Francisco Ferrer Sales

SLAM and Localization of People with a Mobile Robot using a RGB-D Sensor

July/2014

· U C ·

UNIVERSIDADE DE COIMBRA

FCTUC

University of Coimbra

Faculty of Sciences and Technology

Department of Electrical and Computer Engineering

SLAM and Localization of People with a Mobile Robot using a RGB-D Sensor

Francisco Ferrer Sales

A dissertation presented for the degree of
Master of Science in Electrical and Computer Engineering

Coimbra, 2014

FCTUC

SLAM and Localization of People with a Mobile Robot using a RGB-D Sensor

Supervisor:

Prof. Doutor Rui P. Rocha

Co-Supervisor:

Doutor David Portugal

Jury:

Prof. Doutor Hélder Araújo

Prof. Doutor Paulo Menezes

Prof. Doutor Rui P. Rocha

Report written for the dissertation subject, included in the Electrical and Computer Engineering Course, submitted in partial fulfilment for the degree of Master of Science in Electrical and Computer Engineering.

Coimbra, 2014

Acknowledgments

Several people played a fundamental role in the development of this project. I would like to thank them for everything they gave me during this process. Not only colleagues but also, and especially family and friends.

First of all, I would like to thank my supervisor, Prof. Dr. Rui Rocha, and my co-supervisor Dr. David Portugal for their guidance, support, and motivation. They were always extremely active, organized, and I felt that at every point I knew where I was at and where I was going to. It was a pleasureful experience and I hope that this professional and personal relationship does not cease here.

I would like to thank my family and close ones who were as always supportive and were a safe haven whenever I felt down. Also to my wonderful girlfriend Francisca who felt more closely the effects that this project had on me and was comprehensive and supportive.

A mention goes to my laboratory colleagues, João Santos, Gonçalo Martins, Pedro Trindade, Luís Santos, Pablo Lanillos, Micael Couceiro, and Rohit Chandra for the enjoyable work environment and also all suggestions and help they provided. Finally, I am also thankful to my colleagues during these years of the course.

Abstract

The evolution of Robotics lead us to aspire replacing human beings by autonomous robotic systems in the most discomforting or dangerous situations, and even to have them performing tasks that humans are incapable of. Such situations are often encountered when facing a catastrophic scenario and human rescue teams put themselves at risk in their heroic attempt to rescue victims. In nowadays society, the human life is highly valued and, consequently, Search and Rescue (SaR) missions became of extreme importance, precisely because their accomplishment can save several human lives. For that reason, rescue teams are heavily trained and, since they also comprise human beings, heavily equipped. However, they often lack technological tools to perform their mission better and in safety.

The main objective of this work is to study, develop and validate a fully-integrated robotic system to eventually explore catastrophic scenarios, by providing a map of the area and the people it detected while performing its task. This is a vital information in SaR missions, and this work aims at achieving this without risking a human life in the process. To that end, a mobile robot with a Red-Blue-Green and Depth (RGB-D) sensor mounted on its top was used. Furthermore, all approaches and methods were implemented in Robot Operating System (ROS) and thoroughly analyzed to assess their accuracy and performance. However, due to hardware limitations, the results are comprehensively not optimal, but they still provide useful information that might be of great help in SaR missions. Moreover, the study and analysis of this type of sensor and its usability are valuable and are now available for upcoming developments. This work is part of the Cooperation between Human and rObotic teams in catastroPhic INcidents (CHOPIN) project, which intends to develop a support system for small-scale SaR missions in urban catastrophic scenarios by exploiting human-robot cooperation.

Keywords: People Detection, Mapping, RGB-D sensor, ROS.

Resumo

A evolução na Robótica leva-nos a aspirar pela substituição de seres humanos por sistemas robóticos autónomos em situações desconfortáveis ou perigosas, e ainda, que estes executem tarefas irrealizáveis por seres humanos. Estas situações são frequentemente encontradas quando se enfrenta um cenário catastrófico e as equipas de salvamento humanas arriscam a sua integridade na tentativa heróica de salvar vítimas. Na sociedade actual, a vida humana é altamente valorizada e, consequentemente, as missões de busca e salvamento (SaR) tomaram uma importância extrema, precisamente porque o seu sucesso pode salvar diversas vidas humanas. Por esse motivo, as equipas de salvamento são fortemente treinadas e, porque são também compostas por seres humanos, são também bastante equipadas. Contudo, estas frequentemente não dispõem de meios e ferramentas tecnológicas que permitam melhorar o seu desempenho e a segurança com que executam a sua missão.

O principal objectivo deste trabalho é o estudo, desenvolvimento e validação de um sistema robótico totalmente integrado para eventualmente explorar cenários catastróficos, fornecendo um mapa da área juntamente com a localização das pessoas que detectou ao realizar a sua tarefa. Esta informação é vital para as missões de busca e salvamento e este trabalho pretende obtê-la sem recorrer ao risco de vidas humanas. Para este fim, foi utilizado um robô dotado de um sensor Red-Blue-Green and Depth (RGB-D) montado no seu topo. Para além disto, todos as abordagens e métodos utilizados foram implementados no Robot Operating System (ROS) e detalhadamente analisados para avaliar a sua precisão e desempenho. Contudo, devido às limitações do hardware, é compreensível que os resultados obtidos não sejam os ideais. No entanto, o estudo e análise deste tipo de sensores e a sua usabilidade são valiosos e disponíveis para ser utilizados em futuros trabalhos.

Este trabalho está inserido no projecto CHOPIN que tem como objectivo o desenvolvimento e um sistema de suporte para missões SaR de pequena escala em cenários urbanos de catástrofe, através da cooperação entre humanos e robôs.

Palavras-Chave: Detecção de Pessoas, Mapeamento, Sensor RGB-D, ROS.

Contents

List of Acronyms

AMCL	Adaptive Monte Carlo Localization
AP4ISR	Artificial Perception for Intelligent Systems and Robotics
ASR	Automatic Speech Recognition
CHOPIN	Cooperation between Human and rObotic teams in catastroPhic INcidents
CMU	Carnegie Mellon University
CPU	Central Processing Unit
DET	Detection Error Trade-off
FAST	Features from Accelerated Segment Test
FLANN	Fast Library for Approximate Nearest Neighbors
FoV	Field of View
FPPF	False Positives Per Frame
FPR	False Positive Rate
fps	frames per second
FRR	False Rejection Rate
GEVD	Generalized EigenValue Decomposition
GEVD-MUSIC	MUltiple SIgnal Classification based on Generalized EigenValue Decomposition
GHDSS	Geometric High-order Decorrelation-based Source Separation
GPU	Graphics Processing Unit

HARK	HRI-JP Audition for Robots with Kyoto University
HMM	Hidden Markov Models
HOG	Histogram of Oriented Gradients
HOD	Histogram of Oriented Depths
HRLE	Histogram-based Recursive Level Estimation
ICP	Iterative Closest Point
IR	Infrared
ISR	Institute of Systems and Robotics
KTP	Kinect Tracking Precision
LIDAR	LIght Detection And Ranging
LRF	Laser Range Finder
MA	Microphone Array
MFT	Missing Feature Theory
MRL	Mobile Robotics Laboratory
MUSIC	MUltiple SIgnal Classification
ORB	Oriented Binary Robust Independent Elementary Features
PCL	Point Cloud Library
PR	Precision-Recall
RANSAC	RANdom SAmple Consensus
RGB	Red-Blue-Green
RGB-D	Red-Blue-Green and Depth
ROC	Receiver Operating Characteristic
ROS	Robot Operating System
SaR	Search and Rescue

SIFT	Scale-Invariant Feature Transform
SIFTGPU	Scale-Invariant Feature Transform on Graphic Processing Unit
SLAM	Simultaneous Localization and Mapping
SNR	Signal-to-Noise Ratio
SURF	Speeded Up Robust Features
SVM	Support Vector Machine
TORO	Tree-based netwORk Optimizer
TPR	True Positive Rate

List of Figures

List of Tables

Chapter 1

Introduction

This dissertation describes and discusses a research work on “SLAM and Localization of People with a Mobile Robot using a RGB-D Sensor”. The work was carried out in the Artificial Perception for Intelligent Systems and Robotics (AP4ISR) team of the Institute of Systems and Robotics (ISR) University of Coimbra. The main goal of this project is to study the use of the Kinect in mobile robotics and use it to assemble an integrated system capable of building a map of the environment, and localizing people with respect to the map using visual and audio cues.

There were four main work phases. The first step was studying and testing solutions for mapping and navigation with a RGB-D sensor, the Kinect. The following phase was implementing a system capable of detecting and localizing people from the point cloud provided by the Kinect, allowing the execution of further tasks on the system, *i.e.* taking into account the computational load. The third step was studying the use of sound sources for people detection and the improvement they can introduce in the system. Finally, the integration of the previous modules was conducted, as well as the experimental evaluation and validation of the integrated system.

The current chapter presents the context and motivation, the main goals, and the outline of the dissertation.

1.1 Context and Motivation

Although robots are intended to be systems that perform tasks autonomously or interactively, they differ in terms of characteristics and capabilities. One of their most popular relevant applications is replacing humans in unpleasant situations, such as repetitive industrial tasks or in dangerous environments, as exemplified by D.Portugal and R.P.Rocha [61] with multi-robot patrolling of infrastructures. Even though most robots for industrial purposes are static, this is generally not the case in dangerous environments, such as those found in Search and Rescue (SaR) missions. In such harsh scenarios, robots must be able to navigate, explore, map, detect people, assist human first responders in rescue operations, and perform other related tasks.

Such missions are critical and of extreme importance because their successful accomplishment allows saving several lives. As a consequence, human rescue teams are often subject to heavy and specialized training. However, they usually face a lack of technological equipment and risk themselves in this process. Thus, Robotics plays a fundamental role by reducing this risk, and can be a great resource to human rescue teams.

Exploring, navigating, mapping the environment, and detecting people are key tasks in Robotics for SaR missions and other relevant applications. Since these environments are usually dangerous, mobile robots must be endowed with appropriate locomotion skills, provide accurate results, and the whole system should be affordable due to the risk of losing robots in such harsh environments.

In order to navigate in different environments, a robot must be able to sense and perceive its surroundings. This can be done by processing the information gathered by sensors. The most commonly used sensors in mobile robotics are active sensors, such as Laser Range Finders (LRFs), which can make the system too expensive because of their high price. However, in late 2010 Microsoft launched the Kinect, a low-cost sensor developed for gaming that integrates a 3D sensing system and a 2D camera system along with a Microphone Array (MA). Since the Kinect is affordable for any low-budget project, it has become popular in the Robotics research community due to the potential of Red-Blue-Green and Depth (RGB-D) sensors in mobile robot applications. The Kinect can be used in a variety of applications, either in complement to other sensors or to replace them.

A key requirement for a mobile robot that operates in an unknown environment is to autonomously localize itself and navigate. Thus, the depth data provided by the Kinect can be very helpful in map creation and localization, as recent research work has proven, *e.g.* [13]. Furthermore, it also provides images from the Red-Blue-Green (RGB) camera, which can be used to implement visual odometry in combination with the depth map, and implement useful visual navigation methods.

Besides navigation, mobile robots performing SaR missions need to detect and localize people. This task can be difficult in cluttered and noisy environments and still represents a challenge. However, research results show that the Kinect can be of great help due to its RGB-D system, which allows human detection and localization from color images and depth information [72]. Furthermore, the MA might improve the results in scenarios where the visual and depth cues are noisy, corrupted or non-existent, *e.g.* in the case of occlusions.

1.2 Objectives

This work addresses the following subjects: mapping, navigation and exploration, people detection with RGB-D cameras, and people detection with sound sources (MA). With this purpose, the Robot Operating System (ROS) framework and a Kinect mounted on top of a mobile robot (Pioneer 3-DX) were used.

Considering that the Kinect is a cheap, low precision sensor, the first objective is to assess the usability of the Kinect to explore, navigate and map indoor, cluttered environments without the use of any further sensors, and investigate how the environmental factors, such as light and low featured scenarios, affect the results.

The second main goal is the study and implementation of a lightweight people detection system based on the RGB and depth cameras. In the context of people detection, another goal of this project is this work is studying the contribution of the auditory information provided by the MA in the task of people detection and, if it is found valuable, studying and implementing a solution for people detection based on sound. This stage should also include a careful evaluation of the method.

The final and foremost objective is the integration of the aforementioned modules and the development of a mobile robotic system that autonomously navigates and maps indoor, cluttered environments. The final system should be able to assist human responders in small-scale urban SaR missions, and also locate potential victims on the provided map of the incident zone. The system should be tested with a physical mobile robot in different test scenarios.

1.3 The approach followed in this dissertation

This project addresses the problem of the lack of information about the incident zone in SaR missions. It is of extreme importance to provide the first responders with information that might fasten their assistance task to people in these harsh environments. However, the tasks of navigating, exploring, mapping, and detecting people both from visual and audio cues can be computationally heavy and difficult under these conditions.

The need to study and develop an integrated, low cost and fairly reliable system has arisen. In this dissertation, the chosen sensor is the RGB-D sensor Kinect, that also provides access to a MA. Despite the use of a sensor that provides 3D information, such information might be projected down to 2D to lower the computational burden and allow the performance of all the desired tasks.

1.4 Outline of the dissertation

The document is organized in six chapters. Each chapter refers to a distinct phase of the work developed. This first chapter introduced the context and motivation of this project, the objectives, and the approach followed in the dissertation.

Chapter 2 presents the subject of RGB-D mapping, 2D mapping, people detection based on visual, and people detection based on audio cues. It also details the techniques and algorithms used in the aforementioned tasks.

Chapter 3 introduces the selected sensor, the Kinect and outlines its characteristics, sensing mechanisms, accuracy rates, and its applications. It also provides insight on the software already available to use with the Kinect on the selected framework ROS.

The proposed system, its implementation, and all its structure are detailed in Chapter 4. This chapter presents all the modules that compose the system and describe them thoroughly. The ROS main nodal structure of the system is also presented.

In Chapter 5 all the experimental data is reported and discussed. It starts by addressing the results obtained for each module and proceeds to the real world tests conducted.

Finally, Chapter 6 sums up the work, drawing final conclusions and providing guidelines for future research.

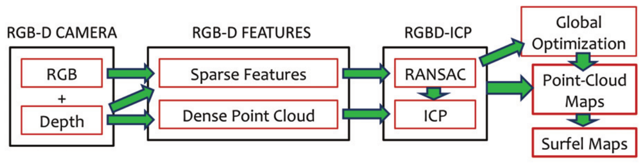
Chapter 2

Related Work

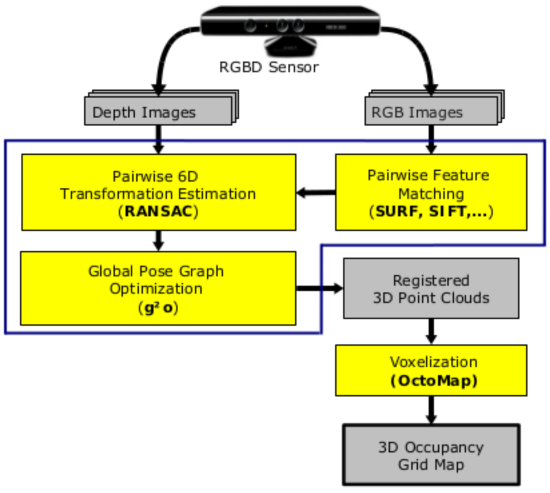
2.1 Mapping

Several relevant applications in obotics require the ability to create a model of the environment and to estimate the robot's pose regarding this model. For a mobile robot, for example, it is essential to know its location in order to navigate safely to a specific target. With the arrival of cheap RGB-D sensors, an approach for Simultaneous Localization and Mapping (SLAM) can be made, combining both depth information and visual features from the RGB camera to create dense 3D environment representations. However, 3D SLAM is a task with high computational burden, and consequently, in certain applications where the mobile robot's motion is constrained to a plane, *e.g.* SaR mission in a facility where a leakage of toxic substances occurred, the 3D information is projected to 2D to perform these tasks.

2.1.1 RGB-D Mapping



(a) RGB-D SLAM by Henry *et al.* [31].


RGBD Sensor
Depth Images
RGB Images
Pairwise 6D Transformation Estimation (RANSAC)
Pairwise Feature Matching (SURF, SIFT,...)
Global Pose Graph Optimization (g²o)
Registered 3D Point Clouds
Voxelization (OctoMap)
3D Occupancy Grid Map

(b) RGB-D SLAM by Endres *et al.* [20].

Figure 2.1: Overview of two different RGB-D SLAM approaches.

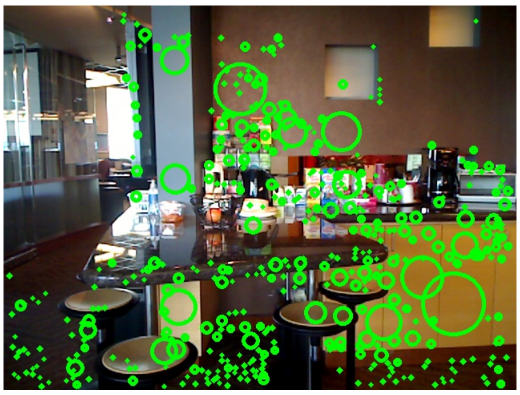
RGB-D mapping consists of generating dense 3D models of indoor environments using both the RGB images and their corresponding per-pixel depth. Several approaches have been presented in order to obtain those models.

Figure 2.1a illustrates the approach presented by Henry *et al.* [31]. Later on, Endres *et al.* [20] presented a similar approach, as shown in Figure 2.1b, differing on the fact that they discarded the Iterative Closest Point (ICP) that Henry *et al.* found often unnecessary, using the g^{2}o framework instead, presented by Kuemmerle *et al.* [40] to optimize the 3D graph pose. Also, on the post-processing step, where Endres *et al.* used a volumetric voxel representation that can be directly used for robot localization, planning and navigation, Henry *et al.* used Surfel Maps. The latter is a concise representation of the map that incorporates all the information from the RGB-D frames and it is based on surfels [59]. A surfel consists of a location, a surface orientation, a patch size and a color.

2.1.2 Different Stages in RGB-D SLAM

1 - Extracting visual features

In order to extract visual features out of the RGB images, *i.e.* pieces of information which are relevant to solve the computational task, several algorithms can be used. Endres et al. [20] evaluated the three most commonly used ones: Scale-Invariant Feature Transform (SIFT) [44], Speeded Up Robust Features (SURF) [11], Oriented Binary Robust Independent Elementary Features (ORB) [66] and Scale-Invariant Feature Transform on Graphic Processing Unit (SIFTGPU). An example of SIFT features on a RGB image can be seen in Figure 2.2a. An overview of these methods is summarized in Table 2.1.


(a) RGB image with SIFT features.

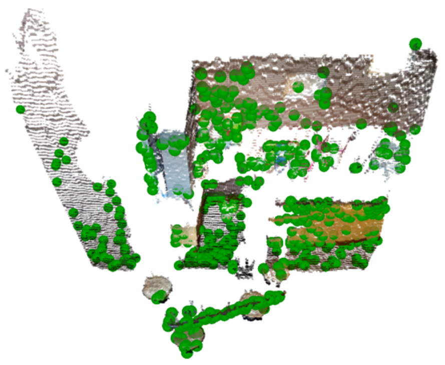

(b) Same SIFT features and their 3D position in the point cloud.

Figure 2.2: Example of RGB-D feature extraction [31].

In terms of accuracy, SIFTGPU results were similar to the ones delivered by SIFT and SURF. In contrast, ORB was less accurate and reliable. Considering its performance, SIFTGPU showed to be the best option, by performing about 3.5 times better than SURF. Despite being faster, ORB produces errors in several of the reported experiments.

Later on, Henry *et al.* showed that the Features from Accelerated Segment Test (FAST) [65] feature detector combined with the Calonder *et al.*'s feature descriptor [15] provided faster and more reliable results.

Table 2.1: Feature Detectors.

Detector	**Description**
SIFT [44]	Uses the result of difference of Gaussians function applied in scale space to a series of images to localize key points.
SURF [11]	It is based on sums of 2D Haar wavelet responses and makes an efficient use of integral images.
SIFTGPU	Same as SIFT but implemented on a GPU.
FAST [65]	Detects and analyses the neighbourhood of key pixels.
ORB [66]	Based on Binary Robust Independent Elementary Features, which directly compute binary strings from image patches, and SIFT.

2 - Feature Matching and Motion Estimation

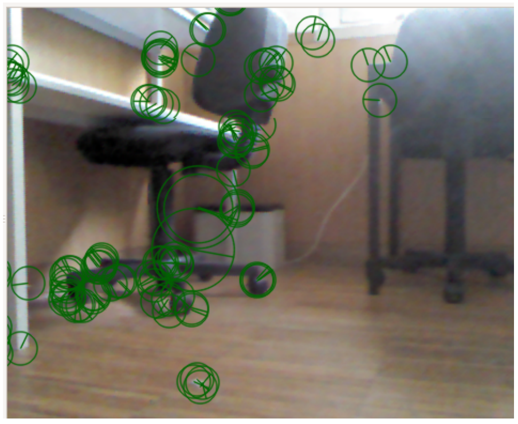


Figure 2.3: Features represented by green circles, and motion estimation represented by lines inside the circles.

Each image frame contains several features. In order to recognize the scenario, features from previous frames must be identified on the current frame, *i.e.* feature matching. When the features are localized in both frames, a motion vector can also be estimated by analyzing the position of the corresponding features. Figure 2.3 shows features and motion estimation represented on an image.

Feature matching and motion estimation must be computed at least once per frame. However, if the current frame is only matched against a predecessor, errors will be accumulated over time. Though it is costly to compute for many frames, the system must be able to detect loop closures in order to be accurate (see Section 2.1.3 on page 9 for more information).

Thus, this process must be optimized and, once again, Endres *et al.* [20] found out that the Fast Library for Approximate Nearest Neighbors (FLANN) [47] reduces this cost by about a factor of two in comparison with a brute force approach. FLANN is a library for fast approximate nearest neighbor searching in high dimensional spaces. It contains algorithms for that purpose and a system that automatically chooses the best algorithm.

3 - Estimation of the Optimized Relative Transform

A RGB-D sensor is composed by two cameras: the RGB camera and the Infrared (IR) camera. RGB-D SLAM makes use of information from both cameras, however the cameras are not in the same position in space. Considering that features from the RGB image must be matched with the information from the depth image, an optimized relative transform must be estimated. One of the most popular algorithms used over time for estimation of the relative transformation between frames is RANdom SAmple Consensus (RANSAC), which was introduced by Fischler and Bolles [25]. Henry *et al.* [31] first used the 3D version of RANSAC for RGB-D sensors, but concluded after further research that RE-RANSAC, a re-projeçtion error method, improved the results with this type of sensor.

4 - Pose Graph Optimization

Some SLAM methods use a graph whose nodes correspond to robot's poses at different points in time, the pose graph; methods falling in this category are usually known as GraphSLAM. For the robot to be able to map an environment, it must be able to estimate accurately its trajectory from relative pose measurements, which is the objective of pose graph optimization. The optimization of small pose graphs can be done in real time, *i.e.* for every frame, since it is fast enough. Over time, the process takes longer with densely connected poses. However, if the motion estimates are reliable enough, the global optimization, *i.e.* for several frames, is not required in every step.

The most commonly used methods are g^{2}o [40], Tree-based netwORk Optimizer (TORO) [29] and Sparse Pose Adjustment [39]. The g^{2}o framework optimizes nonlinear graphs using a least-squares approach. In contrast, TORO applies a stochastic gradient descent to minimize the error introduced by constraints. On the other hand, sparse pose adjustment is based on the well-known Levenberg-Marquardt method [46].

2.1.3 The Loop Closure Problem

Very often, when a robot navigates within an environment, it may return to a position where it has already been before. In terms of SLAM, this may represent the cause of estimation errors if the robot is not able to recognize places that it has already visited previously.

The alignment between successive frames is a solid method for tracking the position over a short distance, *i.e.* scan matching. However, errors in the alignment of two frames or noise may cause an inaccurate estimation of position over time. Consequently, when the sensor returns to a place it has already visited, the map may have two representations of the same region in different locations. This is known as the loop closure problem.

In order to deal with this problem, Henry *et al.* suggested to break it down into two parts: 1) loop closure detection and, 2) correcting the map to merge duplicated regions.

To detect loop closures, they suggest selecting keyframes, which are a subset of the aligned frames. A keyframe is selected whenever a frame is not within a threshold defined for matching, *i.e.* when the frame does not have enough 3D feature point matches with the previous keyframe. Each time a new keyframe is created, it is compared with previous keyframes and if enough geometrically consistent 3D feature point matches are recovered, a loop closure is detected. Then, the TORO algorithm is used to deal with this constraint.

2.1.4 2D Mapping

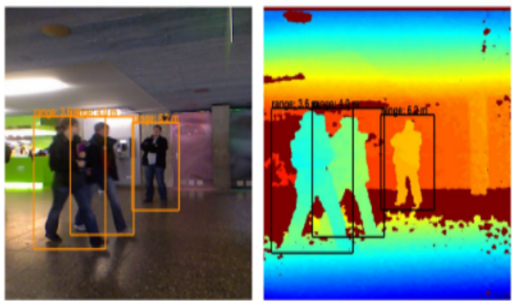
Although RGB-D sensors allow the creation of 3D models of the environment, it is a task with high computational burden, and even the fastest, state of the art algorithms represent a considerable computational cost. Therefore, in robotic applications that besides mapping, also need to perform other tasks and its movement is restricted to a 2D plane such as indoor scenarios, a projection of the 3D information to 2D might be considered, thus allowing the use of 2D SLAM algorithms. Such algorithms require a much lower slice of the computational systems' capabilities.

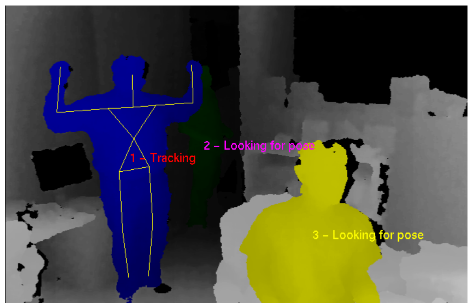
The most popular 2D SLAM algorithms rely on probabilities to cope with noise and estimation errors. There are several implementations based on Kalman Filters and Particle

Filters [18]. An alternative approach is graph-based SLAM, using pose graph optimization (see page 8). In this case, algorithms use the data to build a graph composed of estimated poses, local maps and their relations, in order to compute a consistent global map. ROS, the robotic framework used on this project, already comprises a set of 2D SLAM algorithms, such as *GMapping*, *HectorSLAM*, *KartoSLAM*, etc. [68].

HectorSLAM heavily relies on fast scan matching and does not use odometric information from the robot which makes it a viable option for aerial robotics [38]. KartoSLAM is a graph-based algorithm that employs a highly optimized decomposition for sparse linear systems [77]. *GMapping* [30] is the most used SLAM algorithm in robotics and it is a Rao-Blackwellized Particle Filter SLAM approach. It takes the movement of the robot and previous observations into account, achieving interesting performance with a low number of particles, *e.g.* 30 particles.

2.2 People Detection from Visual and Depth Data


(a) People Detection with RGB-D Cameras [72].



(b) People Tracking with RGB-D Cameras [1].

Figure 2.4: People Detection and Tracking using RGB-D cameras.

People detection is important in diverse Robotics applications, such as in human-machine interaction. Much effort has been put in human-robot interaction for the past few years so that robots can engage and interact with people in a friendly way [24]. Detecting and localizing people is essential before initiating such interaction. However, some of this research has relied solely on 2D visual information provided by cameras, *e.g* [45]. Some methods involve statistical training based on local features, such as Histogram of Oriented Gradients (HOG) [17] and Edge Orientation Histogram (EOH) [41], while other methods involve extracting interest points in the image, such as SIFT features. Recently, with the popularization of 3D sensors, much research has been done on people detection from visual and depth information. Another important application is in intelligent vehicles domain, when avoiding collisions. In this context, there is interesting work, such as [62], [36], [42].

Based on other sensory information, Arras *et al.* [8] suggested a system which segments a complete scan from a LRF into small clusters and then classifies the cluster in both human or not human. Spinello *et al.* [71] extended this idea by extracting a fixed set of 2D vertical scan lines from a full 3D point cloud. Then, the detection is performed separately in each layer. Later, the layers are fused.

Such methods present drawbacks in certain scenarios where the conditions are not ideal, such as cluttered environments or human subjects with articulated poses. Due to these constraints, depth information retrieved from RGB-D cameras is of great use, because objects may not have consistent color and texture but must occupy an integrated region in space. Moreover, they are usually inexpensive and suitable for typical human environments and consequently, have eased the research on this particular subject.

Subsequently, it has been shown that depth images are robust to moderate changes in color and illumination. Additionally, they provide further 3D information [35]. A relevant approach using depth information was proposed by Satake *et al.* [69], where template matches (depth templates) are used to detect the upper human body. On the other hand, Bajrachaya et al. [10] reduced the point cloud to a 2.5D map to preserve the low computational effort and then people detection is based on different 2D features.

Most recently, L.Xia and Aggarwal [79] introduced a new method that is based solely on the depth image of RGB-D cameras; they specifically used the Kinect, as summarized in Figure 2.5. Given the input depth array, a pre-processing step is made to reduce the noise. Then, the 2D chamfer distance matching algorithm is used to scan across the image and give possible regions that may contain people. These regions are examined using 3D head model. If found, the parameters of the head are extracted from the depth array and used to generate the 3D head model. Finally, a region growing algorithm is applied to extract the whole body contour, which is then segmented from the background.

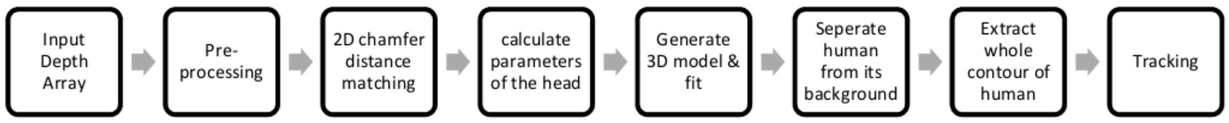

Input Depth Array
Pre-processing
2D chamfer distance matching
calculate parameters of the head
Generate 3D model & fit
Seperate human from its background
Extract whole contour of human
Tracking

Figure 2.5: Overview of human detection method [79].

Spinello et al. [72] introduced a method that combines both depth information and color images to detect people. HOGs are used to detect human bodies from image data and they introduced Histogram of Oriented Depths (HOD), a method for dense depth data that derives from HOG. Finally Combo-HOD is also presented, which probabilistically combines HOG and HOD.

HOD Detector

Based on the idea of the use of HOG [17], which allows the detection of human bodies in RGB images, HOD works similarly for dense depth data, see Figure 2.6. It considers a subdivision of a fixed window into cells, then computes a descriptor for each cell, and collects the *oriented depth gradients* into 1D histograms. The key idea is that an array of local depth changes can robustly characterize the local 3D shape and appearance. The resulting HOD features are used for training a soft linear Support Vector Machine (SVM) as proposed by Dalal and Triggs [17].

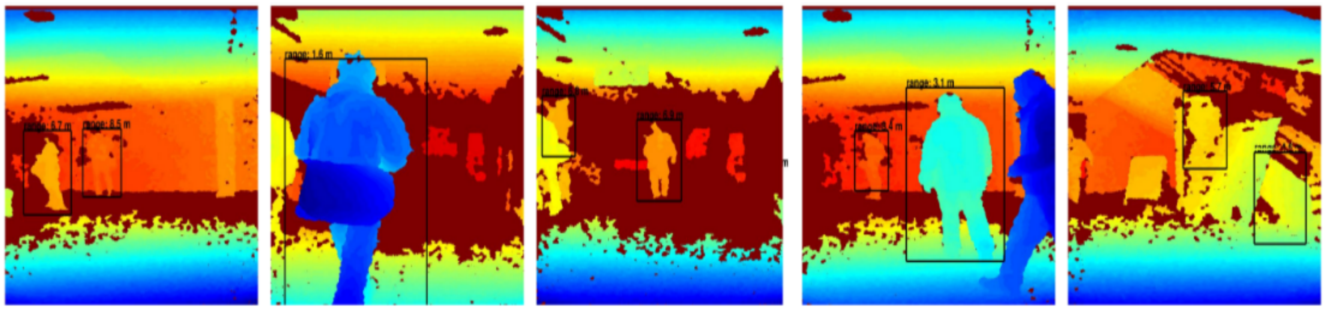


Figure 2.6: Qualitative results of people detection with HOD. Ranges are varying and occlusions may occur [72].

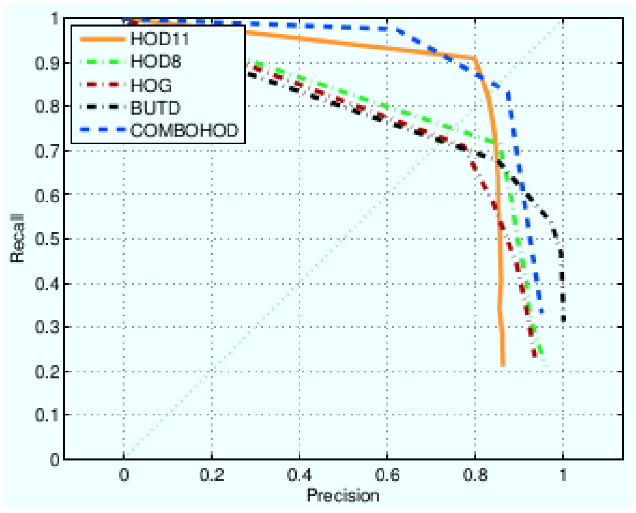


Figure 2.7: Precision-Recall curves for depth-based, image-based, and combined RGB-D based detection models. HOD is evaluated at different depth discretizations, 8bits and 11bits [72].

Combo-HOD Detector

The two detection approaches that use HOG and HOD consider information either from image or depth data. In order to take advantage of the richness of RGB-D data, Spinello *et al.* [72] presented Combo-HOD, a method for detecting people that combines both sensory cues. This method applies separately the HOG-based detector to the image and the HOD-based detector to the depth image. When no depth information is available, Combo-HOD degrades to regular HOG-based detector. However, it requires a proper calibration in order to match correctly both images. When both descriptors are classified, the information is fused using a probabilistic method. Results and comparisons to other methods can be seen in Figure 2.7. Combo-HOD outperforms other methods and even though it can run at 30 *fps*, it relies on a heavy Graphics Processing Unit (GPU) implementation due to the fact that this algorithm densely scans the whole frame.

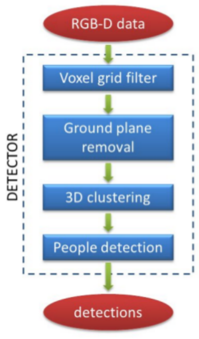


Figure 2.8: Overview of detection system by Munaro *et al.* [49].

Recently, Munaro *et al.* [49] presented a method that does not require a GPU implementation and still presents fast and accurate results. The only drawback is the assumption that people stand on the ground plane, consequently it does not present accurate results for people that stand considerably above or below that plane, *i.e.* performs poorly for people climbing stairs or sitting behind a table.

The system is depicted in Figure 2.8. The voxel grid filter downsamples the acquired point cloud. For each frame, the space is divided in voxels and all points contained by each voxel are represented in the coordinates of the centroid. After the process of downsampling the point cloud, an estimation of the ground plane is made using a RANSAC-based least square method. Then, the ground plane points are removed from the downsampled point cloud, allowing the separation of the sub-clusters that might contain people. For each of this sub-cluster, a HOG-based people detector, such as presented in [17], is applied to the corresponding part of the RGB image, defining weather it is a person or not.

2.3 People Detection from Sound Sources

In scenarios with low visibility, when human-robot interaction is required, it is crucial to analyze the environment in order to perform tasks such as sound source localization, sound source separation and recognition. Computational Auditory Scene Analysis [14] aims exactly at analyzing an auditory scene. It has mainly been studied for indoor use. However, outdoor use might be of great importance, for example in case of disaster to detect human victims that may cry for help under debris or not in line of sight, where visual information cannot provide enough relevant data or is corrupted by smoke, fog or dust.

The first issue to address in these or any other catastrophic scenarios is the high noise, *i.e.* extremely low Signal-to-Noise Ratio (SNR). A technique commonly used to localize sources and deal with high noise situations is MUltiple SIgnal Classification (MUSIC) [70]. Okutani *et al.* [56] suggested extending MUSIC for robot audition by introducing Generalized EigenValue Decomposition (GEVD), proposing the so-called MUltiple SIgnal Classification based on Generalized EigenValue Decomposition (GEVD-MUSIC). Later on, they proposed a new method, incremental MUltiple SIgnal Classification based on Generalized EigenValue Decomposition (iGEVD-MUSIC) [56], which can deal with dynamical changes of noise by estimating a noise correlation matrix incrementally. This method can be used to localize sound sources as is our purpose.

After localizing the sound sources, if there are several simultaneous emissions of sound, they must be separated. One of the most common and accurate method is Geometric High-order Decorrelation-based Source Separation (GHDSS) [52]. With the localization of the sound sources, GHDSS allows to separate the sounds and obtain the audio that corresponds to each localization, based on the direction of the sound sources and the number of the sources present on the audio data.

Finally, with a localization assigned for each separated sound, speech recognition systems might be used to determine if the sound source is human or irrelevant noise.

2.3.1 HARK

Nakadai et al. [50] released a powerful open-source robot audition software that provides a set of useful algorithms, the HRI-JP Audition for Robots with Kyoto University (HARK). In [50], the authors identify noise as the most relevant problem in Automatic Speech Recognition (ASR), and noise perceived by a robot can be categorized as: directional noise, diffuse noise, reverberation, and ego-noise.

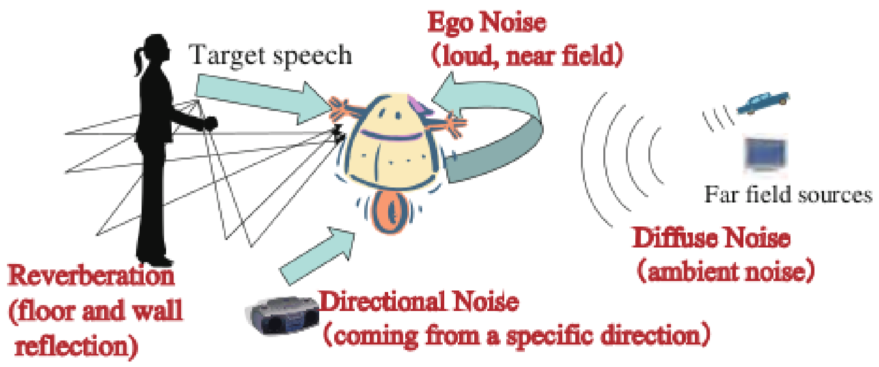
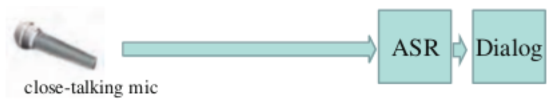
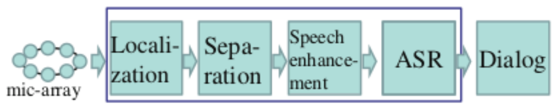


Figure 2.9: Sound sources surrounding a robot [50].

Figure 2.9 depicts the different types of noise. In order to deal with noise, and the consequent low SNR, for ASR applications, HARK includes sound source localization, separation and speech enhancement, as seen in Figure 2.10b.



(a) Processing flow for a conventional speech dialog system.



(b) Processing flow for the HARK-based speech dialog system. The HARK-related part is inside the rectangle.

Figure 2.10: Difference between conventional system and HARK-based systems [50].

Sound Source Localization

Sound source localization consists in detecting the source of the sound. The basic idea of the process is to analyze the input of the same sound signal in different microphones, knowing their relative spatial location, and determine the position of the sound source with respect to the MA. To this end, HARK provides access to the aforementioned MUSIC and MUSIC-based algorithms which are adaptive beamformers and provide noise-robust sound source localization.

Sound Source Separation

In order to recognize more accurately the human voice in a input sound signal, it is necessary to separate it from the remaining data in the audio signal. There are two major sound separation methods: blind source separation [58], [9] and beamforming [28]. HARK provides hybrid algorithms of beamforming and blind separation, such as the Geometric High-order Decorrelation-based Source Separation with Adaptive Stepsize [52].

Speech Enhancement

Speech enhancement consists of improving speech quality by using various algorithms to obtain better results in speech recognition. There are several techniques such as:

- Filtering Techniques: Spectral Subtraction Method [12], Wiener Filtering [23], Signal Subspace Approach [21];
- Spectral Restoration: Minimum Mean-Square-Error Short-Time Spectral Amplitude Estimator [57];
- Speech-Model-Based, *e.g.* [43].

To reduce the noise and improve the results of speech recognition, the Histogram-based Recursive Level Estimation (HRLE) [51] is also available in HARK, and also leakage-noise estimation methods.

Performance and Applications

Nakadai *et al.* [50] implemented HARK, and assessed its performance. For most activities, the total process time was less than 10ms, revealing that HARK is reliable and fast.

Another interesting feature of HARK in the context of the current work is that it is compatible with ROS and the Kinect. Also, considering that it provides basic functions for robot audition, such as sound source localization, sound source separation, speech enhancement, and automatic speech recognition, HARK can be used in parallel with other software in order to to perform multiple tasks, including mapping, victim detection and localization.

For victim detection, HARK provides sound source localization algorithms that allow the estimation of the sound source localization in space. It also provides sound source separation that allows the separation of directional sound such as human voices from the rest of the data in the sound stream. And finally, it includes an ASR system that applies the Missing Feature Theory (MFT) [73]. This way, HARK allows estimating the localization of a sound source, it's separation from the rest of the sound data, and then asserting whether it is or not a human voice.

2.3.2 CMU PocketSphinx

Although HARK provides an ASR system based on the MFT, it is neither as accurate nor lighter in computational load than the CMU PocketSphinx [34].

The PocketSphinx is an open-source real-time continuous speech recognition system light enough to work on hand-held devices. It is a porting and optimization of the *CMU Sphinx II* [33]. Therefore, it is based on Hidden Markov Models (HMM) [64] because a speech signal can be viewed as a piecewise stationary signal or a short-time stationary signal, *i.e.* in a short time-scale (*e.g.* 10 ms), speech can be approximated as a stationary process. Additionally, it inherits its application programming interface, and consequently, it is available to use and it is one of the most powerful and lightweight software in this particular field of study.

2.4 Summary

In the current Chapter, the related work that served as an inspiration to this project was presented. The subject of mapping was addressed in Section 2.1, comprising both RGB-D mapping and 2D mapping. The following addressed subjects were people detection from visual and depth data in Section 2.2, and people detection from sound sources in Section 2.3, which comprised the presentation of HARK and CMU PocketSphinx.

Chapter 3 presents the RGB-D sensor that was used, its characteristics, its suited applications, and the software already available to work with it on the selected framework, ROS.

Chapter 3

The Microsoft Kinect

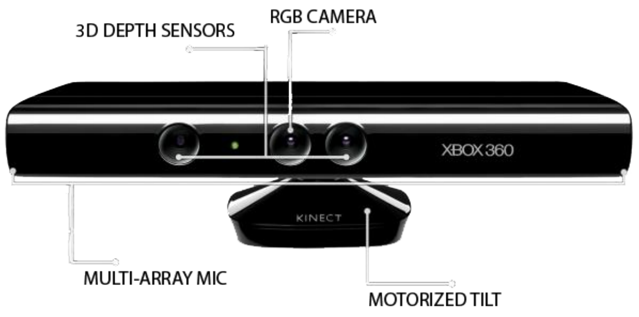


Figure 3.1: The Kinect and its main components [81].

The Microsoft Kinect is a revolutionary sensor designed by PrimeSense and launched in 2010, being widely used in the gaming industry. However, due to its interesting features, it is also suitable for robotic use, specifically indoor navigation, mapping, people detection and tracking. It has become popular because of its low price and its capability to provide RGB images and depth information simultaneously and even combine them to create a coloured point cloud. The basic parts of the Kinect are shown in Figure 3.1. These are: the MA, the motorized tilt system, the RGB camera, and the 3D depth sensors composed by an IR projector and an IR camera.

3.1 Depth Measurement

The depth measurement system comprises an IR projector and an IR camera. Depth measures are accomplished through a structured light technique. Cruz *et al.* [16] showed that the projector emits a single beam that is split into multiple beams in order to create a pattern of speckles projected in front of the Kinect. The emitted pattern is then captured by the IR camera and is correlated against a previously defined reference pattern. Figure 3.2 shows and example of the pattern captured by the IR camera.

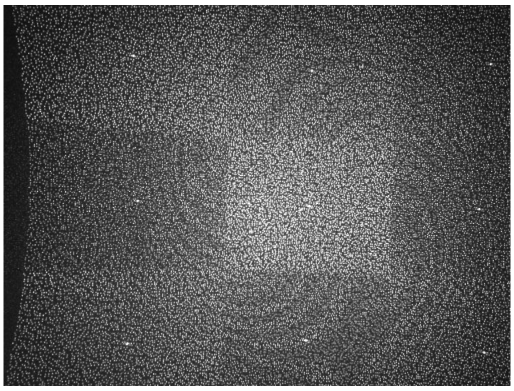


Figure 3.2: Emitted pattern of speckles by the Kinect [16].

When a speckle is projected on an obstacle whose distance to the sensor is not the same as the reference plane with known distance, its position on the captured IR image will be shifted accordingly. By calculating the shift of all speckles, a disparity image is obtained. Then, for each pixel, given the known depth of the reference plane and the disparity, the distance to the sensor can be computed using a triangulation model, as Khoshelham and Elbernik described in [37].

3.1.1 Triangulation model

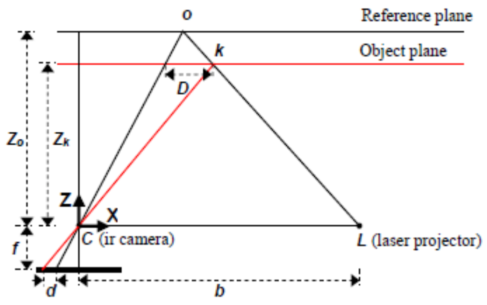


Figure 3.3: Schematic representation of depth-disparity relation [37].

In order to compute the depth of each pixel with information of disparity and the distance between the reference plane and the sensor, a triangulation method is used.

In Figure 3.3, k is the point in the object plane, b is the base length, f is the focal length of the IR camera, D is the displacement of the point k, Z_k is the depth of the point and Z_o is the distance to point k's projection on the reference plane. Z_o, f, and b are constant parameters obtained by calibration.

Figure 3.3 illustrates the relation between the distance of an obstacle point k to the sensor and the reference plane's projection of the same point k and their corresponding disparity d. In this schematic, we consider a depth coordinate system with its origin at the perspective center of the IR camera. The Z axis is orthogonal to the image plane and points towards the obstacle and the X axis is perpendicular to the Z axis in the direction of the baseline b between the IR camera center and the IR projector.

Considering the similarity of triangles, we have:

$$\frac{D}{b} = \frac{Z_o - Z_k}{Z_o}, \tag{3.1}$$

$$\frac{d}{f} = \frac{D}{Z_k}. \tag{3.2}$$

Thus, we can obtain the depth from equations 3.1 and 3.2:

$$Z_k = \frac{Z_o}{1 + \frac{Z_o}{fb} d}. \tag{3.3}$$

As a result, equation 3.3 provides the mathematical background to compute depth from the constant parameters Z_o, f, and b.

With the output of these calculations along with further processing for representation, and combining with the information from the RGB camera, the Kinect provides coloured point clouds.

3.2 Kinect Data Accuracy

The Kinect, as well as other low-cost range sensors[1], is an attractive alternative to expensive LRFs and LIght Detection And Ranging (LIDAR) systems. However, its accuracy has to be analyzed to assess whether it is suited for mobile robotics' use. The Kinect captures depth images at a frame rate of about 30 frames per second (fps), simultaneously with color images. Combining both information results in a coloured point cloud that contains about 300,000[2] points for each frame.

3.2.1 Precision and Resolution

The resolution of the IR camera determines the point spacing of the depth data. The Kinect's IR camera produces a constant 640×480 pixels per frame, which led Khoshelham and Elbernik to conclude in [37] that the depth resolution is inversely proportional to the

[1]Asus Xtion Pro is also a RGB-D sensor.

[2]The Kinect captures frames with 640×480 pixels, thus $640 \times 480 = 307,200$ points.

distance to the sensor, *i.e.* the depth information is less precise for objects farther away from the sensor, and for mapping applications the range should be 1 to 3 meters. Viager concluded that the Kinect is, in general, fairly accurate for a range of 0.6 to 4.6 meters [76].

Table 3.1: Kinect specifications [76].

Specification	**Value**
RGB and Depth images resolution	640×480
Frame rate	30 fps
Ideal operation range	$0.6m - 4.6m$
Ideal operation range for people detection	$\approx 1m - 3m$

Table 3.2: Approximate values of the Kinect Field of View (FoV) in degrees [76].

	RGB	**IR**
Horizontal	62	58
Vertical	48	44
Diagonal	72	69

3.3 Applications in Mobile Robotics

The information provided by the Kinect is of great use in the mobile robotics field. Tolgyessy and Hubinsky further analysed this sensor to determine which applications it is best suited for, as reported in their work in [75]. Below, three examples of applications are provided.

3.3.1 Obstacle Avoidance and Collision Detection

In order to safely navigate through any environment, a mobile robot must always protect itself from obstacles and other collisions, *e.g.* walls or people. Since the Kinect provides a depth map with good resolution, this information alone can be used to that end. However, the depth measurement system reveals some issues with certain obstacles, such as reflective or transparent materials and in non-ideal environments, such as those with strong lightning. Therefore, combining depth information with RGB data is beneficial. To assure the absence of collisions, it is necessary to program the robot to stop its motion and eventually re-plan its trajectory whenever an obstacle is below a safety distance. This can be achieved through the depth map by verifying the existence of pixels with depth information below the defined threshold.

3.3.2 Localization and Navigation

An autonomous robot should be able to localize itself and navigate to a precise destination. For this purpose, the information from the Kinect's depth map is useful in building a

3D or 2D map and localizing the robot, as well as avoiding obstacles and collisions as Henry *et al.* [31] concluded. Furthermore, the images from the RGB camera can be combined with the depth maps for visual odometry applications, to enable more precise localization.

3.3.3 People Detection

Detecting human bodies in pictures or videos is a challenging subject and several RGB image-based approaches have been proposed. Nevertheless, these methods are not accurate under certain circumstances, *e.g.* cluttered background, articulated human poses, etc.. Another alternative would be to use range scans from LRFs, but the price of a LRF is not always affordable. The same can be said for stereo-vision, along with the fact that it requires further computation.

The Kinect is a cheap sensor that provides both dense depth information and RGB images. The use of depth maps and 3D information is of great help facing this problem, and RGB-D cameras, such as the Kinect, are reliable resources as L. Xia and Aggarwal [79] confirmed.

Moreover, it also provides audio information from its MA, which also allows detecting of people based on auditory cues.

3.4 Kinect on ROS

3.4.1 Robot Operating System (ROS)

Writing software for robots is a difficult task, especially with the highly varied hardware used in Robotics. Furthermore, the software must cover all levels, from driver-level, to perception and abstract reasoning. Over the years, a wide variety of frameworks have been created to ease the software prototyping.

In this context, ROS is a framework devoted to large-scale integrative robotics research and, thus, it is becoming more important as robotic systems grow in complexity. ROS is most likely not an optimal solution for particular robotic systems because of the constraints of developing a large-scale solution. However, it is suitable for virtually every system.

Quigley et al. made an overview of ROS in [63]. According to the authors, the philosophy of ROS is based on several aspects referred below.

- Peer-to-Peer: system consist of processes, called nodes that can run on different hosts;

- Multi-lingual: support various languages such as *C, C++, Python, Octave, Lisp*;
- Tools-based: based on a *microkernel* design complemented with tools;
- Thin: drivers and algorithms built in standalone libraries, *i.e.* easy to reuse;
- Free and Open-Source: all its source code its publicly available and developments are promoted.

3.4.2 ROS and the Kinect

As researchers started to use the Kinect in robotic systems, significant work was made using ROS. Most of which is currently available and ready to use. There are two main driver implementations for the Kinect on ROS: the *freenect_stack* [3], and the most commonly used one *Openni_kinect* [5]. These drivers allow a fast collection of the information provided by the Kinect. There are also several software usable with the Kinect:

- An approach to RGB-D SLAM of the method described in [19] and on [2];
- A navigation system using the approach described in the appendix C.1.2 [4];
- A frontier-based approach for exploration [80] available on [6].

3.5 Summary

The current Chapter presented the selected RGB-D sensor, the Kinect. It detailed how the sensor acquires depth measurements in Section 3.1, addressed its accuracy in Section 3.2, presented some of its applications in Section 3.3, and described its integration in ROS in Sectio 3.4.

Chapter 4 presents the proposed system in detail and the algorithms and methods applied to obtain the desired results.

Chapter 4

Proposed System

4.1 System Description

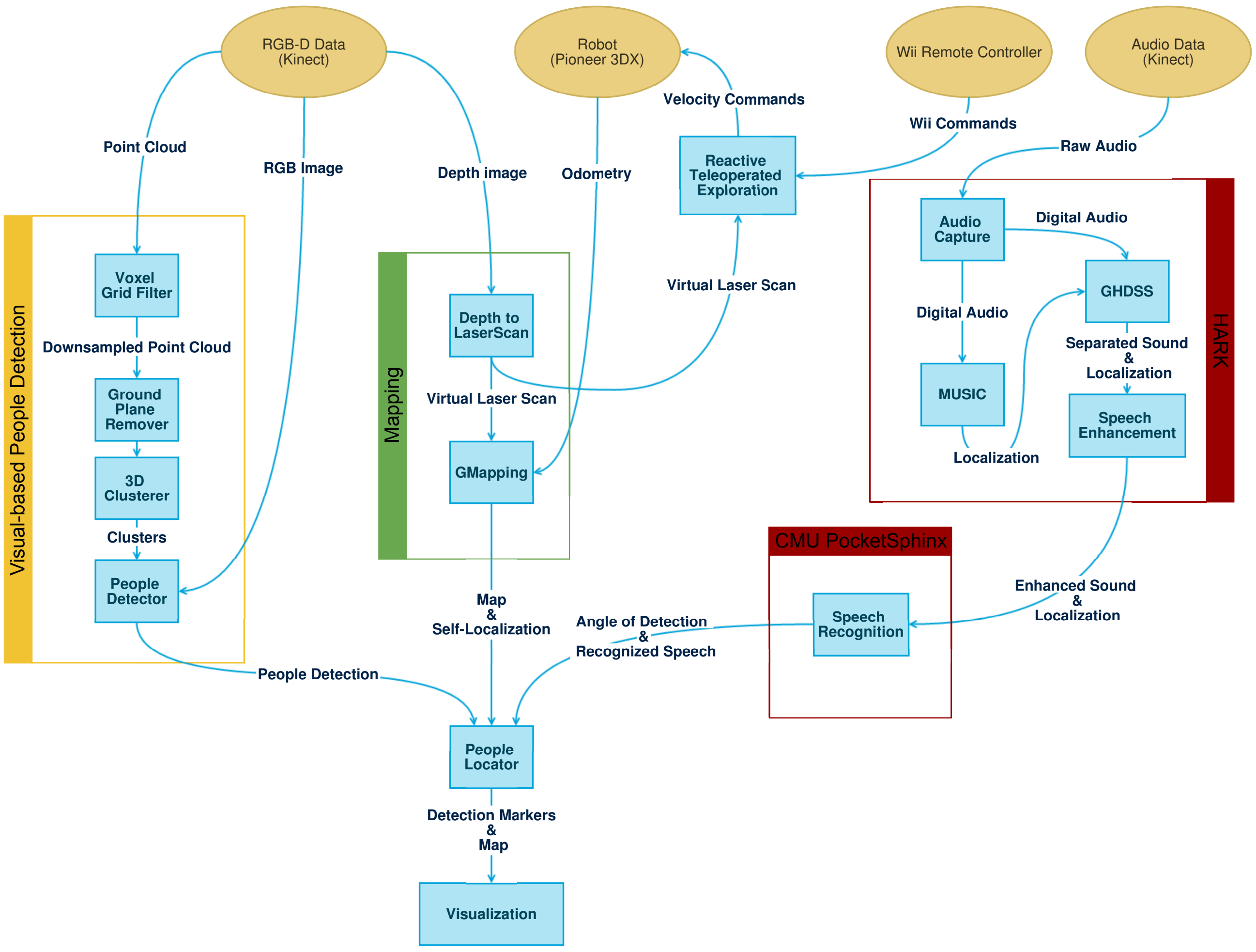


Figure 4.1: System Overview.

The proposed system for people detection and mapping using solely a RGB-D sensor, which also provides a MA, is shown in Figure 4.1. The system comprises four main modules: the visual-based people detection module, the mapping module, and the voice recognition and localization module, which is composed by the HARK module for sound processing and the CMU PocketSphinx module for speech recognition.

The top level of Figure 4.1 is representative of the hardware and drivers used and is further described in section 4.1.1. The mapping module of the system is presented in 4.1.2. Section 4.1.3 describes the proposed approach to exploration with the Kinect. The visual-based people detection is detailed in section 4.1.4. Afterwards, section 4.1.5 comprises the HARK module and the CMU PocketSphinx module and corresponds to the description of the audio-based people detection module. Finally, section 4.1.6 shows how the system handles both detection modules and the map to integrate all the information and present it to the end-user.

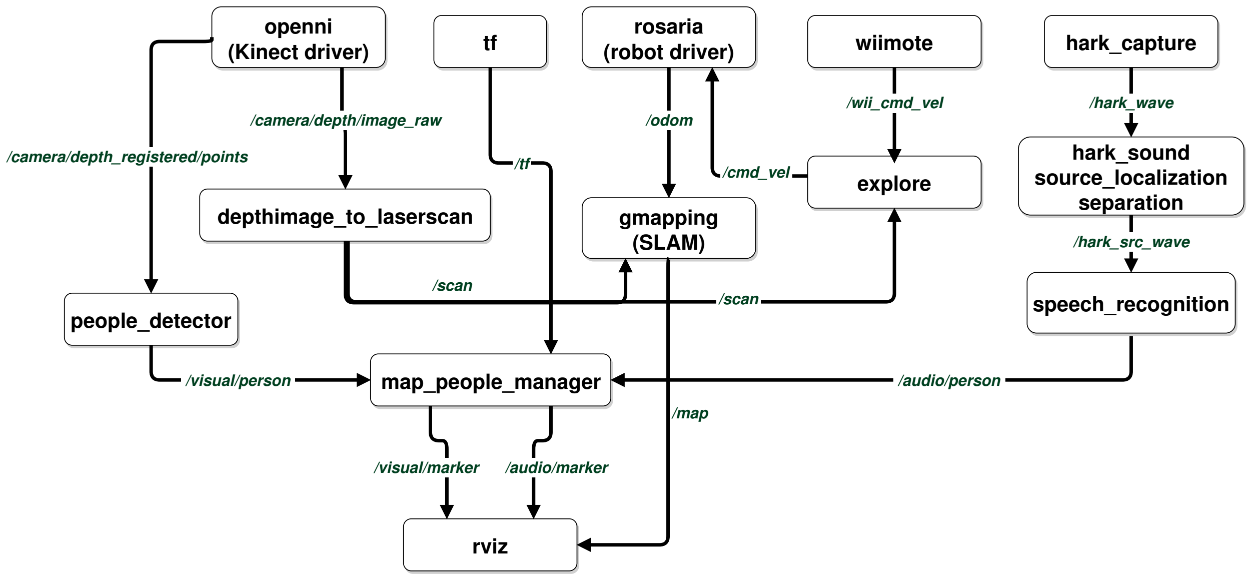


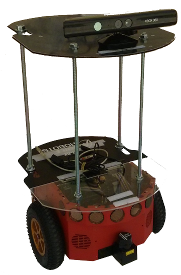
Figure 4.2: Summarized ROS system implementation. Nodes represented on boxes and topics on arcs.

For the implementation of the system, ROS was the selected framework due to its reputation as a Robotics middleware and the set of powerful tools, libraries, driver and other resources that it provides. In addition to an easier development, it also introduces hardware abstraction [63]. In Figure 4.2, a summarized graph of the system's implementation on ROS is presented. The ROS nodes and topics that do not take part in performing the system's main tasks are ommitted.

4.1.1 Hardware and Drivers

The available hardware to build the system is the Microsoft Kinect and the Pioneer 3-DX mobile robot with a custom elevated structure built for the Kinect (see Figure 4.3a). As mentioned in Chapter 3, ROS already provides a large set of tools and software to develop robotic software. In this particular case, the OpenNI driver [5] is used to retrieve all the data from the Kinect, as well as with other RGB-D sensors based on the technology from PrimeSense. Furthermore, for the popular robot base Pioneer 3-DX, there is also a driver available on ROS, *ROSARIA* [7] that collects odometry and sensor data from the robot and also allows sending velocity commands to control it.

4.1.2 Mapping


(a) Photo of the Pioneer 3-DX mobile robot the the custom structure and the Kinect on top of it.

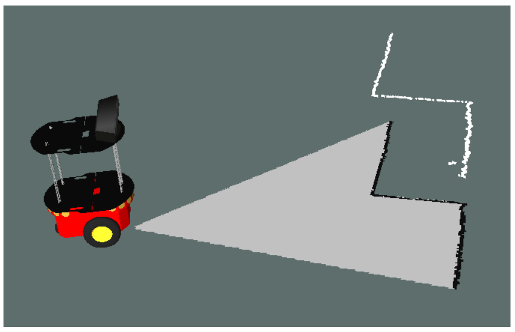

(b) Laser scan output from *rviz*: robot model, map and laser scan in white.

Figure 4.3: Photo of the hardware and the output in *rviz* of the mapping module.

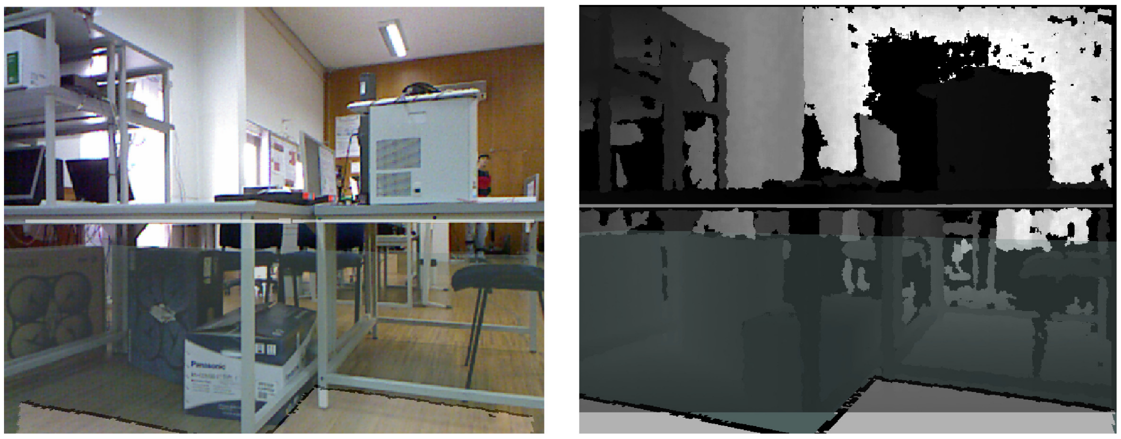


Figure 4.4: RGB image on the left and depth image on the right with overlayed laser scan and map.

Although the Microsoft Kinect allows to perform RGB-D mapping, our goal is to run a SLAM algorithm along with other tasks, such as people detection and autonomous exploration. In preliminary experiments, a RGB-D mapping task along with the operational system and robot's driver resulted in an average CPU load of 81.1%[1]. See Appendix B for details on these experiments.

Considering performance constraints, the option was to project the depth measurements provided by the sensor in the floor plane and simulate a 2D laser scan. This is represented

[1]The system used was a laptop equipped with an Intel i7-4700MQ and 16Gb of RAM, running Ubuntu 12.04 and ROS Hydro.

by the *Depth to LaserScan* block in Figure 4.1 and by the *depthimage_to_laserscan* ROS node in Figure 4.2. It processes the columns of the image matrix and creates a vector with the minimum depth value per column, thus originating a vector of 640 distance readings, *i.e.* a 2D scan.

The 2D range measurements are then used as an input to the *GMapping* SLAM algorithm [30], available in ROS, along with odometry information provided by the robot's driver.

This SLAM algorithm was selected for several reasons. Firstly, considering the performance constraints, it does not present a high computational burden. Secondly, the Microsoft Kinect has a low Field of View (FoV), which can cause problems in scan matching, therefore the mobile robot's odometry can greatly improve results. Finally, it was shown to be robust in testing and experiments, when compared to other SLAM approaches [68].

Despite the result being a 2D map, using the Kinect yields an advantage the map contains 3D information, since the projection from the depth image to a simulated laser scan uses the depth information available from the sensor. In Figures 4.4 and 4.3b it is visible that even though the Kinect is also sensing the empty space under the table top, it uses the table top information of depth to create the map. In contrast, a LRF would only be able to do so, if it was continuously tilting in the same plane as the table top.

4.1.3 Navigation and Exploration

As described on Appendix C.1.2, the Kinect allows us to perform navigation, although that navigation might not be as safe as desired, due to the constraints imposed by the hardware characteristics reported in Section 3.2, *i.e.* narrow range (4.6 meter maximum) and FoV (58°).

In order to test the navigation system, we firstly used an *a priori* known map to assess the reliability of the navigation system based solely on the Kinect data under different environmental constraints. In these experiments, we used the approach described in Appendix C.1.2 [4]. The videos of the experiments are available in:

- Kinect-based Navigation with known map on ISR[2].
- Kinect-based Navigation with known map on a small arena[3].

As seen on the videos, navigation with a known map using the Kinect is fairly reliable and the algorithm is able to safely navigate the robot from one point the another. However,

[2]`http://home.isr.uc.pt/~fsales/nav_amcl_isr.mp4`
[3]`http://home.isr.uc.pt/~fsales/nav_amcl_arena.mp4`

this method requires a known map and a reasonably static environment because dynamic changes close to the robot might fall on the blind spot of the Kinect, which may become unnoticed by the robot and, consequently, originate a collision.

Moreover, in the context of SaR missions, a representation of the environment is not usually available *a priori*. Therefore, it may be necessary for the robotic system to explore an environment without colliding or putting itself at risk. To achieve that, it is necessary to conduct conservative path planning taking into consideration the fact that the environment is unknown and the need to acquire knowledge about it. Path planning is then translated into goals, which are sequentially sent to the navigation system described in Appendix C.1.

A classical way to achieve that is to consider a frontier-based approach, as described in [80]. Based on the detection of frontiers, which are regions on the border between known open space and unexplored space, the system is allowed to autonomously explore the unknown area. Using this information other methods were developed, *e.g.* [**?**].

An implementation of this frontier-based exploration approach [6] was tested with the Kinect. Five runs were made and four out of the five experiments ended in a collision situation and had to be shut down, the most successful one is available on video[4].

The above result shows that even though there are reliable approaches to exploration, the constrains imposed by the Kinect's low range and FoV do not provide enough reliability for autonomous navigation in unknown spaces. A possible workaround for this problem is to develop a semi-autonomous approach that navigates in a more aggressive way and its trajectory can be corrected with inputs from the user.

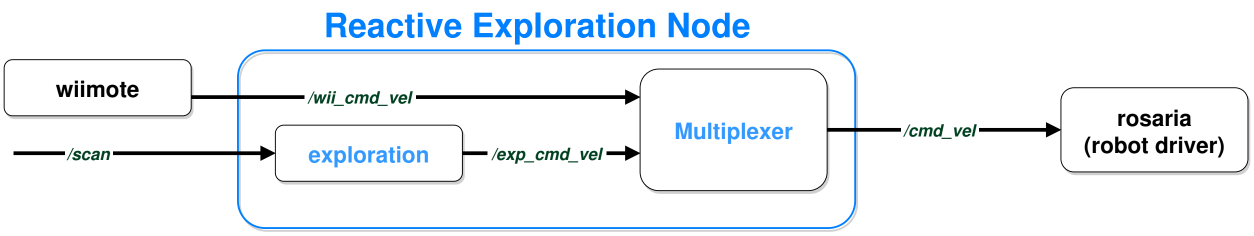


Figure 4.5: Block diagram of exploration node.

Therefore, a ROS node was programmed to use the simulated laser scan and inputs from the Wii remote controller in order to explore by moving the robot and avoiding collisions, also enabling the user to override the exploration and to teleoperate the robot to assure safety and trajectory correction. The information in the current map is not used because in an exploration task, where the map is not known *a priori*, it does not comprise much

[4]`http://home.isr.uc.pt/~fsales/kinect_exploration.mp4`

information due to the narrow FoV of the Kinect. This limitation, and the dimensions of the robot (about 40cm wide) justify the option of the simulated laser scan being considered as a whole for collision avoidance purposes, and split in two so as to determine which way the robot should turn when an obstacle is perceived. This node is represented by *explore* in Figure 4.2 and by the *Reactive Teleoperated Exploration* block of Figure 4.1. The node analyzes each received simulated laser scan, computes metrics and tries to identify dangerous situation to initiate a rotational behavior until safe conditions for moving forward are found. The algorithm is described in pseudo-code as follows:

Algorithm 1: Reactive Exploration

```
Input: sensor_msgs::LaserScan scan, geometry_msgs::Twist wii_cmd
Output: geometry_msgs::Twist cmd_vel
Data: min_dist, avg_left, avg_right, unknown_elem, unknown_r,
      unknown_l, unkn_rate, unkn_rate_right, unkn_rate_left
1  foreach element(i) of scan do
2      if element(i) $<$ min_range or element(i) $>$ max_range $- 1$ then
3          - Set reading invalid and sum unknown counters;
4      else
5          range($i$) = element($i$);
6      endif
7      - Update minimum valid range reading min_dist, and sum of right and left
       sided readings sum_right, sum_left;
8  endfch
9  - Compute unknown rates and range averages;
10 if wii_cmd != 0 then
11     cmd_vel = wii_cmd;
12 else
13     if min_dist $< 0.6$ or unknown_r $>= 0.15$ then
14         - Start rotational behavior based on unknown rates and range averages
           (danger detected). Rotate to the side which has higher range average, if
           averages are below 1 meter, rotate to the side which has higher unknown
           rate (possibility of open space);
15     else
16         cmd_vel = "Move forward";
17     endif
18 endif
19 publish(cmd_vel);
```

Although this reactive exploration approach is considerably safe, its safety can be increased by introducing a human in the loop. Therefore, the Wii remote controller is used to override the exploration movement and assume control of the robot to avoid dangerous situations, or having the robot stuck in local optima.

4.1.4 Visual-based People Detection

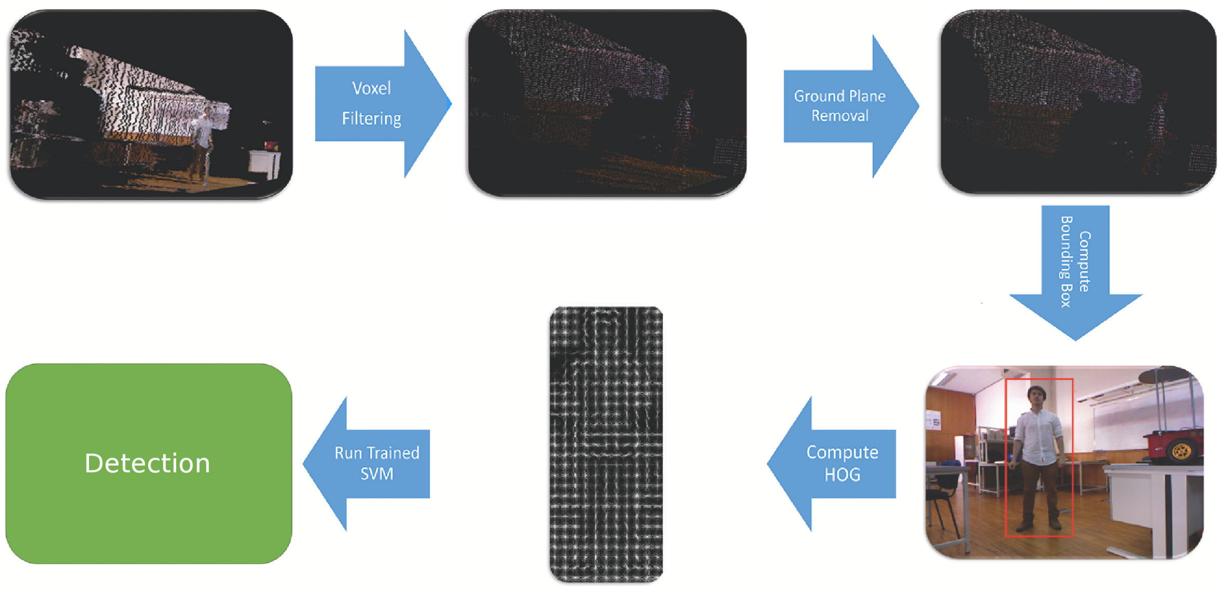


Figure 4.6: Scheme of the visual-based detection system.

Many people detection algorithms do not take into consideration 3D information, while others use that information to improve results. However, in [49] an algorithm that uses the generated point cloud to lower the computational load of classical people classifiers was proposed. Furthermore, ROS provides access to the Point Cloud Library (PCL) [67] which contains algorithms to process point clouds, which can be acquired with RGB-D sensors, such as the Kinect. Therefore, the technical implementation of this algorithm becomes much simplified. The algorithm is summarized in Figure 4.6.

The algorithm firstly processes the point cloud, dividing the space into volumetric pixels (voxels) with an edge of 0.06 m, which size is related to the density of the point cloud, and reduces the 3D points contained on each space into a common point according to the voxel's centroid. With this process, a reduced number of points is obtained and also approximately a constant point density in the point cloud, avoiding its variation with the distance from the sensor, simplifying the process of clustering.

With a filtered point cloud, and considering the assumption that people stand on the ground, the ground plane's coefficients are estimated and updated at every frame using a least squares method, becoming robust to small changes such as those experienced when a mobile robot is moving. At this stage, the points located in the ground plane are removed, by discarding all the points that lie on the estimated ground plane, and also all points having an Euclidean distance to that plane lower than a threshold of 0.06 m (the same as the voxel size). The distance threshold is based on the voxel filtering conducted previously. As a consequence, the remaining clusters become no longer connected by this common plane.

After this first stage of point cloud processing, the different clusters can now be computed by labelling neighboring 3D points on the basis of their Euclidean distances, *i.e.* all the points closer than a considered threshold of twice the voxel size belong to the same cluster. However, this process may lead to errors, *e.g.* dividing partially occluded people into different clusters, or merging different people in the same cluster when they are near each other. For the first issue, the algorithm discards clusters that are distant from the removed ground plane and merges all the clusters that are near in terms of ground plane coordinates, *i.e.* vertically aligned. Regarding the second issue, to detect and separate the merged people in a single cluster, the algorithm uses the position of the heads, that generally are not so close or occluded, to divide these clusters into sub clusters. It defines the head positions by creating a height map, that represents the most distant point to the ground plane per area bin of the cluster, see Figure 4.7 for an example. A search for local maxima is made in the height map and if the maxima have a distance of 0.3 m from each other, the clusters are split.

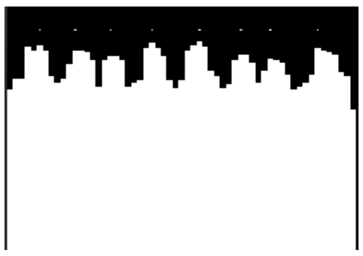


Figure 4.7: Example of a height map computed for 8 people standing side-by-side.

For the clusters obtained earlier, a HOG-based detector [17] is applied to the portion of the RGB image corresponding to the fixed aspect ratio bounding box that contains the whole cluster. This process includes the computation of HOG features and their application to a trained linear SVM[5]. The SVM is a learning model that allows us to classify the data based on its training.

4.1.5 Audio-based People Detection

Generally, the visual-based people detection system described in Section 4.1.4 has proven to be reliable and able to accurately detect and estimate the position of people. However, it relies on the data captured by the Kinect's RGB and depth cameras, which means that the

[5]The SVM was trained using the well known and established *INRIA Person Dataset - (URL:* `http://pascal.inrialpes.fr/data/human`).

detection range is limited to the sensor's operational range.

In section 3.2, it is shown that this range is from 0.6 to 4.6 meters, however, for people detection it is necessary that the point cloud and color image both comprise the person, therefore, the ideal range becomes 1 to 3 meters. Furthermore, in catastrophic scenarios, there is often the need to deal with environmental factors, such as smoke or the interference of other IR sources, that cause the Kinect to retrieve corrupted readings. In addition, people might be trapped under debris or occluded so that they are unable to be seen by the Kinect's cameras or any other visual sensor. Moreover, in these scenarios, power breakages usually occur, which can cause the environment to be under low-light conditions, and causing the RGB camera's data to lack content.

In contrast, the MA is affected by the distance and power of emission of the sound source, the interference of noise, and the composition of the air. Consequently, it is generally more suitable for detection at closer ranges, compensating the Kinect's cameras inability to acquire data at such ranges. Moreover, it is less affected by the mentioned issues that might cause the Kinect depth and color cameras to produce inconsistent data, *i.e.* smoke, interference of IR sources, and occlusions. Considering these factors, the auditive information can play a fundamental role in suppressing issues that visual-based detection approaches commonly suffer from.

HARK

As seen on Chapter 2, HARK is a powerful library for robot audition and audio processing. It provides a network construction concept to create modules in order to interface different methods and algorithms.

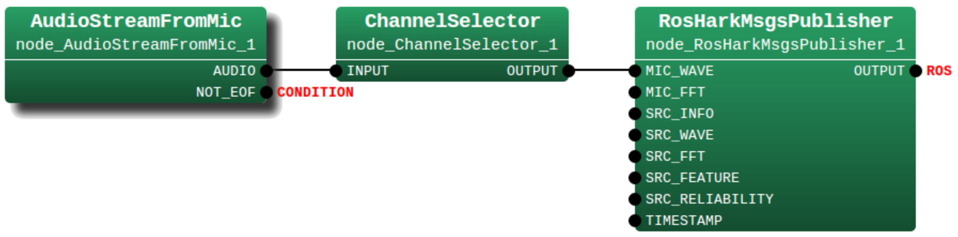


Figure 4.8: HARK network for audio capture from microphone arrays and to publish in an appropriate ROS topic.

In the proposed system, the raw audio of the four microphone channels is captured from the Kinect using HARK (Figure 4.8) and the GEVD-MUSIC algorithm[6], which is used

[6]See Appendix A.1 for a description of the method.

to localize the different sound sources. Despite the low accuracy of this estimation, the direction of arrival of the sound is fairly accurate as seen later on through experimentation in Chatper 5. The direction of arrival is the most important estimation to use in sound separation. In Figure 4.9, the network for the implementation of this method is visible, the input to the node *ChannelSelector* comes from Figure 4.8, *i.e.* the audio captured from the MA.

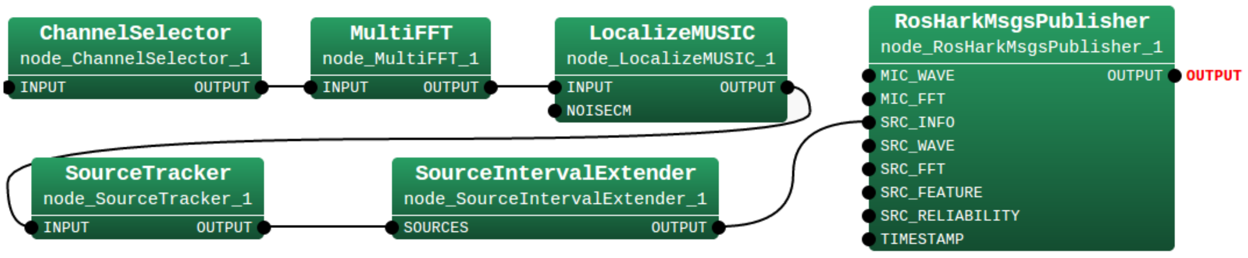


Figure 4.9: HARK network for the use of MUSIC algorithms and to publish the estimation on a ROS topic.

With the estimation of localization, the angle is processed by the GHDSS algorithm (see appendix A.2) along with the audio, so as to separate the sound from each specific source, thus resulting in a single channel audio signal per directional source. The sound separation algorithm used, the GHDSS, considers not only the estimation of the localization, but also the number of sound sources, the relative position of the microphones and the transfer function of the direct sound component. Once again, the HARK network for this method is shown in Figure 4.10, and the inputs derive from Figure 4.9.

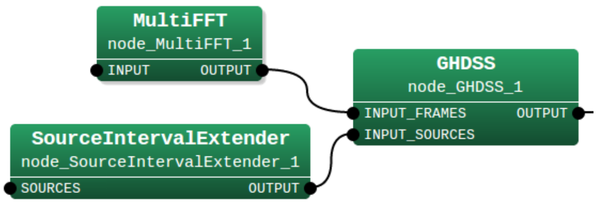


Figure 4.10: HARK network for the use of GHDSS algorithm [53].

The output audio is then processed to remove the stationary noise using a HRLE algorithm for speech enhancement to estimate the noise (further details on [51]). The leakage noise component from the channels is also removed by estimating the optimum gain and then applying a Spectral Gain Filter. Thus, obtaining an estimation of the localization of the sound source and the isolated audio with reduced noise.

In Figure 4.11, the noise is estimated in the HRLE and "EstimateLeak" blocks, and their spectrum is summed in "CalcSpecAddPower". The "CalcSpecSubGain" node computes the

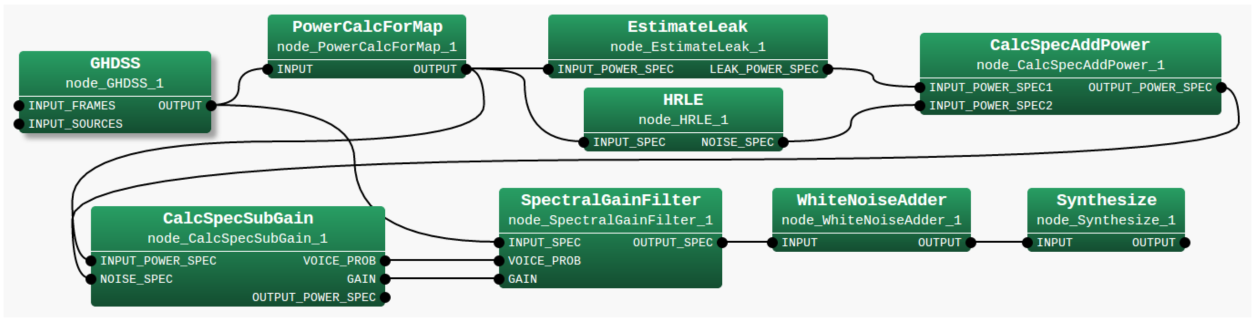


Figure 4.11: HARK network for the noise reduction and speech enhancement.

optimum gain and outputs 1 at the probability of speech presence. Then, the "SpectralGain-Filter" block uses that information and filters the separated sound. Finally, white noise is added to reduce the influence of non-linear distortions and the "Synthesize" block converts the signal from frequency domain into waveforms on time domain. Further details on these methods are available in Appendix A.3.

CMU PocketSphinx

Finally, the audio processed in HARK is further analyzed and processed with Pocket-Sphinx for speech detection. For this purpose, the mean amplitude of the audio signal is computed and a search for intervals of silence and speech is conducted.

From the PocketSphinx analysis, a single word or stream of words is obtained along with a confidence score. Combining the fact that speech is detected with the previously estimated localization, a likely detection is found. In order to validate each detection and avoid false positives originating from noise, a threshold of speech recognition confidence score is set to 60% and a set of key words is defined: **help, rescue, assist, here**. Even though 60% is a low confidence score, the recognition system is used to distinguish between people and background noise, so even if the recognition results in a wrong word, it still recognizes a person and the module serves its purpose despite the error, *i.e.* the 60% rate leads to robustness against noise being falsely recognized as speech.

For further information about this software, the reader should refer to [32] and [34].

4.1.6 Integration

Although all modules run in the same system, the data collected from the detection modules is not represented in the same reference frame as the mapping module. Detections are made on the Kinect sensor frame, which is different from the reference frame of the map. To deal with this issue, an additional ROS node was created (*map_people_manager* in Figure 4.2 of page 25) that subscribes to the detections, transforms their coordinates to map coordinates, using the ROS *tf* library, and manages the detections, avoiding multiple detections of the same person, from the same input module, and in the same position.

Additionally, it publishes the corresponding markers to allow the visualization of the map and the detections on the relative position in the map. If there are two or more people in the same approximate position of 45 cm, the system displays a single marker and prints the number of people that it estimates for that position. Note that, since auditive information is separated according to its source localization, audio-based detections are handled individually and, consequently, this feature is not replicable to these detections.

4.2 Summary

In this Chapter, the proposed system was presented. It started by providing a visual overview of the system and proceeded to detailing each module individually. Section 4.1.1 introduced the used hardware and the drivers used for each one. It was followed by Section 4.1.2, which presented the proposed system's approach for mapping. The subject of navigation and exploration was addressed in Section 4.1.3. Then, in Section 4.1.4, the visual based people detection system was described and in Section 4.1.5 was presented the audio based people detection system. Finally, Section 4.1.6 provideed insight on the technique used to integrate the information provided by all the modules.

In Chapter 5, all the practical experiments, tests, evaluations, and their analysis are presented. The Chapter also reports the tests in real world scenarios.

Chapter 5

Results and Discussion

The system presented in Chapter 4 was modularly implemented. Therefore, after the implementation of each module, a thorough analysis was conducted to assess the accuracy and results provided by the module. Consequently, the current Chapter presents the experimentation setup and then the results obtained for each module and a corresponding analysis. In this Chapter, the real world tests are also reported with results and discussion.

5.1 Experimentation

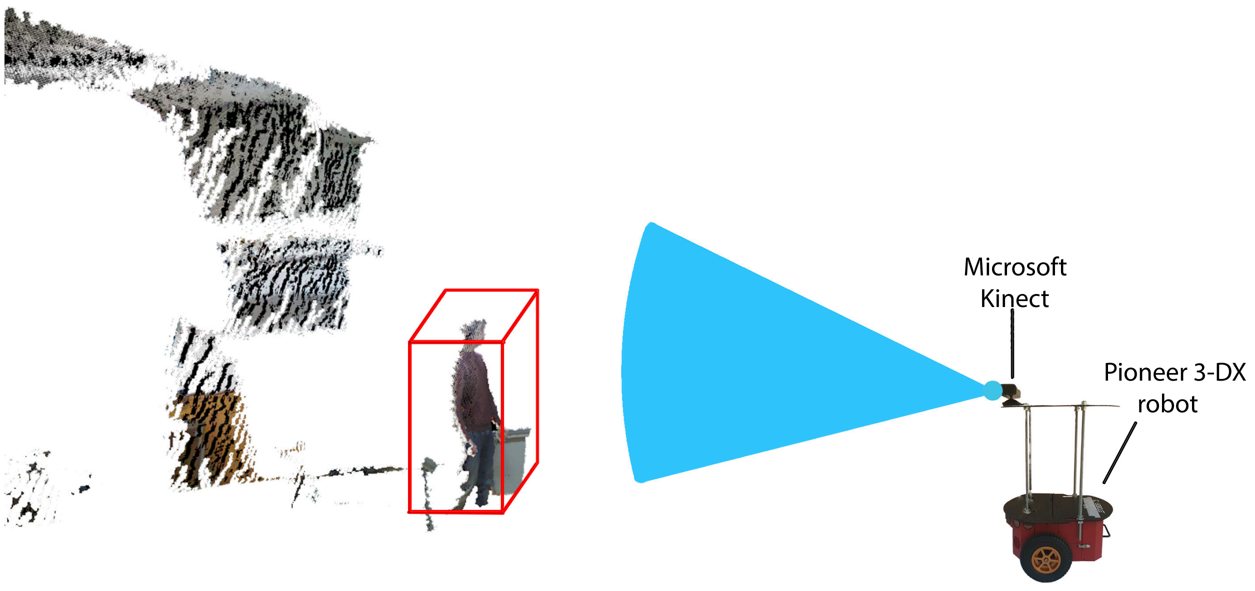


Figure 5.1: System setup.

In all experiments carried out in this chapter, a laptop equipped with an Intel Core i7-4700MQ and 16GB of RAM running Ubuntu 12.04 and ROS Hydro was used. To determine if the integrated system is viable, the CPU load was analyzed at all stages. Note that, for each CPU load analysis, the values were acquired through experimentation, which means that they also include the load of the system processes, the drivers of the Kinect and the robot.

This analysis is also important to leave room for further development on the system and for the use of additional sensors that might allow a more reliable navigation and exploration.

Figure 5.1 shows the typical setup of the practical experiments, with an exception made for the laptop on top of the platform, which is not visible.

5.1.1 Mapping Validation

In order to validate the mapping task with the Kinect sensor, an Hokuyo URG-04LX-UG01 LRF was used to produce maps and compare with the ones obtained using the Kinect and the method described. The environment tested was a lab arena with approximately 4.6×4.0m, as illustrated in Figure 5.2. The robot was teleoperated using an *ssh* remote connection, while running the *GMapping* 2D SLAM algorithm with the simulated laser scans computed from the depth images acquired with the Kinect and the odometry from the robot's driver.

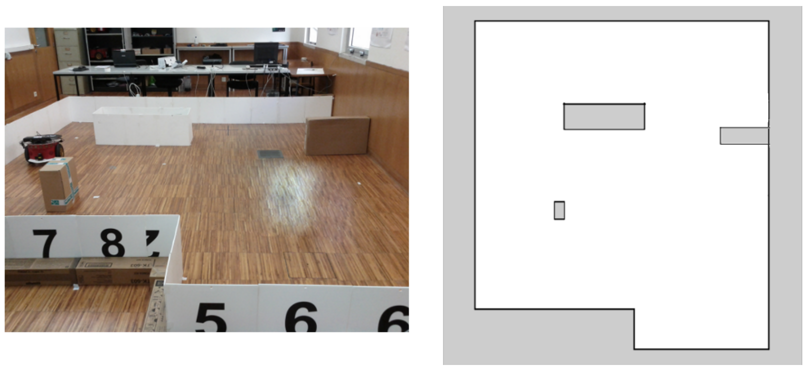


Figure 5.2: Photo (left) and ground truth map of the arena (right).

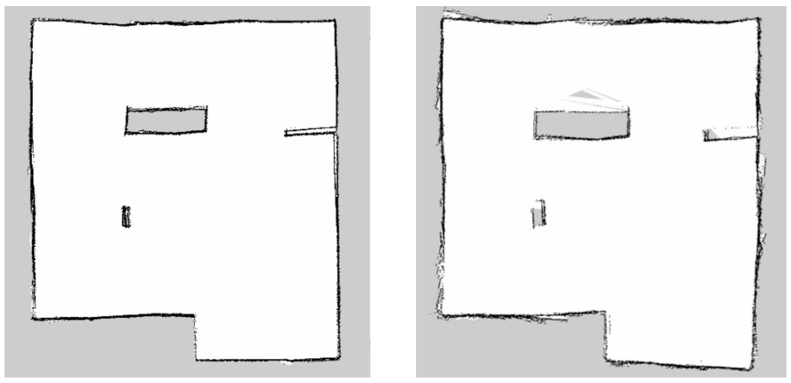


Figure 5.3: Maps produced with the Hokuyo URG-04LX-UG01 LRF (left) and the Kinect (right).

By visually comparing both maps, in Figure 5.3, it becomes evident that the map provided by the Kinect is not as accurate as the one with the LRF, and its limited FoV, range, and the lower accuracy, have a negative impact on the results obtained. Nevertheless, both maps are easily interpretable by the human eye. To assess their quality, the absolute pixel-wise matching of all pixels in the generated maps was computed, and acceptable matching rates were obtained, as shown in Table 5.1. In order to compute the matching metric, the maps obtained and the ground truth were binarized, the best fit alignment by rotation of the maps was calculated, and the pixel-wise match of each pixel in the image was computed.

Table 5.1: Pixel-wise matching rates.

Maps	Matching Rate
Ground truth - Laser	96.9 %
Ground truth - Kinect	94.3 %

From figure 5.4, it is observable that the mapping module is not exactly lightweight, using in average 25.28% of the CPU's load, running along the drivers and the operational system. However, it is reasonable, especially when compared with 3D mapping (see Appendix B), and leaves room to run the people detection modules alongside it.

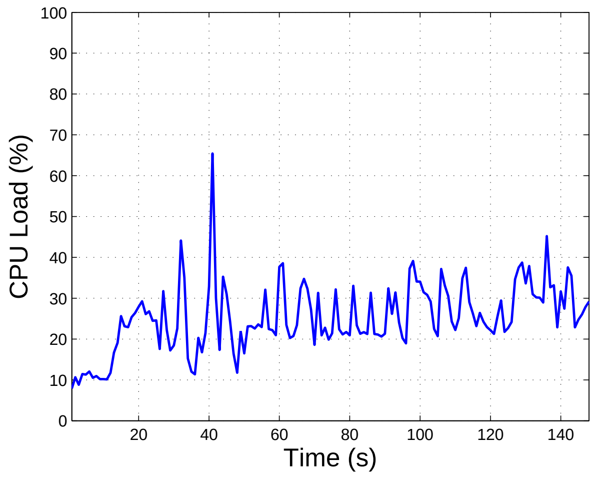
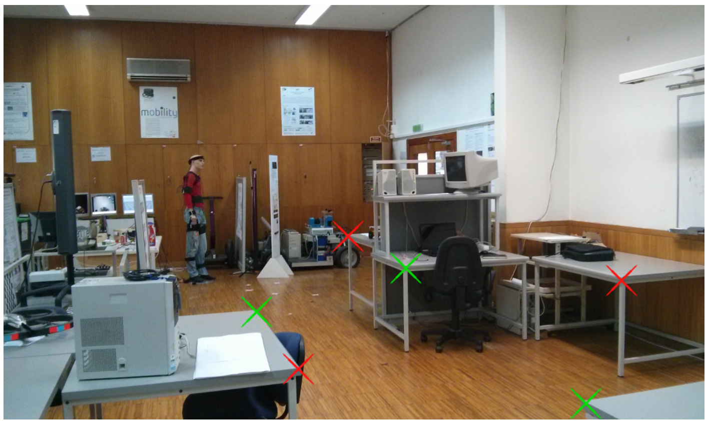


Figure 5.4: Computational load of the mapping module.

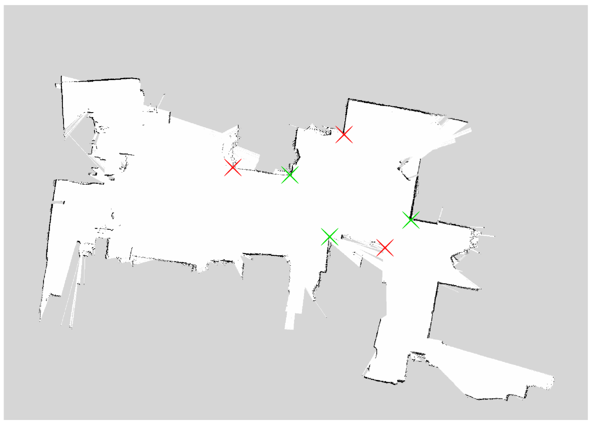
5.1.2 Reactive Exploration

The approach developed for exploration is simple and allows the user to override the robot exploration with the Wii remote controller. Therefore, even if the collision avoidance procedures used in the node fail, the user will be able to take control of the robot and command it back to safety. The reason behind this unsafe behavior is that the Kinect provides limited information about its surroundings, due to its limited FoV and range.

To evaluate the proposed approach, experimental tests with the exploration and mapping modules were conducted in the AP4ISR lab. The accuracy of the obtained map is used to



(a) Photo of the AP4ISR lab.



(b) Map obtained with the proposed exploration approach and the adopted mapping method.

Figure 5.5: Test scenario of the exploration approach.

validate the usability of the exploration approach. A video[1] of a test has been made available. The video shows the whole experiment and the output of the node, which displays a message every time it switches from the autonomous exploration mode to remote controlled mode, and vice-versa. In this particular run that lasted for eight minutes, the user intervened six times, however, two of them might not have been totally necessary.

It is visible that despite the safe behavior, there are still some problems with collision avoidance that originate from the limitations of the Kinect. However, it still represents a good addition to a basic teleoperation system.

5.1.3 People Detection from Visual Cues Validation

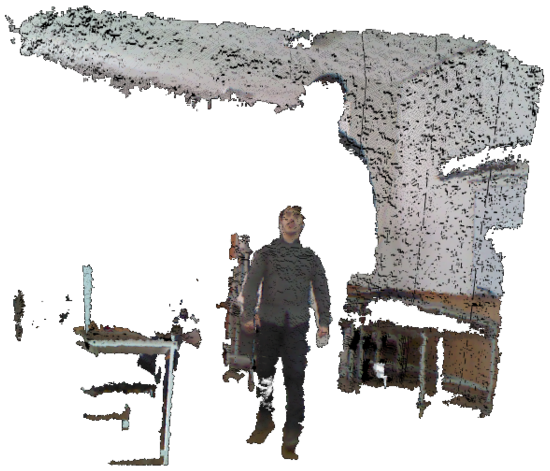


Figure 5.6: Example of a point cloud from the dataset.

In order to validate the people detection method from visual cues, a dataset of about 100

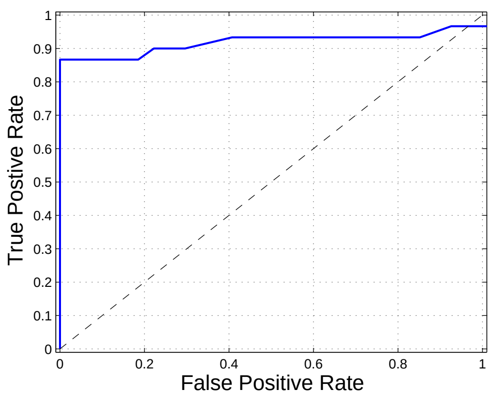
[1]`http://home.isr.uc.pt/~fsales/wii_explore.mp4`

frames that was manually annotated with the people present in each frame was captured. It contains one person walking in several directions at a distance of 1 to 4 meters to the camera frame (an exeinmplary frame is shown in Figure 5.6). Therefore, it is a dataset of binary decisions. This way, the data is in similar conditions to the most frequent situation that occurs on the final intended applications.

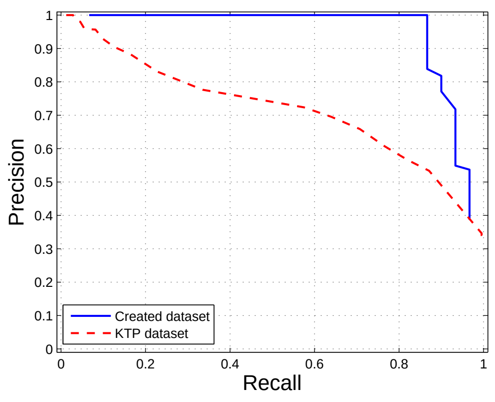
The people detection method was applied using ROS and the point clouds of the dataset were processed, extracting results (true positives, true negatives, false positive and false negatives) and compiling them in Receiver Operating Characteristic (ROC) curves (see Figure 5.7a and 5.7b). The ROC curve is a graphical plot which illustrates the performance of a binary classifier system as the discrimination threshold is varied.

Subsequently, a cross validation test with the Kinect Tracking Precision (KTP) Dataset from [48], which contains sequences of multiple people captured from a static camera, was performed by applying the method on its data.

Figure 5.7a relates the True Positive Rate (TPR) and False Positive Rate (FPR). Perfect results are placed near the top left corner with 0% FPR and 100% TPR. Figure 5.7b shows the precision and recall percentages for each experiment. The ideal result is situated on the top right corner with 100% PR. Figure 5.8 presents the Detection Error Trade-off (DET) curve which relates the False Rejection Rate (FRR) in percentage and the number of False Positives Per Frame (FPPF). The ideal result is located on the bottom left corner.



(a) ROC curve for the created dataset.



(b) PR for the created dataset and for the KTP dbestataset.

Figure 5.7: ROC curve for the created dataset and PR curves for both datasets used.

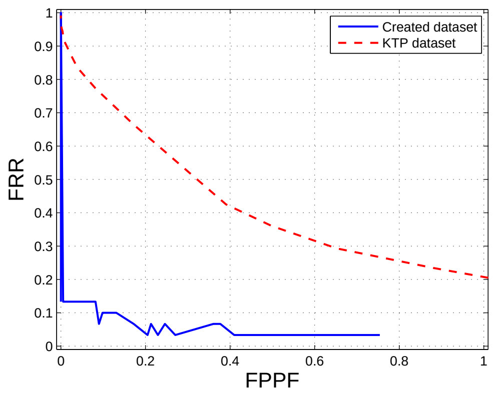


Figure 5.8: DET for the created dataset and for the KTP dataset.

From the results extracted from the created dataset, it can be seen that the method is very robust in terms of false negatives, showing a low FPR for high enough TPR, *e.g.* 86.87% TPR for $\approx 0.0\%$ FPR. This is also visible in the high precision shown in Figure 5.7b, even for high recall values, *e.g.* 100% precision for 86.67% recall. The method is able to achieve an accuracy of 92.98% which shows its reliability.

Note, however, that this analysis is performed independently for each processed point cloud. In real world applications, with the 30fps retrieved by the Kinect, several frames will be captured for each person, which provides a gain in certainty when detecting a person in short time intervals. In the case of SaR missions, it is of extreme importance to lower the false positives as much as possible to avoid wasting resources and time while keeping a high FPR to be capable of detecting all the victims.

The low number of frames and the presence of only one person is clear on the curves and motivated an analysis of the method running on the KTP Dataset. The results obtained on the KTP dataset were comprehensively not as good as the ones on the created dataset, since this is a more complex dataset containing up to five people in the same sequence. Still, the method is accurate enough considering the amount of frames that real scenarios will provide to detect each person. The ROC curve was not computed for this dataset because it aims to assess a binary classification problem and this dataset barely contains binary decisions due to the nature of the tracking problem. Still, it is visible from the PR and DET curves that the results appear to be accurate enough for the intended applications, *e.g.* 72.39% precision for 57.51% recall and 0.39 FPPF for 42.49% FRR.

In figure 5.9, it is shown that the visual detection module is still a bit heavy with an average of 29.25% CPU load. The peak observed at about $t = 75s$ corresponds to a detection made.

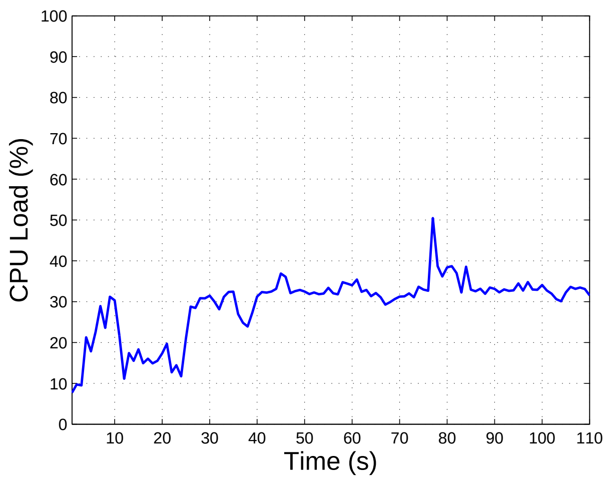


Figure 5.9: Computational load of the visual detection module.

5.1.4 People Detection from Audio Cues Validation

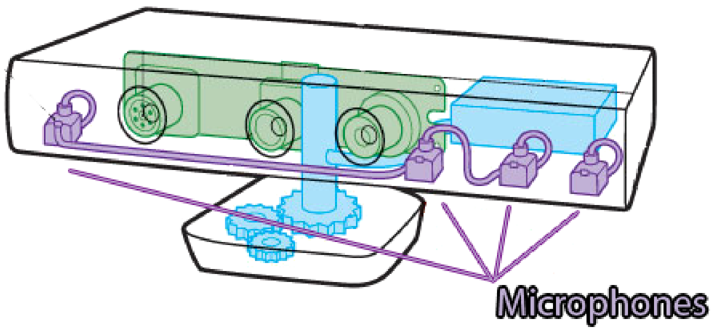


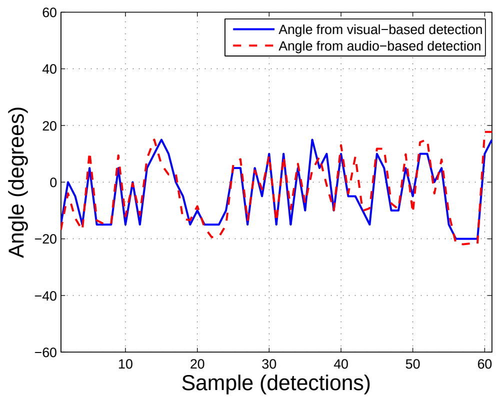
Figure 5.10: Scheme of the Kinect Microphone Array Placement.

Although some MAs can accurately estimate the sound source's localization in relation to itself, the Kinect's MA has a crucial drawback due to the positioning of the microphones. The microphones are placed as Figure 5.10 illustrates, *i.e.* three of them are very close together and they are all co-linear. Their relative position suggests that the estimated localization of the sound source will not be very accurate. However, the estimation of the angle that the input sound source makes with the MA is likely to be accurate enough.

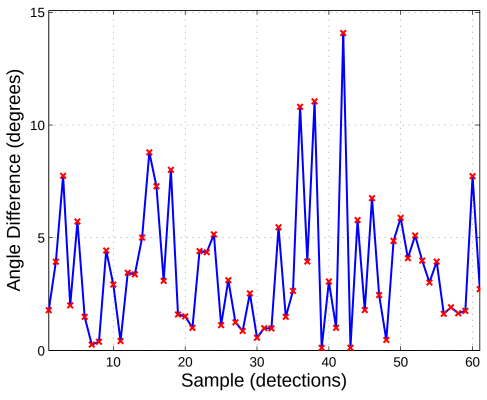
In order to validate the sound source localization of audio containing speech, *i.e.* the localization of people from voice recognized on audio, the results were compared with the ones obtained from visual cues. For each detection computed from the audio data, a corresponding detection based on visual data was found and the estimated localization was stored to be analyzed offline.

The experience was conducted in a laboratory on a regular day and with people working. Consequently, there is a considerable amount of background noise (computer fans, keyboard pressing, chairs movement, etc.).

Angle Estimation



(a) Estimated angle of audio-based detections in red and of visual-based detections in blue.



(b) Absolute value of the difference between the angles obtained via visual and audio localizations.

Figure 5.11: Absolute difference between the estimations of people localization from audio and visual sources.

From figure 5.11a it is visible that, despite the difference in the angle estimations, there is a clear relation between the data from both methods, which suggests that the estimation obtained from audio data is fairly reliable. The detections were all made in approximately a 40° range due to the narrow FoV of the Kinect IR camera, *i.e.* this limitation was not imposed by the MA.

In figure 5.11b, it is observable that the maximum difference of the angle estimations was approximately 14°. However, most of these experiments were conducted on close distances, for which this angle difference does not represent such a significant error in the estimation of the person's position as it would for farther away detections. Although it is still a relevant difference, the average difference is 3.59°. Furthermore, the visual-based detection system also produces an estimation of the position, and consequently the angle is subject to error in the process. Moreover, based on the MA's configuration visible on Figure 5.10 on page 42, it is expectable that the amplitude of this error depends on the direction of arrival of the sound. To confirm it, the estimations from both methods were further analyzed and in Figure 5.12 it is confirmed that there is a relationship between the disparity of both methods and the sound's direction of arrival. This happens because three microphones close together are more sensible to small changes in the angle of the sound because the correlation of the signal between them changes more significantly as opposed to a single microphone. Nevertheless, it is possible to conclude that the estimation of the angle is fairly reliable and usable in the system.

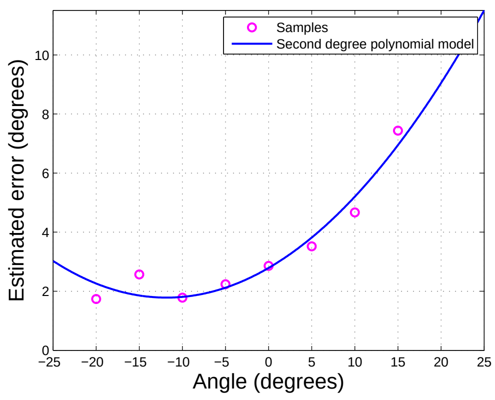


Figure 5.12: Absolute value of the angle difference.

In Figure 5.12 the samples for each angle of arrival are shown with an associated estimated error, based on the visual detection method. The negative angles correspond to the side of the Kinect which comprises the three microphones. The data is well fitted by a second degree polynomial function as follows:

$$f(x) = (7.164 \times 10^{-3})x^2 + (169.7 \times 10^{-3})x + 2.789. \tag{5.1}$$

Distance and Position Estimations

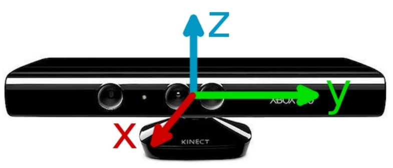
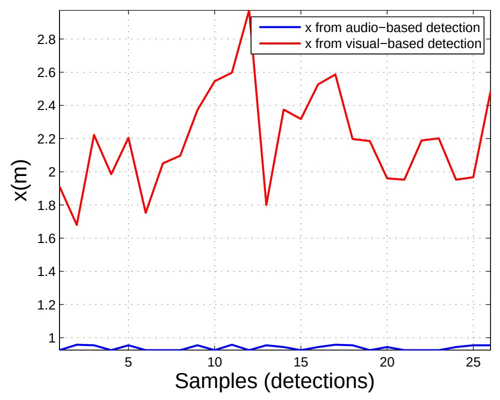


Figure 5.13: Reference frame for the audio-based localization analysis.

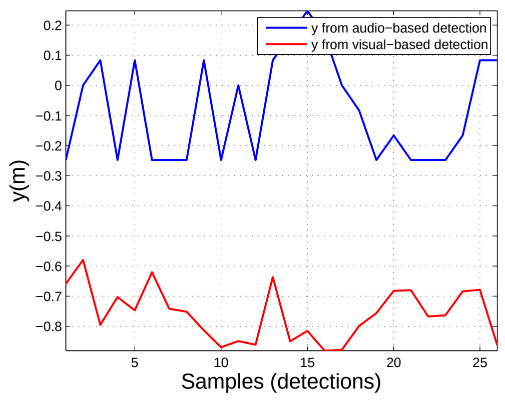
The estimation of elevation, *i.e.* the z coordinate, is not considered for two reasons: it is not as important as the others since the distance from the projection of the voice's point of origin to the projection of the body's mass center is not significant, and the error in the estimation is massive due to the strictly horizontal microphone configuration of the Kinect's MA.

From figure 5.14a it is visible that the estimation of the x coordinate, which points from the center of the Kinect to the front, is not achievable with this hardware. Therefore, all estimations of the localization are approximated to a $0.95m$ distance to the sensor, varying with the angle, see Figure 5.15a and Figure 5.15b, *i.e.* the angle of the sound source is estimated and a fixed distance is used to compute the x and y coordinates.

Although the x estimation is not usable, the y coordinate estimation, which points from the center of the Kinect to the side, appears to reveal some coherence between the data from

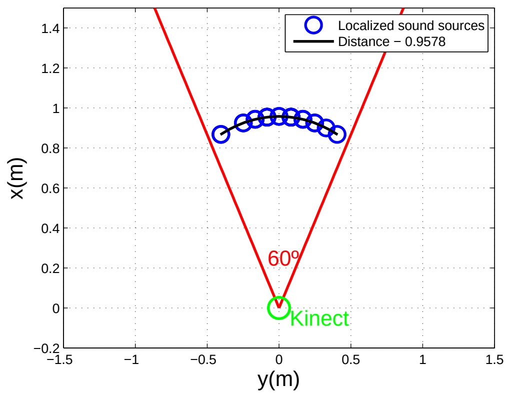


(a) Estimation of the x coordinate from with methods.

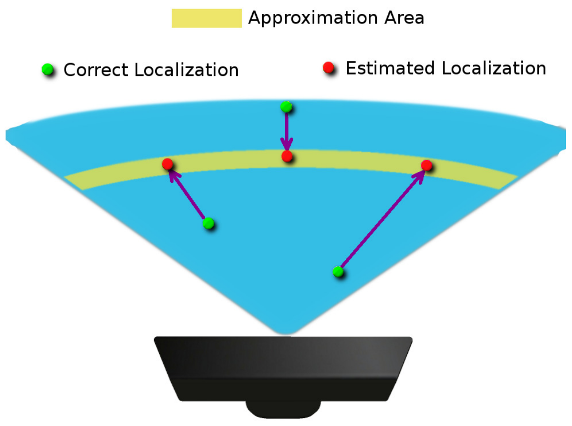


(b) Estimation of the y coordinate from with methods.

Figure 5.14: Absolute difference of people localization from audio and visual sources estimations.



(a) Results from experimentation.



(b) Scheme of sound source localization with the Kinect.

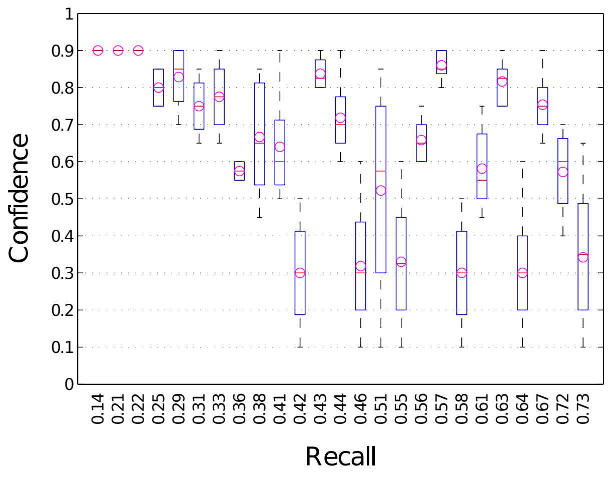
Figure 5.15: Approximation of the x coordinate.

both detection methods is apparent. However, it is related to the correct angle estimation and the approximation made to the distance of the sound source to the sensor which causes the y to be dependent on the angle and the approximation made, *i.e.* the y coordinate estimation is unusable.

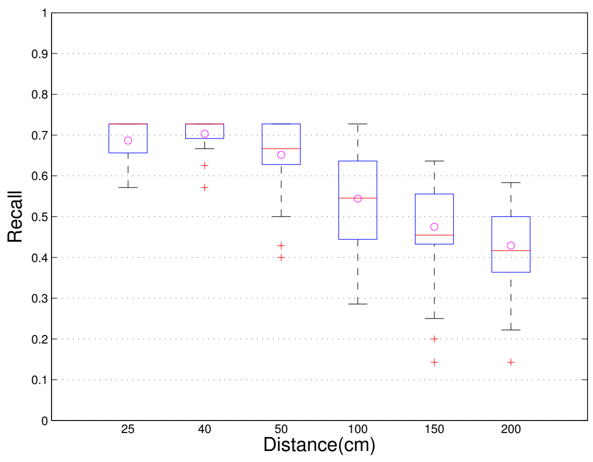
Evaluation

Even though the accuracy of the estimation by sound sources is evidently not as reliable as the one obtained with visual cues, it is useful for situations where no visual data is available or it is corrupted due to environmental factors.

However, the MA system has its limitations and to understand how its information can be introduced in the system an experiment was performed to assess the behavior in response to different sounds and positions in relation to the sensor. To conduct the evaluation, voices were captured from 0.25 to 2 meters at approximately 0° angle with the normal of the sensor. The voice of five different male subjects were used and they were asked to speak a set of 20 known words at approximately a constant pace and power.



(a) Recall-Confidence Box Plot.



(b) Distance-Recall Box Plot.

Figure 5.16: Evaluation of people detection from audio sources.

Considering that PocketSphinx is based on a HMM, it is possible to obtain a posterior confidence measure (for more details read [78]). Despite the low reliability of these confidence values, they can provide insight on how recall rates relate with confidence. In Figure 5.16a this relationship is visible and it is also apparent that confidence tends to degrade for higher recall values, as it would be expectable. Note that the confidence metric is not totally reliable, recall values consider the whole set of words, and results heavily vary from speaker to speaker, which along with the limited amount of data, justifies the high disparity observed in the box plot of Figure 5.16a.

Also, the recognition rates should degrade proportionally with the distance from the sensor, considering that SNR increases with distance due to the sound waves' attenuation on air. Figure 5.16b represents the recall variation with relation to distance, and it is visible that recall is higher near the sensor, as expected. This chart also suggests that for high power signals near the MA, the recognition rates degrade, which is caused by the saturation of the sensor.

In figure 5.17, the computational load is presented, and it is noticeable that the audio module is lighter when compared with the previously analyzed modules. The first peak corresponds to the launching of the system which includes loading the acoustic models and dictionaries, and the following peaks correspond to detections made. Even including the start up peak, the average CPU load is 13.65% .

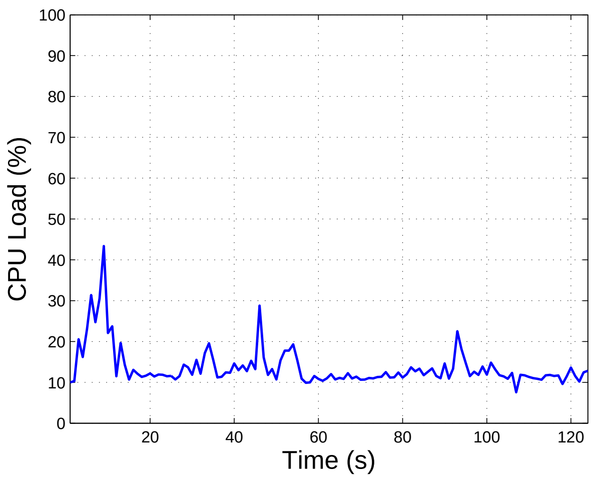


Figure 5.17: Computational load of the audio detections module.

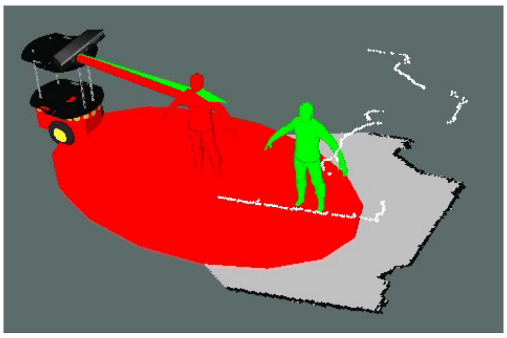
5.1.5 Module Integration

After testing and individually assessing all the composing modules of the system, the next step is to integrate and analyze the behavior of an integrated system that comprises the aforementioned modules. In this context, the same setup of Figure 5.1 is used, and the objective is to assess how the performance degrades with the parallel run of all modules, and the computational load required to perform every task simultaneously.

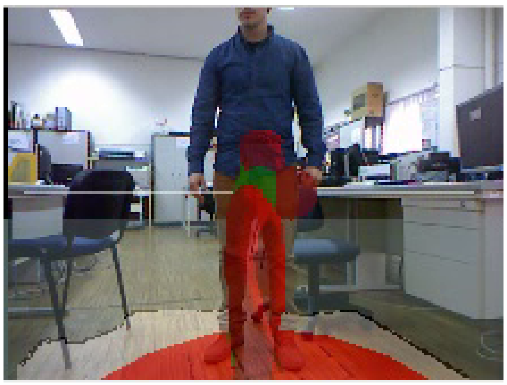
Besides the modules previously described, the system also includes a visualization module based on rviz[2], a module that subscribes to the detection messages, transforms them into the coordinates of the map, stores the data and publishes the visualization markers, and also a module that allows teleoperation with the Wii remote controller and performs reactive exploration using the information gathered by the RGB-D sensor when the user does not input any command. The marker representation is as follows:

- Red arrow - audio-based detection orientation;
- Red marker - estimated localization of audio-based detection (heavily affected by error);
- Red ellipse - region estimated by analysis of error (fixed major axis and minor axis varies with the angle);
- Green arrow - visual-based detection orientation;
- Green marker - estimated localization of visual-based detection.

[2]The rviz is a ROS tool that allows the visualization of the data published on ROS topics.



(a) Output in *rviz*.



(b) Image from RGB camera with the computed information overlayered.

Figure 5.18: System output in *rviz* (figure 5.18a) and a frame captured by the Kinect's RGB camera (figure 5.18b). The ongoing map being built can be seen, as well as the 2D projected laser scan.

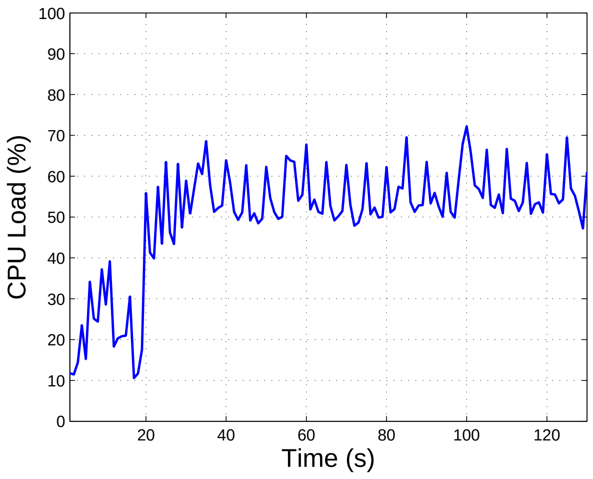


Figure 5.19: Computational load of the complete system.

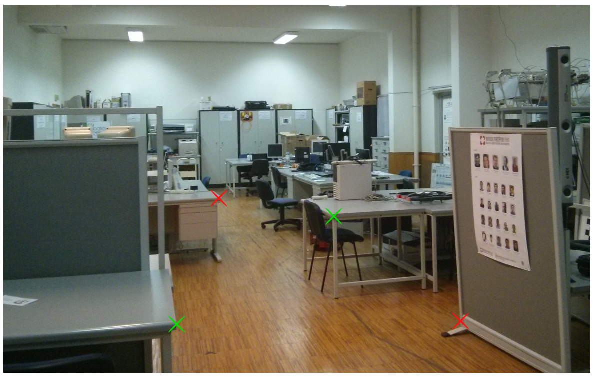
In Figures 5.18a and 5.18b, the results obtained by running all the modules simultaneously can be observable and appear consistent with the individual experiments. The estimated people detections are consistent with the differences previously computed, and both modules were able to detect people in parallel. The keyword used by the subject was "HELP". This experiment shows that the results do not degrade with the integration of the modules, and it is viable to integrate then in a single system.

Figure 5.19 confirms that the computation load increases considerably and its average value is now 50.43%. However, this number includes the system bring up. If this phase is disregarded in the math, the CPU average load rises to 55.66%. Nevertheless, the testing system is able to support all the computation in parallel and still leaves room for further developments.

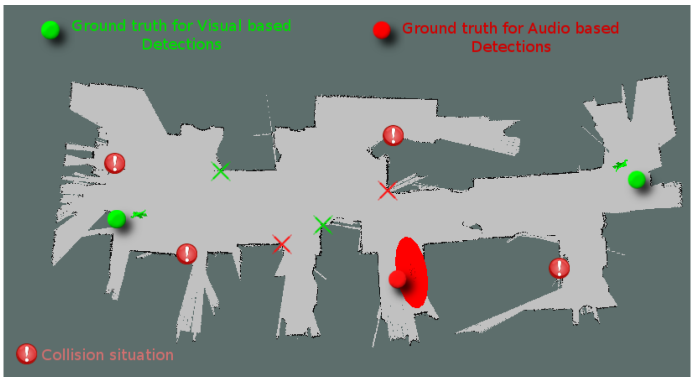
5.2 Results

5.2.1 Scenario 1 - Artificially illuminated indoor laboratory

The first selected scenario to test the developed integrated system was the AP4ISR laboratory with artificial illumination (see Figure 5.20a). The scenarios contained three male subjects, two of them could easily be detected visually, and the other one was occluded but able to speak. All subjects were not static, though they were asked to populate roughly the same area. The environmental noise was moderate, with only the servers, the air conditioner, and the robot itself producing noise. The idea was to use the audio to detect the occluded subject (see Figure 5.21).



(a) Photo of the scenario.


Ground truth for Visual based Detections
Ground truth for Audio based Detections
Collision situation

(b) Resulting output from *rviz* with ground truth.

Figure 5.20: Scenario 1 - Photograph of the scene (left) and resulting output from *rviz* with manually inserted ground truth localization of people (right).

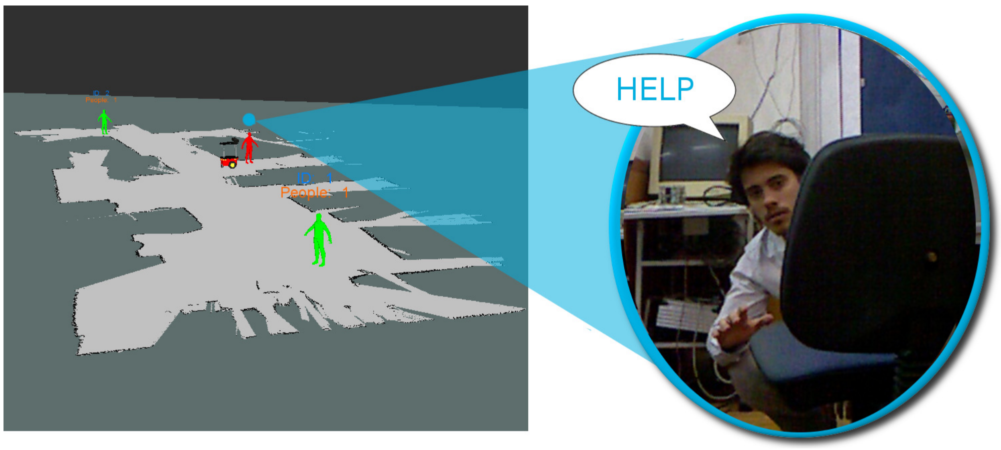


Figure 5.21: Scenario 1 - Occlusion situation handled using auditive information and corresponding image from the RGB camera.

As visible on Figure 5.20b, the scenario was fairly accurately mapped and all three people were detected despite the minor errors in the estimation of their position. The occlusion was suppressed using the auditive information to recognize the test subject's speech. The robot's trajectory was corrected four times to avoid collision. Two of them due to low height obstacles, while the remaining two due to a chair arm and a placard.

5.2.2 Scenario 2 - Indoor laboratory without illumination

The second scenario is the same physical space as the previous one in Section 5.2.1. However, in these experiments all lights were turned off and there was also no natural illumination. The same three people were placed inside the environment and the experiment setup was the same as the previous one. In this test, there was an intention to evaluate the effect of the lack of illumination on the system.

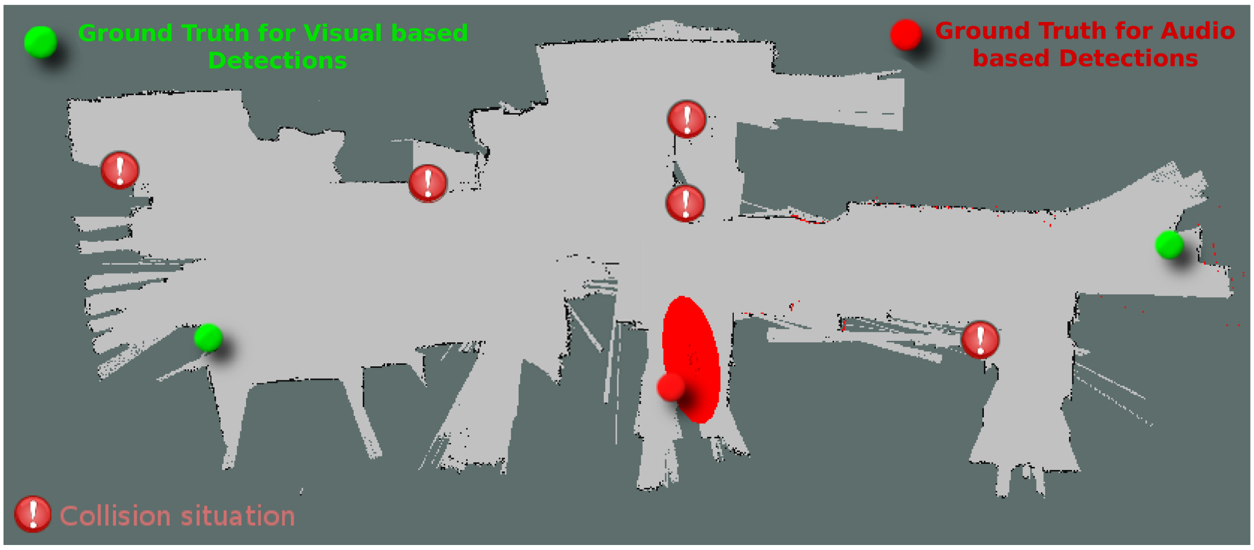

Ground Truth for Visual based Detections
Ground Truth for Audio based Detections
Collision situation

Figure 5.22: Scenario 2 - Resulting output from *rviz* with manually inserted ground truth of people localization.

Unsurprisingly, the changes in lightning conditions did not significantly affect the system's ability to map and due to the nature of structured light sensing. Prevented collisions situations happened when facing a chair, a trash bucket, a wheel of a mobile vehicle, and two tables that were perceived too close to be accurately detected.

Although the results in mapping and exploration did not degrade significantly, the visual-detection method heavily relies on the RGB image. Consequently, as expected, it was not capable of visually detecting the two subjects that did not speak (see Figure 5.23a and Figure 5.23b for an example). Thus, only the voice-based detection was made, as illustrated in Figure 5.22.

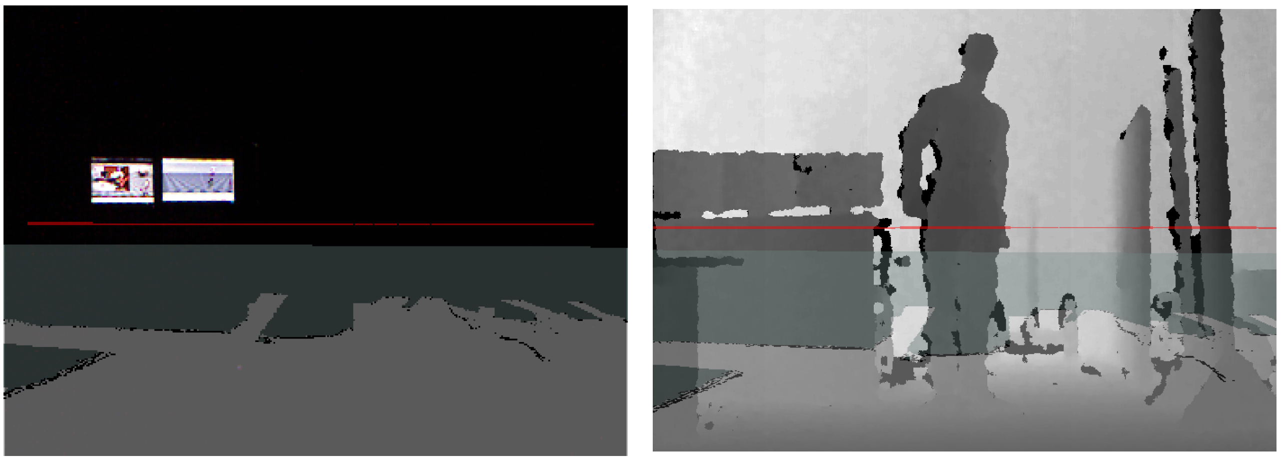

(a) RGB image captured during the experiment. (b) Depth image captured during the experiment.

Figure 5.23: Scenario 2 - Images from the experiment.

5.2.3 Scenario 3 - ISR floor 0 (zero) during night hours

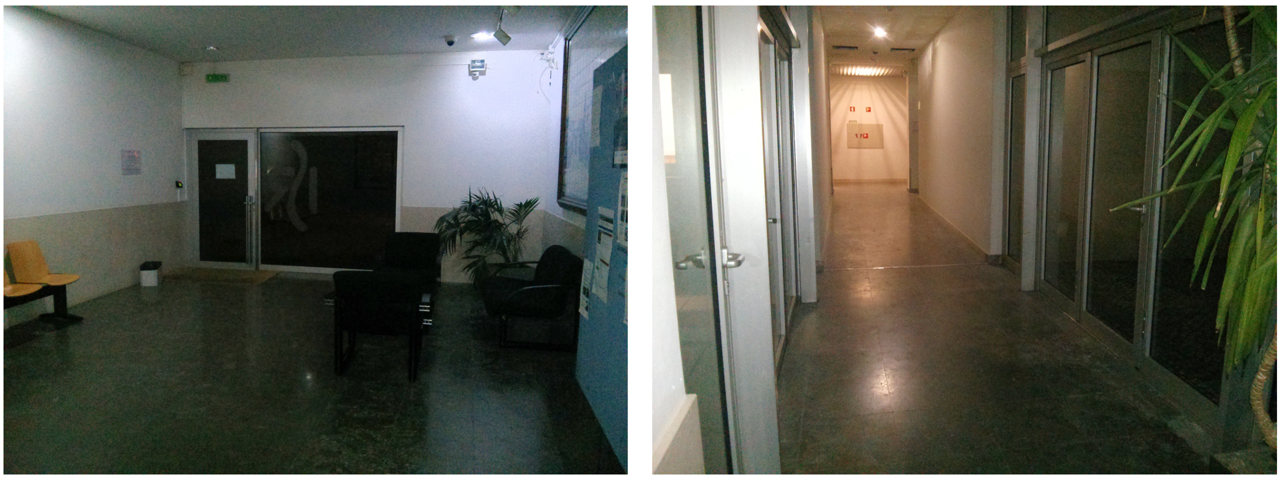
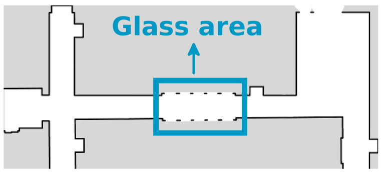

(a) Entrance of the ISR. (b) Corridor of the ISR.

Figure 5.24: Scenario 3 - Photos of the ISR (floor 0 of DEEC).

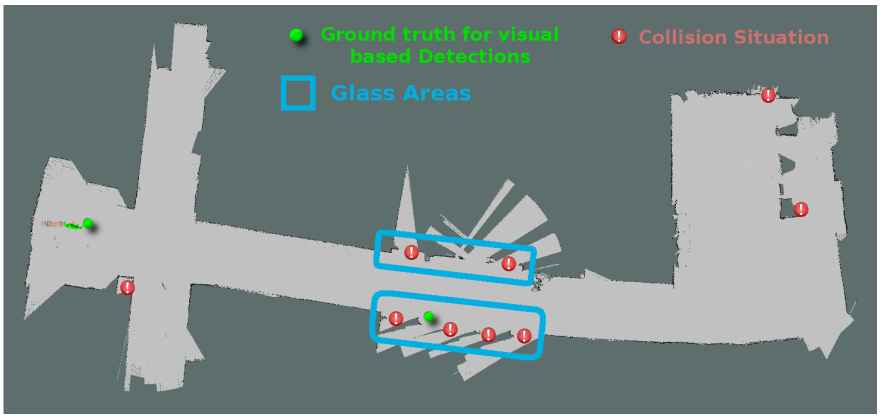
The third scenario was chosen due to its characteristics, which differ from both previous scenarios. It is a large-scale building floor with large areas composed of translucent materials, particularly glasses (see Figure 5.24a and Figure 5.24b). Only a portion of this large environment was selected, more precisely a $26m \times 11.4m$ area. The manually obtained ground truth is shown in Figure 5.25a. There were two people present at this scenario, both were visible.

The main challenges of this scenario were the glassed areas that do no reflect the structured light, thus, the sensor was not able to perceive them, and the large areas with few features that can easily originate errors in mapping.

As expected, most of the problems that arose were associated with the presence of glasses,



(a) Ground truth of the ISR portion.



(b) Resulting output from *rviz* for the ISR floor 0 portion with manually added notes.

Figure 5.25: Scenario 3 - Ground truth of the scenario and resulting output of *rviz* with manually noted ground truth.

e.g. in the middle of the corridor shown in Figure 5.24b, which is marked in the maps of Figure 5.25. At the entrance, the robot moved directly into the glass and had to be interrupted. However, due to the sticker on the glass and the aluminum frames, the system was able to map this area. The corridor area was even more problematic. The glass area is wider and the system was unable to perceive it. Consequently, it attempted to move several times against the glass and even lost its localization which caused significant errors on the map. Due to this unreliable exploration behavior on the corridor, the system also missed a visible person. In conclusion, glass-based walls are a major issue, both because of the robot's inability to correctly perceive them, and because they do not block the IR interference caused by sunlight.

5.3 Summary

The current Chapter presented the experimental results for each module and a detailed analysis for each of them. Section 5.1.1 described and analyzed the results of the mapping module. The test of the reactive exploration module was presented in Section 5.1.2 and its results were criticized. In Section 5.1.3 the people detection from visual cues module was tested with two datasets and the results were compiled and interpretated. The same analysis was conducted for the audio based people detection module in Section 5.1.4, and the last modular test was the integration of the data, which was described in Section 5.1.5. Finally, the real word tests were reported and studied in Section 5.2.

In Chapter 6 conclusions are drawn and interesting future work is suggested based on the results reported on this Chapter and the experience obtained during this project.

Chapter 6
Conclusion and Future Work

6.1 Overview of the System

The main goal of this dissertation was to assemble a fully integrated robotic system built around a RGB-D sensor able to explore, map, detect people, and localize them with respect to the map. Such system was proposed and modularly implemented in order to fulfill this goal. Experimental tests have shown that the sensor used has some limitations, which have a direct impact on the results obtained. Furthermore, computational performance represented a hard constraint, and the option to adopt approaches that downsample the processed data was unavoidable. A preliminary evaluation was conducted for navigation, exploration, and mapping using a RGB-D sensor, and the despite unreliable results in the exploration task based solely on the Kinect, a simple approach was proposed that allows the correction of the trajectory using a human on the loop. Furthermore, a robust people detection system based on the point cloud and the RGB image from the Kinect was implemented in ROS and tested. Additionally, a module that can detect speech in the audio captured from the microphone array and estimate its origin was built. Finally, an integration effort was made to receive, maintain and display the information provided by all modules. Despite the use of a low-cost and imprecise sensor, results show the usability of RGB-D sensors in mobile robotics, more specifically in applications where it is necessary to detect and explore (while mapping) a scenario, *e.g.* in SaR missions.

6.2 Future Work

Although the implementation of a system capable of exploring, mapping, and detecting people from visual and audio cues was possible with the Kinect, some issues are left to solve.

In terms of navigation and exploration, experiments have showed that safe navigation is achievable with the Kinect if the map is available, and an issue was found in the task of exploration without a known map, due to the sensors' constraints. A simple reactive-based exploration approach was proposed, which allows the user to correct the trajectory using the Wii remote controller. Still, this approach only uses the scan readings and does not take into consideration the map obtained thus far and the position of the robot with respect to that map. Leveraging that information may improve the behavior of the autonomous exploration module and allow a more direct search towards the unexplored space.

Concerning people detection based on visual cues, the use of the point cloud to compute the bounding box so as to apply the classical method presented in [17] is accurate and its computational burden is relatively low. However, there is still space to improve. Under certain conditions that can degrade the information of the RGB image, such as reduced illumination or the body image being similar to the background, the depth image is more reliable. Therefore, an analogous method to the one used on RGB images [17] can be used on the depth image to classify the area corresponding to the cluster. There is already work proving that depth images are usable in such way, *e.g.* [72]. The fusion of both method can be done probabilistically, leading to the improvement of the overall results, and allowing detections to be made on scenarios such as the one tested on Section 5.2.2. Furthermore, due to occlusions the body shape is not always visible and only a system that is able to classify body parts may be able to correctly identify the human body. The face is generally not often occluded and contains enough features to be identified.

As for the audio processing, the system shows an acceptable accuracy in terms of sound separation and recognition. Nevertheless, the sound source localization is not accurate in terms of distance, and especially elevation. This can be justified by the hardware limitations of the MA that possesses only four microphones, which are not only co-linear, but also three of them very close to each other. Therefore, the addition of further microphones could provide more precise results, especially if they are sparse is space. Furthermore, an interesting extension to the sound localization method would be to leverage from diverse viewpoints of the robot, assuming a static sound source, thus emulating a microphone array with several more microphones.

Finally, the system displays the results for the audio-based and visual-based detections in different colors on the visualization software. However, information from the different detecting modules could be fused to achieve even better results, instead of separately representing them on the map.

Bibliography

[1] ROS wiki. `http://wiki.ros.org/`, 2013. Accessed: 2014-07-10.

[2] ROS wiki - ccny_rgbd. `http://wiki.ros.org/ccny_rgbd`, 2013. Accessed: 2014-07-10.

[3] ROS wiki - freenect_stack. `http://wiki.ros.org/freenect_stack`, 2013. Accessed: 2014-07-10.

[4] ROS wiki - navigation. `http://wiki.ros.org/navigation`, 2013. Accessed: 2014-07-10.

[5] ROS wiki - openni_kinect. `http://wiki.ros.org/openni_kinect`, 2013. Accessed: 2014-07-10.

[6] ROS wiki - frontier_exploration. http://wiki.ros.org/frontier_exploration, 2014. Accessed: 2014-07-10.

[7] ROS wiki - rosaria. `http://wiki.ros.org/ROSARIA`, 2014. Accessed: 2014-07-10.

[8] Kai Oliver Arras, Óscar Martínez Mozos, and Wolfram Burgard. Using boosted features for the detection of people in 2D range data. *ICRA*, pages 3402–3407, 2007.

[9] Futoshi Asano, S. Ikeda, M. Ogawa, Hideki Asoh, and Nobuhiko Kitawaki. Combined approach of array processing and independent component analysis for blind separation of acoustic signals. *IEEE Transactions on Speech and Audio Processing*, 11(3):204–215, 2003.

[10] Max Bajracharya, Baback Moghaddam, Andrew Howard, Shane Brennan, and Larry H. Matthies. A fast stereo-based system for detecting and tracking pedestrians from a moving vehicle. *International Journal of Robotics Research*, 28(11-12):1466–1485, November 2009.

[11] Herbert Bay, Tinne Tuytelaars, and Luc Gool. SURF: Speeded up robust features. *Computer Vision – ECCV 2006*, 3951:404–417, 2006.

[12] B. Bendjima, Jean Michel Rouvaen, Atika Rivenq, and F. Haine. An adapted spectral subtraction method for enhancing speech signals in a car environment. *SIP*, pages 288–291, 1999.

[13] Joydeep Biswas and Manuela Veloso. Depth camera based indoor mobile robot localization and navigation. *Proceedings of IEEE International Conference on Robotics and Automation*, pages 1697–1702, May 2012.

[14] Guy J. Brown and Martin Cooke. Computational auditory scene analysis. *Computer Speech & Language*, 8(4):297 – 336, 1994.

[15] Michael Calonder, Vincent Lepetit, and Pascal Fua. Keypoint signatures for fast learning and recognition. *In European Conference on Computer Vision*, 2008.

[16] Leandro Cruz, Djalma Lucio, and Luiz Velho. Kinect and RGB-D images: Challenges and applications. *SIBGRAPI Tutorials*, pages 36–49, 2012.

[17] N. Dalal and B. Triggs. Histograms of oriented gradients for human detection. *International Conference on Computer Vision & Pattern Recognition*, 2:886–893, June 2005.

[18] MWM Gamini Dissanayake, Paul Newman, Steve Clark, Hugh F Durrant-Whyte, and Michael Csorba. A solution to the Simultaneous Localization and Mapping (SLAM) problem. *Robotics and Automation, IEEE Transactions on*, 17(3):229–241, 2001.

[19] I. Dryanovski, R.G. Valenti, and Jizhong Xiao. Fast visual odometry and mapping from RGB-D data. pages 2305–2310, May 2013.

[20] Felix Endres, Jürgen Hess, Nikolas Engelhard, Jürgen Sturm, Daniel Cremers, and Wolfram Burgard. An evaluation of the RGB-D SLAM system. *In Proc. of the IEEE Intl. Conf. on Robotics and Automation (ICRA)*, pages 1691–1696, 2012.

[21] Yariv Ephraim and Harry L. Van Trees. A signal subspace approach for speech enhancement. *IEEE Transactions on Speech and Audio Processing*, 3(4):251–266, 1995.

[22] Eitan Marder Eppstein, Eric Berger, Tully Foote, Brian P. Gerkey, and Kurt Konolige. The office marathon: Robust navigation in an indoor office environment. *In Proc. of the IEEE Intl. Conf. on Robotics and Automation (ICRA)*, May 2010.

[23] M. A. A. El Fattah, Moawad I. Dessouky, S. M. Diab, and F. A. E. El Samie. Speech enhancement using an adaptive wiener filtering approach. *Progress In Electromagnetics Research M*, 4:167–184, 2008.

[24] João Filipe Ferreira, Jorge Lobo, Pierre Bessière, Miguel Castelo-Branco, and Jorge Dias. A Bayesian framework for active artificial perception. *IEEE Trans. on Cybernetics (Part B)*, 43(2):699–711, 2013.

[25] Martin A. Fischler and Robert C. Bolles. Random sample consensus: A paradigm for model fitting with applications to image analysis and automated cartography. *Communications of the ACM*, 24(6):381–395, 1981.

[26] D. Fox, W. Burgard, F. Dellaert, and S. Thrun. Monte carlo localization: Efficient position estimation for mobile robots. *In Proc. of the National Conference on Artificial Intelligence (AAAI)*, pages 343–349, 1999.

[27] D. Fox, W. Burgard, and S. Thrun. The dynamic window approach to collision avoidance. *IEEE Robotics & Automation Magazine*, 4(1):23–33, 1997.

[28] O. L. Frost. An algorithm for linearly constrained adaptive array processing. *Proceedings of the IEEE*, 60(8):926–935, 1972.

[29] G Grisetti, C Stachniss, S Grzonka, and W Burgard. TORO: tree-based network optimizer. 2009.

[30] Giorgio Grisetti, Cyrill Stachniss, and Wolfram Burgard. Improved techniques for grid mapping with Rao-Blackwellized particle filters. *IEEE Transactions on Robotics*, 23:2007, 2007.

[31] P. Henry, M. Krainin, E. Herbst, X. Ren, and D. Fox. RGB-D mapping: Using Kinect-style depth cameras for dense 3D modeling of indoor environments. *International Journal of Robotics Research (IJRR)*, 31(5):647–663, April 2012.

[32] Xuedong Huang, Fileno Alleva, Hsiao-Wuen Hon, Mei-Yuh Hwang, Kai-Fu Lee, and Ronald Rosenfeld. The SPHINX-II speech recognition system: an overview. *Computer Speech & Language*, 7(2):137–148, 1993.

[33] Xuedong Huang, Fileno Alleva, Mei-Yuh Hwang, and Ronald Rosenfeld. An overview of the SPHINX-II speech recognition system. pages 81–86, 1993.

[34] David Huggins-Daines, Mohit Kumar, Arthur Chan, Alan W Black, Mosur Ravishankar, and Alex I Rudnicky. Pocketsphinx: A free, real-time continuous speech recognition system for hand-held devices. 1:I–I, 2006.

[35] Himanshu Prakash Jain, Anbumani Subramanian, Sukhendu Das, and Anurag Mittal. Real-time upper-body human pose estimation using a depth camera. *Proceedings of the 5th international conference on Computer vision/computer graphics collaboration techniques*, pages 227–238, 2011.

[36] Christoph G Keller, Markus Enzweiler, Marcus Rohrbach, David Fernandez Llorca, Christoph Schnorr, and Dariu M Gavrila. The benefits of dense stereo for pedestrian detection. *ITS, IEEE Trans. on*, 12(4):1096–1106, 2011.

[37] Kourosh Khoshelham and Sander Oude Elberink. Accuracy and resolution of Kinect depth data for indoor mapping applications. *Sensors (Basel)*, 12(2):1437–1454, 2012.

[38] Stefan Kohlbrecher, Oskar Von Stryk, Johannes Meyer, and Uwe Klingauf. A flexible and scalable SLAM system with full 3D motion estimation. *In Proceeding of Safety, Security, and Rescue Robotics (SSRR), 2011 IEEE International Symposium on*, pages 155–160, 2011.

[39] Kurt Konolige, Giorgio Grisetti, Rainer Kümmerle, Wolfram Burgard, Benson Limketkai, and Régis Vincent. Efficient sparse pose adjustment for 2D mapping. *Intelligent Robots and Systems (IROS)*, pages 22–29, 2010.

[40] R. Kuemmerle, G. Grisetti, H. Strasdat, K. Konolige, and W. Burgard. g2o: A general framework for graph optimization. *Proc. of the IEEE Int. Conf. on Robotics and Automation (ICRA)*, pages 3607–3613, May 2011.

[41] Kobi Levi and Yair Weiss. Learning object detection from a small number of examples: the importance of good features. *Proceedings of the 2004 IEEE computer society conference on Computer vision and pattern recognition*, pages 53–60, 2004.

[42] DF Llorca, MA Sotelo, AM Hellín, A Orellana, M Gavilan, IG Daza, and AG Lorente. Stereo regions-of-interest selection for pedestrian protection: A survey. *Transportation research part C: emerging technologies*, 25:226–237, 2012.

[43] Beth Logan and Tony Robinson. Adaptive model-based speech enhancement. *Speech Communication*, 34(4):351–368, 2001.

[44] David G. Lowe. Distinctive image features from scale-invariant keypoints. *Int. J. Comput. Vision*, 60(2):91–110, 2004.

[45] P. Menezes, L. Brethes, F. Lerasle, P. Danes, and J. Dias. Visual tracking of silhouettes for human-robot interaction. pages 971–976, 2003.

[46] Jorge J. Moré. The Levenberg-Marquardt algorithm: Implementation and theory. 630:105–116, 1978.

[47] Marius Muja and David G. Lowe. Fast approximate nearest neighbors with automatic algorithm configuration. *In VISAPP International Conference on Computer Vision Theory and Applications*, pages 331–340, 2009.

[48] Matteo Munaro, Filippo Basso, and Emanuele Menegatti. Tracking people within groups with RGB-D data. *IROS*, pages 2101–2107, 2012.

[49] Matteo Munaro and Emanuele Menegatti. Fast RGB-D people tracking for service robots. *Autonomous Robots*, pages 1–16, 2014.

[50] Kazuhiro Nakadai, Hiroshi G Okuno, Toru Takahashi, Keisuke Nakamura, Takeshi Mizumoto, Takami Yoshida, Takuma Otsuka, and Gokhan Ince. Introduction to open source robot audition software HARK. *Proceedings of the 29th Annual Convention of RSJ*, September 2011.

[51] Hirofumi Nakajima, Gökhan Ince, Kazuhiro Nakadai, and Yuji Hasegawa. An easily-configurable robot audition system using histogram-based recursive level estimation. *Intelligent Robots and Systems (IROS)*, pages 958–963, 2010.

[52] Hirofumi Nakajima, Kazuhiro Nakadai, Yuji Hasegawa, and Hiroshi Tsujino. Sound source separation adaptable to environmental changes for robot audition. *Journal of the Robotics Society of Japan*, 27(7):774–781, sep 2009.

[53] Hirofumi Nakajima, Kazuhiro Nakadai, Yuji Hasegawa, and Hiroshi Tsujino. Blind source separation with parameter-free adaptive step-size method for robot audition. *IEEE Transactions on Audio, Speech & Language Processing*, 18(6):1476–1485, 2010.

[54] K. Nakamura, K. Nakadai, F. Asano, Y. Hasegawa, and H. Tsujino. Intelligent sound source localization for dynamic environments. pages 664–669, Oct 2009.

[55] Keisuke Nakamura, Kazuhiro Nakadai, Futoshi Asano, and Gökhan Ince. Intelligent sound source localization and its application to multimodal human tracking. *Intelligent Robots and Systems (IROS)*, pages 143–148, 2011.

[56] K. Okutani, T. Yoshida, K. Nakamura, and K. Nakadai. Outdoor auditory scene analysis using a moving microphone array embedded in a quadrocopter. *Intelligent Robots and Systems (IROS), 2012 IEEE/RSJ*, pages 3288–3293, 2012.

[57] Kuldip Paliwal, Belinda Schwerin, and Kamil Wójcicki. Speech enhancement using a minimum mean-square error short-time spectral modulation magnitude estimator. *Speech Communication*, 54(2):282–305, February 2012.

[58] Lucas Parra and Clay Spence. Convolutive blind separation of non-stationary sources. *IEEE Trans. on Speech and Audio Processing*, 8(3):320–327, 2000.

[59] Hanspeter Pfister, Mattias Zwicker, Jeroen van Baar, and Markus Gross. Surfels: Surface elements as rendering primitives. *ACM Transactions on Graphics (Proc. of SIGGRAPH)*, 2000.

[60] Charles Pippin, Henrik I. Christensen, and Lora Weiss. Performance-based task assignment in multi-robot patrolling. *IEEE Robotics and Automation Society (SAC)*, pages 70–76, 2013.

[61] David Portugal and Rui P. Rocha. Multi-robot patrolling algorithms: examining performance and scalability. *Advanced Robotics*, 27(5):325–336, 2013.

[62] Cristiano Premebida, Oswaldo Ludwig, and Urbano Nunes. LIDAR and vision-based pedestrian detection system. *Journal of Field Robotics*, 26(9):696–711, 2009.

[63] M. Quigley, K. Conley, B. P. Gerkey, J. Faust, T. Foote, J. Leibs, R. Wheeler, and A. Y. Ng. ROS: an open-source robot operating system. *ICRA Workshop on Open Source Software*, 2009.

[64] L. Rabiner and B.-H. Juang. An introduction to Hidden Markov Models. *ASSP Magazine, IEEE*, 3(1):4–16, Jan 1986.

[65] E. Rosten and T. Drummond. Machine learning for high-speed corner detection. *In European Conference on Computer Vision*, pages 430–443, 2006.

[66] Ethan Rublee, Vincent Rabaud, Kurt Konolige, and Gary Bradski. ORB: An efficient alternative to SIFT or SURF. *In. Proc. International Conference on Computer Vision*, 11/2011 2011.

[67] R.B. Rusu and S. Cousins. 3D is here: Point cloud library (PCL). *Robotics and Automation (ICRA), 2011 IEEE International Conference on*, pages 1–4, May 2011.

[68] J.Machado Santos, D. Portugal, and R.P. Rocha. An evaluation of 2D SLAM techniques available in Robot Operating System (ROS). *Safety, Security, and Rescue Robotics (SSRR), 2013 IEEE International Symposium on*, pages 1–6, Oct 2013.

[69] J. Satake and J. Miura. Multiple-person tracking for a mobile robot using stereo. *MVA Conference*, pages 273–277, 2009.

[70] R. Schmidt. Multiple emitter location and signal parameter estimation. *Antennas and Propagation, IEEE Transactions on*, 34(3):276–280, March 1986.

[71] L. Spinello, K. O. Arras, R. Triebel, and R. Siegwart. A layered approach to people detection in 3D range data. *Proc. of The AAAI Conference on Artificial Intelligence: Physically Grounded AI Track (AAAI)*, 2010.

[72] Luciano Spinello and Kai Oliver Arras. People detection in RGB-D data. *In Proceedings of Intelligent Robots and Systems (IROS)*, pages 3838–3843, 2011.

[73] Toru Takahashi, Kazuhiro Nakadai, Kazunori Komatani, Tetsuya Ogata, and Hiroshi G. Okuno. Missing-feature-theory-based robust simultaneous speech recognition system with non-clean speech acoustic model. *Proc. of Intelligent Robots and Systems (IROS) 2009*, pages 2730–2735, 2009.

[74] S. Thrun, D. Fox, W. Burgard, and F. Dellaert. Robust Monte Carlo localization for mobile robots. *Artificial Intelligence*, 128(1-2), 2001.

[75] Michal Tolgyessy and Peter Hubinsky. The Kinect sensor in robotics education. *Proceedings of 2nd International Conference on Robotics in Education (RiE 2011)*, pages 143–146, 2011.

[76] Mikkel Viager. Analysis of Kinect for mobile robots. *Technical report*, 2011.

[77] Regis Vincent, Benson Limketkai, and Michael Eriksen. Comparison of indoor robot localization techniques in the absence of GPS. *SPIE Defense, Security, and Sensing*, pages 76641Z–76641Z, 2010.

[78] Daniel Willett, Andreas Worm, Christoph Neukirchen, and Gerhard Rigoll. Confidence measures for HMM-based speech recognition. *in International Conference on Spoken Language Processing*, pages 3241–3244, 1998.

[79] C.C. Chen L. Xia and J. K. Aggarwal. Human detection using depth information by Kinect. *Workshop on Human Activity Understanding from 3D Data in conjunction with CVPR (HAU3D)*, 2011.

[80] Brian Yamauchi. A frontier-based approach for autonomous exploration. *Computational Intelligence in Robotics and Automation 1997,IEEE International Symposium on*, pages 146–151, 1997.

[81] Zhengyou Zhang. Microsoft Kinect sensor and its effect. *IEEE Multimedia*, 19(2):4–10, 2012.

Appendix A
Sound Processing Algorithms

A.1 MUSIC Algorithm on HARK

For further information see [54] and [55].

A.1.1 Transfer Function

In the MUSIC method, the transfer function from sound to each microphone is measured or calculated numerically and it is used as a priori information.

Considering $h_i(\theta, \omega)$, the frequency domain transfer function from sound $S(\theta)$ in direction θ for microphone M_i, the multichannel transfer function can be expressed as follows:

$$H(\theta, \omega) = [h_1(\theta, \omega), ..., h_M(\theta, \omega)] \tag{A.1}$$

In HARK's implementation of this algorithm, this a priori transfer function is computed for multiple parameters so as to compile a database containing a transfer function for every direction of sound. For that reason it is also called steering vector.

A.1.2 Correlation Matrix between Channels

The signal vector in frequency domain is obtained computing a Fourier transform of the input acoustic signal in M channel:

$$X(\omega, f) = [X_1(\omega, f), X_2(\omega, f), X_3(\omega, f), X_4(\omega, f)]^T \tag{A.2}$$

In equation A.2, ω represents the frequency while f stands for the frame index. The correlation matrix between channels of the incoming signal $X(\omega, f)$ is defines for every frame and frequency as follows:

$$R(\omega, f) = X(\omega, f)X^*(\omega, f) \tag{A.3}$$

Theoretically, this correlation matrix is satisfactory, however, practically a stable correlation matrix is obtained by using a time-based average:

$$R'(\omega, f) = \frac{1}{WINDOW} \sum_{i=W_i}^{W_f} R(\omega, f+i) \tag{A.4}$$

In equation A.4, W_i and W_f are defined by $WINDOW$.

A.1.3 Eigenvalue Decomposition

The MUSIC is a eigenvalue (or singular value) decomposition based method. In the proposed system, a GEVD is performed. However, since this operation has a high computational cost, it is only calculated once in several frames, by specifying a *period*:

$$K^{-1}(\omega, f)R'(\omega, f) = E(\omega, f)\Lambda(\omega, f)E^{-1}(\omega, f) \tag{A.5}$$

In equation A.5, the matrix $K(\omega, f)$ is the correlation matrix derived by noise sources. In this model it is considered to be regular, to simply the model and, it most often is. $E(\omega, f)$ consists of singular vector which perpendicularly intersect each other, and $\Lambda(\omega, f)$ represents the diagonal matrix using the eigenvalue to individual eigenvector as the diagonal component.

A.1.4 Calculation of MUSIC spectrum

The MUSIC spectrum for sound source localization is calculated as follows:

$$P(\theta, \omega, f) = \frac{|H^*(\theta, \omega)H(\theta, \omega)|}{\sum_{i=N_s+1}^{M} |H^*(\theta, \omega)e_i(\omega, f)|} \tag{A.6}$$

In the denominator of the right-hand side of equation A.6 is the inner product of the eigen vector and the steering vector. On the space spanned by the eigen vector, since the noise subspace, corresponding to small eigenvalue, and the signal subspace, corresponding to a large eigenvalue, intersect perpendicularly each other, if the transfer function is a vector corresponding to the sound, the inner product will theoretically be 0. Thus, $P(\theta, \omega, f)$ diverges to infinite. In fact, although under the effect of noise it does not diverge, a peak is observed. The numerator is an normalization term.

$P(\theta, \omega, f)$ is the MUSIC spectrum for every frequency, however that is not necessary for the intended application, so it is rewritten as follows:

$$\bar{P}(\theta, f) = \sum_{\omega=\omega_{min}}^{\omega_{max}} W_{\Lambda}(\omega, f) W_{\omega}(\omega, f) P(\theta, \omega, f) \tag{A.7}$$

In equation A.7, ω_{min} and ω_{max} define the minimum and maximum frequencies handled in the MUSIC spectrum, respectively. $W_{\Lambda}(\omega, f)$ is the eigen-value weight and corresponds to the square root of the maximum eigenvalue. $W_{\omega}(\omega, f)$ is the frequency weight.

A.1.5 Sound Sources

Finally, the peaks are detected from the range of θ_{min} to θ_{max} for $\bar{P}(\theta, f)$ and detected by a local maximum search algorithm.

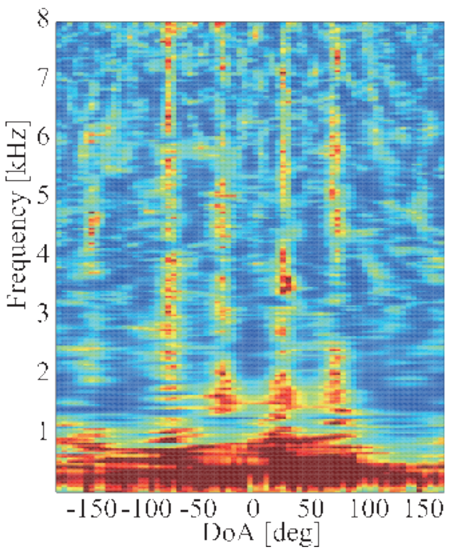


Figure A.1: Example of MUSIC Spectrum for 4 speakers.

Figure A.1 shows the example MUSIC spectrum for a situation with 4 speakers positions at: 75°, 25°, -25°, -75°.

A.2 GHDSS Algorithm on HARK

A.2.1 The Mixture Model

The sound that is emitted from N sound sources is affected in space by $H(k_i)$ and captured by M microphones as the follow equation describes:

$$X(k_i) = H(k_i)S(k_i) + N(k_i) \tag{A.8}$$

In equation A.8, $S(k_i)$ represents the sound source complex spectrum, $N(k_i)$ is the additive noise that acts on each microphone, and $X(k_i)$ is the vector of a microphone observation complex spectrum.

The transfer function $H(k_i)$ depends on the geometry and shape of the room and also on the positional relation between the microphones and the sound sources, and consequently, is difficult to estimate. However, ignoring acoustic reflection and diffraction, and in the case of knowing the relative position of the microphones and the sound source, the transfer function is only limited to the direct sound $H_D(k_i)$ that can be expressed as:

$$H_{D_{m,n}}(k_i) = e^{-j2\pi l_i r_{m,n}}, \tag{A.9}$$

with:

$$l_i = \frac{2\pi\omega_i}{c} \tag{A.10}$$

In equation A.10, c is the speed of sound and l_i represents the wave number corresponding to the frequency ω_i in the frequency bin k_i. Furthermore, $r_{m,n}$ is the difference between the distance from the microphone m to the sound source n and the distance between the reference point of the coordinate system to the sound source n. In short, $H_D(k_i)$ is the phase difference incurred by the difference in arrival time from the sound source to each of the microphones.

A.2.2 The Separation Model

The matrix of a complex spectrum of separated sound $Y(k_i)$ is obtained as follows:

$$Y(k_i) = W(k_i)X(k_i) \tag{A.11}$$

The GHDSS algorithm estimates the separation matrix $W(k_i)$, so that $Y(k_i)$ is close to the sound source complex spectrum $S(k_i)$. The k_i are the frequency bins.

A.2.3 Assumptions of the Model

1. The number of sound sources N;

2. Sound source position (given by MUSIC algorithm);

3. Relative microphone Position;

4. Transfer function of the direct sound component $H_D(k_i)$.

A.2.4 Updating the Separation Matrix

The GHDSS estimates the separation matrix $W(k_i)$ so that the following requirements are met:

1. A high-order decorrelation of the separated signals:

   This corresponds to make the diagonal component of the high-order matrix $R^{\phi(y)y}(k_i) = E[\phi(Y(k_i))Y^H(k_i)]$ of the separated sound $Y(k_i)$ zero. The operator ^{H} is the hermite transpose, $E[]$ is the time average operator, and $phi()$ is the non-linear function. Also, a hyperbolic tangent function is used and is defined as follows:

   $$\phi(Y) = [\phi(Y_1), \phi(Y_2), ..., \phi(Y_N)]^T \tag{A.12}$$

   $$\phi(Y_k) = \tanh(\sigma|Y_k|)e^{j\angle(Y_k)} \tag{A.13}$$

   Where σ is a scaling factor.

2. The direct sound component is separated without considerable distortions.

   Same as making the product of the separation matrix $W(k_i)$ and the transfer function of the direct sound $H_D(k_i)$ a identity matrix:

   $$W(k_i)H_D(k_i) = I \tag{A.14}$$

The upper binary element is matched with an evaluation function as follows:

$$J(W(k_i)) = \alpha J_1(W(k_i)) + \beta J_2(W(k_i)) \tag{A.15}$$

with:

$$J_1(W(k_i)) = \sum_{i \in j} |R^{\phi(y)y}_{i,j}|^2 \tag{A.16}$$

$$J_2(W(k_i)) = W(k_i)H_D(k_i) - I^2 \tag{A.17}$$

In the previous equations, *alpha* and *beta* are weighting factors.

So, the update equation of the separation matrix to minimize equation A.15 is obtained by a gradient method:

$$W(k_i, f+1) = W(k_i, f) - \mu \frac{\partial J}{\partial W^*}(W(k_i, f)) \tag{A.18}$$

In equation A.18, μ is the stepsize, which regulates the quantity of update of the separation matrix.

The initial values of the separation matrix can be computed by:

$$W(k_i) = \frac{H_D^H(k_i)}{M}, \tag{A.19}$$

with M as the number of microphones.

A.2.5 Summary

The process for a time frame f in the GHDSS algorithm is:

1. Acquire a transfer function (direct sound);
2. Estimate the separation matrix W;
3. Perform sound source separation by equation A.11.

A.3 Noise Reduction and Speech Enhancement

A.3.1 Leakage Noise Estimation

Consider $\lambda^{leak}(k_i)$ the power spectrum of leakage noise, obtained by:

$$\lambda_n^{leak} = \alpha^{leak} \sum_{n' \in n} Z_{n'}(k_i), \tag{A.20}$$

with:

$$Z_n(k_i) = \alpha S_n(k_i) + \beta S_n^{leak}(k_i), \tag{A.21}$$

$$\alpha = 1 - (N-1)\alpha^{leak}\beta, \tag{A.22}$$

$$\beta = -\frac{\alpha^{leak}}{1 - (\alpha^{leak})^2 + \alpha^{leak}(1 - \alpha^{leak})(N-2)}. \tag{A.23}$$

In the previous equations $S_n(k_i)$ is the smoothing power spectrum and α^{leak} is the leakage rate for the total of separated sound power, which is computed from the product two other parameters: the leakage rate and the leakage rate weighting factor.

A.3.2 Determination of the optimum gain

The determination of the optimum gain for how much of the estimated noise's power spectrum of the estimated noise is to be removed is fundamental in the process of speech enhancement of a audio signal. Consider $X_n(k_i)$ the power spectrum of the separated sound, $N_n(k_i)$ the estimated noise, n the frame number, k_i the frequency index, and $G_n(k_i)$ the optimum gain.

$$G_n(k_i) = \begin{cases} \alpha \frac{Y_n(k_i)}{X_n(k_i)}, & \text{if } Y_n(k_i) > \beta \\ \beta, & \text{if otherwise} \end{cases} \tag{A.24}$$

with

$$Y_n(k_i) = X_n(k_i) - N_n(k_i) \tag{A.25}$$

A.3.3 Spectral Gain Filter

The spectral gain filter is a module that uses the optimal gain $G_n(k_i)$, the probability of speech presence $p_n(k_i)$ (1 by default), and the separated sound spectrum $X_n(k_i)$ to filter the sound and obtain the separated sound with the speech enhanced $Y_n(k_i)$ as follows:

$$Y_n(k_i) = X_n(k_i)G_n(k_i)p_n(k_i). \tag{A.26}$$

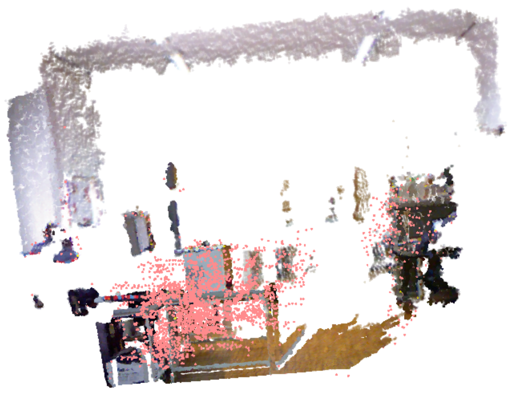
On equation A.24, α is the gain for spectral subtraction and β is the spectral floor.

Appendix B

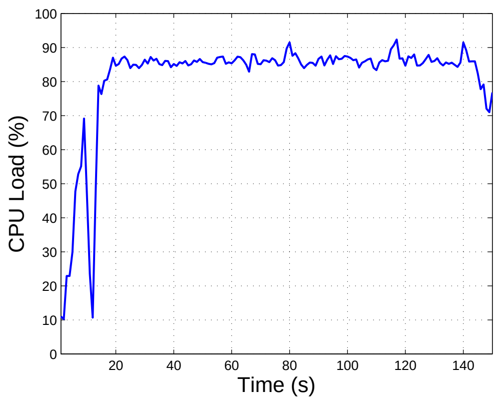
RGB-D SLAM Experimentation

To analyse the computational burden of RGB-D SLAM, the method described in [19] was tested. The implementation on ROS is publicly available in ROS Wiki - CCNY RGBD[1]. A demonstrative video of the method was made available by the authors: `https://www.youtube.com/watch?v=YE9eKgek5pI`

To run the experiments, a laptop equipped with an Intel Core i7-4700MQ and 16GB of RAM running Ubuntu 12.04 and ROS Hydro was used. The referred method run alongside the system and the ROSARIA driver.



(a) Frame of the RGB-D mapping.



(b) CPU load for the RGB-D method tested.

Figure B.1: Images obtained during the test of the RGB-D mapping method.

The results are reliable and accurate, however the computational burden of the method is high to be compliant with a system that also performs people detection. The computed average CPU load was 81.1% including the launching, and 85.71% after that stage.

[1]`http://wiki.ros.org/ccny_rgbd`

Appendix C

Navigation

C.1 Navigation

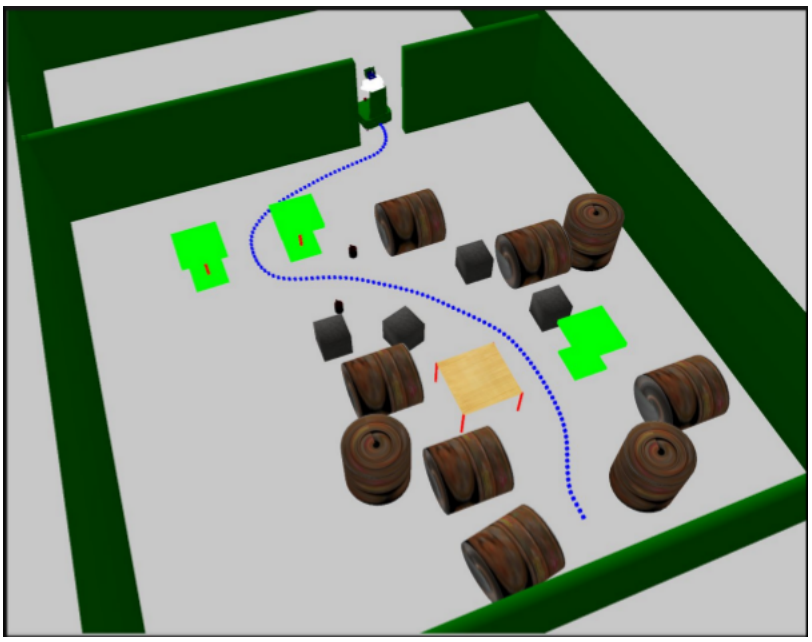


Figure C.1: Simulation of a robot navigating on an indoor cluttered environment [22].

With the arrival of RGB-D sensors, such as the Kinect, and the approaches to RGB-D SLAM, there is an extra interest in studying navigation methods using these cheaper sensors. Marder-Eppstein et al. presented in [22] a navigation system that allowed a robot completing 26.2 miles of autonomous navigation in a real office environment.

The challenge of 3D navigation is to detect and avoid obstacles with non-trivial 3D structure, and still drive through narrow paths. To face this problem it is essential that the robot is able to reason about its environment in 3D, handle unknown space effectively, and navigate in cluttered environments with different obstacles.

The authors use 3D obstacle data to avoid the smaller obstacles the sensor can perceive, and drives through the tightest spaces the robot can fit. The core of their work is Voxel

Grid, a structure that encodes the robot's knowledge of the environment, classifying space as free, occupied or unknown.

C.1.1 Approach

Marder-Eppstein et al. suggested an approach in [22] that is able to deal with the previous problems and navigate safely in cluttered environments. The approached is already available on ROS through the Navigation stack [4]. This section presents the key aspects of this approach.

Sensor Processing Pipeline

Sensors are not totally immune to changes in the environment and, under certain conditions, the data they provide can be corrupted or erratic. Consequently, obstacles may be detected where none exist or be missed when they do. To solve this issue, the authors introduced a pipeline that takes the raw sensor data and applies a number of filters before it is converted to obstacle information and be used for planning. The authors used two Hokuyo laser scanners to collect data, however RGB-D sensors can be used too. In a system with a Kinect as the main sensor, the 3D point cloud provided by it should be converted to a laser scan.

Voxel Grid

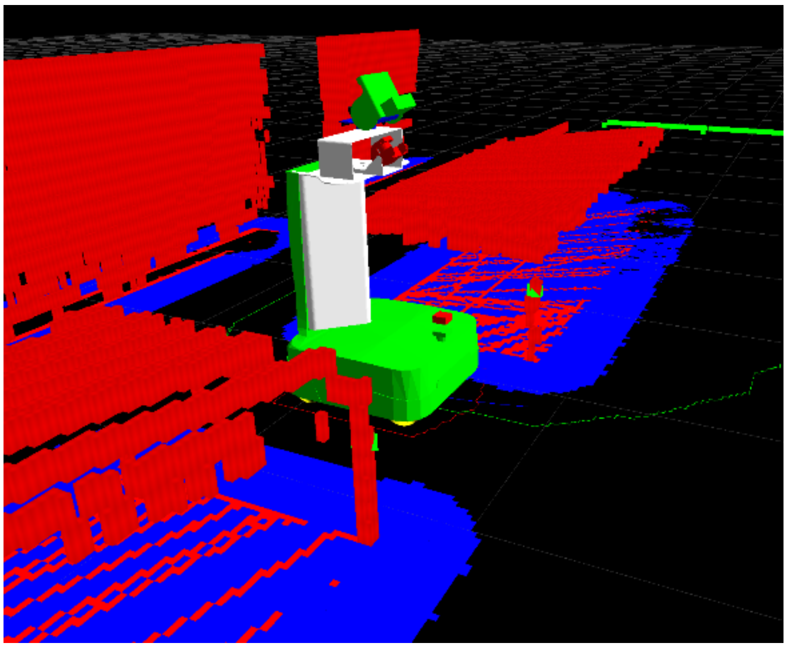


Figure C.2: The Voxel Grid as the robot navigates between two tables [22].

As earlier presented, the Voxel Grid is an efficient 3D occupancy grid that composes the core of this navigation system. Each cell can have three states: free, occupied or unknown. Even though a probabilistic scheme could produce better results, the authors opted not to use it because of performance considerations. There are two main operations that are performed in the grid: *marking* and *clearing*. The former refers to changing the status of an element based on an observation from the sensor. The latter, refers to raytracing in the grid from the origin of a sensor to an observation while asserting free space along the way.

The authors' implementation of the Voxel Grid is remarkably efficient, however it limits the number of grid cells in the Z axis to 16 to run at good rate.

Unknown Space

The Voxel Grid has the important advantage of tracking unknown space in the world, as previously stated. There are several different approaches to handle occlusions in navigation, such as limiting the robot's speed. The author's approach do not directly limit the robot's speed, instead it treats the unknown space as illegal to traverse which results in decreasing the speed.

Ideally, every unknown grid cell would be treated as an obstacle, however that would cause the robot to move at an unreasonable speed. Consequently, the authors used a non-zero tolerance for the number of unknown cells allowed in a Voxel Grid column that are considered to be free. Setting this number low increases safety but causes a slower navigation, and using a higher number results in a faster but riskier navigation.

Costmap

In order to navigate on flat ground, the obstacle data is assembled into a planar Costmap on which planners operate. The Costmap is a 2D structure, however its underlying representation actually consists of the Voxel Grid described earlier. Each column of the Voxel Grid is projected down into two dimensions where it is assigned a cost, computed by the state of the cells on that particular column.

Global Planner

The Global Planner receives the obstacle and cost information from the Costmap, information from the robot's localization system, and a goal in the world. Then, it computes a high-level plan for the robot to follow to reach its goal. It is very important that this process is efficient, so that the navigation system can run reasonably.

Therefore, this planner reduces the robot to a circular configuration and ignores the dynamics and kinematics of the robot. Consequently, it is only used as a high-level guide for navigation.

Local Planner

The Local Planner is responsible for generating velocity commands for the mobile base to safely reach its goal location. The Local Planner receives the plan produced by the Global Planner and attempts to follow it, while taking into account the kinematics and dynamics of the robot, and the obstacle information stored in the Costmap. It makes use of a technique known as the Dynamic Window Approach, a real-time collision avoidance strategy developed by Fox et al. and described in [27].

Localization

It is of absolute importance that the robot is able to localize itself in the maps. To that purpose, there is a probabilistic localization system in ROS based on a particle filter approach for a robot moving in 2D, the Adaptive Monte Carlo Localization (AMCL), further described by Thrun et al. in [74]. It implements the adaptive Monte Carlo localization approach, presented in [26], by Fox et al., which uses a particle filter to track the pose of a robot against a known map.

In this context, it might be useful to convert the point cloud provided by the Kinect to a laser scan to use in the approach earlier described. Then, the task of scan matching between the simulated laser scans and previous information is crucial. It consists of matching the scans of each particle in the AMCL to estimate the position on the local map.

A practical example of this approach is implemented on the TurtleBot[1] that relies solely on the Kinect to navigate safely by down-sampling the point cloud to simulate a laser scan. Studies and experiments have shown that the system is reliable and navigates efficiently to a desired goal location. It was used, for example, by Pippin et al. in the work described in [60].

[1]http://turtlebot.com

C.1.2 Navigating with Kinect

The Kinect provides a dense 3D point cloud and it can be further processed to generate data for the navigation planning. However, this point cloud contains too much information that can be down-sampled in order to achieve better performance for navigation. Though using all the information can add more accuracy in terms of 3D navigation, compiling it and down sampling it to 2D space might increase the system's performance, while not giving up too much accuracy.

Appendix D

Paper accepted to be presented in ICINCO 2014

Real-Time People Detection and Mapping System for a Mobile Robot using a RGB-D sensor

Francisco F. Sales, David Portugal and Rui P. Rocha
Institute of Systems and Robotics, University of Coimbra, Coimbra, Portugal
{fsales, davidbsp, rprocha}@isr.uc.pt

Keywords: People Detection, Mapping, Mobile Robot, RGB-D Sensor, ROS.

Abstract: In this paper, we present a robotic system capable of mapping indoor, cluttered environments and, simultaneously, detecting people and localizing them with respect to the map, in real-time, using solely a Red-Green-Blue and Depth (RGB-D) sensor, the Microsoft Kinect, mounted on top of a mobile robotic platform running Robot Operating System (ROS). The system projects depth measures in a plane for mapping purposes, using a grid-based Simultaneous Localization and Mapping (SLAM) approach, and pre-processes the sensor's point cloud to lower the computational load of people detection, which is performed using a classical technique based on Histogram of Oriented Gradients (HOG) features, and a linear Support Vector Machine (SVM) classifier. Results show the effectiveness of the approach and the potential to use the Kinect in real world scenarios.

1 INTRODUCTION

One of the main use of robots is to replace humans in unpleasant situations, such as repetitive manufacturing tasks and dangerous environments. In these harsh scenarios, robots are usually mobile and should be able to explore, map and detect people, *e.g.* in the case of Search and Rescue (SaR) missions and after an industrial accident, involving the leakage of toxic substances, they can be used to assist human first responders (Rocha et al., 2013).

Such missions are critical and of extreme importance because their accomplishment might save many lives. As a consequence, human rescue teams are often subject to specialized training. However, they usually face a lack of technological equipment and risk themselves in this process. Thus, Robotics plays a fundamental role by reducing this risk, and can be a great resource to human rescue teams.

Detecting people and mapping the environment are key tasks in Robotics for SaR missions and other applications. Since these environments are usually dangerous, mobile robots must be endowed with appropriate locomotion skills, provide accurate results, and the whole system should be affordable due to the risk taken in such harsh environments.

In this work, we use a RGB-D sensor, the Kinect, on top of a mobile robotic platform running Robot Operating System (ROS) (Quigley et al., 2009), to map the environment and detect victims from visual cues. To do so, we project depth measurements to 2D and run a 2D Simultaneous Localization and Mapping (SLAM) algorithm. At the same time, we pre-process the point cloud and compute 3D clusters that might contain people. Afterwards, we run a HOG-based classifier on a corresponding portion of the coloured image to assess the presence of people. Finally, we associate the obtained map and the detections to localize people in the map. Although we use a Kinect sensor, our approach can be applied with any RGB-D sensor. Note however that, for outdoor scenarios, the Kinect is unusable due to infra-red interference induced by the sun, but if the depth measures are made available by more capable sensors under those conditions, our approach is still applicable.

This paper is organized as follows: Section 2 reviews important related work; Section 3 presents the proposed system; Section 4 describes the experimental setup and validates the mapping and people detection modules; Section 5 presents and discusses the results of the integrated system; and in Section 6 we draw conclusions and suggest future work.

This work has been supported by the CHOPIN research project (PTDC/EEA-CRO/119000/2010) funded by "Fundação para a Ciência e a Tecnologia".

2 RELATED WORK

There has been considerable research on SLAM and people detection with laser range finders (LRFs), stereo cameras and, recently, with RGB-D sensors. Surprisingly, it is not common to integrate both efforts, *i.e.* building a map of the environment, localizing the robot with respect to the map and, simultaneously, identifying people within the environment. A recent approach was proposed in (Soni and Sowmya, 2013). However, in contrast to our approach, it is not built around a single RGB-D sensor, which requires performance-oriented approaches to be able to conduct these tasks in near real-time while being able to obtain sufficiently accurate results.

2.1 Mapping

Most popular 2D SLAM algorithms rely on probabilities to cope with noise and estimation errors. There are some popular implementations based on Kalman Filters and Particle Filters (Dissanayake et al., 2001). An alternative approach is graph-based SLAM. In this case, algorithms use the data to build a graph composed of estimated poses, local maps and their relations, in order to compute a consistent global map. ROS, the robotic framework used on this paper, has already available a set of 2D SLAM algorithms, such as *GMapping*, *HectorSLAM*, *KartoSLAM*, etc.

More recently, 3D mapping has also been intensively studied. However, it often relies on stereo cameras (Konolige and Agrawal, 2008), range scanners (Triebel and Burgard, 2005), (May et al., 2009), or monocular cameras (Clemente et al., 2007), thus requiring heavy computation, including aligning consecutive frames, detecting loop closures, and the globally consistent alignment of all processed frames.

The approach used for frame alignment depends on the data to process. However, the Iterative Closest Point (ICP) algorithm is a popular technique for 3D mapping applications (Droeschel et al., 2009). For stereo cameras, Scale-Invariant Feature Transform (SIFT) features (Lowe, 2004), as well as fast descriptors based on random trees (Michael Calonder et al., 2008) computed for keypoints, such as Features from Accelerated Segment Test (Rosten and Drummond, 2006), are often applied. Also sparse feature points can be aligned over consecutive frames via RANdom SAmple Consensus (RANSAC) (Fischler and Bolles, 1981).

Regarding the loop closure problem, most techniques rely on image matching between keyframes. In graph-based techniques, whenever a loop closure is detected, the correspondence between data frames can be used as a constraint in the pose graph, which represents the spatial relationship between frames. The optimization of these pose graphs originates a globally aligned set of frames. In this context, bundle adjustment (Triggs et al., 2000) simultaneously optimizes the pose graph and a map. Other alternatives have also been explored, such as the g2o framework (Kuemmerle et al., 2011).

With the recent massification of RGB-D sensors, most SLAM approaches were adapted to be used with sensors providing 3D dense depth data. This adaptation was required due to the limitation of the field of view (FoV), usually around 60°, and less precise depth measurements. The first constraint can cause problems in the ICP alignment due to the lack of spatial structure, and only a few approaches have been presented that can deal with this particular issue, *e.g.*, the combination of a time-of-flight camera and a CCD camera makes viable to localize the robot (Prusak et al., 2008).

Recently, with the popularization of RGB-D sensors, an approach was presented which uses sparse keypoint matches between consecutive RGB images as an initialization to the ICP algorithm (Henry et al., 2010). However, it has been concluded through experimentation that expensive ICP is not always required. Still, 3D mapping has clearly shown to require more computational effort than 2D mapping.

2.2 People Detection from Visual Cues

People detection is important for various Robotics applications. Much effort has been put in human-robot interaction for the past few years so that robots can engage and interact with people in a friendly way (Ferreira et al., 2013). Detecting and localizing people is essential before initiating such interaction. However, some of this research has relied solely on 2D visual information provided by cameras (Menezes et al., 2003). Some methods involve statistical training based on local features, such as HOG (Dalal and Triggs, 2005), Edge Orientation Histogram (EOH) (Levi and Weiss, 2004), while other methods involve extracting interest points in the image, such as SIFT features. Recently, with the popularization of 3D sensors, much research has been done on people detection. This is also important for intelligent vehicles to avoid collisions. In this context, there is interesting work, such as (Premebida et al., 2009), (Keller et al., 2011), (Llorca et al., 2012).

Another relevant approach using 3D information was proposed by (Satake and Miura, 2009), wherein depth templates are used to detect the upper human body. In (Bajracharya et al., 2009), a reduction of the point cloud to a 2.5D map is performed to preserve the low computational effort so that detection is based on different 2D features.

Later on, a method that combines both depth information and color images to detect people was introduced (Spinello and Arras, 2011). A HOG-based detector is used to identify human bodies from image data and the Histogram of Oriented Depths (HOD) method is introduced for dense depth data that derives from HOG; and, finally, Combo-HOD probabilistically combines HOG and HOD.

Recently, a method that does not require a Graphics Processing Unit (GPU) implementation and still presents accurate and real-time results was presented (Munaro et al., 2012). The only drawback is that they assume people stand on the ground plane, consequently it does not present accurate results for people that stand considerably above or below that plane, *i.e.* performs poorly for people climbing stairs or sitting behind a table. It processes information from the point cloud by downsampling it. Then, it estimates the ground plane with a RANSAC-based least square method so that it can be removed, thus separating clusters that might contain people. For each of these clusters, a HOG-based people detector is applied to the corresponding part of the RGB image.

2.3 Statement of Contributions

In this work, we aim at providing an insight on performing SLAM and human detection and localization simultaneously, while achieving reliable results and acceptable performance using solely one RGB-D sensor in the mobile robot. Even though much research has been conducted on mapping and people detection with RGB-D sensors, both subjects are not often integrated to assemble a functional system for applications such as SaR missions, where providing rescue teams with a map of the environment and localizing possible victims is of inestimable value.

3 SYSTEM OVERVIEW

As seen in Fig. 1, the proposed system uses a Kinect sensor and comprises two major modules: the People Detection module and the Mapping module. Additionally, it runs under ROS which is the most widely used robotics framework, providing a set of tools, libraries, drivers and other resources that make easier developing robot applications, and provide hardware abstraction (Quigley et al., 2009). The data from the Kinect was retrieved using the OpenNI driver[1] and the driver used for the Pioneer 3-DX mobile robot was ROSARIA[2], both already available in ROS.

3.1 Mapping

Although the Kinect allows to perform RGB-D mapping, our goal is to run a SLAM algorithm along with other tasks, such as people detection, and eventually autonomous exploration.

We opted to project the depth measurements provided by the sensor in the floor plane and simulate a 2D Laser Scan in order to reduce the computational cost. This is represented by the "Depth to LaserScan" block in Fig. 1. It processes the columns of the matrix and creates a vector with the minimum depth value per column, thus originating a vector of 640 distance measures, *i.e.* a 2D scan.

The 2D range measurements are used as an input to the *GMapping* algorithm (Grisetti et al., 2007), already available in ROS, along with odometry information provided by the robot's driver.

This SLAM algorithm was selected for several reasons. Firstly, considering our performance constraints, it does not present a high computational burden. Secondly, the Kinect has a low FoV, which can cause problems in scan matching, therefore the mobile robot's odometry can greatly improve results. Finally, it was shown to be robust in testing and experiments, when compared to other SLAM approaches.

3.2 People Detection

Several people detection algorithms do not take into consideration 3D information, while others use that information to improve results. However, the authors of (Munaro et al., 2012) proposed an algorithm that uses the point cloud generated to lower the computational load of classical people classifiers. Furthermore, ROS provides access to the Point Cloud Library (PCL) (Rusu and Cousins, 2011), which contains algorithms to process 3D data from RGB-D sensors. Therefore, the technical implementation of the algorithm becomes much simplified.

The algorithm firstly processes the point cloud, dividing the space into volumetric pixels (voxels) with an edge length of 0.06m, and reduces the 3D points into a common voxel according to the voxel's centroid. Therefore, we obtain a reduced number of

[1]ROS Wiki - openni_kinect, http://wiki.ros.org/openni_kinect (Accessed: 2014-06-21)

[2]ROS Wiki - ROSARIA, http://wiki.ros.org/ROSARIA (Accessed: 2014-06-21)

points and also a point cloud with approximately constant point density, avoiding its variation with the distance from the sensor.

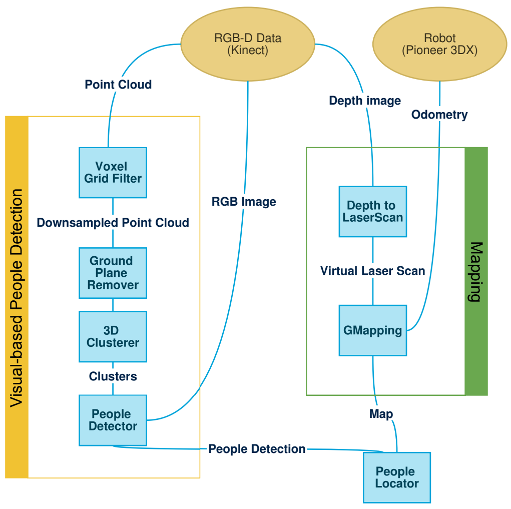


Figure 1: System Overview.

With a filtered point cloud, and considering the assumption that people stand on the ground, the ground plane's coefficients are estimated and updated at every frame using a least square method, therefore it is robust to small changes such as those experienced when a mobile robot is moving. At this stage, points located in the ground plane are removed, by discarding every point located at a distance to the estimated ground plane lower than a threshold of 6cm. As a consequence, the remaining clusters become no longer connected by this common plane.

After this first stage of point cloud processing, the different clusters can now be computed by labelling neighbouring 3D points on the basis of their Euclidean distances. In our case, we started by considering that points closer than a threshold of 2 times the voxel edge belonged to the same cluster. However, this process may lead to errors, *e.g.* dividing partially occluded people into different clusters, or merging different people in the same cluster when they are near each other. As for the second issue, the algorithm uses the position of the heads, that generally are not so close and occluded, to divide these clusters into subclusters, so that people merged previously in a single cluster are separated into different clusters.

For the clusters obtained earlier, a HOG-based detector (Dalal and Triggs, 2005) is applied to the portion of the RGB image corresponding to the fixed aspect ratio bounding box that contains the whole cluster. This process includes the computation of HOG features and their application to a trained linear SVM[3]. The SVM is a learning model that allows us to classify the data based on its training.

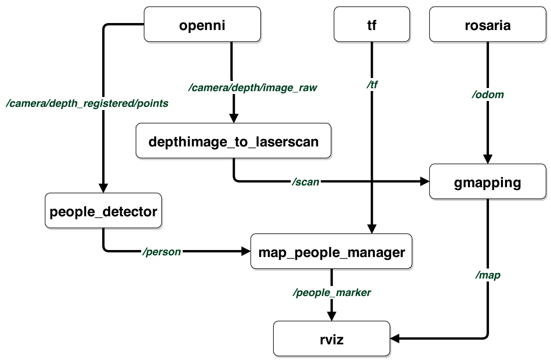


Figure 2: Summarized overview of the system in ROS. Boxes refer to ROS nodes and arcs to topics.

3.3 Integration

Although both modules run in the same system, their data is not in the same reference frame: detections are made on the Kinect frame which is different from the reference frame of the map. To deal with this issue, we created a an additional ROS node (see node *map_people_manager* in Fig. 2) that subscribes to the detections, transforms their coordinates to map coordinates, using ROS tools, and manages the detections, avoiding multiple detections of the same person in the same position. Also, it publishes the corresponding markers to allow the visualization of the map and the detections on the real relative position in the map. The significant portion of the *rqt_graph* is of the ROS system is presented if 2.

4 EXPERIMENTAL SETUP AND VALIDATION

In order to validate the people detection and mapping solution, we used the experimental setup depicted in Fig. 3, with the addition of a laptop on the robot's platform. The Kinect sensor was tilted 8° up so that the operating range for people detection is not affected by the relative position to the ground plane, *i.e.* point clouds will contain the whole person instead of half body at closer distances. The test scenario was indoor and was located in AP4ISR lab of the Institute of Systems and Robotics of the Univ. of Coimbra (ISR-UC). Our experimental work was divided into

[3]The SVM was trained using the well known *INRIA Person Dataset.* (URL: http://pascal.inrialpes.fr/data/human)

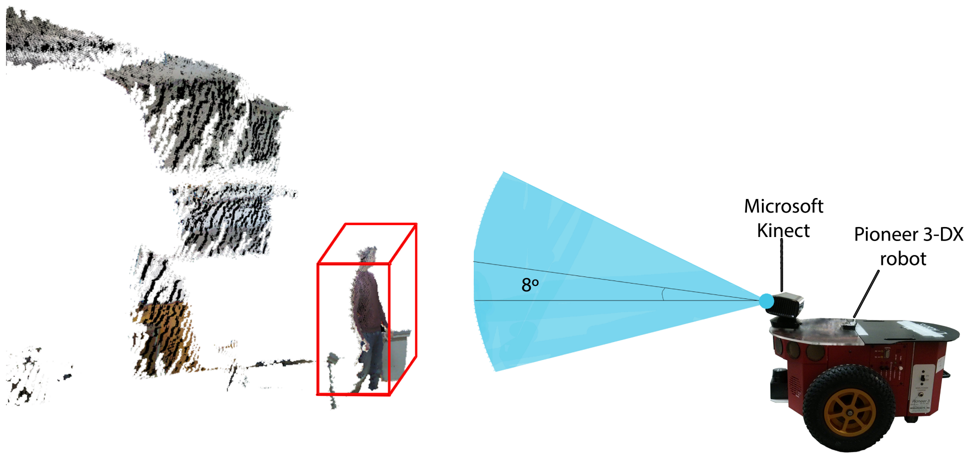


Figure 3: Mobile Robot (Pioneer 3-DX) with a Microsoft Kinect mounted on top.

three stages: mapping method validation, people detection validation, and integrated system validation.

4.1 Mapping Validation

In order to validate the mapping task with the Kinect sensor, we attached to our robot a Hokuyo URG-04LX-UG01 LRF to produce maps to be compared with the ones obtained using the Kinect and the method described in sec. 3.1. The environment tested was a lab arena with approximately 4.6×4.0m, as illustrated in Fig. 4. The robot was teleoperated using an *ssh* remote connection, while running *GMapping*.

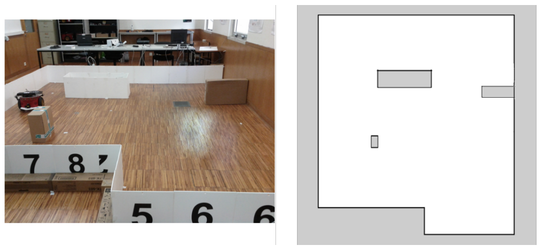


Figure 4: Photo of the test area (left) and ground truth map (right).

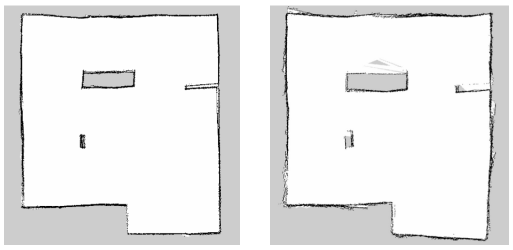


Figure 5: Maps produced with the Hokuyo URG-04LX-UG01 LRF sensor (left) and the Kinect sensor (right).

By visually comparing both maps, in Fig. 5, it becomes evident that the Kinect is not as accurate as the LRF and that its limited FoV, range, and lower accuracy have a negative impact on the results obtained. Nevertheless, both maps are easily interpretable by the human eye. We computed the absolute pixel-wise matching of all pixels in the maps generated to assess their quality, and obtained acceptable matching rates, as shown in Table 1. In order to compute the matching metric, we binarized the maps obtained and the ground truth, calculated the best fit alignment by rotating the maps, and computed the pixel-wise match of each pixel in the image.

Table 1: Pixel-wise matching rates.

Maps	Matching Rate
Ground truth - Laser	96.9 %
Ground truth - Kinect	94.3 %

4.2 People Detection Validation

Despite the availability of some datasets, they do not comply with the constraints and our hardware setup in Fig. 3, mostly because the Kinect is only 24cm above the ground, so it is tilted up to acquire visual information containing people. In order to validate our people detection method, we captured a dataset of about 100 frames that was manually annotated with the people present in each frame. It contains one person walking in several directions at a distance of 1 to 4 meters to the camera frame (an example is shown in Fig. 6). Therefore, we have a dataset of binary decision. This way, we were able to acquire data in similar conditions to the final intended applications.

We applied the people detection method implemented in ROS to process point clouds of the dataset, and extract results (true positives, true negatives, false positive and false negatives) in Receiver Operating

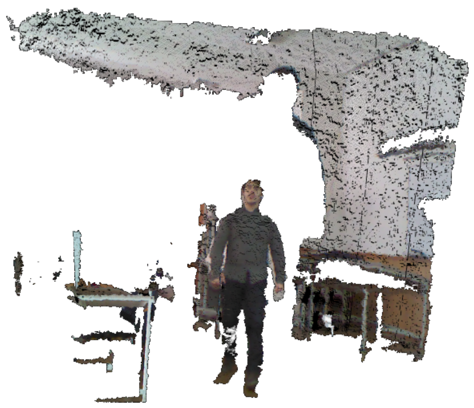


Figure 6: Example of a point cloud from the dataset.

Characteristic (ROC) curves (see Fig. 7 and Fig. 8). The ROC curve is a graphical plot which illustrates the performance of a binary classifier system as the discrimination threshold is varied.

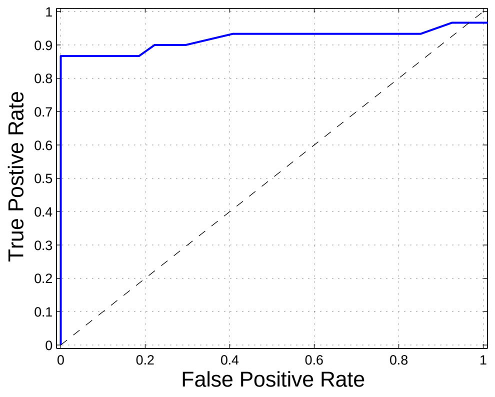


Figure 7: ROC curve on our dataset.

Afterwards, we applied the method to cross validate with the Kinect Tracking Precision (KTP) Dataset from (Munaro et al., 2012), which contains sequences of multiple people captured from a static camera.

Fig. 7 relates the True Positive and False Positive Rates (TPR and FPR). Perfect results are near the top left corner with 0% FPR and 100% TPR. Fig. 8 shows the precision and recall percentages for each experiment. The ideal result is situated on the top right corner with 100% recall and precision. Fig. 9 is the Detection Error Tradeoff (DET) curve which relates the False Rejection Rate (FRR) in percentage and the number of False Positives per Frame (FPPF). The best result is located on the bottom left corner.

We observed in the results obtained with our dataset that the method is very robust in terms of false negatives, showing a low FPR for high enough TPR,

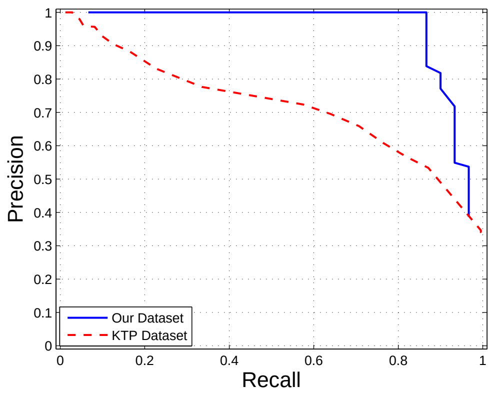


Figure 8: Precision-Recall for our dataset and for the KTP dataset.

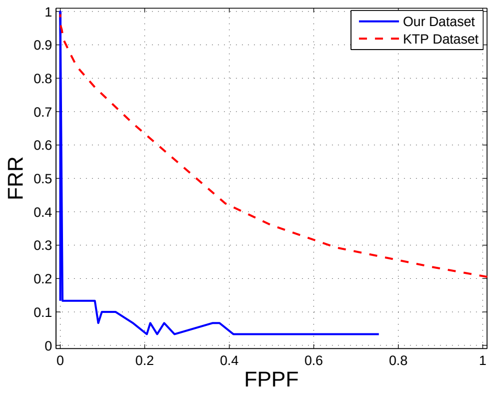


Figure 9: Detection Error Tradeoff (DET) for our dataset and for the KTP dataset.

e.g. 86.87% TPR for ≈ 0.00% FPR. This is also visible in the high precision shown in Fig. 8, even for high recall values, *e.g.* 100% precision for 86.67% recall. We were able to achieve an accuracy of 92.98% which shows the reliability of the method.

Note however that this analysis is performed independently for each processed point cloud (each frame provided by Kinect). In real world applications, with depth data from Kinect at 30 *fps*, we will capture several frames for each person, which allows us to gain certainty when detecting a person in short time intervals. In the case of SaR missions, it is very important to lower the false positives as much as possible to avoid wasting resources and time while keeping a high FPR to be capable of detecting all the victims.

The low number of frames and the presence of only one person is clear in the curves and led us to run the method with the KTP dataset. The results obtained on the KTP dataset were comprehensively not as good as the ones with our dataset, since the former is a more complex dataset containing up to 5 people in the same sequence. Still, the accuracy of the method for a single frame is enough considering the amount of frames available that we can process for detecting each person. We did not compute the ROC curve for this dataset because it aims to assess a binary classification problem and it barely contains binary decisions due to the nature of the tracking problem. Still, we can conclude from the Precision-Recall and DET curves that the results are accurate enough for our intended applications, *e.g.* 72.39% precision for 57.51% recall and 0.39 FPPF for 42.49% FRR.

5 EXPERIMENTAL TESTS WITH A MOBILE ROBOT

To validate the system, we used the previously presented setup in an indoor uncontrolled and cluttered environment in the AP4ISR lab of ISR-UC. The environment contains desks, tables, placards, chairs, hardware and all kinds of objects (see Fig. 10). The systems performed all the computation and also displays the results in real-time.

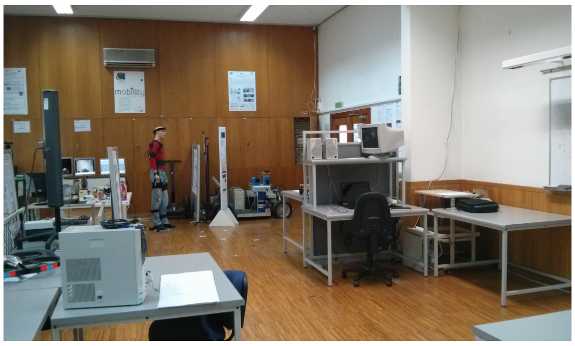


Figure 10: Photo of the environment used to test the whole system.

The environment was reliably mapped (see Fig. 11). We note that the 2D mapping method uses the desks top edges in the mapping so the space under them are not considered due to the point cloud downsampling. On the other hand, for people detection purposes, the whole space is considered, therefore if a detection is made under a table, it would still be represented in the map.

In our experiments, we used three subject standing on different locations, two real humans (one male and one female) and a human model (seen in Fig. 10). In all experiments, the robot was able to map fairly accurately and detect all of them. Fig. 11 shows a picture of the ROS visualization software *rviz* with the ongoing construction of the map and the detections made so far. The experiment depicted lasted 4,3 minutes. The robot was teleoperated with a Wii Remote Control connected via bluetooth to the laptop mounted on top of the robot. The system depicted in Fig. 1 and in Fig. 2 was run on a laptop with an Intel Core i7-4700MQ CPU, 16GB of RAM, Ubuntu 12.04, and ROS Hydro. We computed the average CPU load along the experiment, which resulted in 44.71% of CPU usage and an average of 16.07 *fps* was processed. This frame rate could be increased through the parallelization of the code in a GPU. Also the results demonstrate that the system performs well in real word scenarios and its computational load leaves room to incorporate further modules in the system, such as additional sensory cues, *e.g.* audio input, and perform other tasks in parallel, such as autonomous navigation and exploration.

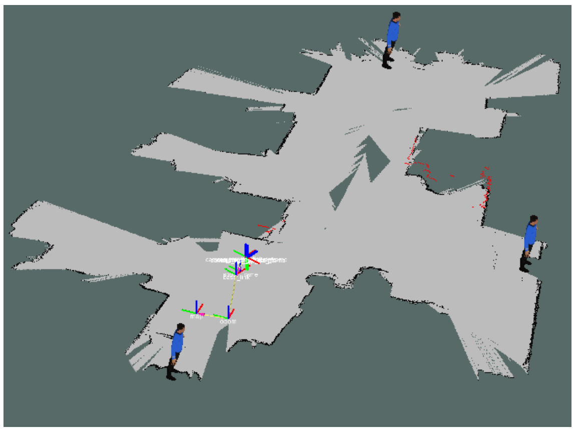


Figure 11: Map obtained and people detected.

6 CONCLUSION

This paper proposed an integrated system that is able to successfully map the environment, localize the robot with respect to the map and, simultaneously, detect and localize people within the environment, while relying solely in a RGB-D sensor. However, we intend in our future work to have a system that is also able to autonomously explore the environment. This will probably require an upgrade of the current hardware setup of Fig. 3, due to the narrow FoV of the Kinect which may cause unreliable navigation. Also, our goal is a system that can be used to perform SaR missions, the robot should be able to navigate towards victims, to eventually interact with them. We intend to study the use of a second Kinect sensor to achieve a wider FoV. This improvement does not imply great costs and should yield a safer navigation.

In the future, we also intend to take advantage of other capabilities of the Kinect sensor, such as processing audio information from its microphone array to improve people detection results. Furthermore, we would like to test the system in other applications such as automated patrolling and surveillance with robotic teams (Portugal and Rocha, 2013).

REFERENCES

Bajracharya, M., Moghaddam, B., Howard, A., Brennan, S., and Matthies, L. H. (2009). A fast stereo-based system for detecting and tracking pedestrians from a moving vehicle. *Int. Journal of Robotics Research*, 28(11-12):1466–1485.

Clemente, L., Davison, A., Reid, I., Neira, J., and Tardós, J. D. (2007). Mapping large loops with a single hand-held camera. *Proc. Robotics: Science and Systems Conf.*

Dalal, N. and Triggs, B. (2005). Histograms of oriented gradients for human detection. *Int. Conf. on Computer Vision & Pattern Recognition*, 2:886–893.

Dissanayake, M. G., Newman, P., Clark, S., Durrant-Whyte, H. F., and Csorba, M. (2001). A solution to the simultaneous localization and map building (SLAM) problem. *Rob. & Automation, IEEE Tr. on*, 17(3):229–241.

Droeschel, D., May, S., Holz, D., Ploeger, P., and Behnke, S. (2009). Robust ego-motion estimation with ToF cameras. *European Conf. on Mobile Robots*, pages 187–192.

Ferreira, J. F., Lobo, J., Bessière, P., Castelo-Branco, M., and Dias, J. (2013). A Bayesian framework for active artificial perception. *IEEE Trans. on Cybernetics (Part B)*, 43(2):699–711.

Fischler, M. A. and Bolles, R. C. (1981). RANdom SAmple Consensus: a paradigm for model fitting with applications to image analysis and automated cartography. *Comm. of the ACM*, 24(6):381–395.

Grisetti, G., Stachniss, C., and Burgard, W. (2007). Improved techniques for grid mapping with Rao-Blackwellized particle filters. *IEEE Trans. on Robotics*, 23:2007.

Henry, P., Krainin, M., Herbst, E., Ren, X., and Fox, D. (2010). RGB-D mapping: Using depth cameras for dense 3D modeling of indoor environments. *Experimental Robotics*, 79:477–491.

Keller, C. G., Enzweiler, M., Rohrbach, M., Fernandez Llorca, D., Schnorr, C., and Gavrila, D. M. (2011). The benefits of dense stereo for pedestrian detection. *ITS, IEEE Trans. on*, 12(4):1096–1106.

Konolige, K. and Agrawal, M. (2008). FrameSLAM: From bundle adjustment to real-time visual mapping. *Robotics, IEEE Trans. on*, 24(5):1066–1077.

Kuemmerle, R., Grisetti, G., Strasdat, H., Konolige, K., and Burgard, W. (2011). g2o: A general framework for graph optimization. *ICRA*, pages 3607–3613.

Levi, K. and Weiss, Y. (2004). Learning object detection from a small number of examples: the importance of good features. *CVPR*, pages 53–60.

Llorca, D., Sotelo, M., Hellín, A., Orellana, A., Gavilan, M., Daza, I., and Lorente, A. (2012). Stereo regions-of-interest selection for pedestrian protection: A survey. *Transportation research part C: emerging technologies*, 25:226–237.

Lowe, D. G. (2004). Distinctive image features from scale-invariant keypoints. *Int. Journal Computer Vision*, 60(2):91–110.

May, S., Droeschel, D., Holz, D., Fuchs, S., Malis, E., Nüchter, A., and Hertzberg, J. (2009). 3D mapping with ToF cameras. *Journal of Field Robotics, sp. issue on 3D Mapping*.

Menezes, P., Brethes, L., Lerasle, F., Danes, P., and Dias, J. (2003). Visual tracking of silhouettes for human-robot interaction. pages 971–976.

Michael Calonder, Vincent Lepetit, and Pascal Fua (2008). Keypoint signatures for fast learning and recognition. *In European Conf. on Computer Vision*.

Munaro, M., Basso, F., and Menegatti, E. (2012). Tracking people within groups with RGB-D data. *IROS*, pages 2101–2107.

Portugal, D. and Rocha, R. P. (2013). Distributed multi-robot patrol: A scalable and fault-tolerant framework. *Robotics & Auton. Syst.*, 61(12):1572–1587.

Premebida, C., Ludwig, O., and Nunes, U. (2009). Lidar and vision-based pedestrian detection system. *Journal of Field Robotics*, 26(9):696–711.

Prusak, A., Melnychuk, O., Roth, H., Schiller, I., and Koch, R. (2008). Pose estimation and map building with a time-of-flight camera for robot navigation. *Int. Journal Intell. Syst. Technol. Appl.*, 5(3/4):355–364.

Quigley, M., Conley, K., Gerkey, B. P., Faust, J., Foote, T., Leibs, J., Wheeler, R., and Ng, A. Y. (2009). ROS: an open-source robot operating system. *ICRA Workshop on Open Source Software*.

Rocha, R. P., Portugal, D., Couceiro, M., Araujo, F., Menezes, P., and Lobo, J. (2013). The CHOPIN project: Cooperation between Human and rObotic teams in catastroPhic INcidents. *SSRR*, pages 1–4.

Rosten, E. and Drummond, T. (2006). Machine learning for high-speed corner detection. *In European Conf. on Computer Vision*, pages 430–443.

Rusu, R. and Cousins, S. (2011). 3D is here: Point Cloud Library (PCL). *ICRA*, pages 1–4.

Satake, J. and Miura, J. (2009). Multiple-Person Tracking for a Mobile Robot Using Stereo. *MVA Conf.*, pages 273–277.

Soni, B. and Sowmya, A. (2013). Victim detection and localisation in an urban disaster site. *ROBIO*, pages 2142–2147.

Spinello, L. and Arras, K. O. (2011). People detection in RGB-D data. *IROS*, pages 3838–3843.

Triebel, R. and Burgard, W. (2005). Improving simultaneous localization and mapping in 3D using global constraints. *AAAI*, 20(3):1330.

Triggs, B., Mclauchlan, P., Hartley, R., and Fitzgibbon, A. (2000). Bundle adjustment – a modern synthesis. *Vision Algorithms: Theory and Practice, LNCS*, pages 298–375.