model from all the varieties of furniture are available online.

Having analysed the works summarized in this chapter, it can be observed the need to extract some features in order to improve the detection of objects on a 3D space (specially for larger objects like furniture) and also the importance of some geometrical entities like planes to represent the environment or some parts of the objects.

Chapter 3

3D Data for Robotics

The recent incorporation of cheap 3D sensors to the robots have lead to the creation of new algorithms and libraries dedicated to process this information efficiently. In this chapter it will given a brief introduction to some of these algorithms as well as the software used in our approach.

3.1 ROS

The Robot Operating System (ROS) is an open source flexible framework for writing robot software [16]. It is not an operating system in the traditional sense, rather it provides a communications layer with the host operating system.

It started in 2007, mainly supported by the start-up Willow Garage, and nowadays is a widely used platform in the robotics research community. From the beginning, ROS has been developed by many researchers around the world (and for multiple robotics platforms) encouraging always the development of collaborative robotics software. In this way, every research team can contribute to the tasks that are their speciality, and everyone can build upon each other's work.

One of the main concepts in ROS are the “nodes”. A node is a process that executes a certain task. Normally, many nodes are running simultaneously. The communication between these nodes is performed by "messages". The messages in ROS are a strictly typed data structure that can support the standard primitive types as integer, floats, arrays, etc., they can even be composed of other messages or array of messages.

For a node to communicate something, a *message* must be created and then publish it to a specific "topic". For a *node* to receive a *message*, it must be subscribed to the corresponding *topic*. A *node* can publish to one or more *topics*, and can be subscribed to one or more *topics*. A *topic* can receive and broadcast messages for many *nodes*.

ROS also provides libraries and tools for common robot tasks, for example mapping, localization and navigation, only to name a few.

As it was previously mentioned, there are several nodes in ROS, created by different research teams. A few of the most common nodes used by any mobile platform will be described next.

The node **slam_gmapping** creates a 2D occupancy grid map. To use this node is necessary a robot equipped with a laser range-finder mounted horizontally and it also requires the odometry information of the robot. To create the map this node uses an efficient Rao-Blackwellized particle filter [15], where they take into account the most recent observation and the movement of the robot to reduce the uncertainty.

In order to localize a robot, ROS provides a node call **amcl.** This is a 2D probabilistic localization system. It is based on the Monte Carlo localization approach [12]. The node calculates the robot's pose estimations based on the data from a laser, a laser-based map (like the one obtained with gmapping) and the robot's transformations.

There is a library in ROS called **tf** (transformation), that it is very helpful, because thanks to it, it is possible to keep track of the reference frame of every moving part of the robot and recover transformations between all them. It is its job to maintain the information for every joint in the robot, and update them (capable of doing it at rates of hundreds of Hertz). This library contains some nodes to automatically update and broadcast the transformations to the system.

3.2 3D data and PCL

3D sensors have been around for some years, but it was until the launch of the Microsoft Kinect that its popularity increased. The low price of the Kinect, compared to other 3D sensors, made it possible for many robots to be equipped with a 3D sensor.

Along with this popularity and a bigger demand for techniques to process 3D sensing, it was created the Point Cloud Library (PCL). This library can work with multi-dimensional point clouds and the 3D geometry processing. It also contains state of the art algorithms, operating on point cloud data, for many tasks, such as: filtering, registration, model fitting, segmentation and others.

A point cloud is a data structure that represent a set of multi-dimensional points. Normally the points are in a 3D coordinate system defined by the triplet (X, Y, Z). Point clouds can be obtained from different sensors like stereo cameras, 3D scanners and RGB-D cameras.

PCL uses its own format for point clouds. The Point Cloud Data (PCD) file format is not the only or the first format for point clouds, but it tries to complement, other existing files. The point cloud stored at PCD files can be of different dimensions, for example, if the points have only (XYZ) data, it would stored in three dimension, but it the points also have color they would be four dimensions, or if the points also have the surface normal components, it would be six dimension $(X, Y, Z, R, G, B, normalX, normalY, normalZ)$.

Another characteristic for the PCD is the way the points are stored. The PCD format contemplates the width and height of the point cloud. If the point cloud has a width and a height different of one, the cloud is considered organized. But, if the total number of points of a point cloud is specified on the width and has a height of one, it is considered as an unorganized point cloud. This means that the points in an organized cloud are stored in a structure very similar to a 2D image, where the data is split in rows and columns. This gives an important advantage because it simplifies the search of a point's neighbors, resulting on a more efficient computation.

3.3 Ransac Algorithm

RANSAC (Random Sample Consensus), is an iterative method for estimating a mathematical model from a data set. This method is capable of detecting outliers in the data set and to estimate a desired model without using them.

The algorithm works, by selecting a subset of the data set, adjusting a model with them and then testing this model to determinate the number of outliers. This process is repeated until some ending criteria is met, for example a small number of outliers, or the number of iterations.

Ransac has become a popular algorithm to detect planes on a point cloud. This algorithm has been implemented on PCL (along with variations of it) and it can be used to adjust a point cloud to different types of models for example: planes, circles, sphere, cylinders, cones, etc.

3.3.1 Common problems with Ransac

After some tests of the Ransac algorithm (from PCL) in our environment, a few problems were detected.

First, the computational time. If the algorithm runs on a full point cloud, normally it will the very slow. To get a faster computation it is needed to perform a downsampling to the point cloud.

Second, the segmentation of the planes. Ransac will return all the points that fit the selected model, even if they belong to different objects. For example, when Ransac detects the horizontal plane of a table, it will also return the points from the walls that are at the same heigh and which fit the model, but without being on the corresponding plane or structure.

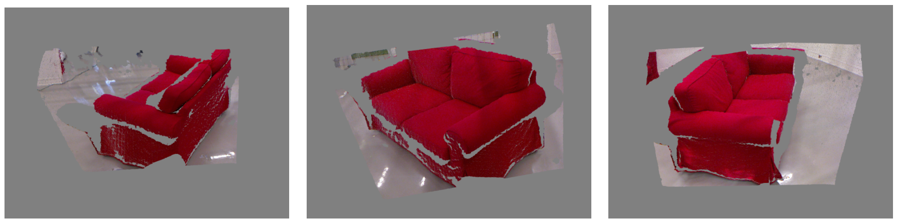

(a) (b) (c)

Figure 3.1: In a), b) and c) point clouds from different points of view for the couch

Another example is the segmentation of slightly curved surfaces that can be consider as planes. For example the couch, its seat has certain curvature, so the detected plane could have a considerable inclination. For our approach it will be preferable to have a segmentation closer to the human perception of the plane, which dismisses the inclination.

3.4 Model construction

As, our goal is to detect pieces of furniture on an indoor scene, we need to have a model of the furniture we wish to detect in order to compare the objects present on the scene. Nowadays there are many repositories online for 3D models with many different formats, however, on one hand, we can not be sure to find all the models we need and on the other, these models contain surfaces not visible for our robot, therefore we have decided to create our own models.

Our model is constructed from several partial-views images taken with the robot's camera. For the acquisition the robot moved around each piece of furniture saving the corresponding 3D images. An example of this partial views of a couch are show on Fig 3.1. Since the robot was localized in the environment reference frame, all the PCD's where transformed to the world reference frame.

Before combining the 3D images, the object was segmented with a clustering algorithm, that eliminates the floor, walls or other objects present on the images.

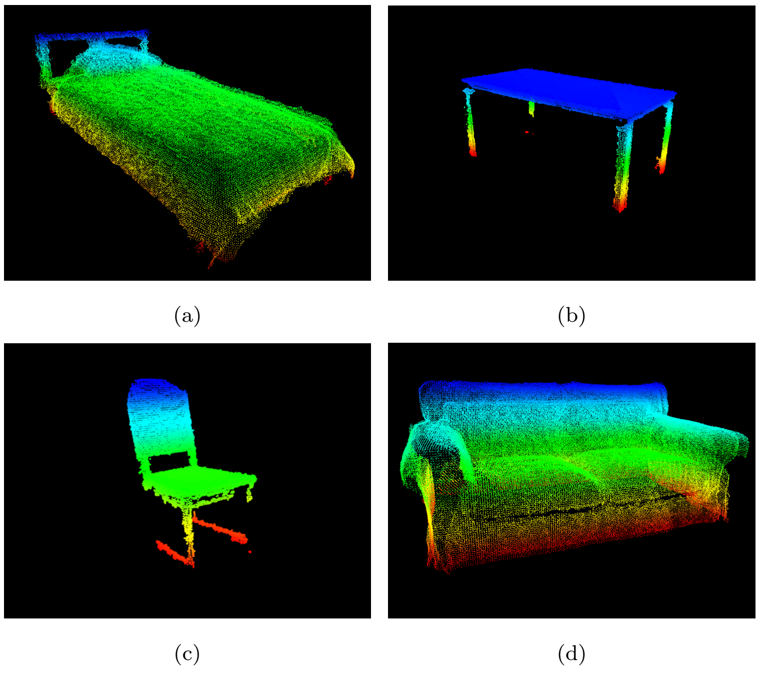


Figure 3.2: Examples of the PCD model created for the furniture, in a) the bed, in b) the table, in c) the chair and in d) the couch

The next step was joining all the images in one single point cloud. There are algorithms like ICP (iterative closest point) used to register a pair o a series of images. Since the images were transformed to the same reference frame, we get a good approximation but we use ICP to refine the transformations and obtain a better result.

3.4.1 ICP algorithm

The main idea for the ICP algorithm is to find the correct transformation between a set of points, the source, to another set, the target. This is a very popular algorithm for register 3D point clouds. [40]

The process starts with an initial guess of the transformation, where a correspondence between points in the source and the target is established. If the correspondences are correct, the correct transformation will be generated. If not, the algorithm iterates to refine this transformation by repeatedly generating new pairs of corresponding points. If the two set of points are close to each other, the algorithm will converge.

Diverse improvements have been proposed where they change the selection or the matching or the points, to the way the transformation is adjusted, etc.

Chapter 4

Geometric Components

Common household furniture is composed of various parts, most of them could be associated to a geometric component, i.e. the legs of a table or the horizontal surface from the seat of a chair.

Planar surfaces are very common in our house environments, an extraction of this planes can create a good representation. This chapter details an approach for some geometrical components extraction based on histograms of 3D points, particularly horizontal and vertical planes.

We begin with a revision of some techniques used and then our approach will be described.

4.1 3D Points Histograms

Histograms are a powerful tool to analyze the distribution of numerical data. Since in this case, we will be working with point clouds, the histograms will be used to analyze the distribution for points over a 3D space. More specifically, we are interested in points that belong to a piece the furniture.

In order to analyze a given point cloud, it is important to be sure that the point cloud is aligned with reference to a specific reference frame. In other words, motions from robot's head and therefore of the camera in it, should be corrected; particularly this means that any tilting in the camera position has to be corrected.

We propose to make a histogram of heights, because it allow us to extract information about the distribution of the points. For each horizontal plane there should be a peak on the histogram.

4.1.1 Peak Detection

A naive approach to detect the peaks over histograms, can be selecting the higher point and then selecting, both sides, descendant values until a minimum is reached. However this approach is very sensitive to noise and adjacent small peaks.

To reduce the noise and have a better peak segmentation detection, the histogram curve is simplified with the Douglas-Peucker algorithm. Once the curve have been simplified, the higher points are selected as well as both neighbours, the right and left, corresponding to the inferior boundaries for the peak. In the next the Douglas-Peucker algorithm used will be described.

4.1.2 Douglas-Peucker algorithm

This algorithm, also known as the Ramer-Douglas-Peucker algorithm (RDP), is a curve simplification algorithm, which takes a polygonized curve with n vertices and outputs an approximation curve with m vertices, where $m < n$. This is one of the most popular curve simplification algorithm, commonly used on cartography and computer vision. It was proposed in the early seventies and since then a several improvements and implementations have been proposed.

The algorithm takes a list of points and an error threshold. The process starts with an straight line between the initial and final point. Then, a search is performed over the list of points to find the farthest point from this line. If the distance from this point to the line is inferior to the error threshold, the

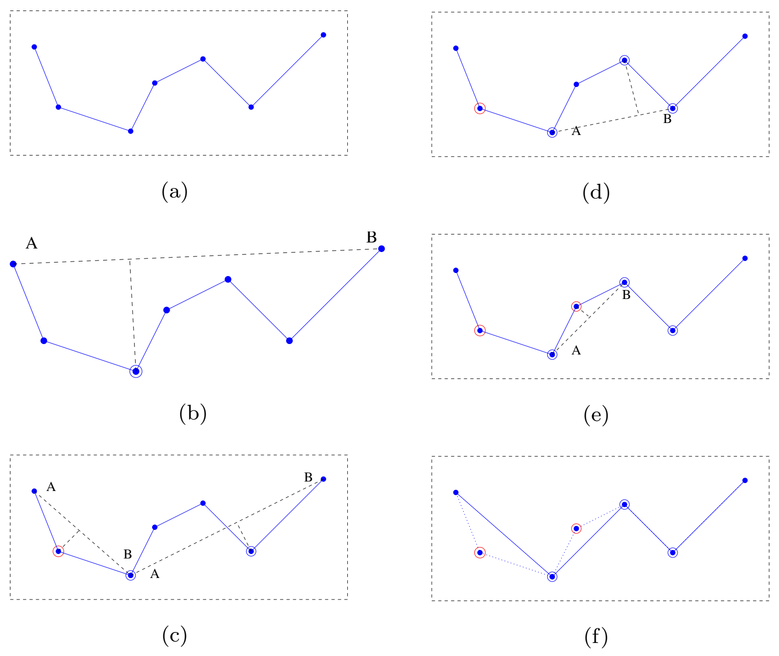


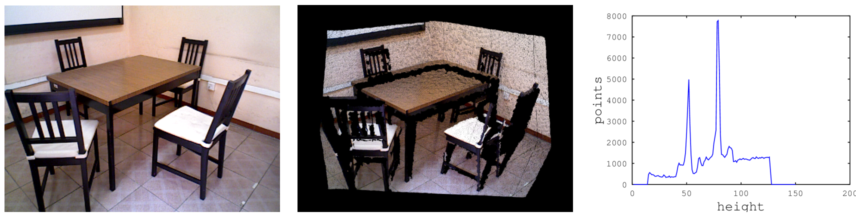
Figure 4.1: Example of Douglas-Peucker algorithm

line is accepted as a good approximation; otherwise, the algorithm is applied recursively to both segments generated: the first one between the initial and the farthest point and the second, between the farthest and the final point.

The figure in 4.1.2 shows an example of the results from the algorithm. The Figure 4.1(a) shows the initial polyline, from Figure 4.1(b) to 4.1(e) show the evaluation of the points, and the Figure 4.1(f) shows the simplified polyline.

According to [19], the algorithm is not optimal, but generally produces high quality approximations compared to other heuristic algorithms.

We have used this algorithm to recover the limits of the peak over the 3D points height histogram.



(a) (b) (c)

Figure 4.2: Example of a height histogram

4.2 Histograms from furniture

As it was mentioned before, histograms will be used to extract and analyze the geometric components of each piece of furniture. The figure 4.2 shows an example of a 3D points height histogram from an indoor scene. The figures 4.2(a) and 4.2(b) show the RGB image and the point cloud respectively and the figure 4.2(c) shows the corresponding 3D point heigh histogram.

As expected, it can be observed that the horizontal plane from the dining table generates one peak, and those from the seats of the chairs generate another.

The resolution or size of the bins to create the histograms is an important parameter. This size should be big enough to eliminate small variations on the curve but not too big as it to eliminate the relevant characteristics of each curve. The figure 4.3 shows an example the previous histogram with three different bin sizes. Looking back at the scene in figure 4.2, if the bin size was selected too small then the peaks on the histogram from the seats of the chairs, could not be adapted to small variations on the cushions of the chairs. And if the bin size were too big then the peaks will also contain points from the legs and the back of the chairs, making a posterior filtering needed.

Different views of an object can produce different histograms, because not all the surfaces are always visible.

So, if the histograms will be used to characterize a given piece of furniture, it is important to create an histogram that captures all the furniture characteristics. The figure 4.4 shows the histograms from different views (in

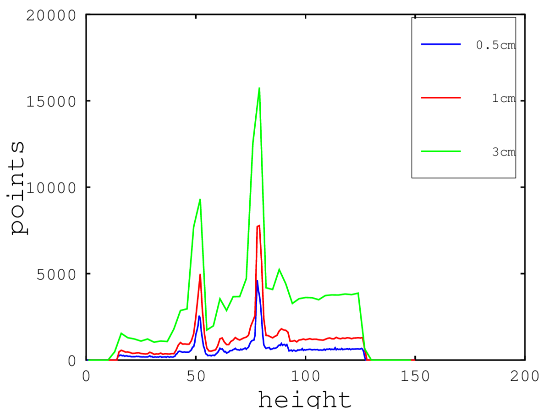


Figure 4.3: Histograms with a different bin size. It can be observed that a bigger bin size reduces small peaks in the curves, but it also incorpotate more points to the bigger peaks

color blue) from a chest of drawers. It can be observed how the partial views are similar between them; therefore the mean of the different views can be consider a good approximation for the piece of furniture.

The figure 4.5 shows the histograms of six different pieces of furniture, it can be observed how all of them present a different curve.

All the furniture's histograms present an outstanding peak which correspond to the horizontal planes of the furniture.

The detection and extraction of these peaks can be helpful to detect the horizontal planes on a scene with multiple pieces of furniture. In the following, it will be explained how this is achieved.

4.3 Plane Detection based on histograms

For detecting planes, instead of using a common technique like Random Sample Consensus (RANSAC), we have decided to use a different technique based on histograms. This approach works for detecting horizontal planes and can be extended to detect also vertical planes and allows us a faster plane detection with

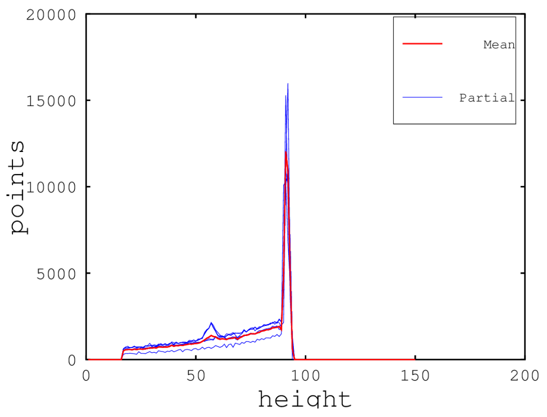


Figure 4.4: Histograms from different views and their mean of a chest of drawers

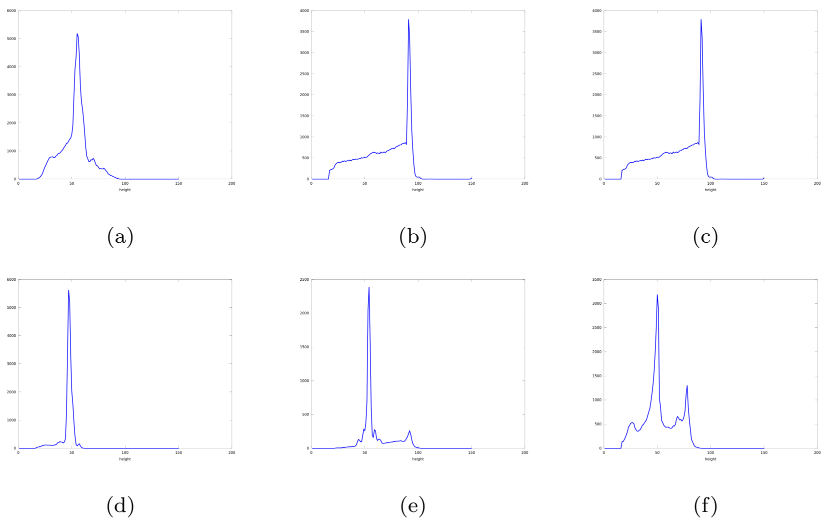
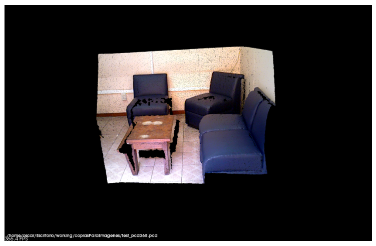
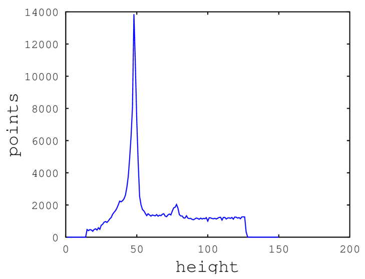


Figure 4.5: Height histograms for different pieces of furniture. In a) the bed, in b) the table, in c) the chest of drawers, in d) the center table, in e) the chair and in f) the couch



(a)



(b)

Figure 4.6: Example of a height histogram

a complete point cloud and the resulting planes could have certain curvature which adapts better to certain furniture surfaces.

4.3.1 Horizontal Planes

For a given point cloud from a scene, the first step to detect the horizontal planes is the construction of the 3D points height histogram.

We must remember that, the point cloud is transformed to the robot's base reference frame, so horizontal planes will concentrate a considerable amount of points at the same heigh and create a peak at the histogram.

When the height histogram is constructed, each bin stores its corresponding points, so when the peaks from the histogram are extracted, it is easy to recover the corresponding points.

The figure 4.6 shows an example of a point cloud from a scene and its corresponding height histogram. It can be observed there are three couches and a center table in the scene. In this case the seat of the couches and the surface of the table are practically, at the same heigh, so there is only one outstanding peak in the histogram.

The extracted points correspond to a horizontal plane, however in some cases (like this one) one peak can contain more than one plane. In order to detect and separate these cases, all the points from a peak are projected to the

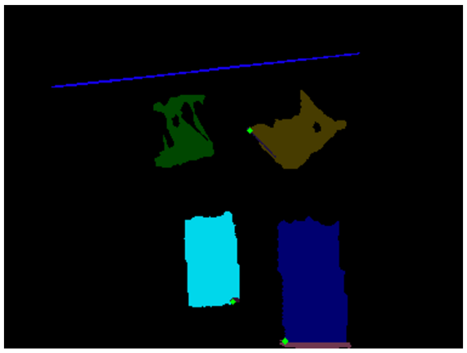


Figure 4.7: Projection to the points extracted

floor plane and an image is created from this projection. In the same way, as the histogram bins stores the corresponding 3D points in them, each pixel from this image projection also stores its corresponding points. Then, this image is analyzed in order to separate the different horizontal planes by doing contour extraction and clustering.

The figure 4.7 shows the projection image from this example, it can be observed the projections from the four different pieces of furniture in the scene and also a line from the wall's point that have the same hight. In order to eliminate this kind of projections and other coming from that can be generated by noise on the point cloud, the contours with a small area are eliminated. The points are extracted from the remaining contours and each one is returned as a horizontal plane found on the scene.

Since, this is a relax approach for plane detection, it allow us to identify as "planes" certain surfaces with a small curvature like the seat of a couch or an irregular surface like a bed.

Another advantage of this approach is that when it fit the plane equation, it does not add points that belong to another object, like the points from the wall in the previous example.

The next subsection explains how this approach can be adapted to extract other types of components.

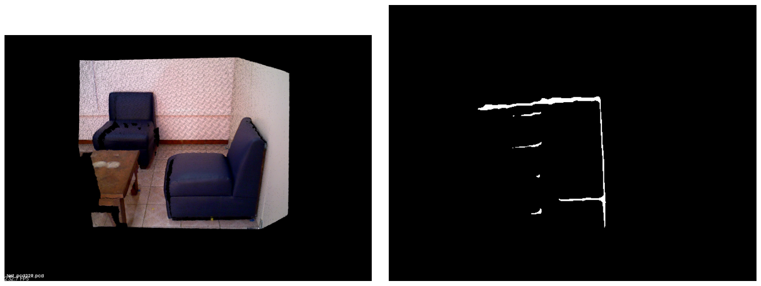

(a) (b)

Figure 4.8: Example of a 2D histogram

4.3.2 Vertical Planes

For detecting the vertical planes, we follow a similar approach that the one for detecting the horizontal planes, based on histograms. This time we create a 2D histogram where the point cloud will be projected to the floor plane, so the vertical planes will generate a high concentration on the same coordinates forming a line. The extraction of these lines will extract the vertical planes.

A grayscale image will be created based on this projection, where the higher amount of points in one coordinate will correspond to white and where there are no points correspond to black. This image will be pre-processed in order to highlight the lines before extract them. The figure 4.8 shows an example of a point cloud from an scene and its 2D histogram.

First the projection image is thresholded to eliminate the small values and then morphological filters are applied to smooth the lines. Since the lines are not always completely straight and with a variable width, a common algorithm to detect lines, like the Hough algorithm, is not appropriate for this case.

A clustering algorithm was used to detect the lines. the resulting clusters are approximated to polygons so we can simplify them and reduce the its number of points. Then a convex hull is calculated to determinate if the cluster is one single line (for example a wall) or if it is composed by two o more lines together, for example two walls in a corner or a piece of furniture placed against the wall.

If the cluster is not a single line, then the intersection points are localized

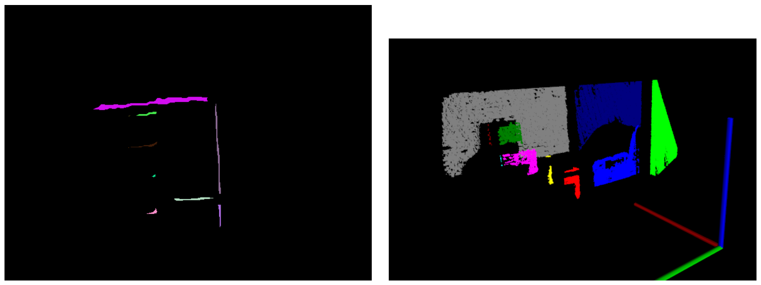

(a) (b)

Figure 4.9: Example for vertical planes detected

in order to separate the lines. Once all the lines are separated, the points from each coordinate from the cluster are retrieve. The figure 4.9 shows the contours found on the image and the vertical planes extracted form them.

4.4 Comparison of the detection of planes

How this approach for detecting horizontal and vertical planes based on the analysis of the distribution of the points compares to the traditional algorithms for plane detection? First, this approach only detects horizontal and vertical planes, it can not detect inclined planes. Since the main goal is the detection of furniture, where horizontal and vertical planes are predominant, this is not consider a drawback for this approach.

The planes detected by this approach are closer to the human perception of a plane, compared to the planes from a RANSAC algorithm. For example, if a RANSAC algorithm is executed for detecting planes on a PCD scene from a dinning room, when the horizontal plane from the table is extracted it will also extract some points from the walls that mathematically satisfy the plane equation, It is preferable to extract only the points from the table.

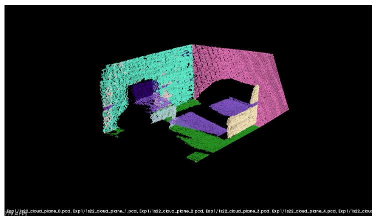
Finally the time of execution, for the RANSAC algorithm it can increase significantly depending on the complexity of the scene while for this approach it is almost stable, depending only on the size of the PCD. For most of the cases, this approach is faster than a RANSAC algorithm when they are computed

	Full PCD			Downsampled		
PCD	# points	# planes	t(ms)	# points	# planes	t(ms)
1s02	307200	6	212 529	56985	3	70.8411
1s08	307200	9	443 996	91907	5	188.536
1s22	307200	13	924 436	77495	8	307.6721
1s25	307200	16	996 850	98226	6	176.7132
1s32	307200	8	539 632	96164	5	126.41
1s36	307200	10	755 982	75436	5	149.2531
1s41	307200	15	745 595	78399	6	195.037

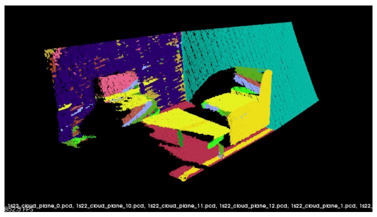
Table 4.1: Table for RANSAC algorithm plane detection

	Full PCD				Downsampled			
PCD	# points	# H	# V	t(ms)	# points	# H	# V	t(ms)
1s02	307200	1	3	350.607	56985	1	2	200.866
1s08	307200	1	3	273.514	91907	1	4	223.605
1s22	307200	3	8	383.694	77495	3	8	328.837
1s25	307200	1	4	339.507	98226	1	3	274.755
1s32	307200	1	5	325.256	96164	2	3	282.252
1s36	307200	3	9	444.23	75436	4	6	368.704
1s41	307200	3	4	338.192	78399	2	4	264.416

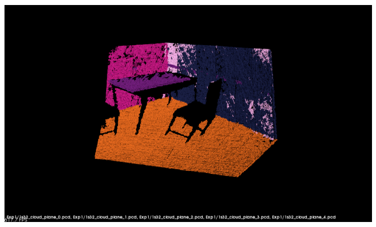
Table 4.2: Table for this approach plane detection



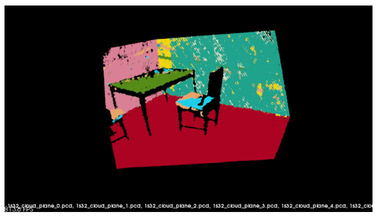
(a)



(b)



(c)



(d)

Figure 4.10: Example of the planes detected on some scenes by both approaches. In a) and b) the results from a living room scene for our approach and for Ransac, respectively. In c) and d) the results for a living room, in c) the results for our approach and in d) for Ransac

with the full PCD, if the point cloud is downsampled, the RANSAC algorithm is faster. The maximal error for the planes detected by the RANSAC algorithm can be expected to be smaller compared with the one from the planes detected by our approach since the detected planes can incorporate small curvatures to the surfaces, which has already be explained as a helpful characteristic for our approach. Tables 4.1 and 4.2 there are the number of planes detected and execution times for both approaches.

This chapter gave an overview on how to extract different geometrical components, the next one will explain how to characterize them in order to characterize the furniture based on the combination of the geometrical components.

Chapter 5

Graphs Representation

In this chapter we describe how geometrical components are characterized and merged into a graph, in order to provide an useful representation for each piece of furniture. These graphs are used also for the furniture detection.

5.1 Geometrical Components

Given a piece of furniture, we can decompose it into different parts that roughly can approximate geometric shapes. For example, a table can be decomposed into a horizontal plane and (regularly) four legs that can be approximated to cylinders or rectangular prisms These are the parts, which we will refer to as *geometrical components.*

There can be different types of geometrical components, in the previous example of the dinning table there are two types: the horizontal plane and the legs, but there can be more types, for example the backrest of a chair will be approximate to a vertical plane.

We define a set $\mathcal{G}c$, which contains N_k different types of geometric components. Each geometric component Gc^k can be composed by a non homogeneous set of characteristics that describe it, defined as:

$$Gc^k = \{ft_1^k, ft_2^k, ..., ft_{n_{ft^k}}^k\} \tag{5.1}$$

where k designate an element from the set $\mathcal{G}c$. Then a geometric component Gc^k is composed by a set of n_{ft^k} characteristics ft^k.

Each types of geometrical components can have its own set of characteristics or they can share some or all the characteristics.

5.1.1 Characteristics of Geometrical Components

Characteristics or features of a geometrical component can be of various types or sources, for example, for a horizontal plane, theses characteristics can be: their height, area and relative measures.

Every geometrical component must be characterized in order to describe the complete piece of furniture. By simplicity, at this point, all the geometrical components have the same characteristics, but more can be added or replaced in the future. Features or characteristics of the geometrical components considered are:

- **Height:** The average height of the points belonging to the geometric component.
- **Height Deviation:** Standard height deviation of the points in the peak or region.
- **Area:** Area covered by the points.

In Figure 5.1, are shown the values of each characteristic for the main horizontal plane of some furniture models; the parallelepipeds represent the uncertainty (computed as variations) for each variable. It can be observed how the selected types of furniture are fully separable with this three characteristics.

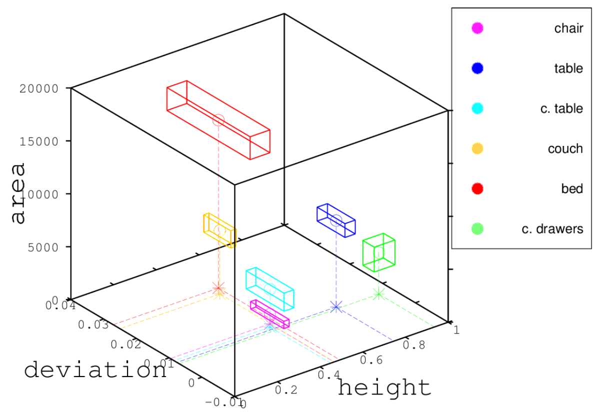


Figure 5.1: Characteristics of the main geometrical component (horizontal plane) for diverse pieces of furniture

5.1.2 Similarity Between Geometrical Components

In order to compare, geometric components from models to geometric components extracted from a scene, we have proposed a similarity measure.

In general, a similarity measure s_{Gc} of two geometric components Gc^k and $Gc^{k\prime}$, both of the same type k, has been defined as:

$$s^k_{Gc}(Gc^k, Gc^{k\prime}) = 1 - \sum_{i}^{n^k_{ft}} w^k_{Gi} d(ft^k_i, ft^{k\prime}_i) \tag{5.2}$$

where k represents the type of geometric component, w_{Gi} are weights and $d(ft^k_i, ft^{k\prime}_i)$ is a function of differences for the i^{th} feature of the geometric components Gc^k and $Gc^{k\prime}$, defined as follow:

Be

$$\delta\varphi = \frac{|ft^k_i - ft^{k\prime}_i| - \epsilon_{ft_i}}{ft^k_i} \tag{5.3}$$

$$d(ft_i^k, ft_i^{k\prime}) = \begin{cases} 0, & \delta\varphi < 0 \\ \delta\varphi, & 0 \leq \delta\varphi \leq 1 \\ 1, & \delta\varphi > 1 \end{cases} \tag{5.4}$$

where ϵ_{ft_i} is a measure of uncertainty related to the i^{th} characteristic. This function normalize the difference to assure that the result will be between zero and one; zero when the two characteristic are practically equal (considering the uncertainty) and one when they are totally different. Another expression for Eq 5.4 is

$$d(ft_i^k, ft_i^{k\prime}) = max(min(\frac{|ft_i^k - ft_i^{k\prime}| - \epsilon_{ft_i}}{ft_i^k}, 1), 0) \tag{5.5}$$

The uncertainty will reflect the changes that the characteristics could have. This is primarily due, by the noise of the sensor, but also by other factors, as a floor not regular, calibration imprecisions or so on. In other words, small variations can be expected between different executions. In the next section there will be an explanation on how these uncertainties are calculated.

5.2 Graph Representation

Taking into account, the common definition of a graph, as an ordered pair $G = (V, E)$ comprising a set V of vertices or nodes, and a set E of edges or arcs; a piece of furniture F^i has been defined with a graph representation as follow:

$$F^i = (V^i, E^i), \quad \text{with } i = [1, ..., N_f] \tag{5.6}$$

where F^i is an element from the set of furniture models $\mathcal{F}$; the sets V^i and E^i contains the vertices and edges associated with the i^{th} class and N_f is the number of models in the set $\mathcal{F}$.

The set of vertices V^i and the set of edges E^i are described by lists as follows:

$$V^i = \{v_1^i, v_2^i, ..., v_{n_v^i}^i\} \tag{5.7}$$

$$E^i = \{e_1^i, e_2^i, ..., e_{n_e^i}^i\} \tag{5.8}$$

where n_v^i and n_e^i are the number of vertices and edges, respectively, for the i^{th} piece of furniture.

The functions $V(F^i)$ and $E(F^i)$ are used to recover the corresponding lists of vertex and edges of the graph F^i .

An edge e_j^i is the j^{th} link in the set, joining two nodes or vertex for the i^{th} piece of furniture. As connections between nodes are a few, it is possible to use a simple list to store them. Thus, each edge e_j^i in the list is described by:

$$e_j^i = (a, b) \tag{5.9}$$

where a and b correspond to the linked vertices $v_a^i, v_b^i \in V^i$, such that $a \neq b$; and since the graph is an undirected graph $e_j^i = (a, b) = (b, a)$.

5.2.1 Furniture Graphs

Once we have the models of the furniture and the geometric components has been extracted and characterized, it is possible to construct a graph to represent each piece of furniture.

Be F^* a piece of furniture, for example a dinning table, therefore as stated on (Eq. 5.6), the graph is described by:

$$F^* = (V^*, E^*)$$

where V^* has five vertices and E^* four edges, in other words, $n_v = 5$ and $n_e = 4$. The five vertex correspond to: one vertex for the horizontal surface of the dining table and one for each of the four legs. As have been stated, the horizontal plane is selected as the main vertex. The edges correspond to the union between each leg and the horizontal plane.

In Figure 5.2(a), it is presented the graph corresponding to a dinning table. In a similar way, Figure 5.2(b) represent the graph of a chair. On last case, there is one more vertex, which is an vertical component corresponding to the backrest of the chair.

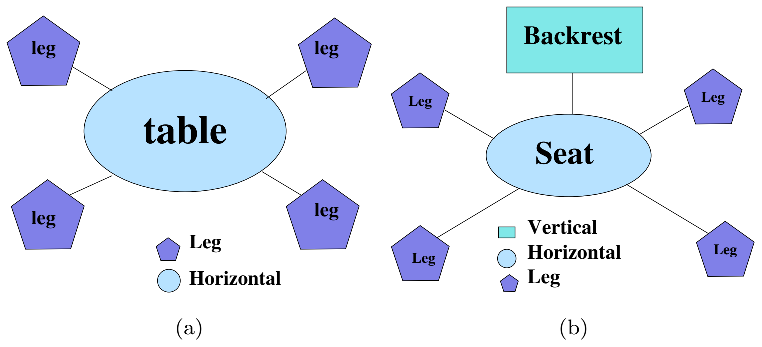


Figure 5.2: Example of complete graphs models for: a) a dinning table and b) a chair; different shapes represent different types of geometric components

An example for list of vertices and edges for the dinning table would be:

$$
\begin{aligned}
V^{table} &= \{table, leg1, leg2, leg3, leg4\} \\
E^{table} &= \{(table, leg1), \\
&\quad (table, leg2), \\
&\quad (table, leg3), \\
&\quad (table, leg4)\}
\end{aligned}
$$

And for the chair :

$$\begin{aligned}
V^{chair} &= \{seat, backrest, leg1, leg2, leg3, leg4\} \\
E^{chair} &= \{(seat, backrest), \\
&\qquad (seat, leg1), \\
&\qquad (seat, leg2), \\
&\qquad (seat, leg3), \\
&\qquad (seat, leg4)\}
\end{aligned}$$

5.2.2 Partial Views

Considering the different viewpoints from which a robot can observe a furniture. It is clear that its perception is a reduced version of the proposed representation since not all geometrical components are visible from any position.

Graphs representation correspond to the whole piece of furniture, then some subgraphs should be generated corresponding to different views that a robot can have.

Considering the following definition, a graph H is called a subgraph of G, such that $V(H) \subseteq V(G)$ and $E(H) \subseteq E(G)$.

Then a partial view Fp^i for a furniture is a subgraph from F^i, described by:

$$Fp^i = (\tilde{V^i}, \tilde{E^i}) \tag{5.10}$$

such that $\tilde{V^i} \subseteq V^i$ and $\tilde{E^i} \subseteq E^i$. There are as many partial views as subsets can be generated from F^i, however not all partial views are useful.

To generate a small set of sub-graphs corresponding to partial views, different points of view have been grouped in four quadrants, so there are: two graphs for the front left and right views and two more for the back view left and right (Figure 5.3). However, due to symmetry and lack of occlusions, the set of subgraphs can be reduced, for example for the dinning table, there is only

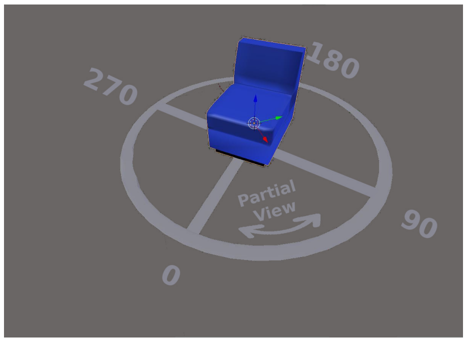


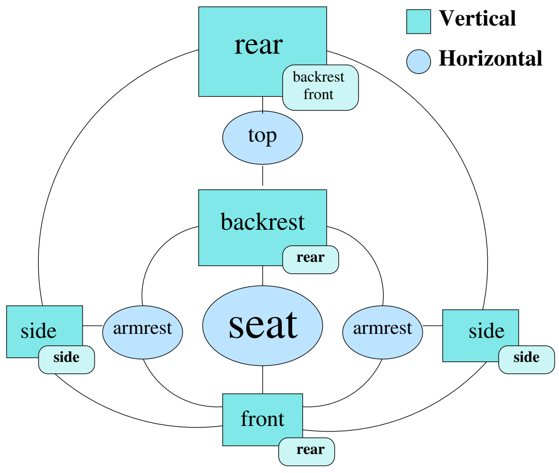
Figure 5.3: Visualization the different points of view used to generate partial views.

one graph without subgraphs, as its four legs can be seeing from many points of view. This subgraphs help us to improve and reduce the search space when we compare a scene graph with a model graph. If a piece of furniture has some opposite sides, the robot should not expect to view both.

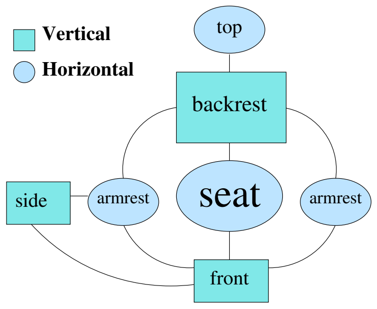
Consequently graphs require also to specify which planes are on opposite sides (if there are any), because this information is important in order to specify which components are visible from every view. Visibility of a given plane is encoded at the vertex. For example, for a chest of drawers is not possible to see the front and the back at the same time. It is not possible to see the right and the left side of a couch together. But, it is possible to see, at least partially, the four legs of a table.

The Figure 5.4 shows an example of a graph model and some sub-graphs for a couch graph model. It can be observed on Figure 5.4(a), the small rectangles to the side of the nodes indicating the opposite nodes. The sub-graphs (Figures 5.4(b) and 5.4(c)) represent two sub-graphs of the frontal views left and right respectively. The reduction of the graph can be observed, for example, in these cases the backrest and the front nodes are present, so the rear node is not, since it would not be visible for the robot from a front view of the furniture. Thus, a subgraph avoid to compare components which are not visible from a given point of view.

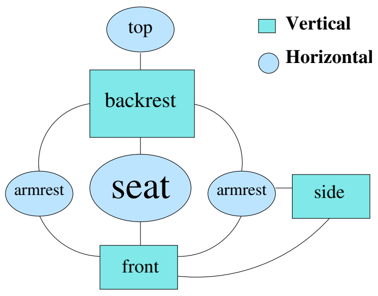
In order to match, robot perception with generated models, some simi-



(a)



(b)



(c)

Figure 5.4: In a) the graph corresponding to the complete couch graph and in b) and c) two sub-graphs for the front left and right views

larity measurements for Graphs and Geometric Components should be defined. At the following, these measurements will be defined.

5.2.3 Similarity Between Graphs

The similarity s_F of two furniture graphs (or partial graphs) F^i and $F^{i\prime}$ has been defined as:

$$s_F(F^i, F^{i\prime}) = \sum_{j}^{n_v} w_{Fj} s_{Gc}^k(Gc_j^k, Gc_j^{k\prime}) \qquad (5.11)$$

where w_{Fj} are weights, corresponding to the contribution of the similarity s_{Gc} between the corresponding geometric components j to the graph model F^i.

In next section, values for proposed equations (5.2) and (5.11), in a specific context and environment will be provided.

5.3 Determination of values for models and geometric components

In order to validate the proposed approach, a home environment has been used. This environment is composed of six different pieces of furniture ($N_f = 6$), which are:

$$\mathcal{F} = \{dinning\ table, chair, couch, center\ table, \\ bed, chest\ of\ drawers\}$$

To represent the components of those pieces of furniture, it has been selected the following three types of geometric components:

$$\mathcal{G}c = \{horizontal\ plane, vertical\ plane, legs\}$$

and the features of the geometric components are the described on section 5.1.1.

5.3.1 Weights for the geometrical components comparison

In order to compute the proposed similarity s_{Gc} between two geometrical components, it is necessary to determine the corresponding weights w_{Gi} in (Eq. 5.2). At this stage of the work, those values have been determined empirically, as follows:

From a set of scenes taken by the robot, it was selected a set of them where each piece of furniture were totally visible. Then, geometric components were extracted following the methodology proposed on section 5.1. The weights then were selected according to the importance of each feature to a correct classification of the geometric component.

Table 5.1, shows the corresponding weights for the three geometric components.

	w^k_{G1} (height)	w^k_{G2} (h. deviation)	w^k_{G3} (area)
w^H_{Gi} (horizontal)	0.65	0.15	0.25
w^V_{Gi} (vertical)	0.5	0.2	0.3
w^L_{Gi} (legs)	0.5	0.2	0.3

Table 5.1: Weights for similarity estimation

5.3.2 Uncertainty

Uncertainty values in Eq 5.3, were estimated also by an empirical process. From some views selected for each piece of furniture, where it was fully observable, it was calculated the differences with its correspondent model; in order to have an estimation of the variation of corresponding values, with the complete geometric component. Then the biggest difference for each characteristic was selected as the uncertainty. As it can be seen on the graphic of Figure 5.1, the use of characteristics as: height, height deviation and area; are enough to classify main horizontal planes. Moreover, over this space, characteristics and uncertainty from each horizontal plane makes regions fully classifiable.

There are other features of the geometrical components not used for similarity computation, but that are helpful to define the type of geometrical component or their relations, so they have also been added to the vertex structure; these features are:

- **Center:** The 3D point center of the points that make up the geometrical component.

- **PCA eigenvectors and eigenvalues:** Eigenvector and eigenvalues resulting of a PCA analysis to the region points.

These values are used to verified the type of geometric component, particularly to discriminate between vertical planes and poles, because they have been extracted with the same process.

5.3.3 Weights for the graphs comparison

In a different way, as weights were determined for (Eq. 5.3), the weights for the similarity between graphs (Eq. 5.11) were calculated based on the total area of models for each piece of furniture.

Given the projected area of each geometric component of the graph model the total area is calculated. Then, the weights for each vertex (geometric component) has been defined as the percentage of its area in comparison to the total area. Moreover, when dealing with a sub-graph from a model, the total area is determined by the sum of areas from the nodes from that particular view (sub-graph). Thus, there is a particular weight vector for each graph and subgraph in the environment.

In Table 5.2, are shown the values of computed areas for the chest of drawers corresponding to the graph and subgraph of the model (Figure 5.5).

	side	front	side	back	top
area	1575.5	9155.5	1575.5	9155.5	1914
Model %	6.74	39.16	6.74	39.16	8.19
Partial View %	12.45	72.41			15.13
Partial View %		72.41	12.45		15.13

Table 5.2: Example of the weights for the comparison of the chest of drawers graph, based on the area of each geometric component

Table 5.3 shows the weights in the case of the dinning table, which does not have sub-graphs (Figure 5.2(a)).

In this chapter we have explained the graph representation proposed to

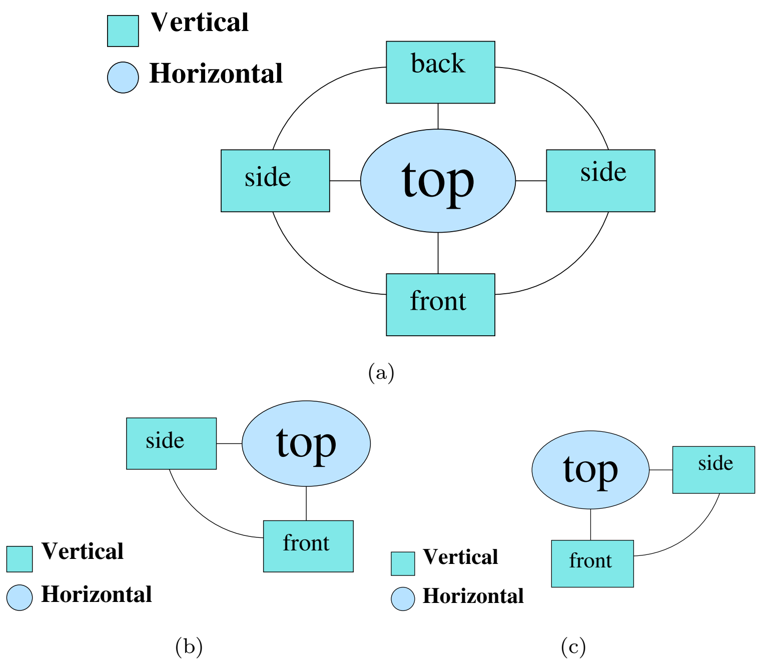


Figure 5.5: Graphs models for the chest of drawers, in a) the graph for the full model, in b) and c) graphs for partial front views, left and right, respectively

characterize the furniture and how the comparisons will be made. In the next chapter, we will present the evaluation of propose approach and the results obtained.

	table	leg	leg	leg	leg
area	8159	947	947	947	947
Model %	68.31	7.92	7.92	7.92	7.92
Partial View %	68.31	7.92	7.92	7.92	7.92

Table 5.3: Example of the weights for the comparison of the dining table graph, based on the area of each geometric component

Chapter 6

Evaluations

Considering an observation of a scene, where geometrical components has been extracted, by applying the methods described on section 5.1.

Be O the set of all geometric components observed on a scene, then:

$$O = \{O^1, ..., O^{N_k}\} \tag{6.1}$$

where O^k are the subsets of geometrical components of the type k.

In this way, observed horizontal geometrical components found on the scene are on the same subset, lets say O^*, then it is possible to extract each one of them in the subset and then compare them to the main nodes for each furniture graph.

Once the similarity between the horizontal components on the scene and the models has been calculated, all the categories with the similarity higher than certain threshold are chosen as probable models for each horizontal component. A graph is then constructed for each horizontal component, where adjacent geometrical components are merged to it. Then this graph is compared with the sub-graphs of the probable models, previously selected.

The scene in Figure 6.1 is composed by a dinning table and a chair. After the geometrical components are extracted, the two horizontal planes cor-

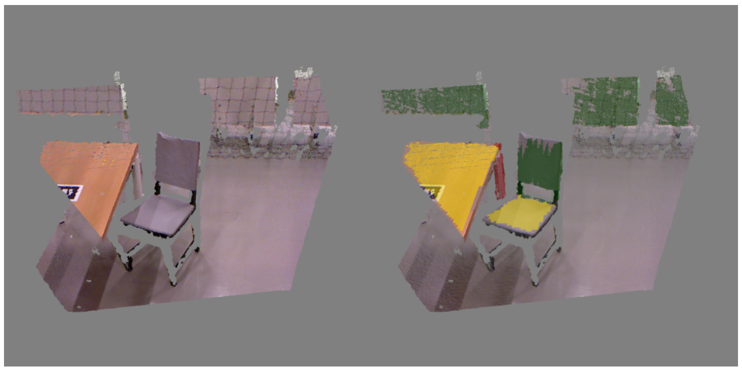


Figure 6.1: Example of the geometric components detected, in a) a scene from a living room and in b) the geometric components coloured .

responding to the dinning table and the chair are selected.

A comparison of those horizontal components to each one of the main nodes of the furniture graphs is performed. The similarities computed can be observed in table 6.1. It can be observed for the plane labelled as "H0" that there are two similarities higher than the previously defined threshold of 0.7, the table and the chest of drawers. For the plane labelled "H01" there are three probable models (chair, center table and couch).

	table	chair	bed	couch	c of drawers	c table
H00	**0.8743**	0.0	0.4780	0.5128	**0.8014**	0.6270
H01	0.5794	**0.9891**	0.6977	**0.7277**	0.5929	**0.8895**

Table 6.1: Similarities measures of two geometrical components

Next, graphs are constructed for each horizontal plane (the main node) and adding its adjacent components; in this case, both graphs have only one adjacent node (Figure 6.2). Those planes previously detected that are not attached to any graph (like the walls) are now discarded.

The Figure 6.3(a) shows the generated graph ("G0") from the plane "H00" on the scene. Figures 6.3(b) and 6.3(c) shows and the partial-views graphs from the selected probable models. Fig 6.3(b) for the table and Fig 6.3(c).

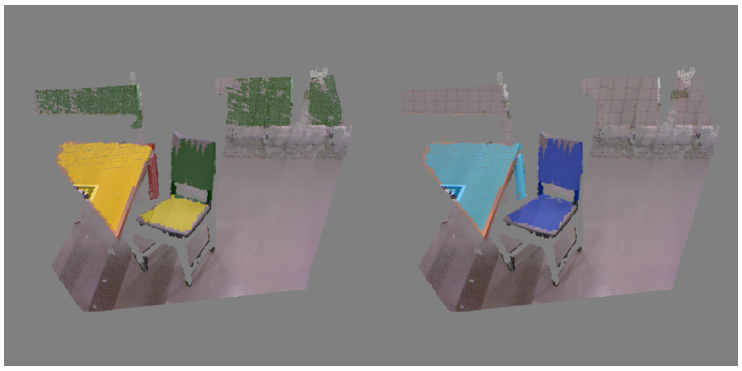


Figure 6.2: Example of graphs created, in a) the different geometric components coloured and in b) two graphs created with those geometric components.

It can be observed that "G0" has an adjacent leg node, so it can only be a sub-graph for the table graph since the chest of drawer graph has only adjacent vertical nodes.

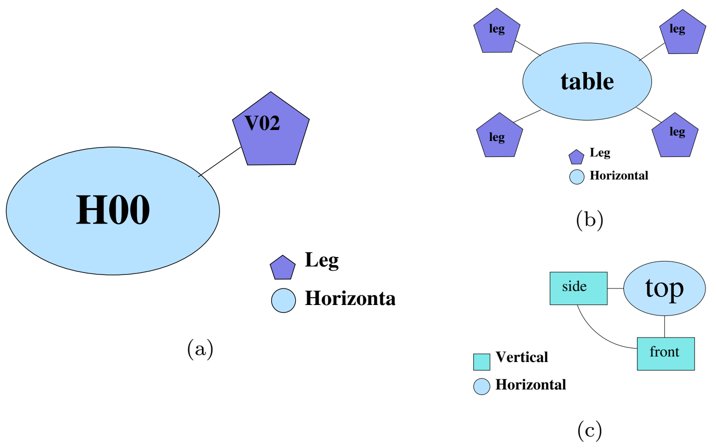


Figure 6.3: Graph comparison. In a) one of the graph generated from the scene in Fig 6.5(b), in b) and c) partial graph from the table and the chest of drawers

The graph "G1", created from the plane "H01" is shown in Figure 6.4(a). Figures 6.4(b), 6.4(c) and 6.4(d) show the partial view graph for the selected probable models; chair, table and couch, respectively.

It can be observed that the graph "G1" has a vertical node, so it is matched to the backrest node of the chair and of the couch (Figs. 6.4(b) and 6.4(d)); and there are no match with the center table graph (Figure 6.4(c)).

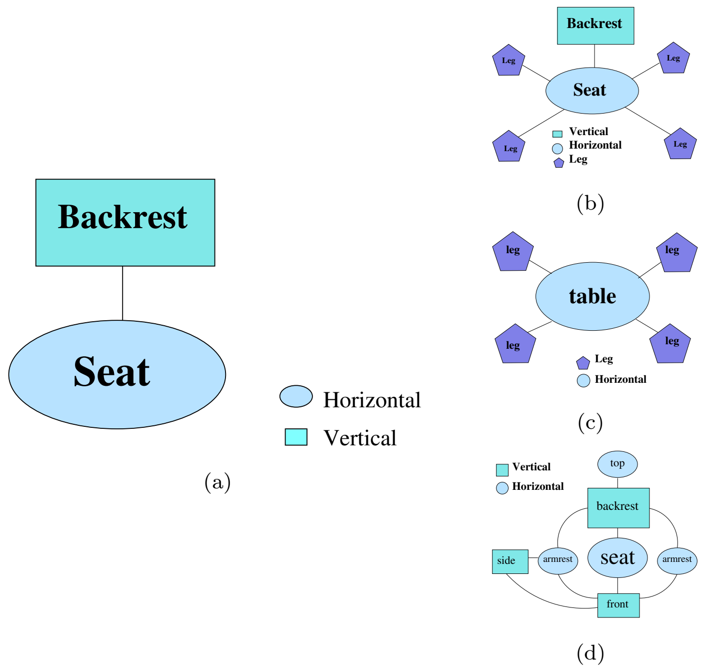


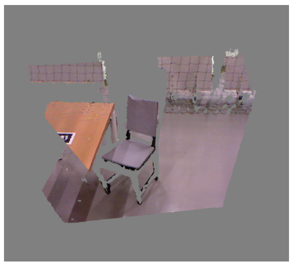
Figure 6.4: Graph comparison. In a) one of the graph generated from the scene in Fig 6.5(b), in b), c) and d) partial graph from the chair, the center table and the couch.

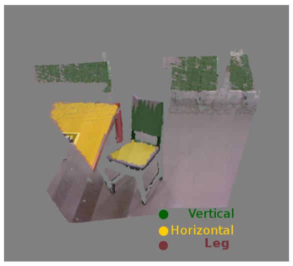
The similarity for adjacent nodes are noted in the table 6.2. Graph similarity is calculated with Eq. 5.11 and shown in the last column of the table. The "G0" is selected as a table and "G1" as a chair (Figure 6.5(c)). It should be noted that the "G0" is selected as a table even when its detected horizontal planes is incomplete, however the amount of points in this case was enough to discriminate it from the other models.

Figure 6.5 shows the results of applying the described procedure to different scenes with different types of furniture. The first column (Figs. 6.5(a), 6.5(d), 6.5(g) and 6.5(j)) shows the point clouds from the scenes. The column at the center (Figs. 6.5(b), 6.5(e), 6.5(h) and 6.5(k)) show the geometrical components found on the corresponding scene. Finally the last column (Figs. 6.5(c), 6.5(f), 6.5(i) and 6.5(l)) show the generated graphs that correctly classified.

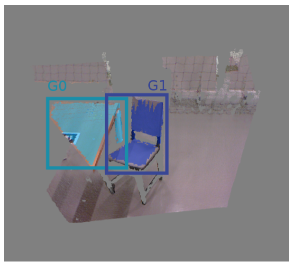
Main Node	Main Node Sim.	Adjacent Nodes	Adjacent Node Sim.	Graph Similarity
H00	Table 0.8743	V02	Table leg 0.7015	0.0.6670
H00	Chest of Drawers 0.8014	V02	No Match	X
H01	Chair 0.9891	V01	Backrest 0.8878	0.5740
H01	Center Table 0.8895	V01	No match	X
H01	Couch 0.7277	V01	Backrest 0.7150	0.3204

Table 6.2: Example for graph classification

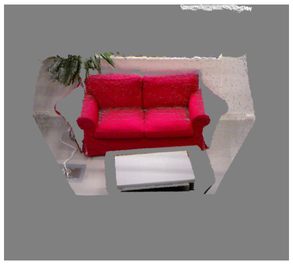

(a)

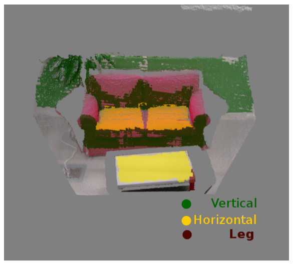


(b)

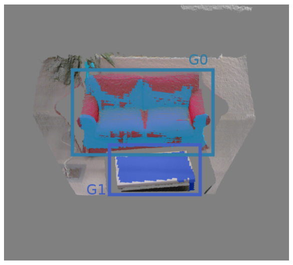


(c)

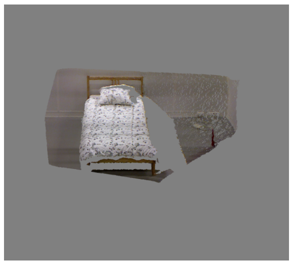

(d)

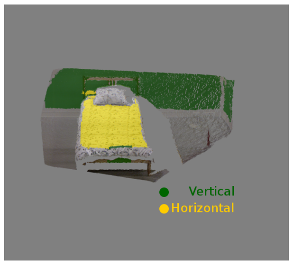


(e)

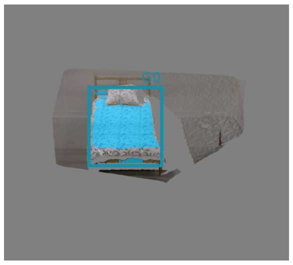


(f)

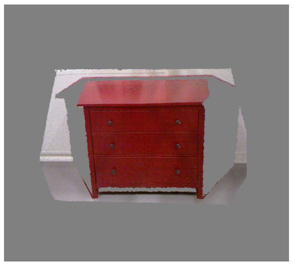

(g)

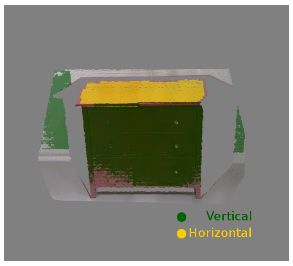


(h)

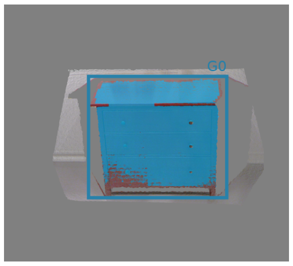


(i)


(j)



(k)



(l)

Figure 6.5: Results for the furniture detection, each row shows one scene. In a), d), g) and j) the original point clouds. In b), e), h) and k) are shown the geometric components found, the vertical planes are in color green, the horizontals in yellow and the legs in red. The bounding boxes in c), f), i) and l) show the graph generated that were correctly identified as a piece of furniture

Chapter 7

Conclusions

The obtained conclusions for this work can be listed as:

- We propose a graph representation for pieces of furniture, based on geometrical components. Where each one of these geometrical components correspond roughly to a different part of the furniture. This representation proved to be useful for furniture detection.
- Representation of geometrical components based on characteristics extracted from 3D data
- We develop a method for extracting geometrical components. The algorithm proposed for the extraction of 3D horizontal planes is based on an analysis of height histogram. This approach allows a linear computational time which is an improvement from the common plane extractions algorithms. For vertical planes a similar approach is used based on 2D histograms.
- We propose two effective similarity measures. One for comparing geometrical components and another for comparing graphs for models and scenes
- To validate this approach we perform several evaluations in two different house-like environments at the

In the future, other geometrical components or characteristics could be added, this could be done easily thank to the nature of our representation. Having more geometrical components would allow us to also incorporate new types of furniture.

Bibliography

[1] Radhakrishna Achanta, Appu Shaji, Kevin Smith, Aurelien Lucchi, Pascal Fua, and Sabine Süsstrunk. Slic superpixels compared to state-of-the-art superpixel methods. *IEEE transactions on pattern analysis and machine intelligence*, 34(11):2274–2282, 2012.

[2] Luıs A Alexandre. 3d descriptors for object and category recognition: a comparative evaluation. In *Workshop on Color-Depth Camera Fusion in Robotics at the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), Vilamoura, Portugal*, volume 1, page 7, 2012.

[3] Oscar Alonso-Ramirez, Antonio Marin-Hernandez, Michel Devy, and Fernando M Montes-Gonzalez. Indoor home furniture detection with rgb-d data for service robots. In *2014 International Conference on Electronics, Communications and Computers (CONIELECOMP)*, pages 172–177. IEEE, 2014.

[4] S. Y. An, L. K. Lee, and S. Y. Oh. Fast incremental 3d plane extraction from a collection of 2d line segments for 3d mapping. In *2012 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 4530–4537, Oct 2012.

[5] G. Arbeiter, S. Fuchs, R. Bormann, J. Fischer, and A. Verl. Evaluation of 3d feature descriptors for classification of surface geometries in point clouds. In *2012 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 1644–1650, Oct 2012.

[6] Herbert Bay, Andreas Ess, Tinne Tuytelaars, and Luc Van Gool. Speeded-up robust features (surf). *Computer vision and image understanding*, 110(3):346–359, 2008.

[7] Paul J Besl and Neil D McKay. Method for registration of 3-d shapes. In *Sensor fusion IV: control paradigms and data structures*, volume 1611, pages 586–606. International Society for Optics and Photonics, 1992.

[8] T. Birdal and S. Ilic. Point pair features based object detection and pose estimation revisited. In *2015 International Conference on 3D Vision*, pages 527–535, Oct 2015.

[9] L. Bo, X. Ren, and D. Fox. Depth kernel descriptors for object recognition. In *2011 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 821–826, Sept 2011.

[10] Pedro MR Caleiro, António JR Neves, and Armando J Pinho. Color-spaces and color segmentation for real-time object recognition in robotic applications. *Revista do DETUA*, 4(8):940–945, 2007.

[11] C. Choi, Y. Taguchi, O. Tuzel, M. Y. Liu, and S. Ramalingam. Voting-based pose estimation for robotic assembly using a 3d sensor. In *2012 IEEE International Conference on Robotics and Automation*, pages 1724–1731, May 2012.

[12] Frank Dellaert, Dieter Fox, Wolfram Burgard, and Sebastian Thrun. Monte carlo localization for mobile robots. In *ICRA*, volume 2, pages 1322–1328, 1999.

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

[14] Bertram Drost, Markus Ulrich, Nassir Navab, and Slobodan Ilic. Model globally, match locally: Efficient and robust 3d object recognition. In *CVPR*, volume 1, page 5, 2010.

[15] Open Source Robotics Foundation. gmapping - ros wiki.

[16] Open Source Robotics Foundation. Ros.org powering the world's robots.

[17] Cipriano Galindo, Alessandro Saffiotti, Silvia Coradeschi, Pär Buschka, Juan-Antonio Fernandez-Madrigal, and Javier González. Multi-hierarchical semantic maps for mobile robotics. In *2005 IEEE/RSJ international conference on intelligent robots and systems*, pages 2278–2283. IEEE, 2005.

[18] Martin Günther, Thomas Wiemann, Sven Albrecht, and Joachim Hertzberg. Model-based furniture recognition for building semantic object maps. *Artificial Intelligence*, 247(Supplement C):336 – 351, 2017. Special Issue on AI and Robotics.

[19] Paul S Heckbert and Michael Garland. Survey of polygonal surface simplification algorithms. Technical report, CARNEGIE-MELLON UNIV PITTSBURGH PA SCHOOL OF COMPUTER SCIENCE, 1997.

[20] Jose-Juan Hernandez-Lopez, Ana-Linnet Quintanilla-Olvera, José-Luis López-Ramírez, Francisco-Javier Rangel-Butanda, Mario-Alberto Ibarra-Manzano, and Dora-Luz Almanza-Ojeda. Detecting objects using color and depth segmentation with kinect sensor. *Procedia Technology*, 3:196–204, 2012.

[21] Dirk Holz, Stefan Holzer, Radu Bogdan Rusu, and Sven Behnke. Real-time plane segmentation using rgb-d cameras. In *Robot Soccer World Cup*, pages 306–317. Springer, 2011.

[22] A. E. Johnson and M. Hebert. Using spin images for efficient object recognition in cluttered 3d scenes. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 21(5):433–449, May 1999.

[23] Andrew E Johnson. Spin-images: a representation for 3-d surface matching. 1997.

[24] Yohei Kakiuchi, Ryohei Ueda, Kei Okada, and Masayuki Inaba. Creating household environment map for environment manipulation using color range sensors on environment and robot. In *2011 IEEE International Conference on Robotics and Automation*, pages 305–310. IEEE, 2011.

[25] Kevin Lai, Liefeng Bo, Xiaofeng Ren, and Dieter Fox. A large-scale hierarchical multi-view rgb-d object dataset. In *2011 IEEE international conference on robotics and automation*, pages 1817–1824. IEEE, 2011.

[26] Marc Levoy. The stanford 3d scanning repository, 2014.

[27] Ziyuan Liu, Wei Wang, Dong Chen, and Georg von Wichert. A coherent semantic mapping system based on parametric environment abstraction and 3d object localization. In *2013 European Conference on Mobile Robots*, pages 234–239. IEEE, 2013.

[28] David G Lowe. Distinctive image features from scale-invariant keypoints. *International journal of computer vision*, 60(2):91–110, 2004.

[29] Guido Manfredi, Sandra Devin, Michel Devy, and Daniel Sidobre. Autonomous apartment exploration, modelling and segmentation for service robotics. *IFAC-PapersOnLine*, 49(15):120–125, 2016.

[30] Zoltan Csaba Marton, Radu Bogdan Rusu, Dominik Jain, Ulrich Klank, and Michael Beetz. Probabilistic categorization of kitchen objects in table settings with a composite sensor. In *IROS*, pages 4777–4784, 2009.

[31] Julian Mason and Bhaskara Marthi. An object-based semantic world model for long-term change detection and semantic querying. In *2012 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 3851–3858. IEEE, 2012.

[32] Michael E Moran. Evolution of robotic arms. *Journal of robotic surgery*, 1(2):103–111, 2007.

[33] O. M. Mozos, Z. C. Marton, and M. Beetz. Furniture models learned from the www. *IEEE Robotics Automation Magazine*, 18(2):22–32, June 2011.

[34] Matthias Nießner, Michael Zollhöfer, Shahram Izadi, and Marc Stamminger. Real-time 3d reconstruction at scale using voxel hashing. *ACM Transactions on Graphics (TOG)*, 32(6):169, 2013.

[35] Andreas Nüchter and Joachim Hertzberg. Towards semantic maps for mobile robots. *Robotics and Autonomous Systems*, 56(11):915–926, 2008.

[36] Dejan Pangercic, Benjamin Pitzer, Moritz Tenorth, and Michael Beetz. Semantic object maps for robotic housework-representation, acquisition and use. In *2012 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 4644–4651. IEEE, 2012.

[37] Charles R. Qi, Hao Su, Matthias Niessner, Angela Dai, Mengyuan Yan, and Leonidas J. Guibas. Volumetric and multi-view cnns for object classification on 3d data. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2016.

[38] Md Alimoor Reza and Jana Kosecka. Object recognition and segmentation in indoor scenes from rgb-d images. In *Robotics Science and Systems (RSS) conference-5th workshop on RGB-D: Advanced Reasoning with Depth Cameras*, 2014.

[39] M. Ruhnke, B. Steder, G. Grisetti, and W. Burgard. Unsupervised learning of 3d object models from partial views. In *2009 IEEE International Conference on Robotics and Automation*, pages 801–806, May 2009.

[40] S. Rusinkiewicz and M. Levoy. Efficient variants of the icp algorithm. In *Proceedings Third International Conference on 3-D Digital Imaging and Modeling*, pages 145–152, 2001.

[41] R. B. Rusu, N. Blodow, and M. Beetz. Fast point feature histograms (fpfh) for 3d registration. In *2009 IEEE International Conference on Robotics and Automation*, pages 3212–3217, May 2009.

[42] R. B. Rusu, Z. C. Marton, N. Blodow, A. Holzbach, and M. Beetz. Model-based and learned semantic object labeling in 3d point cloud maps of kitchen environments. In *2009 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 3601–3608, Oct 2009.

[43] Radu Bogdan Rusu, Gary Bradski, Romain Thibaux, and John Hsu. Fast 3d recognition and pose using the viewpoint feature histogram. In *2010 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 2155–2162. IEEE, 2010.

[44] R. F. Salas-Moreno, R. A. Newcombe, H. Strasdat, P. H. J. Kelly, and A. J. Davison. Slam++: Simultaneous localisation and mapping at the level of objects. In *2013 IEEE Conference on Computer Vision and Pattern Recognition*, pages 1352–1359, June 2013.

[45] Agnes Swadzba and Sven Wachsmuth. A detailed analysis of a new 3d spatial feature vector for indoor scene classification. *Robotics and Autonomous*

Systems, 62(5):646 – 662, 2014. Special Issue Semantic Perception, Mapping and Exploration.

[46] A. J. B. Trevor, J. G. Rogers, and H. I. Christensen. Planar surface slam with 3d and 2d sensors. In *2012 IEEE International Conference on Robotics and Automation*, pages 3041–3048, May 2012.

[47] T. Varvadoukas, E. Giannakidou, J. V. Gomez, and N. Mavridis. Indoor furniture and room recognition for a robot using internet-derived models and object context. In *2012 10th International Conference on Frontiers of Information Technology*, pages 122–128, Dec 2012.

[48] Wei Wang, Lili Chen, Dongming Chen, Shile Li, and Kolja Kühnlenz. Fast object recognition and 6d pose estimation using viewpoint oriented color-shape histogram. In *2013 IEEE International Conference on Multimedia and Expo (ICME)*, pages 1–6. IEEE, 2013.

[49] Zhirong Wu, S. Song, A. Khosla, Fisher Yu, Linguang Zhang, Xiaoou Tang, and J. Xiao. 3d shapenets: A deep representation for volumetric shapes. In *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1912–1920, June 2015.

[50] M. Wunstel and R. Moratz. Automatic object recognition within an of .ce environment. In *First Canadian Conference on Computer and Robot Vision, 2004. Proceedings.*, pages 104–109, May 2004.