KTH Computer Science
and Communication

Defining and evaluating qualitative spatial relations for describing desktop scenes

ADRIÀ GALLART DEL BURGO

Master's Thesis at NADA
Supervisor: Patric Jensfelt / John Folkesson
Examiner: Stefan Carlsson

November, 2013

Abstract

In an everyday scene such as an office, one can find lots of objects with different sizes and shapes. Normally, these objects are placed following a certain order. Describing the different spatial relations between the objects in an environment is a simple task for a human being but it is not for a robot. This is due to the nature of these relations being abstract and this requires certain degree of intelligence in a robot.

This thesis investigates ways for a robot to find descriptions in a scene using spatial relations such as "the mouse is to the right of the keyboard". To deduce these relationships, accurate information about the position and orientation of the objects is needed and to achieve this purpose a software was developed. In order to use this information to extract descriptions of a scene, a system was implemented to quantify to what degree certain relations hold. Finally, several scenes were analyzed, by obtaining its respective description, trying to extract some patterns.

Referat

Utformning och utvärdering av kvalitativa spatiala relationer för att beskriva av skrivbordsmiljöer

I vardagliga miljöer som t.ex. ett kontor, kan man hitta en mängd olika objekt i olika storlekar och former. Normalt är dessa objekt placerade enligt vissa mönster. För oss människor är det enkelt att beskriva de rumsliga relationerna mellan objekt, men det är mycket svårt för en robot. Dessa relationer är abstrakta och kräver en relativt hög nivå av resonerande för att kunna hanteras. Denna rapport utreder en robots möjligheter att själv kunna generera beskrivningar för en scen med hjälp av rumsliga relationer som t.ex. att "musen är till höger om tangentbordet". För att härleda dessa relationer är korrekt information om position och orientering för föremålen väldigt viktigt. För att få fram denna information har en mjukvara utvecklats som kan användas för att i insamlad data markera var objekt finns. Data från kontorsmiljöer har samlats in och olika relationer har definierats och undersökts för olika scener.

Acknowledgements

First, I would like to express my gratitude to my supervisors, *Patric Jensfelt* and *John Folkesson*, for giving me the opportunity to develop my thesis with them. Thank you for your guidance, your advices, your supervision and your patience.

I would like to thank all the people of the Computer Vision and Active Perception Lab for their kindness and hospitality. Especially to *Akshaya Thippur* for helping me a lot. Also to all the people who played innebandy every Wednesday for teaching me how to play this amazing sport. To my "Farsta friends", *Matteo*, *Maddi*, *Jonas*, *Jan*, *Denis*, *Clarisse*, *David*, *Gabi*, *Timo*, *Büsra* and *Marco* for all the great moments we had. I will never forget any moment we spent together playing Bang and Werewolves, skating, going out, travelling, etc.

Last but not least, I want to thank my family; to my parents and brothers for understanding all my decisions and giving me the chance to study abroad. Thank you for all your love and support.

Finally, I want to thank *Cris*, for always being by my side. For your constant love in good and bad moments.

Adrià Gallart del Burgo
Stockholm, November 2013

List of Acronyms

QSR Qualitative Spatial Relations

CVAP Computer Vision and Active Perception Laboratory

RCC Region Connection Calculus

PCL Point Cloud Library

PCD Point Cloud Data

ROS Robot Operating System

NADA Numerical Analysis and Computer Science

RGB Red, Green and Blue

RANSAC RANdom SAmple Consensus

GUI Graphical User Interface

Contents

Chapter 1

Introduction

The scope of Robotics is expanding, from the traditional industrial applications to service robots. In the last few years robotics is making more effort on the progress of the service robots rather than in industrial robots due to the great development already made in previous years. Service robots - robots that work in an autonomous way with the aim to help humans in a wide variety of purposes - have to work in unpredictable environments. So if we want to have robots with the ability to work in such environments, they need to know how to represent and understand them, and to do that, we believe *spatial relations* play an important role.

Spatial relations are abstract concepts that humans use to describe the location and configuration of objects. If we endow robots with the ability to describe an environment using spatial relations we have made an advance on giving more intelligence to robots. This progress would be done whenever a robot will be able to describe a scene by saying things like: "The mouse is to the right of the keyboard", "The pen is on the book" or "The mug is near the lamp". However, an abstract concept such as spatial relations is not easy to implement in a machine as there is not an exact and specific definition of how humans interpret all different relations. There are a lot of factors that depend on the interpretation of spatial relations between objects in a human being such as shapes, distances or angles.

If we give robots the ability to observe a scene and describe it in terms of relations between all relevant objects, we get several improvements:

- Robots will have the possibility to find usual behaviours (patterns) in terms of distribution. These could allow a robot to help a human, for instance by alerting whenever there is some anomaly in the distribution of an environment.
- Improve the interactivity and communication with humans. Thinking as humans do in terms of spatial relations could give to the robot the ability to communicate using sentences that humans normally use.
- A robot will be able to understand how all the objects are located in a *human-*

oriented environment.

Social robots have always been seen by the majority of humans as funny machines with a bit of intelligence that can do things such as moving through a place without colliding, recognizing a person using its sensors or manipulating a set of objects with talent. Nevertheless, giving intelligence to a robot, with the previous mentioned abilities, can change the vision of these people and make them believe that a robot is more like a human than a machine. Imagine a robot that can search for objects given the instructions of a person and this person asks the robot to find an object that, for instance, is to the right of the laptop. How the robot can search the object if he does not know what "to the right of" means?

Robots have to be able to transform the data observed from the sensors into human expressions and vice versa, and this is not an easy job. How can we give robots the capacity of reasoning? How can we find and quantify spatial relations?

1.1 Problem statement

This work is a part of an European project called STRANDS[1], where the final goal is to develop robots capable of running autonomously during long periods of time in a human environment. These robots have to be able to explore and understand the 3D space of their worlds. If we achieve this purpose, a robot can be able to detect changes in the distribution of the environment and it will be able to alert if something is wrong. This ability can be used to create intelligent security robots. The acknowledge of the changes in an environment can be also used for a robot to avoid situations that can make it less productive. Imagine that a robot is working in a hospital and some corridors of the hospital, depending on the moment of the day, are more concurred. If the robot knows this behaviour, it could be able to avoid passing through this different places in order to be faster in its task.

The overall goal of our work is to develop methods that allow robots to work with qualitative spatial relations (QSR), which describe the relationship between objects in space. To obtain a good description of a scene, we should select a set of spatial relations that will allow us to describe a scene with sufficient details to make it useful. This selection will be made after analyzing some scenes trying to find a more suitable selection.

An important thing to define such qualitative spatial relation is to think about which are the parameters that can be used to define them and how this could be extracted from the data obtained by a robot. Thus, after the selection of a set of relations and parameters used to define them, we will develop novel approaches to

[1] `http://strands.acin.tuwien.ac.at/index.html`

evaluate all these relations.

After obtaining several descriptions of different scenes, it is possible to extract some information about objects distribution. This could be done by obtaining patterns observed in the data.

1.2 Contributions of the thesis

This thesis makes some contributions on the representation of a specific scene, a typical tabletop, using qualitative spatial relations. The most suitable qualitative spatial relations for encoding knowledge about this kind of scenarios (tabletops) was investigated. A novel approach was developed by formalizing different spatial relations that allowed us to obtain a description of a scene using these relations between objects. After analyzing several scenes, a set of spatial relations were selected. Specifically, the qualitative spatial relations chosen to define our scenes were "in front of", "to the right of", "to the left of" and "behind".

In order to evaluate the model we needed to collect data. This data was obtained using the typical sensors that are used for robots. Specifically, for each scene the data obtained was a point cloud, which provide the robot the possibility to perceive the environment with details. This data was gathered in the Computer Vision and Active Perception Laboratory (CVAP) at the Royal Institute of Technology (KTH) in Stockholm, Sweden. A total of 42 scenes were collected and annotated.

Before getting the spatial relations between the objects, the robot has to find the location of them. An important contribution of this thesis was the development of an annotation tool which allows the user to annotate all the objects of a scene represented as a point cloud. This tool, named 3D Annotation Tool, was developed from scratch and it is available online at `https://github.com/adria-gallart/3d_annotation_tool`. The 3D Annotation Tool works with the point cloud data and it is capable of doing things such as plane detection, plane segmentation or object annotation.

Using the information of every scene, a set of experiments were performed to find how suitable a relation is to describe the position of two objects. Analyzing all descriptions extracted, some patterns on the location of objects were found.

1.3 Outline

The report is organized as follows. The second chapter, Related Work, describes work done by other people. Chapter 3 presents what is needed to find qualitative spatial relations of a scene and the formalization of models used to define some of these relations. Chapter 4 deals with the data collection, detailing the systems used to gather the data and how it is obtained. Chapter 5 presents the design and the implementation of the annotation tool developed called 3D Annotation Tool. Chapter 6 describes the data obtained. Chapter 7 shows the results of the experiments and Chapter 8 presents the conclusions of the thesis and suggests some future work.

Chapter 2

Analysis of Related Work

There has been research on how to represent the environment using qualitative spatial relations. To see a general vision of this, [1] cites and comments some of the techniques used for qualitative spatial representation and reasoning. It explains different areas of applications that qualitative spatial reasoning can be used for. [1] also focuses on which are the aspects (sizes, shapes, orientation or distance) used for different kinds of spatial relations. At the end of this work, the usage of spatial reasoning is expanded by speaking about how to reason utilizing spatial change.

Describing scenes using qualitative spatial relations between objects using prepositions such as "in front of", "behind" or "to the right" is not easy due to the difficulty to find some formal way of describing these prepositions. [2] suggests that spatial relationships can be defined using three variable: places, directions and distance. It is also suggested which of these factor are considered to define relations such as "below", "near" or "behind".

Some research has focused on finding qualitative spatial relations in 2-dimensional domain. One of the formalism used in lots of works referred to qualitative spatial reasoning is the *Region Connection Calculus* (RCC). RCC, first explained in [3] and [4], try to describe how regions are related depending on their position in an abstract way. We can see the two-dimensional examples used for the RCC-8 (eight base relations) in Figure 2.1, where DC is disconnected, EC is externally connected, EQ is equal, PO is partially overlapping, TPP is tangential proper part and NTPP is non-tangential proper part. All the possible combinations can be built using these basic relations.

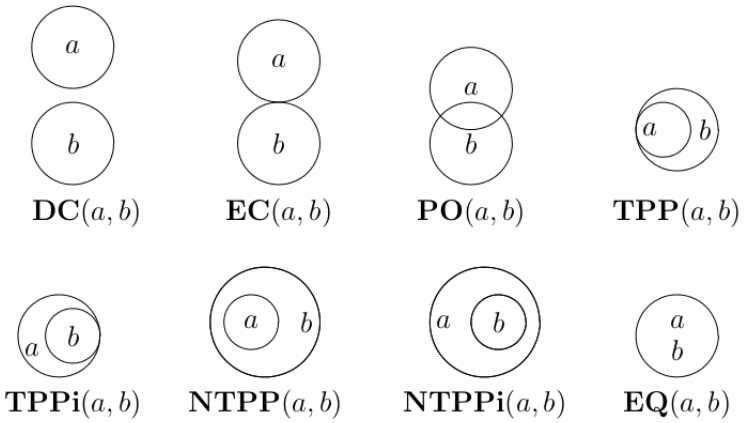


Figure 2.1: RCC-8 base relations (image from [5])

In [6], Lockwood *et al.* presented a model that can differentiate the relation between two figures inside an image between "in", "on", "above", "below" and "left". It is explained how the model is developed and trained using hand-drawn sketches as a input. Also two experiments are performed.

[7] describes how is possible to use video input to generate qualitative reasoning. Their aim is to apply qualitative spatial-temporal reasoning techniques to real world. For this purpose they used the data obtained from a static camera located in a highway. After analyzing their results, one of the conclusions they made is that they can classify a lot of events by only selecting a small number of qualitative spatio-temporal descriptions. In [8], Terry Regier and Laura A. Carlson proposed a model to measure quantitatively the spatial relation "above" between two objects. This model, *Attention Vector Sum*, is chosen after comparing with other models such as *bounding box model*, *proximal and center-of-mass model* or *hybrid model*.

The aim of [9] is to obtain new relationships between objects from some other known relationships. Sistla *et al.* worked with a pictorial database and they considered the following spatial relationships: "left of", "right of", "in front of", "behind", "above", "below", "inside", "outside" and "overlaps". They assume that in the database, for each picture there is a description about the objects and some relations between them. In addition, it is shown that the given set of rules is incomplete for two-dimensional systems but it is complete for three dimensional systems.

Focusing on research done using three-dimensional data, [10] decided to work with point clouds. It used the relationships "on" and "adjacent" and it is presented an algorithm that tries to extract the structure of the objects and find the spatial relationships between them. They create a structure where it is possible to see if

objects are touching. In Figure 2.2, it is shown how they represent a scene using this algorithm. In [11], Kasper *et al.* analyzed the real world trying to find spatial relationships between objects. Specifically, they focused on working in an office space. The process of data acquisition and how this data is annotated is explained. In their results, they evaluate things such as probability of appearance, object sizes, euclidean distance and relative position between objects.

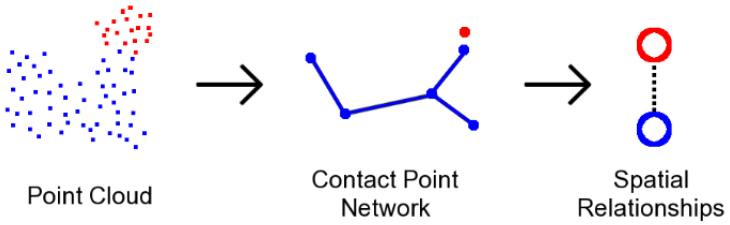


Figure 2.2: An illustration of the layered scene representation (image from [10])

In [12], Sjöö *et al.* proposed a system that deals with the representation of three-dimensional scenes using qualitative spatial relations. They focused on two topological spatial relations such as "on" and "in". They also presented a formalization for these relations. This work is based on work published in [13] and [14]. In [13] there is a more detailed explanation of the spatial relation "on", describing how it is possible to get a measurable value for this relation. Also some experiments are performed in order to check that the model implemented represents the desired behaviour. In [14], it is presented how a robot can use spatial relations between objects in order to do some search actions.

There are some research focussing on the analysis of the language expressions referring to space in a more linguistic way. [15] has defined three types of elements that classify spatial expressions in different groups depending on the meaning. These three elements are:

- Geometrical properties.
- Functional properties.
- Pragmatic principles.

There is also some bibliography which analyzes how people think and feel about space. In [16], among multiple other things, we learned how humans tend to use appropriate descriptions about space. An example to show that could be the answer to a question such as "Where is the laptop?". It will be difficult to imagine a situation in the real world where the answer would be "the desk is under the laptop" rather than "the laptop is on the desk".

Chapter 3

Qualitative Spatial Relations

Spatial relations are specifications about how an entity is located in space in relation to some reference. For us, these relations will be between objects. Before a robot will be able to describe a scene using qualitative spatial relations, it should be determined how the environment is going to be captured and defined. This chapter explains how is the structure used to obtain a description of a scene using QSR. It gives details about which information is needed to define a spatial relation and also it is explained all the possible qualitative spatial relations presents in our scenes and a suitable selection of them to describe the scenes using these relations.

3.1 Methodology

Before starting with the definition of what qualitative spatial relations are and how we can obtain them, we are going to see an overview of how we are going to proceed to achieve our goal. In the figure below it is shown how a robot will be able to obtain a description of a scene using QSR.

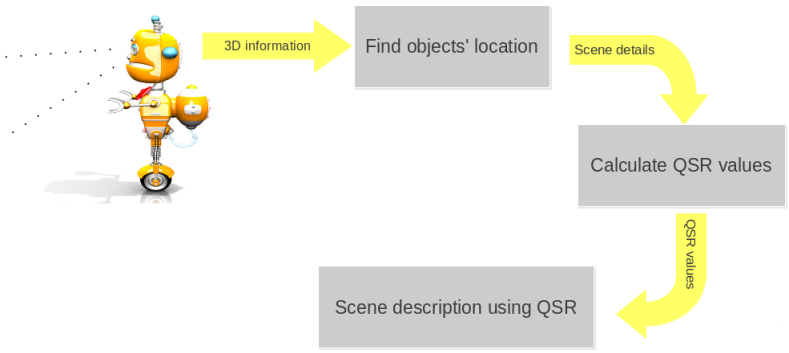


Figure 3.1: Structure to obtain scene descriptions using QSR

Figure 3.1 shows us that to give our robots the ability to describe the environment using different qualitative spatial relations, the following is needed:

- Data capable to represent three-dimensional information. This data should be easy to obtain through robot sensors. How this data is obtained and represented will be explained in Chapter 4 but it is needed to think which area of study will be used. It was decided to limit our area of study to a tabletop because it is a common place where there are lot of different objects with different sizes. Also it is a dynamic place as we can find a lot of changes in the position of objects during a normal day in this kind of environment.
- From the data obtained, it is needed to extract some essential information as the position and the orientation of the objects. This is done using an annotation tool developed specially for this work, which will be explained in Chapter 5.
- Using the acknowledge of the location of all objects, the robot calculates different values that represents if spatial relation holds given the pose (position and orientation) of the objects. Using these values for each QSR, the robot can obtain a description of a scene.

3.2 Qualitative Spatial Relations

Describing scenes by using qualitative spatial relations is one of the aims of this thesis. Before describing all the possible relations it will be necessary to define the main concepts that are going to be used to identify objects when referring to them while they are being described. All objects are treated as a cuboid with the size of the respective real objects. This cuboid is called the *bounding box*. The decision of using a bounding box to represent the objects was made because it is a simple way to describe an object in space and a bounding box can be defined for all objects regardless of its shape.

An object is defined by specifying the pose and the dimensions of the bounding box that can efficiently encapsulate the points belonging to the object in that scene. The pose of all objects should be referred to the same point of reference and we have selected the lower left corner of the table as a reference. The position and dimensions of the bounding box is defined as it is shown in Figure 3.2.

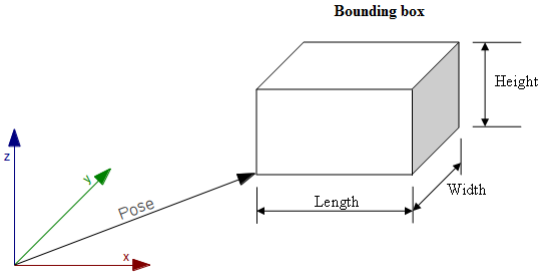


Figure 3.2: Bounding box and its pose

Also the orientation of the bounding box is defined. A 3D body can be rotated about three orthogonal axes, as shown in the figure below.

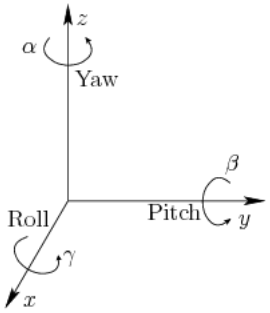


Figure 3.3: 3D rotation

Borrowing aviation terminology, these rotations will be referred to as yaw, pitch and roll:

1. A *yaw* is a counterclockwise rotation of α about z-axis.
2. A *pitch* is a counterclockwise rotation of β about the y-axis.
3. A *roll* is a counterclockwise rotation of γ about the x-axis.

To achieve a satisfactory description of the scenes, it is needed to analyze in depth all the possible relations present in our environments of study. This should be done in order to select the appropriate set of relations that could make our system obtain a detailed description of a scene in terms of spatial relations.

In this report the following terms that are used in cognitive semantics to define spatial relations will appear: landmark and trajector.

Landmark Entity of reference to define a relation. The trajector is always refered to the landmark.

Trajector Entity whose location related with the landmark is described as a spatial relation. The trajector is the focus of the relation.

Thus, in a sentence like "The keyboard is in front of the monitor", the keyboard is the trajector and the monitor is the landmark. We can see how the trajector is the focus of the relationship because the landmark only acts as a reference. The terms trajector and landmark date back to the Gestalt psychology [17] where scenes were divided into the main object (*figure*) and the background (*ground*). Then, we can see how there is a parallelism between figure and trajector, and between ground and landmark. In language there are two different classes of relations, depending on the nature of the relationship. These two categories are topological and projective.

Projective spatial relations are defined depending on the direction where the trajector is located with respect to the landmark. This direction may depend on the view point of the observer if the landmark does not have perfectly defined faces, i.e. if the landmark is a television humans have clear concept of which is the front face but if this landmark is a box where all the faces looks like the same, the relation will definitely depend on the point of view. Examples of projective relations are "in front of", "to the right of" and "behind".

Topological relations do not depend on the location of the observer so a direction is not a factor to take into account to define these relations. In this kind of relations, terms such as proximity, remoteness, contain or support are considered. Examples of topological relations are "inside", "near" and "on".

It is important to analyze which objects can be defined as landmark and which can only be defined as a trajectors. As the landmark acts as a reference, normally the bigger objects take this role. For instance, humans normally use a sentence like "The mouse is in front of the laptop" and not "The laptop is behind the mouse". Therefore, small objects such as pen, pencil or mug are always going to be treated as a trajector and never as a landmark in this work.

Analyzing some of the scenes gathered, it is possible to extract the possible relations between all the objects. Focusing on the objects that usually appears on a desk, it is possible to extract the most common relations that we can find in such scenes. Following, there is a list with these spatial relations:

To the left of The trajector is located to the left side of the landmark.

To the right of The trajector is located to the right side of the landmark.

Near The distance between two objects is insignificant with respect to the size of the table.

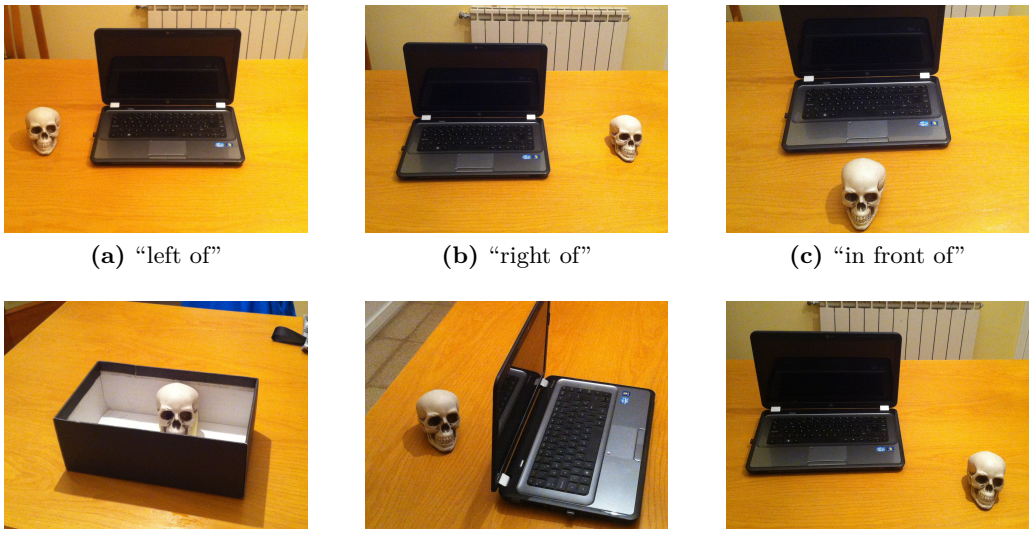
Far The distance between two objects is significant with respect to the size of the table.

In The landmark contains the trajector.

On The landmark supports the trajector.

In front of The trajector is located at the frontal side of the landmark.

Behind The trajector is located at the back side of the landmark.



(a) "left of" **(b)** "right of" **(c)** "in front of"

(d) "in" **(e)** "behind" **(f)** "near"

Figure 3.4: Representation of some qualitative spatial relations using a skull as a trajector

In Figure 3.4, we can see a simple representation of some relations described earlier. As mentioned before, the choice of trajector and landmark matters: *The skull is in front of the laptop* does not mean the same thing as *The laptop is in front of the skull*.

3.3 Analysis

The aim of this thesis is not to define all the spatial relations present in our scene, rather to obtain a robust model of them that will allow a robot to use them in the real world. To find the typical objects that are usually placed on most of the desks and the patterns referred to the relations between objects, the analysis of the data will be performed by observing the following characteristics of the objects:

- Frequency of their presence in the scenes.
- Relations between them.

It was possible to recognize some objects that usually appear in our scenes. It was chosen to analyze all the relations between four objects: monitor, keyboard, mouse and mug. This was done to simplify the task of describing a scene and this also allowed us to annotate more scenes given that the less objects to annotate we have the less time the annotation requires. The first three of them appears mostly in all the scenes and the latter (the mug) in some of them.

A good description of a scene can be obtained by using some of the previously mentioned relations. Given the set of objects selected before, we found it appropriate to implement four qualitative spatial relations: "to the left of", "to the right of", "in front of" and "behind". After analyzing some scenes, we found that relations such as "on" and "in" were not present between this objects. With these picked relations it is possible to find mostly all the relations between our selected objects of study and achieve a useful description of such scene.

3.4 Formalization

Klaus-Peter Gapp said in [18] that to find a spatial relation, the most important factors are angle, distance and shape. In this section the model implemented to evaluate the different spatial relation is explained. The goal is to implement a model trying to produce results in accord with human intuitions of "to the left of", "to the right of", "in front of" and "behind". Developing these models will allow our robot to describe scenes using these qualitative spatial relations, otherwise it has no way of knowing how to describe a scene. To define these relations, different parameters (distance and angles) were computed using the information obtained from the data annotation.

Given a pair of objects, one used as a trajector and the other as a landmark, each relation is calculated and can attain values in the range [0, 1]. If the value obtained is close to 1 it means that the pair of objects accomplish that relation and if it is close to 0 the relation does not hold otherwise it holds to some degree. To obtain this value associated with each relation, three parameters are calculated (one distance and two angles).

These three different parameters were chosen to model our qualitative spatial relations because they are the principal factors taking into account for a human when describing our spatial relations. The computation of the angles are needed to know the direction where the trajector is located with respect to the landmark and this is an important factor. The distance evaluates how near/far the trajector is to the landmark, thus this give us the possibility to control when an object is close enough to hold a relation.

The distance is calculated from the center of the desired face (depending on the

relation) of the landmark to the center of mass of the trajector. The center of mass of the trajector is taken as the geometrical centroid of the bounding box that defines the object. This distance is shown in Figure 3.5, where the landmark is located on the left part of the image.

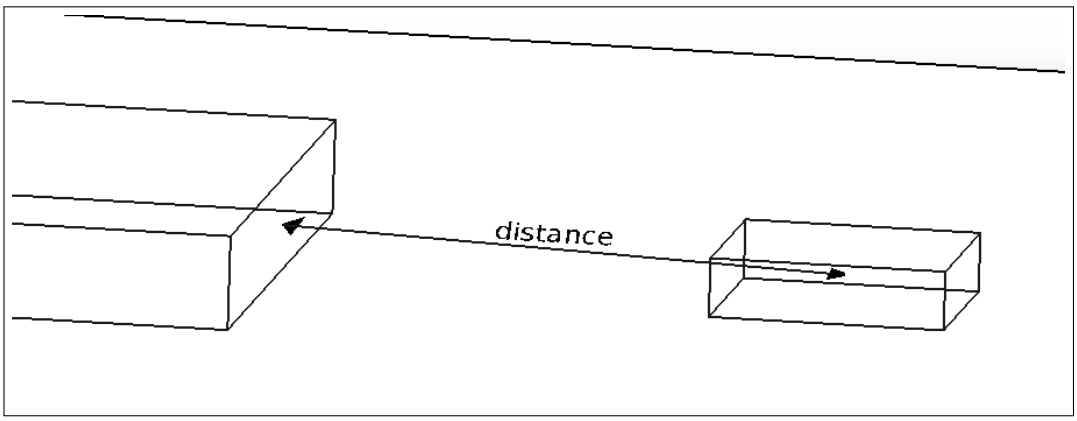


Figure 3.5: Distance between the landmark and the trajector

The other parameters to obtain the evaluation of the qualitative spatial relations are two angles. To calculate these angles we have used each lower corner of the face of the landmark and the projection of the center of mass in the table plane. How these angles are calculated is shown in the following image.

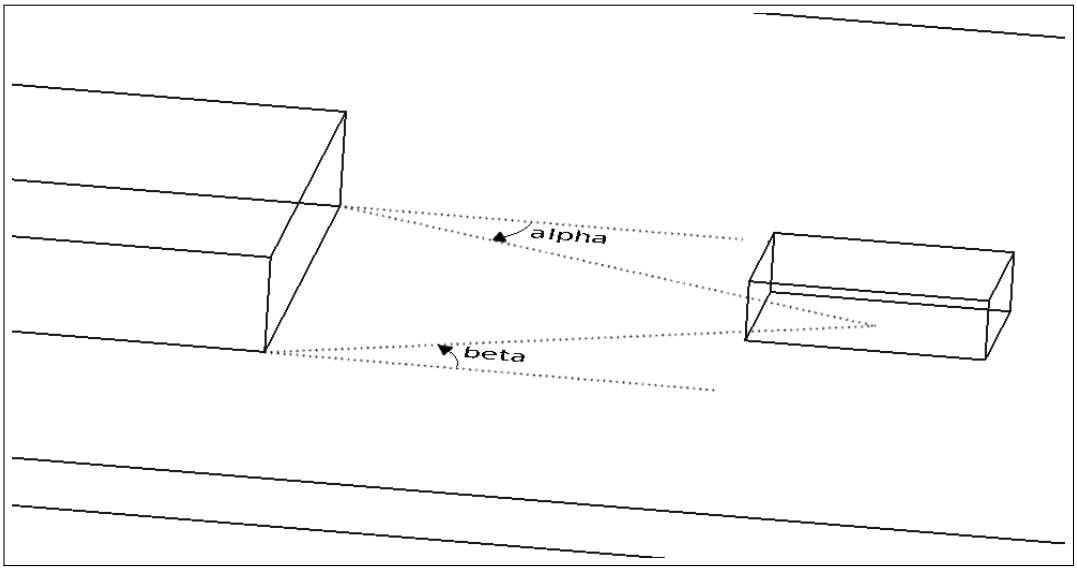


Figure 3.6: Angles calculated between the landmark and the trajector

Depending on which relation is calculated, the parameters will be obtained from a different face of the reference object (landmark). For instance, the relation calcu-

lated in Figure 3.5 and Figure 3.6 is "to the right of". Therefore, all the parameters used in this example are calculated from the right face of the landmark.

To calculate the value of each relation, two functions are used employing the previously mentioned parameters (distance and angles). Both functions take values from 0 to 1 and are multiplied obtaining the final result. The same functions are used to calculate the value of each four relations, so the difference between them is the orientation. The schema of how it works is shown in the image below.

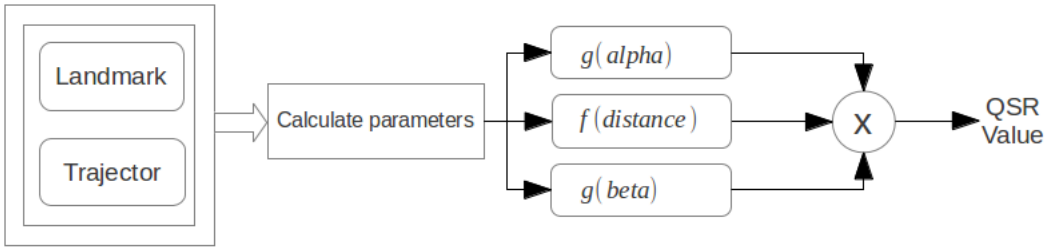


Figure 3.7: Schema to calculate QSR values

For each pair of objects (landmark and trajector) four different QSR values are calculated, corresponding to each qualitative spatial relation. The value obtained is not a probability but it is possible to use it to decide if an object accomplish more or less a relationship. Knowing these QSR values, the robot can decide which is the relation that satisfy the position of both objects, for instance if the value is greater than a previous decided threshold.

The function for the angle, $g(angle)$, has a high value when an angle is between 0 and $\pi/2$, otherwise this value is smaller or equal to 0. The reason for such behaviour is that we would like to give importance to objects that are located in the correct direction to hold a spatial relation. To achieve this purpose, a piecewise function was implemented as shown below. A graphical representation of this function is shown in Figure 3.8.

$$g(angle) = \begin{cases} 0 & -\frac{\pi}{2} \leq angle < -\frac{\pi}{6} \\ 1 - \frac{36}{\pi^2}\left(angle\right)^2 & -\frac{\pi}{6} \leq angle < 0 \\ 1 & 0 \leq angle < \frac{\pi}{2} \\ 1 - \frac{36}{\pi^2}\left(angle - \frac{\pi}{2}\right)^2 & \frac{\pi}{2} \leq angle < \frac{2\pi}{3} \\ 0 & \frac{2\pi}{3} \leq angle < \frac{\pi}{2} \end{cases}$$

It was chosen to use $g(\alpha)$ and $g(\beta)$ together to give a value close to 1 when the object is located on the side of the corresponding spatial relation. Therefore, to get a high value for a specific relation the position of the object is restricted to be on the correct side of the landmark. For instance, if the QSR value that is going to be calculated is "in front of" and the trajector is physically located in front of the landmark (independent which is the distance between them), this calculation give a high value. So the distance function will evaluate if an object is close enough to think that a relation is possible or not.

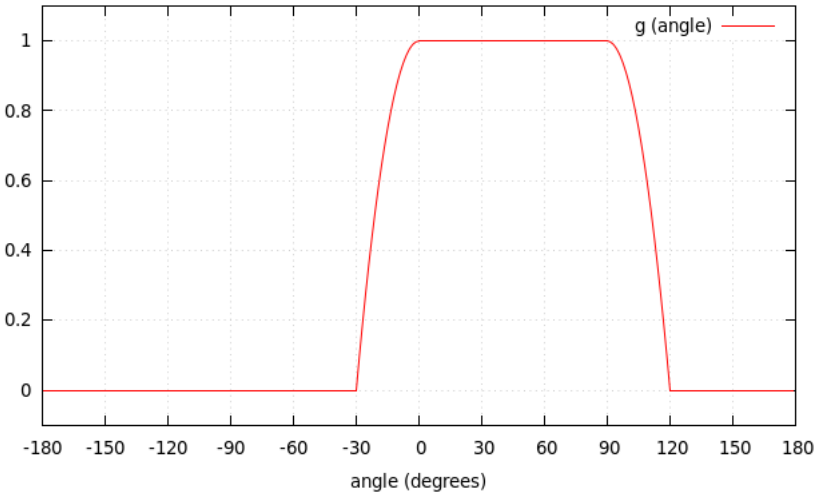


Figure 3.8: Function used for the angle

For the distance, a function was implemented that is close to 1 when the distance is small and decreases exponentially when the distance between the trajector and the landmark increases:

$$f(distance) = \begin{cases} 1 & 0 \leq distance < 0.2\,m \\ \exp\left(-\left(\dfrac{distance - 0.2}{0.4}\right)\ln 2\right) & distance \geq 0.2\,m \end{cases}$$

This tries to reproduce that the farther the trajector is from the landmark, the lower the value of a projective spatial relation should be. This function has been created for our desktop environments so if we would like to generate relations between other objects (e.g. tables, beds or cupboard), this functions should be changed. This can be easily done by generating a different function depending on the size of the objects. In Figure 3.9 it is shown the behaviour of this function, if the distance is less than 20 cm the function returns 1 and if it is greater the function decreases.

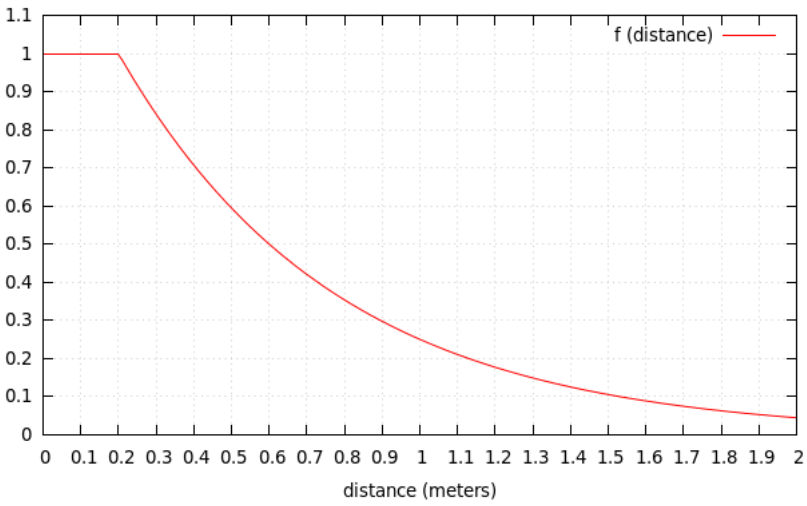


Figure 3.9: Function used for the distance

Using this system makes it possible to obtain values about the different qualitative spatial relations explained earlier. As an example, it is explained how the value of "to the right of" and "in front of" between two objects is calculated given a specific scene shown in Figure 3.10. The keyboard is going to be the landmark and the mouse the trajector.

Figure 3.10: Scene of the example

As shown in Figure 3.7, given a landmark and the trajector, the first thing to do is to calculate the three parameters needed to define a relation. Afterwards the functions described before will be used to compute the final value. In this case we obtain:

To the right of

- $distance = 11.2\,cm \longrightarrow f(distance) = 1$
- $\alpha = 62.9^{\circ} \longrightarrow g(\alpha) = 1$
- $\beta = 13.2^{\circ} \longrightarrow g(\beta) = 1$

In front of

- $distance = 31\,cm \longrightarrow f(distance) = 0.83$
- $\alpha = 260^{\circ} \longrightarrow g(\alpha) = 0$
- $\beta = 92^{\circ} \longrightarrow g(\beta) = 0.99$

So we can easily see that the $QSR(mouse\ to\ the\ right\ of\ the\ keyboard) = 1$ and $QSR(mouse\ in\ front\ of\ the\ keyboard) = 0$. Therefore, we can extract that the qualitative spatial relationship between the mouse and the keyboard, using the mouse as a trajector, is “to the right of”.

Chapter 4

Data Collection

To obtain the data needed, the first step is to choose which type of data is the most suitable for our purpose and how this data is going to be collected. The goal is to collect all the information related to the objects present on a desk, so the robot will be able to perceive the environment with details. Therefore, the data needed should be able to represent a three-dimensional environment. The data structure chosen was the point cloud, which represents the real environment and can be acquired from hardware sensors such as Microsoft Kinect, Asus XtionPRO and PrimeSense 3D cameras. Additionally with the 3D data obtained from the depth sensors, the RGB data from the camera is also captured. Furthermore, the point clouds are gaining prominence as a representation of the world in a number of new mobile robots.

The point cloud was chosen for our work because is a simple type of data that can be obtained from the typical sensors used in robotics. Another advantage is that lot of people is working with this kind of data in robotics, therefore there are multiple different functionalities already available. Also the libraries used to deal with this data are open source, which facilitates the development of new tools. Given that point cloud can mix data coming from the depth sensors and the RGB camera, the obtained representation of an environment is similar to what humans perceive.

Having a good dataset is important for this project for different reasons. The first one is that during the annotation of all the objects, they should be represented clearly in order to be able to annotate them. Therefore, the data should have enough resolution to allow that. Using the point cloud as a data structure we ensure that the data is good enough for this purpose. Besides, obtaining data from several different scenes is needed for different reasons:

- It is important to run our system with different distributions of objects to check if the results we obtain are reasonable and plausible.
- Having a lot of data from different desks will allow us to extract some patterns on the distribution of a desk.

- If some of the data comes from the desk of the same person in different times of the day, we could try to find some patterns about how the position of the objects changes over time.

To deal with the point cloud data, the Point Cloud Library (PCL)[1] is used. To convert what the robot perceive through the sensors to the point cloud data, we used the Robot Operating System (ROS)[2] which is fully integrated with PCL. This chapter begins with the explanation of how to represent the data and finishes giving details about the collection.

4.1 Point Cloud Library

PCL is an open C++library to work with point clouds [19]. The PCL framework includes a wide variety of different tools structured in several libraries such as filters, features, keypoints, registration, kdtree, octree, segmentation, sample_consensus, surface, range_image, io and visualization. These algorithms facilitate the work of a programmer when dealing with point clouds because the most common actions needed are already implemented.

In March 2010, the development of the Point Cloud Library started. Everything started in a well-known robotics company, Willow Garage[3]. Nowadays, Willow Garage also maintains the OpenCV computer vision library[4]. The development of the PCL is continuous by a large number of people around the world for the reason that PCL is an open source software.

4.1.1 Point Cloud Data

Point Cloud Data (PCD) is the file format used for the Point Cloud Library[5], though PCL give the possibility to save and load data in other file formats such as PLY (Polygon File Format), STL (STereoLithography) and OBJ. However, in this thesis only the PCD file format was used. The usage of this file format (PCD) offers some advantages compared to other file formats:

- For robotics applications, this type of data allows us to store and process organized point clouds.
- The PCD format is used a lot in robotics applications, thereby a lot of research has been done about its usage.
- The PCD file is more adapted to PCL than other formats, thus this improves the efficiency of the PCL applications.

[1] `http://pointclouds.org/`
[2] `http://www.ros.org/`
[3] `http://www.willowgarage.com/`
[4] `http://opencv.org/`
[5] `http://pointclouds.org/documentation/tutorials/pcd_file_format.php`

```
# .PCD v.7 - Point Cloud Data file format
VERSION .7
FIELDS x y z rgb
SIZE 4 4 4 4
TYPE F F F F
COUNT 1 1 1 1
WIDTH 213
HEIGHT 1
VIEWPOINT 0 0 0 1 0 0 0
POINTS 213
DATA ascii
0.93773 0.33763 0 4.2108e+06
0.90805 0.35641 0 4.2108e+06
0.81915 0.32 0 4.2108e+06
0.97192 0.278 0 4.2108e+06
0.944 0.29474 0 4.2108e+06
0.98111 0.24247 0 4.2108e+06
0.93655 0.26143 0 4.2108e+06
0.91631 0.27442 0 4.2108e+06
```

Figure 4.1: Snippet of a PCD file

As it is shown in Figure 4.1, the PCD file contains a header that identifies and declares properties of the point cloud data stored. In the version 0.7, the PCD header contains the following entries:

Version Specifies the version of the PCD file.

Fields Specifies the name of each field that a point can have. In our case, four fields are present: x, y, z and rgb.

Size Indicates the size of each dimension in bytes.

Type Defines the type of each dimension as a char.

Count Specifies how many elements does each dimension have.

Width Indicates the width of the point cloud dataset in number of points. This field can specify the total number of points in the cloud if the data is not organized.

Height Specifies the height of the point cloud dataset in the number of points. It is set to 1 for unorganized datasets.

Viewpoint Indicates an acquisition viewpoint for the points in the dataset specified as a translation (tx ty tz) + quaternion (qw qx qy qz).

Points Specifies the total number of points in the cloud.

Data Defines the data type of the point cloud data stored (*ascii* or *binary*).

After the header, the next bytes are considered part of the point cloud data. Each point has the value of the component x, y and z together with the RGB value.

4.1.2 Modules

PCL is organized into several small code libraries. Some of them are listed below:

Filters
This library contains different filters doing modifications to the point clouds such as noise removal or downsample.

Registration
It is used to combine more than one datasets into a global model. The idea is to identify common points in the different data sets and try to find a transformation to align all of them.

Segmentation
This library contains algorithms for segmenting a point cloud.

Sample Consensus
It holds SAmple Consensus (SAC) methods like RANSAC (explained in details in Section 5.3.1) and models like planes and cylinders. This module was used in this thesis to detect the dominant plane in our scenes.

IO
The io library contains classes and functions for reading and writing point cloud data (PCD) files. Also some functions to capture point clouds from different sensing devices.

Visualization
This library allows the visualization of the point clouds. It is based on VTK [20] that offers support for rendering 3D point clouds. This will be used in our annotation tool to develop the main widget (the visualizer).

Common
The common library contains all the data structures and methods used by the majority of PCL libraries. This includes the PointCloud class (used in this thesis) and a multitude of point types [1]. For our point clouds, the points used were the *PointXYZRGB*. PCL also give the possibility to the user to create its own point type.

[1] `http://pointclouds.org/documentation/tutorials/adding_custom_ptype.php#what-pointt-types-are-available-in-pcl`

4.2 Robot Operating System

ROS is an open-source robot operating system that provides libraries and tools to help software developers create robot applications [21]. ROS was originally developed in 2007 under the name *switchyard* by the Stanford Artificial Intelligence Laboratory (STAIR)[1]. As of 2008, development continues primarily at Willow Garage, a robotics company [22].

Using ROS with robots facilitate the programmer's job in terms of integration and usage. With ROS it is easy to integrate multiple tools, already implemented, from different areas. ROS was usefull for us because the most commonly sensors used for a robots are compatible with this operating system and there are already multiple different functions implemented. Also, we wanted to create a new tool and ROS give supports in four very different languages: C++, Python, Octave, and LISP.

Software in ROS is organized in packages, covering areas such as:

- Perception
- Control
- Grasping
- Motion
- Object Identification

The goal of these packages is to provide useful functionality so that software can be easily reused. In our work we are interested in the perception of the robot so other packages such as grasping, motion or control are not used. One of the available packages in perception is the `pcl_ros`[2] package which provides interfaces and tools for bridging a running ROS system to the Point Cloud Library.

4.3 Data collected

The data collected is essential in this thesis because without data we are not able to know how our environment is. To gather the data, as mentioned before, we used the `pcl_ros` package, which allows the robot to receive, process and save the 3-dimension data that devices such as Kinect or Asus Xtion provides. Specifically, the tool used to save the point cloud as a PCD file was a modification of the `pointcloud_to_pcd`, which is implemented inside the `pcl_ros` package.

[1] `http://stair.stanford.edu/`
[2] `http://wiki.ros.org/pcl_ros`

Two sensor were available to gather the data: Asus Xtion PRO Live or Kinect. The sensor used was the Asus Xtion[1] instead of the Kinect because Asus are USB-only, smaller and easier to mount. The fact that Asus Xtion is USB-only helped us to simplify our work when connecting the sensor to a laptop because we did not need to plug the sensor to a power supply as the Kinect requires.

Figure 4.2: Asus Xtion PRO Live.

As the aim is to deal with different distributions of desks and it is desired to see the changes in the objects' distribution, a lot of data was gathered. The desks chosen to collect were located in the CVAP department at KTH. During one week the desk environment from six different people in different moments of the day was collected. As a result, a total of 42 scenes were collected and can be found in `http://www.cas.kth.se/qsr-data/`.

[1] `http://www.asus.com/Multimedia/Xtion_PRO/`

Chapter 5

3D Annotation Tool

To get the spatial relations between objects in a specific scene, first of all the robot has to know where each object is located. This is done by finding the pose (position and orientation) of all the objects, which is hard to do in general and we wanted to be able to annotate manually. If fully automatic was possible we would not need the tool but nowadays we can not find a robust system that give as all the details of a scene in an automatic way. We needed to develop this software given that such a tool was not found. It was desired to build a software to annotate objects, present in the desks of the data collected, in an easy and interactive way. Therefore, a software called 3D Annotation Tool was developed.

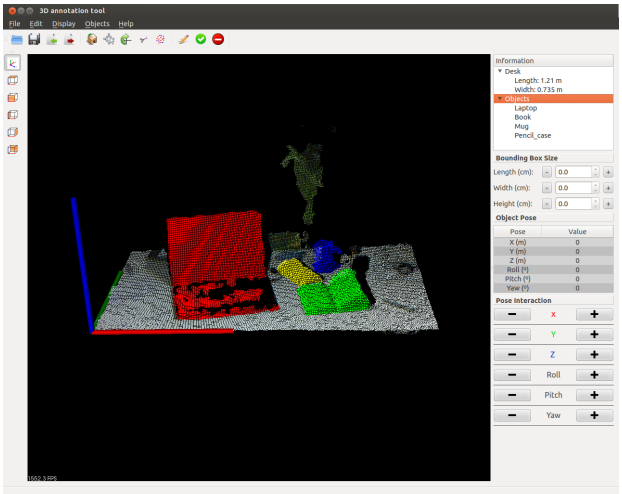


Figure 5.1: Main Window of the 3D Annotation tool

The application can be retrieved and installed on a computer from the software repository `https://github.com/adria-gallart/3d_annotation_tool`. This chapter will explain briefly the most relevant characteristics of the software. For more information about the development, implementation and usage of the application

see Appendix A.

5.1 Purpose of the tool

The aim of this annotation tool is to obtain the location of the objects that are present in our scenes in a way that we can use it to extract qualitative spatial relations. To achieve this purpose, the performance of different processes is needed:

- Find the location of the table plane. Given a scene, finding the table plane will allow us to know where the objects must be situated.
- Set a reference point to which all the objects are referred. The pose of each object is going to be defined from that point of reference.
- Once the scene is prepared for the annotation (acknowledge of plane location and fixed a reference point), we should be able to annotate objects in an easy way. In this process it is important to acquire all the parameters needed to define an object: position, orientation and dimensions (explained in Section 3.2).
- Obtain an understandable representation of all the information.

All these processes have the same importance because without one of them the others are useless. In the following sections, how everything has been solved is explained.

5.2 Tool requisites

To make this application useful for our purpose it had to accomplish different requisites:

- Interactivity: it was essential to show the point cloud and see all the modifications done during the annotation process. For this reason a graphical user interface (GUI) was built.
- Software performance: the software should be able to process the point cloud doing things, previously mentioned, such as detection of the table plane, segmentation of the point cloud, object annotation and save the annotated information.
- Incremental: the user must be able to modify a previously annotated scene. So if a scene is annotated and saved but after that the user wants to change (adding or deleting) some objects, this should be done in an easy way without losing any information.

5.3 Software performance

To annotate a scene as it is desired, several actions were programmed in the 3D Annotation tool. To annotate all the objects, previously it is needed to define the table plane, define the reference point and segment the point cloud. Once the annotation is done, it is desired to save that information. Some other actions were programmed such as the downsampling, undo or change viewpoint but they are not explained in this chapter (this information can be found in the Appendix A). This section is focused on explaining the main functions of the annotation tool.

5.3.1 Plane definition

Whenever a point cloud is loaded in the annotation tool, it is essential to know the location of the table because all our work depends on the objects that are on that table. This will help us to implement other functionalities of the software such as the plane segmentation or the object annotation. Therefore, the first thing to do is to detect the plane of the table and this can be done automatically and manually. Once the plane definition is completed, the point cloud is rotated and translated such that the plane is parallel to the x-y plane of the reference spatial coordinate system. This will be the first step to set the reference point, which is desired to locate on the lower left corner of the table.

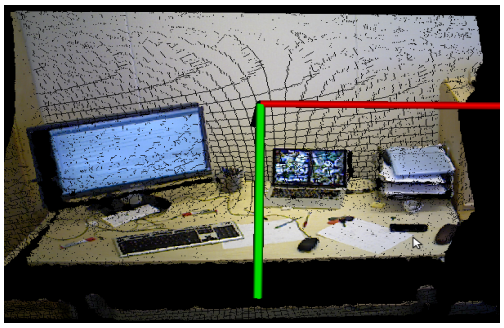


Figure 5.2: Before plane definition

Figure 5.3: After plane definition

In order to see changes in the point cloud, in Figure 5.2 and Figure 5.3 we can see the coordinate system of the point cloud on the screen. The x axis corresponds to the red axis, y is the green axis and z is the blue axis.

When using the automatic plane detection, the software performs a plane detection using the RANSAC method [23]. RANSAC is an abbreviation for "RANdom SAmple Consensus". It is an iterative method that uses the following things as input: the data (set of points), the model (plane for us) and other parameters such as number of iterations or number of data to fit the model. Through all the iterations, the algorithm tries to find if the model is present in the data.

When trying to find a model in a set of points, two kinds of data can be differentiate. Data that fits the distribution of a model or *inliers*, and data that do not fit the model (*outliers*). The usage of the RANSAC method is better than other ones to fit a model because the model is computed choosing only inliers and not outliers. In the figure below we can see a simple example of how RANSAC fits a line given some points.

Figure 5.4: Data with many outliers

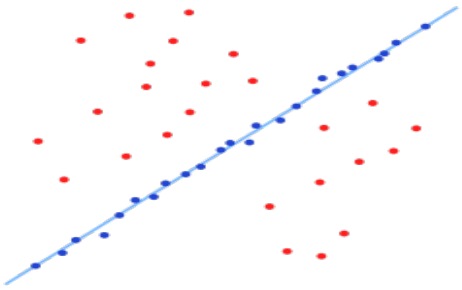


Figure 5.5: Fitted line using RANSAC

A plane in space has an equation as follows:

$$Ax + By + Cz = D$$

So the definition of the plane is done by computing the four coefficients A, B, C and D. The RANSAC method is used to obtain the coefficients of the fitted plane. Once these coefficients are obtained the rotation and translation needed to transform the point cloud can be calculated. Using this transformation the point cloud is modified performing the effect shown in Figure 5.2 and Figure 5.3.

Depending on the point cloud, sometimes the automatic plane detection cannot exactly detect the dominant plane as humans perceive it semantically. For instance, in some cases, the wall is detected as the dominant plane instead of the table. The solution chosen to solve that issue was to define the dominant plane by selecting three points belonging to it.

5.3.2 Reference placement and plane segmentation

In Section 3.2 it was explained that we selected the lower left corner of the table to place our point of reference. When loading a point cloud the coordinate origin of the data is located in a position not desired for us. It is desired to transform the point cloud to get the coordinate origin in the lower left corner of the table. This makes the annotation easier in the way that the coordinates x, y and z of the pose will be directly the x, y and z values of the lower left point belonging to the front face of the object. Therefore, after doing this action we would like to have our point cloud as in the following figure.

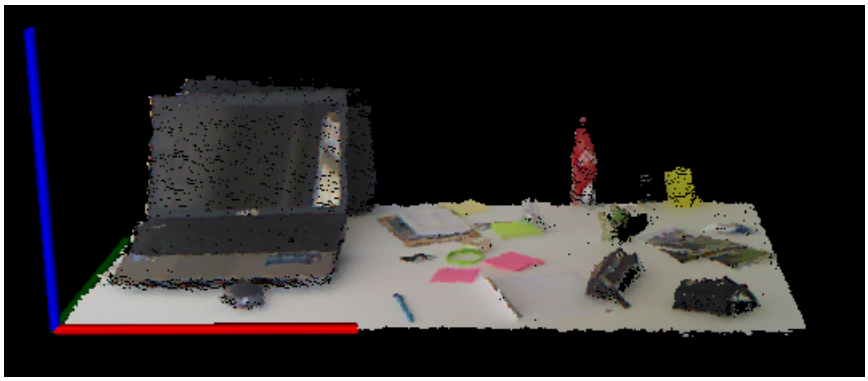


Figure 5.6: Desired location of the coordinate system

The action of setting the reference point is done after the plane definition. To find the transformation, the 3D Annotation tool uses its interactivity asking the user to select three significant points of the scene:

1. Lower left corner of the table.
2. Lower right corner of the table.
3. Point on the upper edge of the table.

Once these points are selected, the point cloud is transformed as it was desired using different functions of PCL. Moreover, this gives us the dimensions of the table and permits us to segment the scene. The segmentation removes all the points outside and under the table to eliminate unneeded points, which makes the PCD file smaller and permits to save space when storing it.

5.3.3 Object annotation

When the user wants to annotate an object, an initial rough bounding box is inserted by picking 4 particular points. The front side of the bounding box is drawn in red and the points inside the bounding box are highlighted (Figure 5.7). To fit all the points belonging to the object, the bounding box can be modified changing all the parameters that defines an object:

1. **Bounding Box Size:** The length, width and height of the bounding box.
2. **Object Position:** The position of the object referred to the reference point.
3. **Object Orientation:** The yaw, pitch and roll angles.

After doing the necessary modifications on the bounding box to fit the object, all these parameters are saved. These parameters together with some other useful data is used to represent the scene and it will be detailed in the following section.

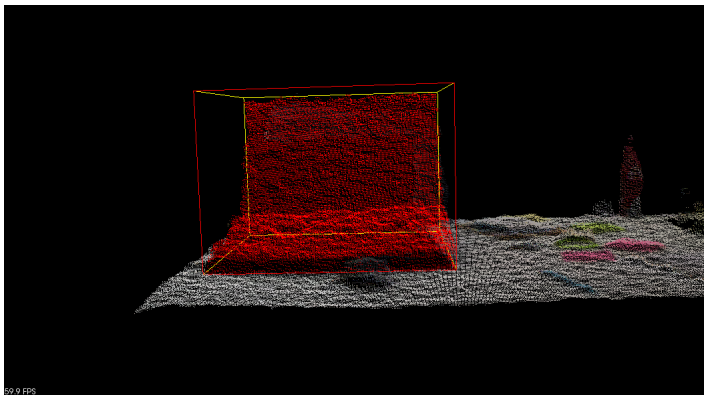


Figure 5.7: Inserting an object

5.3.4 Export object information

Once the annotation of all the objects is done, it is needed to save all this information. This is the information that is going to be used to extract the qualitative spatial relations between all the objects. To do that, for each annotation an .xml file is created with all details about the scene and its objects. This is the information saved:

1. Type of the scenario (desk in our case).
2. Point cloud file name for which the annotation was performed.
3. Dimensions of the table (length and width).
4. Number of objects annotated.
5. For each annotated object the following is saved:
   - Name of the object.
   - Highlight color associated with that object.
   - Absolute pose of the bounding box with respect to the fixed reference spatial coordinate system: x-y-z coordinates and the roll, pitch and yaw angles.
   - Dimensions of the bounding box: length, width and height of the bounding box.
   - Indices: all the indices of the points contained in the defined bounding box for this object.

Following, it is shown an example of the output file obtained:

```
<scenario>
    <type>desk</type>
    <annotatedFrom>/data/example_segmentated.pcd</annotatedFrom>
    <dimensions>
        <length>1.21264</length>
        <width>0.741531</width>
    </dimensions>
    <allObjects>
        <numberOfObjects>3</numberOfObjects>
        <object>
            <name>Laptop</name>
            <color>red</color>
            <pose>
                <x>0.293258</x>
                <y>0.0700548</y>
                <z>0</z>
                <roll>0</roll>
                <pitch>0</pitch>
                <yaw>0.211196</yaw>
            </pose>
            <dimensions>
                <length>0.38</length>
                <width>0.38</width>
                <height>0.27</height>
            </dimensions>
            <indices>3014 3015 3026 3027 3028 3029 3030 3031 3032 3033
                 3034 3035 3036 3037 3038 3039 3040 3041 3042 3043 3044
                 3055 3056 3057 3058 3059 652 48653 48654 48655 48656
                </indices>
        </object>
        <object>
            <name>Mug</name>
            <color>blue</color>
            <pose>
                <x>0.787483</x>
                <y>0.460005</y>
                <z>0</z>
                <roll>0</roll>
                <pitch>0</pitch>
                <yaw>0.00677016</yaw>
            </pose>
            <dimensions>
                <length>0.12</length>
                <width>0.1</width>
                <height>0.11</height>
            </dimensions>
            <indices>10122 10123 10124 10125 10313 10314 10315 10316
                 10317 10318 10319 10506 10507 10508 10509 10510 10511
                 10512  </indices>
        </object>
        <object>
            <name>Pencil_case</name>
            <color>yellow</color>
```

```
49          <pose>
50              <x>0.745578</x>
51              <y>0.325982</y>
52              <z>0</z>
53              <roll>0</roll>
54              <pitch>0</pitch>
55              <yaw>0.596957</yaw>
56          </pose>
57          <dimensions>
58              <length>0.11</length>
59              <width>0.21</width>
60              <height>0.05</height>
61          </dimensions>
62          <indices>18318 18319 18320 18321 18322 18323 18694 18695
                18696 18697 18698 18699 18700 18701 19062 19063 19064
                19065 19068 19069 19070 19071 19072 19073 19074 19075
                19076 19077 19078 19439 19440 19441  </indices>
63      </object>
64    </allObjects>
65 </scenario>
```

Chapter 6

Dataset

For this thesis we have chosen a typical tabletop (desk) as area of study. We can find this type of environment in a typical office space. This was chosen because in a desk there are lots of objects with different shapes and sizes. Moreover, the position of all the objects on a desk vary during a normal day and not every person places the objects in the same way. Having an extensive dataset is an important factor in this thesis. This will allow us to generate multiple descriptions of different scenes and therefore the possibility to extract some conclusions about the objects distribution.

The dataset contains 42 different scenes with annotations. As explained in Chapter 4, the scene is represented using a point cloud data and the annotation of each scene is saved in an xml file. This makes a total of 84 files, 42 point cloud data (.pcd) and its corresponding xml file. The point cloud data stored, after the segmentation applied during the annotation process, has more than 100000 points. This data was obtained from 6 different desks during 5 days and in some of these days the data was obtained in the morning and afternoon.

Throughout the 42 scenes a total of 136 objects (monitor, keyboard, mouse and mug) have been annotated. All of the desks did not have all of the objects, some had only 3 and some only 2. We can see an example of a scene in Figure 6.1. We can observe how in this scene there are four objects: monitor (blue), keyboard (green), mouse (red) and mug (yellow).

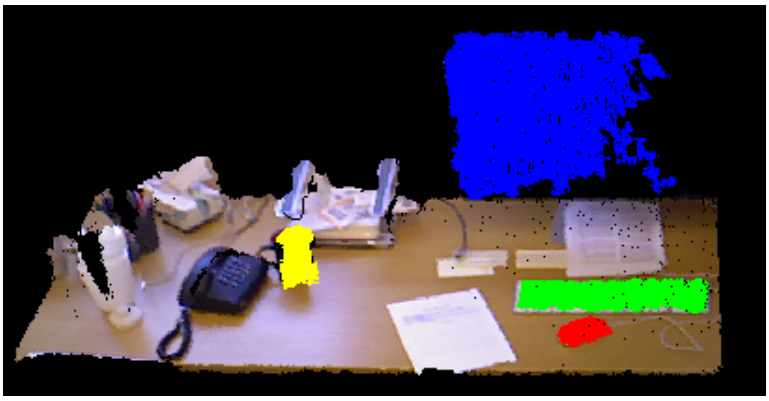


Figure 6.1: Example of scene

From the annotation of this scene. We obtain the following information about the desk:

- Length: 1.49 m
- Width: 0.72 m
- Number of objects: 4

The parameters about all the objects are displayed in Table 6.1. In the table, all the information available is present except the object indices of all the points inside the point cloud.

	Mouse	Keyboard	Monitor	Mug
Color	red	green	blue	yellow
Position: x (m)	1.16	1.07	0.91	0.54
Position: y (m)	0.12	0.21	0.49	0.31
Position: z (m)	0	0	0.08	0
Rotation: roll (rad)	0	0	0	0
Rotation: pitch (rad)	0	0	0	0
Rotation: yaw (rad)	5.28	6.23	0	0.24
Dimensions: length (m)	0.06	0.40	0.51	0.08
Dimensions: width (m)	0.10	0.12	0.15	0.09
Dimensions: height (m)	0.04	0.03	0.44	0.12

Table 6.1: Information about all the objects

Chapter 7

Experiments

This chapter describes the experiments performed for our system using the data collected. All the data available, 42 different scenes, have been tested with our system in order to obtain results that allow us to describe a scene using qualitative spatial relations. We would like to analyze things such as how often objects appear in different scenes, how objects are distributed in space and which are the spatial relations between all of them.

Once all these results are extracted, we can look for patterns that describe typical relations between objects. The patterns found will help a robot to know how the real world is in terms of spatial relations. Thus, it will be capable to reason and extract conclusions about how the objects are located in a human-oriented environment over time.

7.1 Object frequency

Four objects have been taken into account during the process of annotation (Section 3.3). The objects selected need to be present in enough scenes to obtain a satisfactory result from the data annotated. In Figure 7.1 it is shown how many times an object is found in the different 42 scenes.

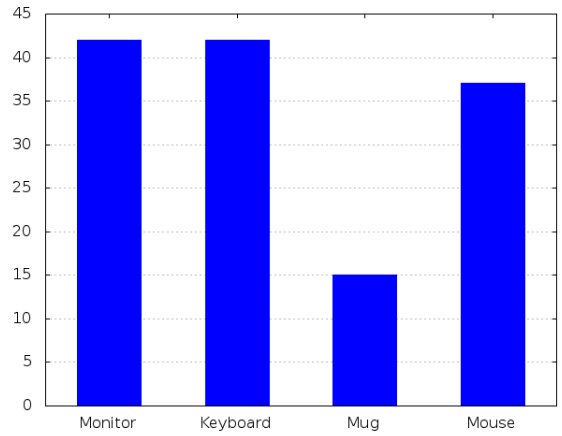


Figure 7.1: Number of times that the different objects appears in the 42 scenes

At least three of the objects appears mostly in all of the scenes. Specifically, monitor and keyboard are found in every scene, mouse appears in all the scenes except 5 and the mug is on a desk in 36% of the scenes. This shows us that monitor, keyboard and mouse are (as everyone would expect) almost always present in such environments and the mug is found sometimes.

7.2 QSR values

Every scene has been evaluated using our formalization of the qualitative spatial relations. Once an annotated scene is passed through our system, we obtain different values of how well a spatial relation holds between two objects. These values are calculated for all the spatial relations selected (*to the right*, *to the left*, *in front* and *behind*). A relation between two objects have two different values depending on if one object is playing the role as a trajector or vice versa. However, not all the objects will play both roles. The mug and the mouse will not develop functions as a landmark in our study (Section 3.2).

We can illustrate how the whole system works by showing an example, seen in Figure 7.2.

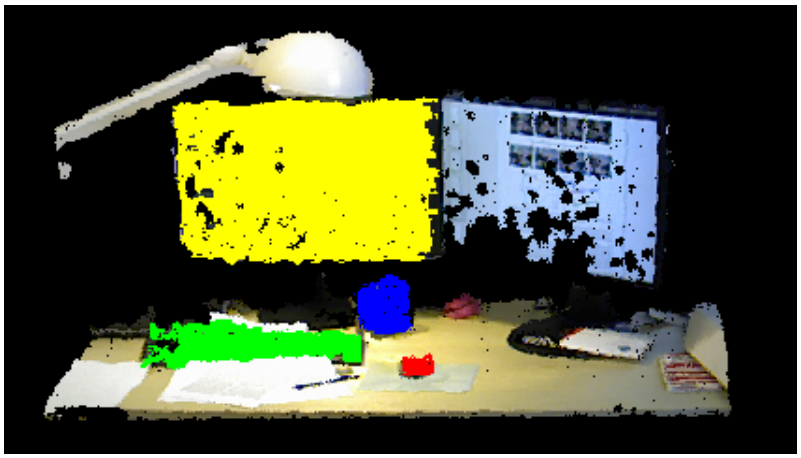


Figure 7.2: Point cloud of the example

As we can see in Figure 7.3 four objects are present in the scene: mouse (red), keyboard (green), mug (blue) and monitor (yellow). To see the placement of all these objects more clearly we can project them into the plane x-y.

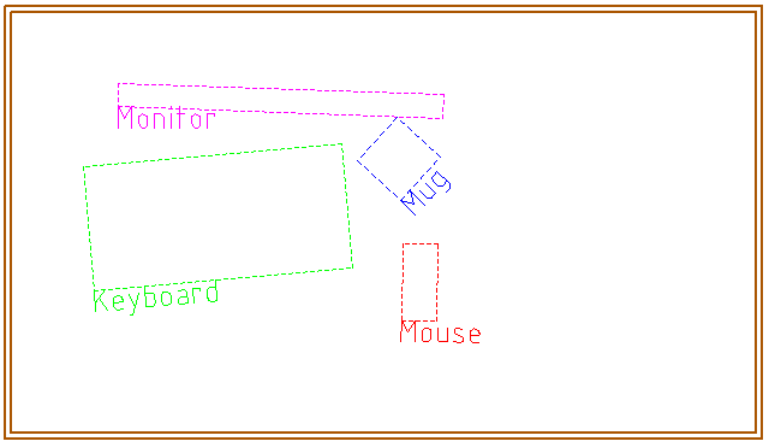


Figure 7.3: Objects projected into x-y plane

All the information found in the annotation file of the scene (position, orientation and dimensions) is used as an input for our system in order to obtain all measures and its correspondent QSR values. The following table represents the values for the spatial relation “to the right of”:

	Keyboard	Monitor
Mouse	0.58	0
Keyboard	X	0
Mug	0.91	0
Monitor	0	X

Table 7.1: Values for the relation "to the right of"

Each value indicates the degree of acceptance of "to the right of", where the corresponding row is the trajector and the column acts as a landmark. An X is displayed when the trajector and the landmark are the same object, therefore a value cannot be calculated. If the value is close to 1 the relation holds. Thus, looking the column of the keyboard and its value with the mug, the value obtained is 0.91. As this is a value close to 1 it is possible to say that "The mug is to the right of the keyboard". In Table 7.2, Table 7.3 and Table 7.4 it is shown the other three relations.

	Keyboard	Monitor
Mouse	0	0
Keyboard	X	0
Mug	0	0
Monitor	0	X

Table 7.2: Values for the relation "to the left of"

The values for the relation "to the left of" are all equal to 0, which means that there are no objects located to the left of any landmark.

	Keyboard	Monitor
Mouse	0	0.62
Keyboard	X	0.77
Mug	0	0.85
Monitor	0	X

Table 7.3: Values for the relation "in front of"

Unlike the previous table, with the relation "in front of" some calculated values are greater than 0, thus it is possible to extract some descriptions from these values.

	Keyboard	Monitor
Mouse	0	0
Keyboard	X	0
Mug	0	0
Monitor	0.81	X

Table 7.4: Values for the relation "behind"

Now we can use all these values to obtain a description of a scene using qualitative spatial relations. We have to set a threshold to decide when a value is suitable to define a relations or not. If the QSR value found is greater than this threshold we will say that both objects holds that relation and if it is lower, this description is not found in the scene. To choose this value would be interesting to run a survey to see different behaviours of how humans interpret if a relation is valid given two objects. However, this was not possible to do in our work due to lack of time. So we analyzed some of the results obtained from different scenes and we found appropriate to set this threshold at 0.5. We found appropriate this value because we saw that QSR values greater than 0.5 give us what a human could considered that is a spatial description and not otherwise.

Using this threshold, the following descriptions can be extracted from the calculated values:

- The mouse is to the right of the keyboard.
- The mug is to the right of the keyboard.
- The mouse is in front of the monitor.
- The keyboard is in front of the monitor.
- The mug is in front of the monitor.
- The monitor is behind the keyboard.

A total of six different description have been extracted using our system in this scene. If we observe the scene, we can confirm that the descriptions obtained are present in the scene. From the above descriptions we can see how there are two sentences that are dual: the keyboard is in front of the monitor and the monitor is behind the keyboard.

7.3 Mouse and keyboard

An interesting case to study is the relation between the mouse and the keyboard. We would expect that the mouse should be located to the right or to the left of the keyboard. Considering that around 10% of the population is left-handed and the rest right-handed, there should be a high value of mouses located to the right of the keyboard rather than to the left.

Analyzing all the scenes where a mouse has been annotated, 37 in particular, we analyzed the relative positions of the mouse with respect to the keyboard. In the following figure, it is shown these locations in polar coordinates. 0° is the direction of the frontal face of the keyboard, thus 90° refers to the right of the keyboard and 270° to the left. The distance from the center of the keyboard is measured in cm.

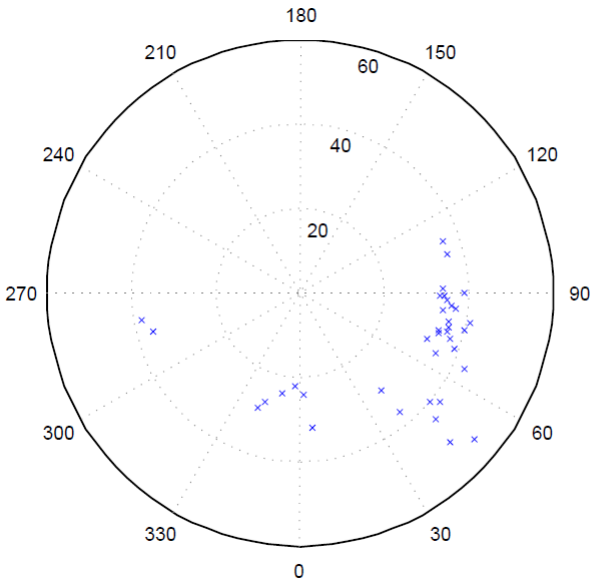


Figure 7.4: Relative positions of the mouse with respect to the keyboard

We obtained that the distribution of the location of the mouse is more dense to the right of the keyboard than to the left. Nevertheless, surprisingly in some cases the mouse was located in front of the keyboard. We can also check this behaviour by studying the different QSR values obtained and get the corresponding description:

- 17 times the mouse was to the right of the keyboard.
- 8 times the mouse was in front of the keyboard.
- 2 times the mouse was to the left of the keyboard.

Exactly as described before, we can see how the mouse is mostly located to the right side of the keyboard. A robot can use this information to detect that something is atypical, for instance, if it finds that the mouse is behind the keyboard.

7.4 Mug

Another important object to study its behaviour in terms of spatial relations is the mug. It is an object that a priori, we do not know if it has any pattern in terms of spatial with the other objects. If we get all the possible descriptions for the mug it is obtained:

- 9 times the mug was in front of the monitor.
- 4 times the mug was located to the right of the keyboard.
- 4 times the mug was behind the keyboard.
- 2 times the mug was found to the left of the keyboard.
- 1 time it was placed to the right of the monitor.

It has been also represented the location of all the mugs found in our scenes:

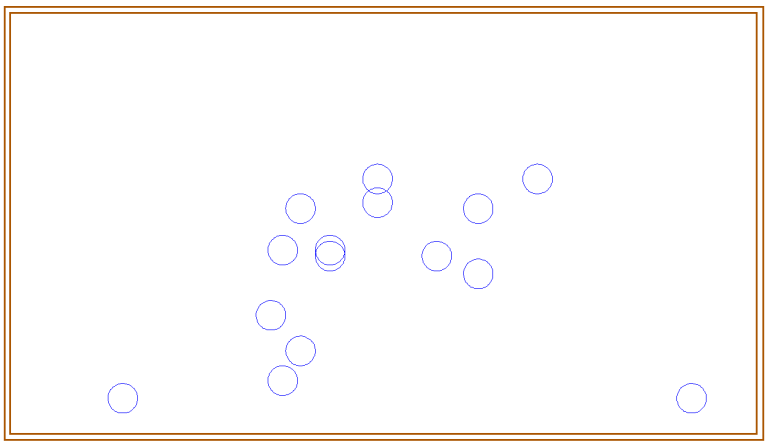


Figure 7.5: Location of all the mugs on a desk

We can see how the mug is commonly situated centered on a desk. From the descriptions obtained, it is possible to extract that the mug can be found around the keyboard and in front of the monitor. This behaviour is reasonable given that when anyone has a mug on the table, this is located in a place that might be easily reachable. As usually a person is close to the keyboard and/or in front of the monitor the mug should be located around them.

7.5 Keyboard and monitor

In a typical desk we expect to find the keyboard in front of the monitor and therefore the monitor behind the keyboard. To check out if this assumption is true in our scenes, it has been calculated the mean and the variance of all values (42 scenes) extracted for these two spatial relations:

- "The keyboard is in front of the monitor":
  - Mean = 0.743
  - Variance = 0.004
- "The monitor is behind the keyboard":
  - Mean = 0.773
  - Variance = 0.007

Indeed we can see how both relationships have a high mean value, greater than 0.7, and therefore both of them fulfill such relation. Since the variance is small, this indicates that all values are very close to the calculated average and therefore in all cases, the keyboard is in front of the monitor and the monitor is behind the keyboard. Then we can say that when one of these relations is true, automatically the other holds as well.

If we have a robot running autonomously and it learns the previously mentioned behaviour between the keyboard and the monitor, it will be able to detect an anomaly of the objects' distribution if the keyboard is not placed in front of the monitor.

Chapter 8

Conclusions and Future Work

This thesis described how a robot can find a reasonable description of a scene in terms of qualitative spatial relations between all the objects. We understand that a description is reasonable if a human can extract the same descriptions by watching the same scene. The usage of qualitative spatial relations might allow a robot to reason about space as humans do. These spatial relations are concepts commonly used for humans to simplify the description of an environment. In our work, the area of study was limited to a tabletop as its dynamics give us various distributions of objects.

To generate a representation of such environment, different things has been achieved in this thesis:

- Make a selection of which objects and spatial relations were used to describe a scene.
- Modelling the different qualitative spatial relations.
- Collect data from our environments for learning and testing.
- Find the position and orientation of the objects in our space.

A total of four spatial relations were chosen to be used in our system: "to the right of", "to the left of", "in front of" and "behind". A novel approach was made to model these relations using the distance and two angles between objects. From the variety of objects present in a desk only a subset were selected: monitor, keyboard, mouse and mug.

The point clouds, gathered with the Asus Xtion, are able to represent 3D scenes and they have enough details to use it for our purpose. With this data and the annotation tool developed, an accurate description of the pose of all objects for each scene was obtained. The 3D annotation tool performs different actions to annotate a scene in an easy and interactive way.

Once all these pieces of the system were implemented, we could extract descriptions using spatial relations for each scene. Therefore, it has been achieved one of the aims of this work. At the end, from the obtained representations of all scenes in terms of spatial relations, some behaviours in regard to spatial relations were found. The different patterns found, give us an idea of how objects are located in a desktop scene. This information should be generated automatically for our robots in order to know how is the typical distribution of an environment. If a robot can know how is the normal distribution of a scene, then it will be able to notice when something is out of the ordinary.

8.1 Future Work

Several directions for future work involving qualitative spatial relations to describe a scene can be found. In our work we used the 3D Annotation tool to obtain the position of each object present in our scene. Despite the fact that this method give us detailed information of the scene, it is desired to make it as automatic as possible without loosing accuracy. So a system that given a point cloud could detect different objects, by performing object recognition, and find the pose of all objects would be desirable.

In this thesis, only a few objects have been used to describe a scene, thus if more objects are chosen a more completed representation could be found. Moreover, finding a more detailed description of a scene implies developing more models for the spatial relations that have not been taken into account in this project. Another interesting experiment can be to perform a spatio-temporal analysis in order to see if the distribution of a scene changes significantly over time (e.g. during a day).

Focusing on the annotation tool, some improvements can be made:

- Right now, when the user wants to annotate an object, an initial bounding box appears after clicking at 4 points on an object. This could be improved by implementing a method such as region growing that only clicking in one point of the object to annotate, a estimated bounding box is generated.
- Using memory about previous annotated scenes can help to predict future actions of the user when annotating a scene.
- Incorporate a real time tool for capturing point clouds.
- Add an object detection system that permits the system find all possible objects present in a scene.

The data used in this work was good enough for our purpose. However, we used single snapshots to take one point cloud each time we captured a scene and we would like to be able to obtain information of larger scenes. This can be done by

taking different point clouds from the same scene and combining them to obtain one global model. We can achieve this, using the registration library found in PCL. Having this kind of data available, it would be possible to obtain bigger scenes and analyze them as well. Therefore, it will be possible to obtain description of entire rooms.

Bibliography

[1] A. G. Cohn and S. M. Hazarika, "Qualitative spatial representation and reasoning: An overview," *Fundamenta Informaticae*, vol. 46, no. 1-2, pp. 1–29, 2001.

[2] J. O'Keefe, *The Spatial Prepositions*, ch. 7. The MIT Press, 1999.

[3] D. Randell and A. Cohn, "Modelling topological and metrical properties in physical processes," in *Proceedings 1st International Conference on the Principles of Knowledge Representation and Reasoning* (R. Brachman, H. Levesque, and R. Reiter, eds.), (Los Altos), pp. 55–66, Morgan Kaufmann, 1989.

[4] D. A. Randell, Z. Cui, and A. G. Cohn, "A spatial logic based on regions and connection," in *KR*, pp. 165–176, 1992.

[5] S. Li and M. Ying, "Region connection calculus: Its models and composition table.," *Artif. Intell.*, vol. 145, no. 1-2, pp. 121–146, 2003.

[6] K. Lockwood, K. Forbus, D. T. Halstead, and J. Usher, "Automatic categorization of spatial prepositions," in *Proceedings of the 28 th Annual Conference of the Cognitive Science Society. Stressa*, pp. 1705–1710, 2006.

[7] J. Fernyhough, A. Cohn, and D. Hogg, "Constructing qualitative event models automatically from video input," *Image and Vision Computing*, vol. 18, pp. 81–103, 2000.

[8] T. Regier and L. A. Carlson, "Grounding spatial language in perception: An empirical and computational investigation," *Journal of Experimental Psychology General*, vol. 130, pp. 273–298, 2001.

[9] A. P. Sistla and C. T. Yu, "Reasoning about qualitative spatial relationships," *J. Autom. Reasoning*, vol. 25, no. 4, pp. 291–328, 2000.

[10] B. Rosman and S. Ramamoorthy, "Learning spatial relationships between objects.," *I. J. Robotic Res.*, vol. 30, no. 11, pp. 1328–1342, 2011.

[11] A. Kasper, R. Jäkel, and R. Dillmann, "Using spatial relations of objects in real world scenes for scene structuring and scene understanding," in *Proceedings of the 15th International Conference on Advanced Robotics*, 2011.

[12] K. Sjöö, A. Pronobis, and P. Jensfelt, "Functional topological relations for qualitative spatial representation," in *ICAR 2011*, June 2011.

[13] K. Sjöö, A. Aydemir, T. Mörwald, K. Zhou, and P. Jensfelt, "Mechanical support as a spatial abstraction for mobile robots," in *2010 IEEE/RSJ International Conference on Intelligent Robots and Systems*, October 2010.

[14] A. Aydemir, K. Sjöö, J. Folkesson, and P. Jensfelt, "Search in the real world: Active visual object search based on spatial relations," in *to appear in Proc. of the IEEE International Conference on Robotics and Automation (ICRA'11)*, 2011.

[15] M. Aurnague and L. Vieu, "A three-level approach to the semantics of space," in *The semantics of prepositions: from mental processing to natural language processing* (C. Zelinsky-Wibbelt, ed.), Natural Language Processing 3, pp. 395–439, Mouton de Gruyter, 1993.

[16] Y. Tuan, *Space and Place: The Perspective of Experience*. Tales and Travels of a School Inspector Series, University of Minnesota Press, 1977.

[17] W. Köhler, *Gestalt Psychology: An Introduction to New Concepts in Modern Psychology*. Black and Gold library, Liveright, 1992.

[18] K.-P. Gapp, "Angle, distance, shape, and their relationship to projective relations," 1995.

[19] R. B. Rusu and S. Cousins, "3D is here: Point Cloud Library (PCL)," in *IEEE International Conference on Robotics and Automation (ICRA)*, (Shanghai, China), May 9-13 2011.

[20] W. Schroeder, K. Martin, and B. Lorensen, "The visualization toolkit: an object-oriented approach to 3d graphics [visualize data in 3d - medical, engineering or scientific; build your own applications with c++, tcl, java or python; includes source code for vtk (supports unix, windows and mac)]," 01 2006.

[21] M. Quigley, B. Gerkey, K. Conley, J. Faust, T. Foote, J. Leibs, E. Berger, R. Wheeler, and A. Ng, "Ros: an open-source robot operating system," in *ICRA workshop on open source software*, vol. 3, 2009.

[22] Willow Garage, "ROS platform." `http://www.willowgarage.com/pages/software/ros-platform`. [Online; accessed April-2013].

[23] M. A. Fischler and R. C. Bolles, "Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography," *Commun. ACM*, vol. 24, pp. 381–395, June 1981.

[24] J. Blanchette and M. Summerfield, *C++ GUI Programming with Qt 4*. Upper Saddle River, NJ, USA: Prentice Hall PTR, 2006.

Appendix A

Using the annotation tool

3D Annotation Tool is an application designed to annotate objects in a point cloud scene. Initially, it was developed with the aim to annotate the objects of a table. The rest of the document is written with the intention of annotating objects in a table scenario without loss of generality. The format of the point cloud supported is .pcd[1].

A.1 Tool development

For the creation and the development of the tool, it has been used the QtCreator[2]. QtCreator is a cross-platform IDE (integrated development environment) for the development of tools. It includes an advanced code editor, visual debugger and GUI designer. The latter, allows the user to rapidly design and build widgets and dialogs using on-screen forms using the same widgets that will be used in the application.

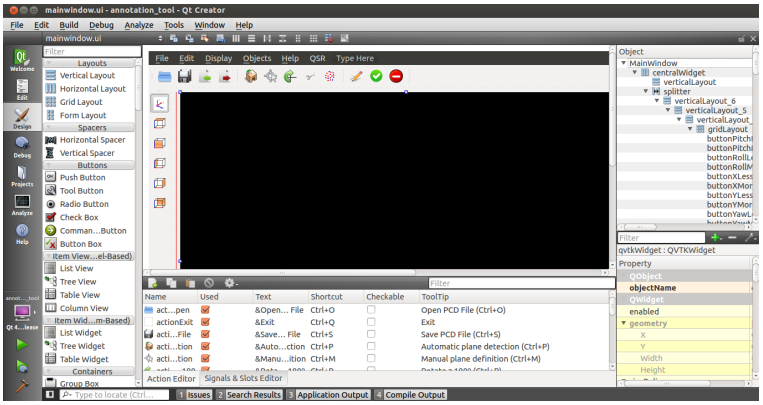


Figure A.1: QtCreator GUI editor

[1] `http://pointclouds.org/`
[2] `http://qt-project.org/`

The QtCreator GUI designer allowed us to build and design the aspect of our application in an intuitive manner by adding and deleting any item (widget) you desire. QtCreator uses a XML file to save the layouts and all the modification done in the designer are automatically changed in this file.

All the other code related with this software has been written in C++[24]. It has been essential the use of the PCL library (Section 4.1) to deal with the point clouds because it offers tools to make easier things such as loading, visualization, saving, rotation, translation and segmentation. It was possible to develop this annotation tool thanks that the PCL visualization library which is based on VTK [20] and it is also possible to integrate it in QtCreator by using the QVTK Widget. Therefore, these libraries permits the interactivity with the user by showing the point cloud and all the modifications done.

A.2 Interface

A.2.1 Main window

An interactive and intuitive interface to do all the desired actions has been developed. In the main window of the software there are different sections for different proposes.

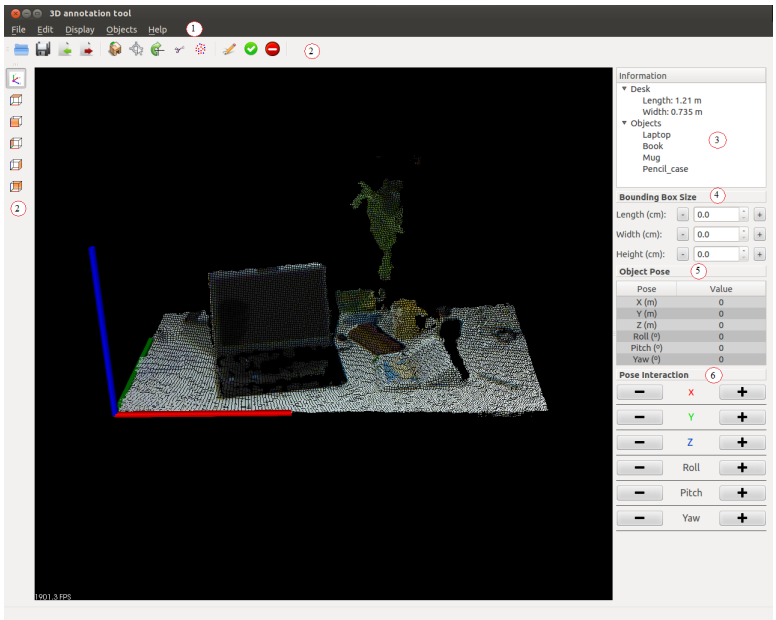


Figure A.2: Main window

The main window of 3D Annotation tool has the following sections:

1. **Menu Bar:** It has access to the functions grouped in File, Edit, Display, Objects and Help.
2. **Toolbar:** Two different toolbars are available giving quick access to some functions.
3. **Information:** This widget gives information about all the objects annotated and the dimensions of the table.
4. **Bounding Box Size:** This widget modifies the dimensions of the bounding box that contains the object of interest (OI).
5. **Object Pose:** This widget displays the pose of the bounding box containing the OI. This field also allows the user to manually modify all the pose parameters (x-y-z and roll-pitch-yaw). The absolute pose parameter values can be manually entered here.
6. **Pose Interaction:** This widget modifies the pose of the bounding box containing the OI. However, this modifies the pose parameters in predefined step sizes only (1 cm or 1 degree accordingly).
7. **3D visualizer:** This is a widget which provides a field to visualize the point cloud.

A.2.2 Information Widget - details

The Information widget contains details about the table size and all the objects annotated. When "Table" is selected or the triangle icon is clicked, it displays the dimensions of the table in the widget. When "Objects" is selected or the triangle icon is clicked, all annotated objects are displayed as a list in the widget. Also, when "Objects" is clicked, all the annotated objects are highlighted in different vivid pre-chosen colors.

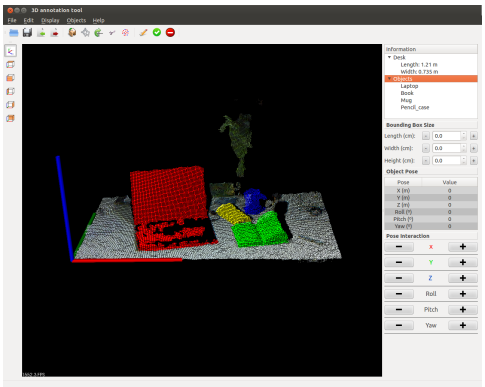


Figure A.3: All objects highlighted in vivid pre-chosen colors

When a particular object in the "Objects" list is selected, only that object is highlighted in the scene and it's bounding box is displayed. Moreover, it is also possible to modify the bounding box's properties by changing the pose and the dimensions of the bounding box A.3.4.

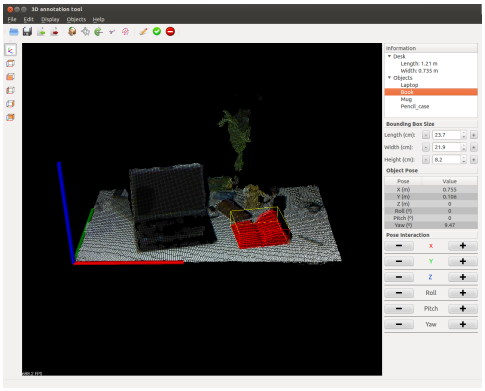


Figure A.4: One object selected and it's bounding box

A.2.3 3D Visualizer Widget - details

In the 3D Visualizer the point cloud loaded by the user is displayed along with the reference spatial coordinate system.

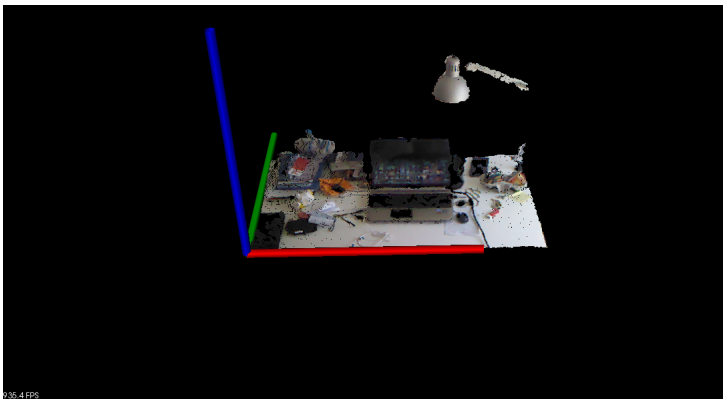


Figure A.5: 3D visualizer

Interactivity

Using the mouse, it is possible to change the point of view. The next modifications can be done:

Rotation Holding down the left mouse performs a rotation the current scene displayed in the 3D Visualizer widget.

Zoom Zoom in/out is done by scrolling the mouse.

Translate Holding down the scroll of the mouse performs a translation of the scene in the widget.

It is also possible to interact with the visualizer using the keyboard and some of the interactions are listed below:

Take snapshot Pressing j or J it is possible to take a .PNG snapshot of the current window view.

Display scale grid A scale grid corresponding to the previously defined imaginary axes is available for toggle by pressing g or G.

Help If h is pressed all the possible keyboard interactions are shown in the terminal.

A.2.4 Toolbar - details

The buttons available in the toolbar are listed below along with their respective actions:

Button	Action	Description
	Open	Section A.3.1
	Save	Section A.3.1
	Import	Section A.3.1
	Export	Section A.3.1
	Plane detection	Section A.3.2
	Plane definition	Section A.3.2
	Rotation	Section A.3.2
	Plane segmentation	Section A.3.2
	Downsample	Section A.3.2
	Coordinate system	Section A.3.3
	Top view	Section A.3.3
	Front view	Section A.3.3
	Left view	Section A.3.3
	Right view	Section A.3.3
	Back view	Section A.3.3
	Insert object	Section A.3.4
	Confirm object	Section A.3.4
	Delete object	Section A.3.4

A.3 Functions

To annotate a scene as it is desired, several actions were programmed in the 3D Annotation tool.

A.3.1 File Menu

Open PCD file

To load a .pcd file the following dialog box is opened to select the desired file:

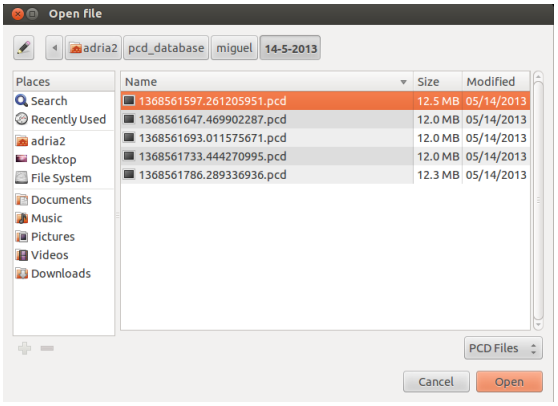


Figure A.6: Load file

Save PCD file

If the point cloud has been modified it is possible to save it.

Import object information

If in a previous session the user has saved the scene's information in a .xml file, this information can be imported and loaded. Then it is possible to modify this file by adding/deleting some objects or change their pose.

Export object information

Exports the annotation information into a .xml file with all the details about the scene. Everything was explained in section 5.3.4.

A.3.2 Edit Menu

The software provides some different tools to modify the loaded point cloud scene such as plane detection, rotation and segmentation.

Undo

This provides the option to be able to undo the actions of Filter and Plane segmentation.

Automatic plane detection and Manual plane definition

The explanation of these action can be found in section 5.3.1.

Roll 180

Sometimes, after having detected the correct dominant plane in the scene, it might be required to rotate the z-axis of the point cloud by 180 degrees to obtain the correct position of the coordinate system. This is achieved using this functionality.

Plane segmentation

This option provides a means to segment the dominant plane in the scene. This also implicitly places the coordinate system to the lower left corner of the dominant plane. This action removes all the points outside and under the dominant plane. Also is asked to the user which is the scenario. Therefore, after doing this action the point cloud has less points (less size) and with only the part of the scene needed.

Downsample

Downsample option gives the user a means to reduce the number of points in the displayed point cloud scene. This is useful when the resolution is higher than required for that scene. If the point cloud resolution is unnecessarily high it causes slowing down of the interactivity of the system. The Downsample works using user inputs to create a lower resolution voxel grid which merges many points in a voxel into a single point.

Preferences

This option provides the possibility to switch off some redundant user information messages during repeated use.

A.3.3 Display Menu

Coordinate system

It is possible to show and hide the axes of the reference spatial coordinate system in the visualizer.

Perspective

The perspective can be chosen from amongst five different options: Top view, Front view, Left view, Right view and Back view.

A.3.4 Objects Menu

The annotation of the objects is the most important feature of this software.

Insert new object

This functionality has been detailed in section 5.3.3

Confirm position

Once the bounding box is constructed around the points of the OI this option confirms the position and saves all the pose and dimension information related to the object.

Delete object

If the user for some reason wants to delete a previously annotated object, it can be done by selecting the name of the object from the annotated list.