3D Position Estimation for Mobile Robots

Levente TAMÁS

Automation Department

Technical University of Cluj-Napoca

2020

Contents

List of Figures

List of Tables

Nomenclature

Greek Symbols

δ	Delta overlap error
ε	Threshold
γ	Polar coordinates angle, tilt
ω	Function
Φ	Function
ρ	Polar coordinates angle, pan
σ	Noise covariance
θ	Heading

Subscripts

n	Discrete point index

Other Symbols

$\mathbf{K}$	Camera matrix
$\mathbf{P}$	Projection matrix

R	Rotation matrix
t	Translation vector
X	World coordinates
x	Image coordinates in pixel
p	Probability
r	Range
u	Control

Acronyms

CNN	Convolutional Neural Networks
CVFH	Clustered viewpoint feature histogram
ESF	Ensemble of Shape Functions
FPFH	Fast point feature histogram
IGE	Intensity Gradient Estimator
ISE	Intensity Spin Estimator
NN	Nearest-neighbors
PC	Principal curvature
PFH	Point feature histogram
RIFT	Rotation invariant feature transform
ROC	Receiver Operating Characteristic
RSD	Radius-based Surface Descriptor

SI	Spin images
VFH	Viewpoint feature histogram
GMM	Gaussian Mixture Model
ICP	Iterative Closest Point
IR	Infrared
LIDAR	Light Detection and Ranging
pdf	Probability density function
RANSAC	RANdom SAmple Consensus
ROI	Region of interest
RRT	Rapidly–Exploring Random Tree
SAD	Sum of absolute differences
SF	Sensor Fusion
TOF	Time of flight
VP	Vanishing point

SCIENTIFIC CONSTRIBUTIONS

Chapter 1

Introduction

This chapter introduces the main domain in which the habilitation thesis is enrolled highlighting the motivation of adopted approach. Besides the general overview of the depth sensing and processing techniques, the special focus on position estimation for mobile robots domain is introduced. After this, the contributions of this work are shortly described. Finally the thesis outline and structure is shown.

1.1 Problem Description and Motivation

The 3D perception of the surrounding environment is still an important research field for both the industrial and research community. There are several potential applications for this domain, mainly from the fields of urban surveillance, path planning and cultural heritage conservation. Each individual application requires a specific handling of the acquired data sets. Although certain applications in the mobile robotics domain require real time data processing, e.g. dynamic perception and planning, the post-processing of data is sufficient for our pipeline.

Several sensors can be used for the acquisition of data, such as stereo cameras, laser range finders, or the recent structured light sensors. These devices have their own special characteristics in terms of precision, range and speed. Thus the way in

which these sensors are chosen depends on the specific requirements of the measurement problem to be solved Scharstein & Szeliski (2002).

Relatively large areas, such as indoor spaces for offices, require several different measurements to be aligned in the same coordinate frame. This kind of problem is well studied in the 2D space, mainly in the image processing domain. Although these 2D algorithms can be adopted for the registration of 3D data, they need special adaptations. Also, characteristics such as range, noise, or distribution have a large influence on the algorithms used for 3D data processing, pose estimation and including the registration of point clouds.

1.2 2D-3D Object Recognition and Position Estimation

1.2.1 2D-3D Object Recogntion

Our target application was an object recognition system based on 2D CNNs Krizhevsky *et al.* (2012) and 3D pose estimation. The combination of 2D state-of-the-art systems with 3D features carries great possibilities Lahoud & Ghanem (2017). Clearly, in order to use this technique, not only the RGB images are needed, but also their corresponding depth images. We implemented a system, that does not rely entirely on 2D or 3D features, rather combines them and extracts the most important segments. This system is able to provide good results even when the available computational resources are limited. Since our main goal was to use it on mobile robots, the processing power cannot be compared to a computer's GPU, such as in case of dedicated servers.
We also tested how do different neural network frameworks perform when the same, lightweight dataset is used. The comparison highlights the advantages and drawbacks of each, regarding the domain of object recognition for mobile robot applications.

1.2.2 2D-3D Data based Pose Estimation

There is a considerable research effort invested in autonomous car driving projects both on academic and industrial side. While for the specific scenarios such as highways there are already a number of successful applications, this topic is still generally not solved for complex environments such as the urban ones Geiger *et al.* (2014); Lin *et al.* (2013). The recent developments in the autonomous driving in urban environment using a great variety of close-to-market sensors including different cameras put into the focus the need for information fusion emerging from these sensors Furgale *et al.* (2013).

One of the most challenging issues for the pose estimation is the fusion of information from different cameras.

In this thesis a short overview is given for the patch based 2D-3D data fusion for non-conventional 2D cameras and 3D lidar sensors emerging from different sources in a single step automatic manner based on our previously published papers Tamas & Kato (2013a); Tamas *et al.* (2014). The main contribution is the formulation of the calibration problem as a general 2D-3D non-linear registration problem which works without special targets or established point matches. The registration is accomplished by solving system of nonlinear equations based on the idea of Domokos *et al.* (2012).

1.3 Thesis Outline

This section presents the thesis structure. The chapters are organized in a linear manner including the depth data processing, 3D mapping, pose estimation and their applications in the robotics field.

Each of the chapters covers a theoretical introduction with simulation / experimental results. Thus, Chapter 2 introduces the 3D sensors for the mobile robot applications. The data processing as well as map creation alogrithms are presented.

The Chapter 5 presents the object recognition related subject for depth only data

and for 2D-3D heterogeneous data as well. The real life experimental validation of the proposed methods is presented at the end of this chapter.

Chapter 5 presents the fundamentals of the pose estimation for 2D-3D sensors based on pathces. This is mainly a theoretical presentation, but at the end of the chapter synthetic and real data results are presented for validating the discussed algorithms.

As a final validation of the ideas presented in the earlier parts, experimental results are shown in Chapter 5 from different robotics related applications including flying, dual-arm and mobile robots as well.

Finally the thesis is concluded with Chapter 5 the discussion upon the results of the experiments and the possible future extension of the topics.

Chapter 2

Spatial Data Processing

This chapter introduces the depth sensors for the mobile robots, through what information about the surrounding word is gathered in form of Eucledian coordinates. Nowadays, due to the increasing complexity of the autonomous systems, the greater demand to achieve good accuracy in the position estimation motivates the interest in the 3D sensing technology. The position measurement of the robot can be performed using various sensors returning either absolute or relative position information.

In the first part of the chapter are presented general concepts regarding the 3D sensing and data processing including the design details of a custom 3D scanner used for the applications as well. Further on, 3D map creation related parts are presented including the keypoint-feature extraction and map registration topics. Finally, the chapter is concluded with practical aspects of the presented algortihms Tamas & Goron (2014).

2.1 Introduction

The 3D perception of the surrounding environment is still an important research field for both the industrial and research community. There are several potential

applications for this domain, mainly from the fields of urban surveillance, path planning and cultural heritage conservation. Each individual application requires a specific handling of the acquired data sets. Although certain applications in the mobile robotics domain require real time data processing, e.g. dynamic perception and planning, the post-processing of data is sufficient for our pipeline.

Several sensors can be used for the acquisition of data, such as stereo cameras, laser range finders, or the recent structured light sensors. These devices have their own special characteristics in terms of precision, range and speed. Thus the way in which these sensors are chosen depends on the specific requirements of the measurement problem to be solved Scharstein & Szeliski (2002).

Relatively large areas, such as indoor spaces for offices, require several different measurements to be aligned in the same coordinate frame. This kind of problem is well studied in the 2D space, mainly in the image processing domain. Although these 2D algorithms can be adopted for the registration of 3D data, they need special adaptations. Also, characteristics such as range, noise, or distribution have a large influence on the algorithms used for 3D data processing, including the registration of point clouds.

Different algorithms can be used for iterative map registration, including keypoint and feature extractors, nonlinear correspondence estimators or odometry based approaches Magnusson *et al.* (2007). Although the data merging can be performed based only on the odometry information, this kind of registration is prune to fail due to the error integration characteristics of the odometers Kaushik *et al.* (2009). Hence a more robust method is applied for the initial alignment phase based on an extracted keypoint–feature data set proposed in the work Zhang *et al.* (2008). A similar version of this approach was adopted for the registration stage in the article at hand focusing on the iterative registration without explicitly making use of the loop closure data. Further on, for the different data sets, specific features and their correspondences among them are evaluated.

A certain environment is scanned from multiple viewpoints. These point clouds are then registered using an ICP-based algorithm applied in two stages: (i) initial

alignment: only for filtered set of correspondences, usually a fast process; and (ii) refined alignment: using the complete data sets, being a time consuming but accurate variant. The assumption is that the registered cloud is not axis aligned, thus having a random coordinate system. Since the approach relies on accurate alignment with real-world axes, this method transforms the cloud in two steps: (i) initial guess: using normals of dominant planes to compute the axes; and (ii) correct alignment: where basic features from indoor environments are used to determine the final axes. During this process the planar surfaces are segmented and the boundary points for each plane are computed. Quadrilateral shapes are then fitted to each set of boundaries. These shapes will tell us the positioning of walls and components such as doors and windows. After inspecting the sizes of these rectangle-like shapes and determining the relationships between them, the method can start assigning a class or label to each point.

The main contributions presented in this chapters are as follow:

- development of a robust framework for iterative map registration based on the combination of different keypoint feature pairs from the main literature;
- validation of the iterative registration method on different data sets including indoor and outdoor variants;
- a straightforward and reliable method of estimating the principal axes of 3D complete indoor data sets;
- improved hierarchical model fitting by clustering the models' inliers, and retaining only consistent clusters as final models;
- a simple and reliable set of rules for labeling indoor components, without the use of any training classifiers;
- procedure for estimating quadrilateral-like forms, which are needed for the proposed labeling process of 3D points.

2.2 Related Work

The perception of environments is a current topic for several research works especially in the field of robotics perception Nüchter & Hertzberg (2008); Rusu & Cousins (2011b); Tamas & Goron (2012). Further more, the data registration issues are treated in different works using various approaches, such as planar patch matching Pathak *et al.* (2010b), keypoint-descriptor pairs for the depth information Rusu *et al.* (2010a) or even heterogeneous data fusion including RGB data in the works Kaushik *et al.* (2009); Tamas & Majdik (2012a).

A good example of creating the surrounding environment is given in Kasinski & Skrzypczynski (2001). The authors are concentrating on the distributed mapping of an environment, where the application is an industrial one, e.g. a shared floor in a warehouse. In comparison to ours – which is intended for understanding the human indoor environment, such as kitchens, offices, hospitals – the application presented in Kasinski & Skrzypczynski (2001) is closer to business and enterprise.

Several research work deal with the estimation of the precision of the registered 3D maps. This is valid for both the loop closure based mapping process (also referred as simultaneous mapping and localization) May *et al.* (2009); Strasdat (2012) and the iterative registration based mapping Hervier *et al.* (2012); Magnusson *et al.* (2009) without closuring the loop during the mapping.

In Nüchter & Hertzberg (2008) the authors propose a so-called scene interpretation method for labeling the walls, floor, and ceiling. This approach relies only on the segmentation of planar patches using the RANSAC *(Random Sample Consensus)* Fischler & Bolles (1981) algorithm. Further identification of indoor components is obtained using a rather complex recognition pipeline.

The authors in (Rusu *et al.*, 2008) are presenting a similar approach to the one described in this article, at Section 2.5. They also use 2D quadrilaterals to detect rectangular shapes, but under the assumption that the points are axes aligned (Rusu, 2009). Based on this assumption the reasoning about the environment becomes more accessible. Also, from the article it was understood that the so-called cuboid

fitting was applied only for that particular kitchen data sets.

Regarding to door detection, there has been done some very interesting work, such as (Murillo *et al.*, 2008) where authors use computer vision to classify pixels using a somehow smaller set of classes that the one presented in this work. Although the results look good, the performance can be easily affected by changing light conditions. Also by using only image processing, robotic applications which require 3D information cannot be supported. Another work on door detection is (Morisset *et al.*, 2009) where the developed system is accurately detecting and opening doors. Unfortunately, the authors have applied it only on doors and handles that are conform to ADA *(American Disability Act)* U.S. law, thus suggesting that the system might not comply for other types of doors.

In addition, there was very little to be found on indoor window detection, this application being more common for outdoor processing of buildings. Nevertheless in (Tuttas & Stilla, 2011) a method is presented for detecting windows from outdoor scans, using indoor points. The authors also rely on the detection of dominant planes, being considered most representative. Using the indoor points, meaning the points which lie inside the buildings, and which were obtained by the laser beam going through the windows into the building, the authors estimate the positions of windows.

In Silva *et al.* (2012) the authors give an extensive overview over the latest applications in the area of intelligent homes, such as child and elderly care, surveillance, or even the optimization of energy usage. Although the applications summarized in Silva *et al.* (2012) are relevant to the applied engineering field, robotic applications are underrepresented in this work. We believe that "smart homes" should also incorporate "smart agents", such as personal or service robots, which can interact autonomously with the environment Papadakis (2013).

2.3 3D Scan Preprocessing

Several range scans are necessary in order to build a 3D elevation map of the environment. To use these scans as a coherent data set, they have to be unified in a common coordinate frame. Unless the position and orientation of the mapping robot is accurately known, the range scan registration needs to be done using specialized algorithms Nagatani *et al.* (2009). Since in our case the robot position could not be determined with a sufficient accuracy between the measurement steps, the registration algorithms were employed for creating the elevation maps Tamas & Goron (2012).

The problem addressed in this section deals with an iterative scan registration without explicitly requiring or performing the loop closure. Similar benchmarking and approaches were tested for fusing different additional sensors in the work of Hervier *et al.* (2012), or dealing with normal based information registration presented in Magnusson *et al.* (2009).

2.3.1 Data Acquisition

There are several possibilities to acquire 3D information from the surrounding environment. The measurement methods can be divided into 3 major categories based on applied sensor and sensing technology: stereo vision (with two or more cameras) Morisset *et al.* (2009), active triangulation Ohno *et al.* (2010) and time of flight measurements. One of the most precise time of flight measurement systems is based on the laser scanners; however it is the most expensive one. A cheaper variant is the stereo camera, with less precision in the depth estimations or the structural light approaches Khoshelham & Elberink (2012).

In our experiments the data sets were recorded at normal light conditions for spaces ranges between a few *cm* up till 80*m*. The scanning of the environment with a custom 3D laser scanner mounted on the *P3-AT* mobile robot was performed in a stop-scan-go fashion. The core part of the 3D scanner is a Sick LMS200 planar scanner augmented with an additional mechanical part in order to gain the third

degree of freedom for the data Tamas & Goron (2012). A single scan takes up to 60 seconds depending on the used sensor configuration. The resolution of the used custom 3D scanner was 0.25° in yaw and 0.5° for the tilting while the depth absolute error was less than $1cm$. All the measured data was integrated in the ROS[1] environment where each logging was timestamped for an easier off-line processing Goron *et al.* (2010a).

2.3.1.1 Data Acquisition from Custom 3D Laser Ranger

For a scanner with pitching actuator the 3^{rd} dimension of a point is given from the pitch angle. The coordinates of one 3D point result from the distance to the surface, the yaw angle of the beam, and the pitch angle of the actuated mechanical part. Thus a scanned point can be represented as a tuple of the form $(\rho_i; \theta_i, \gamma_i)$ where ρ_i represents the depth information from the laser scanner and θ_i, γ_i the yaw and pitch measurements. The forward kinematic transformation taken as original coordinate system to the laser base link is given by:

$$p = \begin{pmatrix} x_n \\ y_n \\ z_n \end{pmatrix} = \begin{pmatrix} \cos\gamma & 0 & \sin\gamma \\ 0 & 1 & 0 \\ -\sin\gamma & 0 & \cos\gamma \end{pmatrix} \times \begin{pmatrix} \rho\cos\theta \\ \rho\sin\theta \\ 0 \end{pmatrix} \tag{2.1}$$

where p is a point in the Cartesian space with the coordinates x_n, y_n and z_n.

The key component of the 3D sensor is the 2D commercial laser scanner for which a custom rotary platform was designed. There are several possibilities to rotate the laser scanner, i.e. around the yaw, pitch or roll axis, thus achieving a yawing, pitching or rolling 3D sensor Wulf & Wagner (2003). Each of these three setups has its own advantage and disadvantage. As for mobile robots the most common approach is the pitching scan, which was adopted for the current system. The mechanical design and prototype are presented in Figure 2.1. The design shown has two parts: one fixed containing the driving servo motor (left) and the rotation

[1] http://www.ros.org/

encoder (right); and the mobile rotary on which the Sick LMS200 is placed. The prototype was built using an iron frame both for the fixed and mobile part Goron *et al.* (2010b).

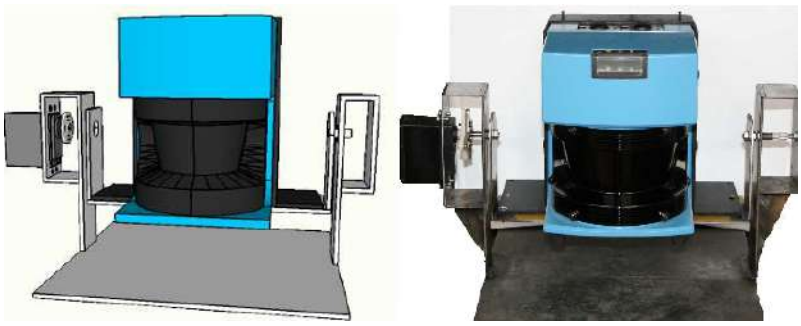


Figure 2.1: The design and prototype of the actuated 3D sensor.

2.3.2 3D Keypoints and Descriptors

There are several possibilities for extracting interest points and descriptors from 2D images including the popular SIFT (Scale Invariant Feature Transform) Lowe (2004a) or the SURF (Speeded Up Robust Features) Bay *et al.* (2008) features. Unfortunately, these rely on local gradients from a unique orientation and therefore are not directly applicable for our approach with 3D data, however some concepts may be inherited from the 2D domain.

In this article the Normal Aligned Radial Feature (NARF) Steder *et al.* (2010) keypoints were adopted for the extraction of interest points from range images. This type of keypoint takes into account the information about the borders and surfaces, ensures the detection from different perspectives and the stability for the descriptor computation. The most important parameter for the NARF extraction is the support size, i.e. the diameter of the sphere in which the interest point characteristics are determined Steder *et al.* (2011). In our case several values for this parameter were tested in order to gain a sufficient number of keypoints for different types of data

sets. After the selection of keypoints was completed, the specific properties are determined, i.e. the descriptors for the set of extracted keypoints.

For our approach we used the optimized version of the FPFH descriptor in order to augment the three dimensional space with pose-invariant local features and also tested the NARF descriptors with Manhattan metrics as fitness score for the same set of keypoints. To compare the two set of descriptors, the runtime (T) in seconds and the initial alignment fitness score (S) was computed for indoor (Id) and outdoor (Od) data sets. The result of the comparison is summarized in the Table 2.1.

Table 2.1: Feature descriptor comparison

$Dataset$	T_{NARF}	T_{FPFH}	S_{NARF}	S_{FPFH}
$Id_{cluttered}$	0.19	45	0.071	0.032
Id_{plane}	0.12	12	0.094	0.057
Od	0.11	26	0.083	0.044

The tests were performed on data sets containing around $10K$ points for which the extracted number of keypoints was in the magnitude of $0.1K$. For computing the runtime the average values were considered for 10 batch runs on an Intel Pentium 4 single core laptop running Ubuntu Linux. Although the run-time of the proposed algorithm is higher than some custom scenario based approaches such as the one presented in the work Kaushik *et al.* (2009), the degree of generality of the current approach is higher.

As observed, NARF descriptors are computed with several orders of magnitude faster than FPFH descriptors, but the latter approach is more robust in terms of estimating correspondences. This would be also the case for scenes which present less clutter or variation, thus having less discriminative features, where the FPFH features ensured a better correspondence between points.

2.3.3 3D Features for Stereo Images

2.3.3.1 Principle of Stereo Image Depth Extraction

The stereo cameras are popular tools for creating 3D coloured data based on two or more images recorded from different camera positions. In order to recover depth information from 2D images correspondences have to be established between pixels of the left and right image of the stereo pair, representing the same object in the scene.

Numerous algorithms are focusing on finding correspondences [1]. Generally speaking to obtain better precision more complex algorithms are applied resulting in a higher computational effort.

In many applications there is no need of a dense reconstruction for all the pixels in the image. In these cases is sufficient to use only some points, namely to apply image feature detectors Majdik *et al.* (2010). Image features capture relevant distinctive information in images that can be repeatable detected under different lighting conditions, angle of view and scale increasing thus the robustness of the algorithm.

One of the fastest methods to compute dense disparity maps form stereo images is based on correlation analyses. Using this method the depth is computed at each pixel, a grey level around the pixel in the left image is correlated with the corresponding pixel in the right image. The disparity of the best match from the correspondence is determined using the Sum of Absolute Differences (SAD).

Image feature descriptors can be also used in order to generate dense depth maps. In such a scenario for every pixel in the image a complete descriptor is computed. Afterwards the corresponding pixel is searched in the other image of the pair by comparing the feature descriptors.

[1] Two-frame stereo algorithms: http://vision.middlebury.edu/stereo/

2.3.3.2 Enhanced Stereo Image Algorithm

From the numerous image feature descriptors presented in the literature Daisy Tola *et al.* (2008) was chosen in order to compute the 3D point clouds presented in Figure 2.3. The Daisy descriptor has several advantages over other popular methods such as: SIFT Lowe (2004a) or SURF Bay *et al.* (2008). It can be computed very rapidly for all the pixel of an image. In our experiments for 640x480 pixels the computational time was around 3.2s using a standard PC. Also the Daisy descriptor has better precision than other image feature descriptors according to the literature Tola & Lepetit (2010).

The processing pipeline consist of several phases as follows: (1) capture the stereo image pair; (2) rectification and aligning of the images; (3) compute the Daisy descriptor for every pixel of the left, respectively right image; (4) for every pixel of the right image search for the corresponding one in the left image, the search is performed on the same line from the minimum disparity (10 in our experiments) to the maximum disparity (respectively 40); (5) select the best match by comparing the descriptors, resulting the disparity (for some pixels the disparity could not be found); (6) using the stereo geometry of the vision system and triangulation compute the real world XYZ coordinate for the pixels; (7) get the RGB colour and save the point cloud shown in Figure 2.3.

2.3.3.3 Robust Coloured Point Cloud Features Correspondence

For the RGB-D datasets SIFT3D type keypoints were considered, while for the local descriptors both the NARF and FPFH variants were tested. As in the case of the laser range data, the FPFH proved to be more robust to the rotation type transformations. For the translations the performance of the two type of descriptors were similar.

The output of the feature estimation and the correspondence estimation for the SAD stereo algorithm is shown in the Figure 2.2. As it can be seen, the correspondences are valid for this type of algorithm, although the quality of the 3D point

cloud is rather noisy.

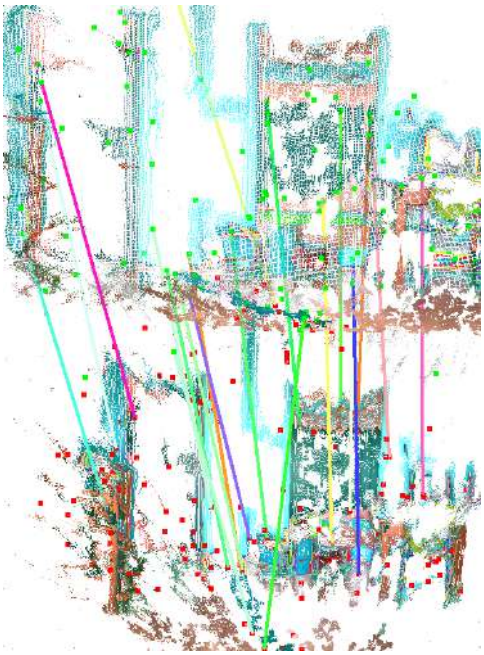


Figure 2.2: Feature matching for coloured point cloud from stereo camera with SAD algorithm

The results of the feature estimation for the Daisy stereo image processing is shown in Figure 2.3. For this case, the generated 3D coloured point cloud has less distortions, although it has depth discontinuousness for ranges above $2m$. Also in this case the keypoint-descriptor pairs have less false correspondences as in the case of the SAD stereo imaging algorithm.

2.3.4 3D Features from Structured Light Camera Images

2.3.4.1 Principle of Depth Measurement via Triangulation

The Kinect RGB-D camera captures infrared and RGB data from the same scenario. It has also an infrared emitter, which is used for structural light projection on the

Figure 2.3: Feature matching for coloured point cloud from stereo camera with Daisy algorithm

scene as this is described by the inventors in Freedman (2010).

The projected pattern is captured by the infrared camera and is spatial correlated with the reference one. The effective distance measurement is done on the principle of computing the distance between the reference points from the emitter and the ones received at the infrared sensor via a triangulation using the known baseline distance between the emitter and the receiver.

Further on the depth image and the RGB data can be fused by using a calibration based on the fixed parameters of the camera. The main error sources for this device are related to the lighting conditions of the measured scene, as for strong lighting (e.g. sun) the projected infrared pattern have low contrast. This is also the main reason for the indoor usage limitation of this type of sensor.

2.3.4.2 Feature Correspondence Filtering for Kinect RGB-D Data

We tested different keypoint-descriptors for this type of RGB-D data including NARF, SIFT3D and FPFH ones. The best computational effort and descriptor performance was obtained for the SIFT3D keypoint and FPFH feature descriptor pair. For this type of setup the result of the algorithm is shown in Figure 2.4, which shows two recordings which were taken from the same scenario with the second scene rotated with 30 deg to the base one.

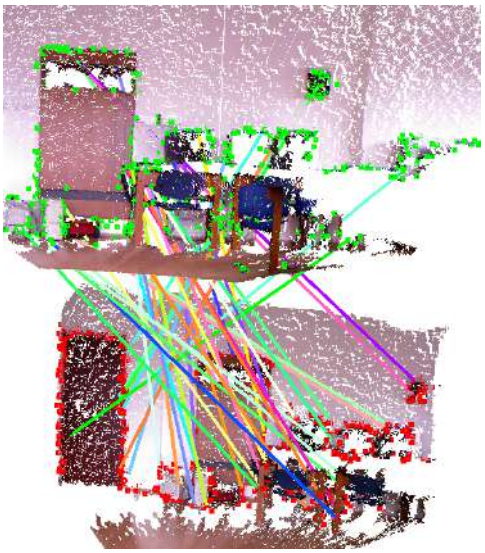


Figure 2.4: Feature matching for coloured point cloud from structured light camera, unfiltered correspondences.

Further on, the SAC based filtering of the correspondences gives a reduced set of feature pairs as this is visible in Figure 2.5. The original correspondence pairs were reduced from 56 to 11. This was done by computing both the back and forth correspondences between the source and target datasets.

For both cases the original RGB-D point cloud was filtered by using a voxel-grid filter in order to reduce the measurement noise in the data and to have a more compact dataset Rusu & Cousins (2011c).

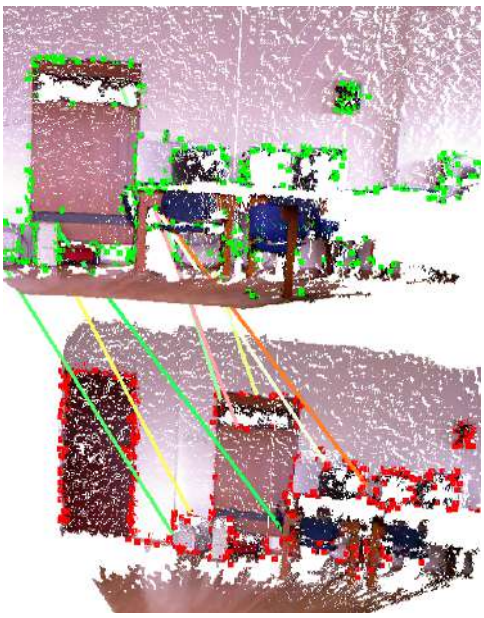


Figure 2.5: Feature matching for coloured point cloud from structured light camera, filtered correspondences

2.4 3D Data Registration

2.4.1 ICP-Based Registration

The registration problem can also be viewed as the optimization of a cost function describing the quality of alignment between different scans. The algorithm determines the rigid transformation which minimizes this cost function Surmann *et al.* (2001). The type of algorithm applied for the frame alignment strongly depends on the measured data set type. For the 3D laser scans the Iterative Closest Point (ICP) and derivatives are popular in the field of robotics Besl & McKay (1992); Nuechter (2009); Pathak *et al.* (2010b). The *ICP* computes the rigid transformation which minimizes the distance among two point sets by associating a point from one frame to the closest point in the target frame. The transformation between two in-

dependently acquired sets of 3D points consists of two components, a rotation $\mathbf{R}$ and translation $\mathbf{t}$. Correspondence points are iteratively searched from the model set of points M (with $|M| = N_m$) in the data set D (with $|D| = N_d$). In case of a valid correspondence we need the transformations $\mathbf{R}$ and $\mathbf{t}$ which minimize the distance between the two points as follows:

$$E(\mathbf{R},\mathbf{t}) = \sum_{i=1}^{N_m}\sum_{j=1}^{N_d} w_{i,j}||\mathbf{m}_i - (\mathbf{R}\mathbf{d}_j + \mathbf{t})||^2 \tag{2.2}$$

where $w_{i,j}$ is assigned 1 if a valid correspondence is found between the ith point from M denoted with $\mathbf{m}_i$ and the jth point from D denoted with $\mathbf{d}_j$.

Different variants were developed in order to increase the robustness and the performance of the algorithm especially for computing the rotational transformation term, which introduces a non-linear term in the minimization problem. A comprehensive overview and qualitative evaluation of different approaches for the registration problem can be found in Salvi *et al.* (2007).

A common approach for boosting the *ICP* robustness is the augmentation of the points with additional features such as point color, geometric features or point histograms Sharp *et al.* (2002). This transposes the optimization problem in a higher order dimensional space search. These features are usually computed only for a certain subset of interest points from the original point cloud, i.e. keypoints in order to reduce the computational effort and enhance robustness.

The use of keypoints is to enable the efficient comparison between different data regions. Our approach for the data registration is based on the correspondence estimation for the extracted keypoint features.

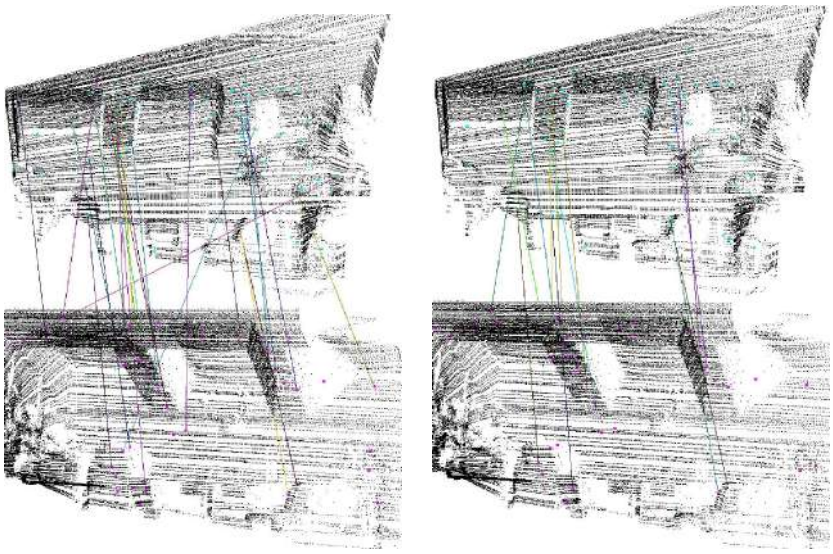
2.4.2 Correspondence Estimation

The next step after determining the keypoints and the descriptors is the estimation of correspondences between the two sets of keypoints with descriptors. There are several methods for the correspondence estimation, such as one-to-one, back and

forth or sample consensus based Pathak *et al.* (2010b).

In our approach the correspondence estimation was performed based on the geometric constrains of the selected points. Thus the nearest point in the high dimensional descriptor space was searched by using a *kd-tree* for enhancing the search speed Bentley (1975). Unfortunately, the brute force search does not ensure a coherent result for the correspondence estimation problem having in many cases a large number of false positives.

For improving the estimation results, the filtering based on sample consensus was adopted. This ensures that after performing the one-to-one search for descriptors, only those correspondences are kept which satisfy a geometrical transformation constrain.



(a) (b)

Figure 2.6: Initial correspondences (a) and filtered correspondences (b).

The comparison of the unfiltered and filtered set of correspondences is shown in Figure 2.6 on an indoor data set. This data set contains two scenes, the original one and the one rotated with 45°. As it can be observed, the initial, unfiltered set of correspondences contains a large number of false positives, which are eliminated,

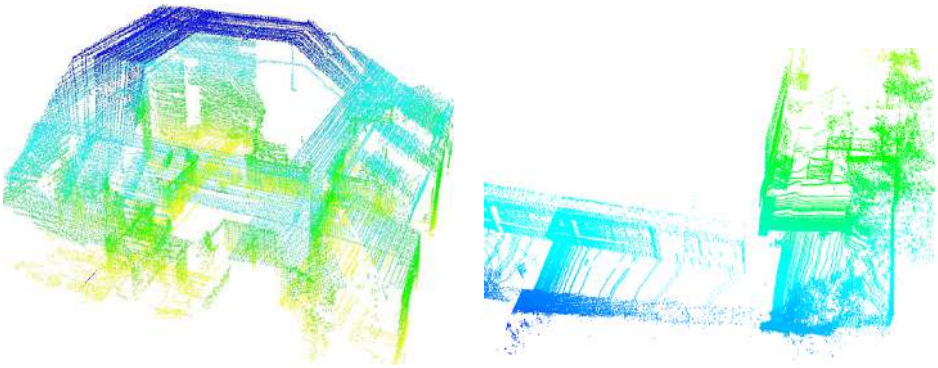
yielding a more consistent estimation. The number of final correspondences depend on the parameters used as a threshold for the sample consensus rejection.

The complete ICP-based algorithm can be found in the works Besl & McKay (1992); Zhang (1992), therefore only a short overview is given, with emphasis on the additional descriptor information for the points. The algorithm has two input point clouds, P_s for the source and P_t for the target. Step 1 and 2 extract the FPFH of the source and target clouds (these two steps can be substituted with arbitrary point cloud feature search), while in Step 3 the initial alignment t^* is determined after the correspondence filtering. In the while statement in each iteration a set of associations A_d is taken for which the best transformation is determined. The loop exit conditions are related to the error variation or to the maximum number of iterations both specified as tuning parameters for the algorithm. Finally, the algorithm returns the computed transformation between the two data sets. Further details regarding the implementation of the *ICP* with initial alignment based on sample consensus can be found in Morisset *et al.* (2009).

The aligned map for the indoor and outdoor environment is presented in Figure 2.7. In both cases the registration was performed using a pair alignment approach and the FPFH descriptors for the computed NARF keypoints. The initial alignment of the scans was performed based on the filtered correspondences of the FPFH descriptors. This alignment was then used for the *ICP* refinement, computed on the last pair of data in the alignment loop.

For the presented scenarios the error convergence of the *ICP* algorithm was monotonically decreasing, a suitable registration error was achieved in less than 100 iterations. This scenario was obtained by considering the maximum distance between two neighbor points to be less than $1m$.

For evaluating the different ranging devices, the data captured needs to be transformed in a common coordinate frame. In our case the mobile robot platform on which the sensors were mounted was considered to be the origin of this common coordinate frame. With this assumption, the change in the orientation and position of the platform needs to be estimated from the different correspondences.



(a) (b)

Figure 2.7: Example of registered indoor map(a) and outdoor registered map(b)

For the correspondence evaluation metric first a rigid transformation based on the correspondences between two different point clouds is determined. This transformation is compared against the reference transformation between the two measurement coordinates. The reference point cloud is determined via the fine tuning of the 3D point cloud from the laser range finder, as this offers a precision under $1cm$ in an indoor environment. The fine tuning is done by applying a iterative closest point (ICP) algorithm to the initially aligned laser point cloud based on the information from the correspondence estimation Besl & McKay (1992).

By taking the normalized point cloud distance between the different rigid transformation estimation, the evaluation of the different feature correspondence estimation is performed. According to this idea, evaluation of the heterogeneous feature sets were tested, by means of fusing the range feature set with the kinect and the ones extracted from the stereo imaging. Also other combinations of the feature set are proposed within this framework. In order to reduce the false correspondences, a sample consensus based filtering is applied to the heterogeneous feature set Pathak *et al.* (2010a).

The reference transformation is computed based on a FPFH type descriptor for the laser data recorded at two different positions as this is listed in the Algorithm

2.1. First the F_s and F_t descriptors are computed for the source and target datasets, which are used for the initial alignment guess for the ICP. The ICP based registration with a sufficiently good initial guess serves a high precision registration for a laser ranger 3D data set. The final ICP transformation t_{ref} is returned as the reference transformation for the feature based correspondence rigid transformation estimation Tamas & Majdik (2012b).

Algorithm 2.1 ICP with initial alignment for reference

Input: L_s, L_t

1: F_s = ComputeFPFH(L_s);
2: F_t = ComputeFPFH(L_t);
3: $(t_{init}, C_f) = InitialAligment(F_s, F_t)$;
4: **while** $(error_{diff} < \varepsilon) or (iteration < iteration_{max})$ **do**
5: $\quad A_d = getClosestPoints(t_{ref}, L_s, L_t)$;
6: $\quad t_{ref} = argmin\left(\frac{1}{|A_d|}\sum_{j\in A_d} w_j \left|t(l_s) - l_t\right|^2\right)$;
7: **end while**
8: **return** t_{ref}

The proposed feature evaluation algorithm is presented in Algorithm 2.2. In the first part of this algorithm the 3D features descriptor D_s and D_t are computed for both input point clouds. In the next step the correspondences are evaluated and filtered out based on a sample consensus framework. Finally, the normalized distance E_d for the estimated transformation t_c and the reference t_{ref} is determined.

Algorithm 2.2 Correspondence estimation

Input: P_s, P_t, t_{ref}

1: D_s = Compute3DFeatures(P_s);
2: D_t = Compute3DFeatures(P_t);
3: (t_c, C_f) = SACFilterCorrespondences(D_s, D_t);
4: $E_d = \frac{1}{|C_f|}\sum_{j\in C_f} w_j \left|t_{ref}(p_s) - t_c(p_s)\right|^2$;
5: **return** E_d

2.4.3 Performance Evaluation on Lidar data

The registration performance evaluation problem is an actual topic for the robotics community. Several recent works are focusing on this problem presenting different benchmarking variants including one iterative registration evaluation Censi (2007); Magnusson *et al.* (2009) or the popular global optimization with loop closure Huitl *et al.* (2012); Kümmerle *et al.* (2009); Strasdat (2012). The covariance estimation for the registration is also an important aspect which gives an information about the confidence of the registration result as this is highlighted in the work of Hervier et al. Hervier *et al.* (2012).

In order to characterize the quality of the registration beside the registered error the covariance of the estimate is also important for mapping purposes. The ICP can be seen also as a least-square estimator which has as error bound the Cramer-Rao bound defined as $N = cov(\hat{x}) = [I(x)]^{-1}$ where $\hat{x}$ represents the estimate of x and the $I(x)$ denotes the Fisher information matrix. The later can be rewritten for as

$$I(x) = \frac{1}{\sigma^2}\left[\sum_i \begin{pmatrix} -A_i^2 & A_i \\ -A_i & I_3 \end{pmatrix}\right]^{-1} \tag{2.3}$$

where σ denotes the sensor noise and A_i is the skew matrix composed from the point coordinates of the i^{th} point as proposed in Hervier *et al.* (2012). This metric is also useful, as the singularities of the skew matrix can give a hint of the axes on which the unobservability condition may arise (e.g. a straight corridor).

As the ground truth is often peculiar to obtain, in order to evaluate the performances of the proposed methods a public dataset was considered as a reference data the Jacobs University campus outdoor data Elseberg *et al.* (2012) with manual marked ground truth data. As their measurement was also based on lidar the covariance of the sensor readings was considered $\sigma = 1cm$. For benchmarking purposes as a stopping criterion the iteration number of the ICP algorithm was considered $n_{iter} = 100$, usually this offering sufficient good accuracy for the registration in a few seconds runtime. In order to compare the performances of the different

keypoint-feature based pre-aligned scans as output the absolute translation error and the covariance defined with (2.3) was considered.

The summary of the evaluation run for different pairs of scans from the public dataset are summarized in the Table 2.2 for the NARF, FPFH and the combined set of keypoints used for initialization:

Table 2.2: Evaluation of the registration on public dataset.

	t_x	t_y	t_z
ICP_{NARF}	2.70 ± 2.96	2.21 ± 2.74	3.9 ± 4.18
ICP_{FPFH}	1.32 ± 3.08	1.59 ± 2.13	2.71 ± 4.06
$ICP_{combined}$	1.21 ± 2.59	1.16 ± 2.97	1.82 ± 3.25

The comparison also in this case reflects the superior results of the initial aligned with the FPFH keypoints over the NARF ones, while the proposed combined variant of these keypoints give the best results. This is due to the different characteristics of the two set of keypoints as well as the larger number, hence more robust initial alignment with the combined keypoint set. Also it is important to observe that the absolute errors computed on different axes tend to have significant differences. This is mainly due to the asymmetrical displacement of the coordinate frames with respect to the axes, i.e. the largest displacement is towards the z ax on which the largest errors were measured as well.

2.4.4 Heterogeneous Feature Evaluation

The main idea of the heterogeneous feature evaluation relies in the simultaneous use of different type of descriptors from different type of data sets, i.e. stereo camera, Kinect or laser. The motivation behind the use of different feature descriptors is the enhancement of the robustness for the initial alignment of the rigid transformation estimation. Also in the case of the stereo imaging - Kinect combination, ensures a larger range of operation, beyond the limits of the Kinect sensor.

In our test scenario, we used the different stereo imaging algorithms and the kinect data for comparing them against the ground truth laser dataset. Four type of descriptor sets were considered: stereo from SAD, stereo from Daisy, the combination of SAD stereo with the Kinect and the combination of Daisy stereo with kinect. In the case of the RGB-D data, SIFT3D keypoints were considered. These showed to be more powerful than the NARF ones for the colored 3D data.

In Table 2.3 the results of the heterogeneous keypoint-descriptor pairs are shown. The initial estimate for the SAD stereo algorithm is compared to the Daisy one and to the combined SAD-Kinect, Daisy-Kinect sets. The Daisy correspondences returned a better transformation than the SAD, while the SAD-Kinect combination over performs both stereo algorithms. The best performance was obtained for the Daisy-Kinect pair in terms of differences from the reference point cloud transformation obtained from the 3D laser scanner considered as reference in our test cases.

Table 2.3: Heterogeneous 3D feature descriptor performance comparison

Name	*SAD*	*Daisy*	$SAD-Kinect$	$Daisy-Kinect$
Error	0.13	0.10	0.094	0.081

The tests were performed on datasets containing around 10^4 points for which the extracted number of keypoints was in the magnitude of 10^2. For computing the runtime the average values were considered for 10 consecutive runs on a $P4$ single core laptop running Ubuntu.

The test datasets were recorded indoor at natural lighting conditions, the transformation between the source and target sets being a rotation plus translation type transformation.

2.5 Indoor Environment Classification

This section describes an integrated method for classifying basic components of human indoor environments based on their physical features. To be more concise,

the system records 3D point clouds and after processing assigns a label to each point according to the class it belongs to. A P3-AT *(Pioneer 3 All Terrain)* and PR2 *(Personal Robot 2)* robot are used for achieving this task. In the following, a comprehensive overview of the proposed method is given.

2.5.1 Labeling Process

A simple yet robust classification process was implemented, which can be easily extended with additional component classes if necessary. The goal of the system was not to rely on any training process but rather use common sense knowledge to label 3D points. To give some examples of what we understand as common sense about indoor human environments, please look into Table 2.4.

Table 2.4: Rules for labeling point of indoor environments.

Criterion	Description
(i)	in the majority of cases the walls inside houses and buildings form rectangular shapes
(ii)	by considering also the floors and ceilings, cuboid shapes are obtained
(iii)	doors are always positioned on the edge of a wall, very close to the floor
(iv)	and if closed, a door is always parallel to its supporting wall; the same for windows
(v)	windows are always located higher from the floor, and closer to the ceiling
(vi)	furniture pieces are usually placed right against the walls, being also aligned accordingly

For the time being, the classification procedure uses ten classes, which are also called labels: (1) *floor*; (2) *ceiling*; (3) *wall*; (4) *door frame*; (5) *door*; (6) *window frame*; (7) *window*; (8) *furniture*; (9) *handle*; and of course (10) *unknown*. It might seem redundant to have such similar classes, e.g. (4) with (5), and (6) with (7),

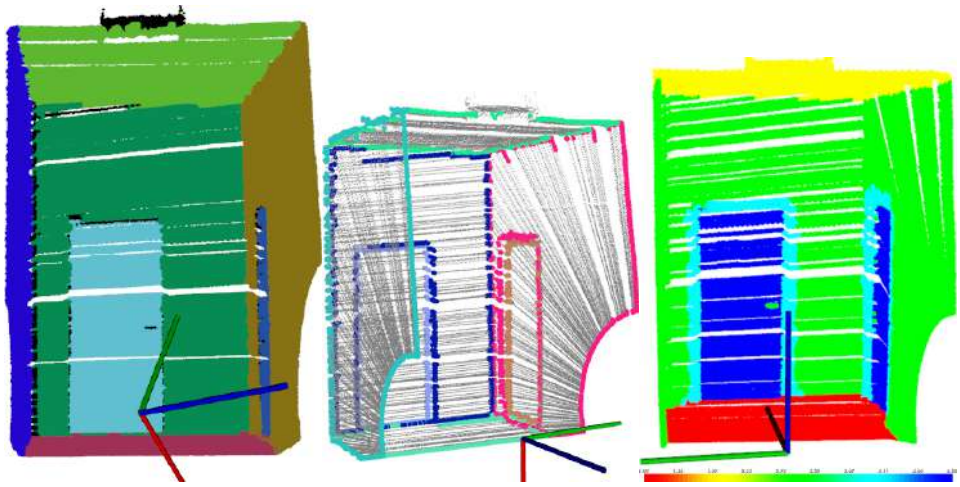
but some components are not rigid and are constantly manipulated. Therefore, it is useful to have a system which can determine the places of those components. The method might not detect the actual door or window, but will identify the location where it should be when closed.

2.5.2 Principal Axes Estimation

In this subsection, a technique is described for estimating the principal axes of indoor point cloud data described in details Tamas & Goron (2014). First, planes are fit using the RANSAC (Fischler & Bolles, 1981) algorithm, by segmenting the inliers of each plane out of the point cloud. For an example of the final fitted planes, please look at Figure 2.8. This is repeated until the remaining points in the cloud fall under a certain threshold. The mentioned threshold is set to an empirical value of 2500 point inliers.

As expected, RANSAC always finds the dominant planes first, meaning the planes with the biggest number of inliers. Also in many indoor environments most of the surface normals coincide with one of the three main axes of the room. This is because most of the walls and furniture surfaces are parallel and/or perpendicular to each other. Therefore, the idea behind estimating the principal axes is to find three planes, which can form a right angle between each others normals. The first step is to compare the normal of the most significant plane with the normal of the second most significant plane and so on. When a match is found, those plane normals are marked as axes for the new coordinate system. After the three planes and their corresponding normals are found, the point cloud is transformed into the new coordinate system. This was the initial guess stage of the estimation procedure.

Although now the points are aligned with the real-world Cartesian system of coordinates, the orientation of the three vectors representing the axes are most probably incorrect, as it can be seen in Figure 2.8. But the method will correct the axes in the second step, which is described in the following subsection. Also, it is important to mention that usually only two normal planes which are at a right angle



(a) (b) (c)

Figure 2.8: Significant stages for the interpretation of indoor environments, where axes X, Y, and Z, are colored in *red*, *green*, and *blue*, respectively: (a) segmentation of planar surfaces, where it can be observed that all major planes are segmented correctly, including the two doors; the coordinate system is not correct, thus having the axes randomly tilted in $3D$ space; (b) axes are aligned with the real-world, but the coordinate system is still not right, i.e. the Z-axis should point upwards, and not towards the reader; computed boundary points for each planar surface are also presented; (c) labeling results of the floor (*red*), ceiling (*yellow*), walls (*green*), door frames (*cyan*), and doors (*blue*), plus the corresponding color scaling; axes are aligned as in the real-world environment.

are needed, since the cross product between the two mentioned normals returns the third axis.

2.5.3 Quadrilateral Detection

A strategy is here detailed for detecting quadrilateral-like shapes and explain the reasoning behind the classification process used.

The work is based on the idea presented in (Rusu *et al.*, 2008) although in this work cuboid fitting was based on four extracted line features while in the current

paper this requirement is relaxed and only three lines are considered for this purpose. This allows the fitting to be performed on lower density or sparser data too. This is especially useful for experiments in non-laboratory conditions or with noisy data such as in our outdoor measurements.

Another major difference between the method presented in (Rusu *et al.*, 2008) and the current one is the scan area that is considered for classification. While in our method a whole room is considered, in the original method only a single scan with limited angle of view is used as data source. Also the number of datasets on which the experiments were performed make the qualitative comparison difficult, as in the original work only a single scenario is analyzed while in this paper more than ten scenes are considered.

The method uses these quadrilaterals to classify points which belong to doors, windows, or frames, based on their physical sizes and positioning inside the scene. By using this approach, the classification can be effortlessly extended for other rectangular-shaped components, e.g. furniture pieces, radiators, or trash bins. But in order to perform quadrilateral fitting, the method would need to execute these steps:

1. compute boundary points for each segmented plane previously found;
2. detect line models inside the sets of boundaries using the RANSAC algorithm;
3. analyze line segments to find candidates which can form rectangular-like shapes.

The planar boundary points are estimated by using the PCL[1] *(Point Cloud Library)* project. For visualizing the boundaries please take a look at Figure 2.8, where each plane has its boundaries colored differently. Then the method continues by fitting lines much similar to the plane segmentation routine presented in the previous subsection. Whereas here the threshold for lines is set to 25 point inliers, which is also a value deduced empirically.

[1] `http://pointclouds.org/`

Quadrilateral fitting is performed by comparing lines between each other to determine if formations of rectangular-like shapes are possible. Finding quadrilateral-like forms which have right angles is done in an iterative fashion. Each iteration, a new line segment is checked and if it satisfies certain conditions, it is added to a quadrilateral configuration as described in Tamas & Goron (2014). A line is added to a current shape, i.e. rectangular, if two conditions are fulfilled:

(a) at least one, if not both, of the segment's ends should be in the proximity to either one of the shape's ends;

(b) the angle between the newly added segment and the existing one, is of approximately 90°, give or take 5°.

A quadrilateral configuration is obtained when a maximum of four line segments are found making up a rectangular shape. On the other hand, shapes which have less than three line segments are rejected. This routine stops when there are no more line segments to be analyzed. Naturally, the results of this routine are influenced by the point cloud density, hence more points result in a better accuracy.

After finding the 2D rectangular shapes, the method checks their sizes, to see if there are any doors or windows. Usually, the doors encountered so far were around 2000*mm* by 800*mm*, and windows around 1200*mm* by 600*mm*, whereas walls have a height of around 3000*mm*. Also, this is not the only criterion by which rectangles are classified as doors or windows. There is also the positioning of those rectangles in their supporting quadrilaterals, i.e. usually walls. Therefore, the method checks also for the relationships between rectangles, i.e. the relative position in comparison with one another. With the help of this information, the correct alignment of the coordinate system – please see Figure 2.8 – can be detected.

As an example, think of a smaller rectangle, which is very close to one of the edges of a bigger rectangle. If that smaller rectangle within the bigger rectangle has a size close to the system's thresholds, then it can be asserted that the smaller rectangle is a door, which lies on the floor, and that the Z-axis should be oriented upwards in the door's direction.

If the normal of the door is then considered, as for example Y-axis, which is at a right angle with the newly found Z-axis, the cross product can be computed and the X-axis is obtained. The point clouds is transformed according to the new axes, and the correction of alignment has been fulfilled. Thus having the real-world coordinate system, the floor, ceiling, and walls can be determined, as shown in Figure 2.8, by using a pass through filter along the Z-axis.

2.5.4 Detection of Furniture

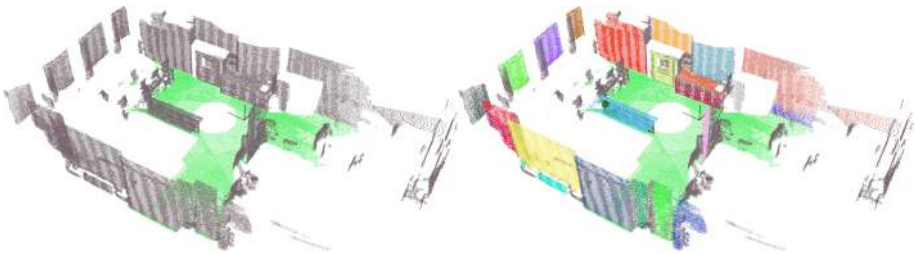
Our method is based on similar concept proposed in (Tuttas & Stilla, 2011) but also incorporates constrains on the position and size of the detected windows. Although windows and other furniture pieces like pictures or mirrors may have the same silhouette, the way that they lay in the plane, i.e. the windows are embedded in the plane, while the other class of objects usually tumble out from the plane.

The number of furnitures classified in the work (Rusu *et al.*, 2008) is lower than in the current work, the focus being in a statistical analysis of the proposed algorithms. The limitation of the current approach are related to the objects classes which were considered at the design phase for the classifier.

The system extracts relevant planes from the registered point cloud, categorizes them as doors or drawers, walls, floor, ceiling, and tables or other horizontal structures. This is achieved by first locating the relevant planar structures, testing for the existence of fixtures, and segmenting the different doors.

As an exhaustive search for all planes is computationally intractable, the method is only searching for those that are aligned with the walls of the room. The orientations of the main walls are determined using a RANSAC-based (Fischler & Bolles, 1981) approach on the normal sphere, as described in (Marton *et al.*, 2010). Since in many indoor environments, most of the surface normals estimated coincide with one of the three main axes of the room, these directions can be used to limit the plane extraction, as it can be seen from Figure 2.9.

After extracting the primary planes, they are classified into floor and ceiling



(a) (b)

Figure 2.9: Segmentation result of human indoor environment, visualized without ceiling: (a) 3D data set with highlighted floor; (b) detection of vertical and horizontal surfaces.

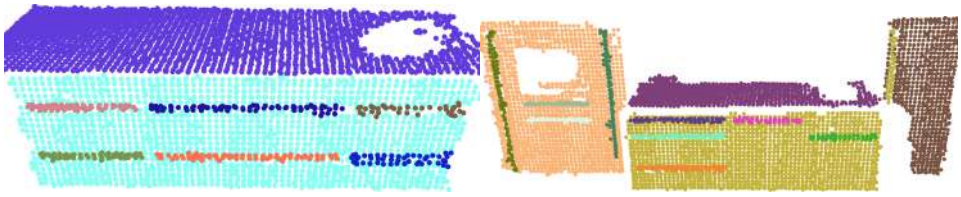
based on horizontal orientation and height, and the walls based on the observation that they are adjacent to the ceiling. The remaining planar connected components – if they exceed a minimum size – constitute candidates for tables or furniture faces as suggested in the algorithm Tamas & Goron (2014). This minimum size is the empirically deduced value of 500 point inliers in herein presented experiments.

Fixtures are detected by first finding point clusters that are within the polygonal prism of the furniture faces using Euclidean distance measure and then fit RANSAC lines or circles to those clusters and thereby differentiate between handles and knobs. A down-sampled version is used, i.e. voxel size 3.5 $[cm]$, of the point cloud for speed considerations and to simplify the computation of patch areas.

(a) (b) (c)

Figure 2.10: Detection of important furniture pieces, inside kitchen areas.

Kitchen appliances, doors and drawers typically have fixtures that allow interaction with them. The existence of fixtures is a good indication to the presence of these objects, so the algorithm searches for clusters of points in the vicinity of detected vertical planar structures. Since the ultimate goal is the manipulation of the handles by the robot, clusters that are too big in diameter relative to the gripper aperture are discarded. These filters are simple enough to be performed for all possible clusters and explain all of the fixtures in typical kitchen environments. The results of this process can be visualized in Figure 2.10.



(a) (b)

Figure 2.11: Obtaining the handles of furniture, which ca be used in grasping applications.

2.5.5 Experimental Results

The proposed system was tested on several point clouds, recorded inside various office and kitchen environments. Table 2.5 presents empirical results for 13 different data sets, out of which 4 are complete scans of rooms, while the other 9 are only partial scans. These data sets were recorded with two different devices; 8 of them using a Hokuyo UTM-30LX sensor, mounted on a PR2 robot; and the rest of 5 were taken with a Sick LMS 200 sensor, mounted on a P3-AT robot. Both scanning devices are fitted with tilting mechanisms in order to attain 3D point clouds.

The obtained results are promising, considering the accuracy values shown in Table 2.5. Detection of indoor components was achieved by identifying planar patches and line segments in the above mentioned data sets. The floors and ceilings

Table 2.5: Detection results presented by individual labels.

No.	Label	Ground Truth	Correct Detection	Accuracy	Present in
1	*floor*	11	11	**100%**	11 scenes
2	*ceiling*	11	11	**100%**	11 scenes
3	*wall*	53	50	**94%**	13 scenes
4	*door*	10	8	**80%**	6 scenes
5	*door frame*	10	8	**80%**	6 scenes
6	*window*	10	8	**80%**	4 scenes
7	*window frame*	10	4	**40%**	4 scenes
8	*furniture*	35	33	**94%**	8 scenes
9	*handle*	27	25	**93%**	2 scenes

are detected flawlessly, as they are the most simple indoor components. Walls and furniture parts, along with their corresponding handles, are also accurately identified; whereas doors and windows, with their frames, are more complicated to detect. Doors can easily blend in as walls, while windows are harder to detect, due to their transparent nature. During the experiments, it was also observed that success rates of handle detection are directly proportional with their physical sizes.

2.6 Conclusions

This chapter covers the necessary steps for a 3D environment perception in different environments with different types of sensors including lidar, stereo and time of flight variants in a robot perception context. As a first step of the perception, the data acquisition and preprocessing part is presented including the scan registration too. For the data aquisition a custom laser scanner is also considered. For the registration algorithms a robust keypoint feature based variant was proposed which proved to provide reliable results for sparse and cluttered environments too.

Further on a system was presented for labeling 3D point clouds taken from human indoor environments by relying on physical features. The labeling classes are as follows: *floor*, *ceiling*, *wall*, *door frame*, *door*, *window frame*, *window*, *furniture* and *handle*. The method was tested on different data sets with promising results.

As research perspectives there are a number of ways to extend this work, both short-term and long-term. In short-term it is intended to enlarge this classification, i.e. by adding new classes to the pipeline, e.g.relevant objects in an industrial context. And for long-term the perception can be improved by incorporating vision into the process, thus contributing to the overall robustness. By using end to end learning pipelines this perception process can be compacted into a single processing chain, which can be customized easily according to changes in the environment.

Chapter 3

3D Object Detection and Recognition

This chapter gives an overview about the 3D object detection and recognition techniques typically used in mobile vehicles assemblies. The first part serves as an introduction to the currently used algorithms for 3D data based recognition, comparing the different algorithms based on their runtime and accuracy. In the second part of the chapter the presented algorithms are illustrated on a land vehicle navigation application in adverse weather conditions Fulop & Tamas (2018); Tamas & Jensen (2014).

3.1 Introduction

The use of 3D perception sensors in the mobile robotics application became popular during the last few years. This is mainly due to the appearance of the affordable depth sensors such as the time-of-flight or projected infrared (e.g Kinect) ones. The research focus in the field of object descriptors, keypoints and classification Ali *et al.* (2014) grow especially for the Kinect like sensors, while for the time-of-flight sensors such as the SwissRanger it is rather limited. Although there are similarities in the characteristics of the two sensors, the quality of depth data acquired with the SR is smoother compared to the Kinect like senors, and it can used daytime in

outdoor environment. Instead of RGB data in the case of the SR camera family for each 3D point the intensity and the confidence information for that point is available. Thus a different approach and behavior is expected for the 3D point descriptors for the data emerging from the SR cameras.

In this paper we propose a thorough analyses for the already implemented feature descriptors in the PCL Rusu & Cousins (2011a) library available with a BSD type license. This analyze is focused on the data acquired with a Swiss Ranger 4000 (SR4K) time-of-flight camera Belhedi *et al.* (2012) in a indoor environment for a large range of objects. As typical error sources the occlusion, subsampling, boundary errors and signal noise are considered as separate test cases. For performance metrics the receiver operating characteristic (ROC) Fawcett (2006) curve is considered based on the true positives and false positives during the recognition tests. Further on, the size and runtime of the different descriptors are considered. Based on these evaluations the best descriptor can be considered for the application dealing with time-of-flight camera data recognition.

3.2 Data Preprocessinga and Recognition

This section presents in detail the data base building as well as the recognition pipeline used in testing phase. The details regarding the data filtering, object segmentation and recognition are highlighted as well.

3.2.1 Data set acquisition

In order to build up a test database for the feature descriptor analysis 15 different sized objects were considered. The list of objects captured with the SR4K camera are listed in Figure 3.1.

These objects vary in size from $0.1m$ to $1.5m$ and are well observable with the time-of-flight sensor, i.e they are not transparent nor lucid surfaces. Also note that, objects from the same category were considered, i.e two different chairs or a drawer

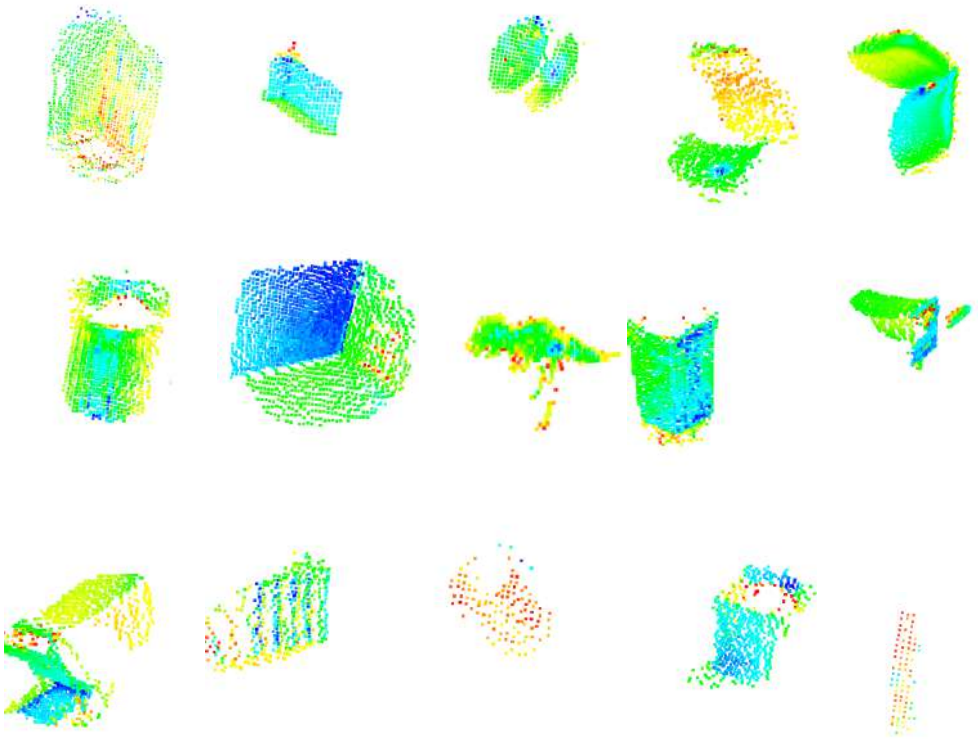


Figure 3.1: Test objects considered during the evaluation (top-left to bottom-right): box, small box, cable-holder, chair1, chair2, cylinder, drawer, dyno, hexagon, open-drawer, open-drawer2, paper-holder, small-cylinder, trash-bin, tube.

with different configurations (with one and with two open doors), which makes the recognition and thus the feature descriptor evaluation challenging. There were more than 300 different captures made for these objects from different views, distances and positions.

3.2.2 Filtering

The first step of the object recognition pipeline was the filtering of the raw data. The main role of this step was to reduce the outliers with a statistical filtering and to get a

compact representation of the data with a voxel-grid type filter. Also a pass-through filter was considered in order to cancel out the false readings from the SRK4 sensor, which are often present like far shadows for the objects. All these filters are part of the PCL library. In our setup the tuned parameters for the filters were as follows: for the voxel grid we considered a grid size of $0.01m$, which is close to the actual resolution of the sensor, while for the statistical outliers we considered 50 points and with the standard deviation threshold of 0.8.

3.2.3 Object segmentation

In the next step of the data preprocessing the major planes are segmented out in order to get the objects from the scene. This is an important step in order to end up with the data containing only the objects from a scenario. In order to achieve this, planar models are fitted to the acquired data, and the largest planes up to a tuned percentage are removed. The plane fitting is performed in a standard sampling consensus (SAC) approach, and for the plane removal parameter we set to 25 percentage of the original data.

3.2.4 Object recognition

The object recognition domain is a well evaluated one especially in the 2D space, special contests being organized for this topic such as the Pascal VOC. In the last few years the recognition in the 3D domain got into the focus with major contributions such as the work Hane *et al.* (2013). These ones often make use of techniques and principles developed in the 2D domain and extrapolated to the 3D case. This is valid also for the point-correspondence based recognition. This technique usually makes use of feature descriptors for the points in order to perform an efficient recognition.

After the feature descriptors are extracted, for the feature comparison different distance metrics can be used also such as the Euclidean, L_1, L_2, Hellinger or χ^2. In

our approach we used the χ^2 metric of two feature vectors $\mathbf{x}$ and $\mathbf{y}$ with length N such as follows:

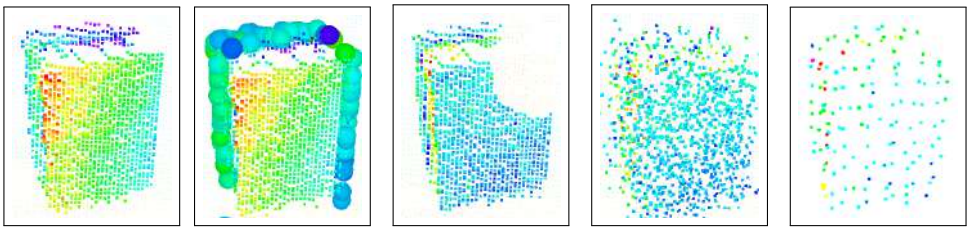
$$\chi_{\mathbf{x},\mathbf{y}} = \sqrt{\frac{\sum_{i=1}^{N}(\mathbf{x}_i - \mathbf{y}_i)^2}{\sum_{i=1}^{N}(\mathbf{x}_i + \mathbf{y}_i)}} \tag{3.1}$$

This metric proved to be stable and good weighting for the different types of features used in our evaluation tests. For the feature correspondence search the nearest-neighbors (NN) search was used with the above mentioned metric. The details regarding the used feature descriptors are presented in the next section.

3.3 Depth Feature Descriptors

The depth feature descriptors as well as the image descriptors are compact representation of data. Beside the fact that the data is represented in a compact form, the features tend to contain rich information extracted from the data. Another important general characteristics is the invariance with respect to certain transformations and disturbances. This invariance is essential in order to use them as discriminators in a recognition type applications.

Two main categories of descriptors are distinguished based on the range of data on which are computed: local and global ones Aldoma *et al.* (2012). Global descriptors are meant to be used to describe a larger set of data containing objects of interest, and they capture the geometric properties of these objects. Local descriptors in contrast are usually computed on a small subset of representative data (keypoints) for already segmented scenes, and are capturing properties around these keypoints. Both types of descriptors have their advantages/disadvantages, which we analyzed in our robustness test benches.



(a) (b) (c) (d) (e)

Figure 3.2: 3D data corrupted with different noise (best viewed in color): (a) the nominal data, (b) the contour segmentation noise, (c) occlusion noise, (d) Gaussian noise, (e) sampling noise

3.3.1 Local variants

3.3.1.1 Rotation invariant feature transform

The rotation invariant feature transform (RIFT) can be applied to 3D data containing intensity information too. Originally it was proposed in the work of Lazebnik *et al.* (2005) as an extension to the SIFT image descriptors Lowe (2004b). The algorithm iterates over each point P_i in the input point cloud and and within a sphere of radius r all the points are considered to belong to a subset P_{ik}. An imaginary circle with 4 bins (rings) are considered perpendicular to the normal at the point P_i. All neighbors of the selected point are assigned to a ring based on the relative distance based on gradient histogram computed with 8 bins using a thresholding. Thus a total number of 32 histograms are computed with this technique, which describe circular features of a point P_i.

3.3.1.2 Intensity Gradient Estimator

The intensity gradient estimator (IGE) uses as input depth data with intensity information. In the first step the normals of the 3D points are computed, and for each point the analytical intensity gradient is computed along the surface direction of the considered region. This technique is similar to the 2D image intensity computation

Shams *et al.* (2007). In the final step the computed gradient is projected onto the surface, and the output of the estimator is the projection gradient vector containing the intensity information computed for each input point.

3.3.1.3 Intensity Spin Estimator

This type of descriptor is based on the work of Lazebnik *et al.* (2005), however the idea of using intensity information as a descriptor was already present in the earlier work Johnson & Hebert (1999). In contrast to the IGE type of descriptor in this case there is no need for explicit normal pre-computation at the input point cloud, which gives a considerable speed-up for this algorithm. As tuning parameters the point distance and intensity distance bins can be set, having the same meaning as in the case of the RIFT descriptor.

3.3.1.4 Spin Image Estimator

The original idea for the spin image (SI) estimation is presented in the work Johnson & Hebert (1999) and can be applied to depth data with pre-computed normals. The algorithm computes two types of distances: the distance of the normals computed at a point and the source normal n and the between the from the considered point along n. The distances larger than a tuning threshold are rejected. From the remaining distance pairs a histogram is built, which represents the occurrence of the discrete distance pairs.

3.3.1.5 Point Feature Histogram

The local point feature histogram (PFH) Morisset *et al.* (2009) descriptor extends the original surflet-pair relation histograms suggested in the work of Wahl *et al.* (2003). The input for this feature descriptor is a pointcloud with normals. In the first step for each point P_i the neighbors within a search radius are computed, denoted with sets P_{ik}. Within these sets point pairs are considered denoted with P_s and P_t with the meaning source and target. For these pairs, the difference of normals are

computed and, described with 3 angles around the axis and a distance. As the distance if varying with viewpoint, this can be left out. Finally, these angles are considered to be sorted in the 125 binned histogram, which is the output of the algorithm for each point.

3.3.1.6 Fast Point Feature Histogram

The fast point feature histogram (FPFH) Morisset *et al.* (2009) is an extension of the PFH yielding to computationally less expensive variant of the PFH. The major difference between PFH and FPFH is that while in the case of PFH all pairs of points are considered in the subsets P_{ik}, in this case only the point pairs between P_i and the rest of the point within P_{ik} are considered. Thus the computation cost drops from $O(nk^2)$ to $O(nk)$. The three angles in this case are binned into a 11 bin histogram, the total length of the obtained histogram is 33 for each point.

3.3.1.7 Viewpoint Feature Histogram

The global viewpoint feature histogram (VFH) Rusu *et al.* (2010b) describes the pointcloud P as containing two components: a component representing the viewpoint of the scene and one containing the FPFH features. In this case the FPFH features are binned into a 45 bin histogram, and the distance between the points is also taken into account, thus a total number of 4 extended FPFH features are stored. The additional view point feature is computed by taking the centroid of the pointcloud denoted with P_c and computing the FPFH for each neighbors. The later histogram is represented using 128 bins, thus the total number of bins for a pointcloud is 308 for this descriptor for the entire pointcloud.

3.3.2 Global variants

3.3.2.1 Clustered Viewpoint Feature Histogram

The clustered viewpoint feature histogram (CVFH) Aldoma *et al.* (2011) is an extension of the VFH in order to handle occlusion or other types of sensor noise. This is mainly important for the VFH, as in case of an occluded view of the same object the histogram of the descriptor varies considerably. The basic idea of the CVFH is the construction of stable regions (clusters) S which step is done by computing compact regions using the region growing approach with thresholding on the normal values. Once these S regions are computed, the VFH for each of them is determined, and an additional shape distribution (SD) is computed as $SD = \frac{(c-p_i)^2}{sup(c-p_i^2)}$, where c is the centroid of the cluster S and the p_i represents the points from this region. This component is also stored in a binned histogram, the total number of descriptor histogram bins for a single point being equal to 308.

3.3.2.2 Ensemble of Shape Functions

The ensemble of shape functions (ESF) type descriptor was proposed in the work Wohlkinger & Vincze (2011) which is based on the A3, D2, D3 shape descriptor functions and extends the D2 type description presented in Ip *et al.* (2002).

The algorithm starts with selecting a subset of 20000 points from the pointcloud, and samples three random points from this P_a, P_b, P_c. The D2 distance is based on the metric distance between the points P_a and P_b. In the next step is verified weather the line connecting the two points are on the surface (in), or out the surface (out) or both (mixed). The corresponding bin for the D2 distance is incremented at the computed bin. This procedure is repeated for the remaining two point-pairs.

3.3.2.3 Radius-based Surface Descriptor

The Radius-based Surface Descriptor (RSD) Marton *et al.* (2011) has as input an oriented pointcloud, i.e. with normals and describes the local geometry of a point

with respect to its neighbors in terms of radii. Each point pair is supposed to lie on a sphere, and the distance of the points d and the angle between the normals at the two points has the relation:

$$d(\alpha) = \sqrt{2r} \cdot \sqrt{1 - cos(\alpha)} \approx r\alpha + r\alpha^3/24 + O(\alpha^5) \tag{3.2}$$

The equation holds for $\alpha \in (0, \pi/2)$, while for an accurate estimation of the radii a linear regression is applied on the extremas of the (α, r) pairs. Also an intuitive geometric interpretation of the obtained radii makes it usable for surface categorization, i.e. the large radii denotes planar surfaces, while small radii is for cylindric objects. For the recognition test we used the radii as histograms computed for individual objects.

3.3.2.4 Principal Curvature Descriptor

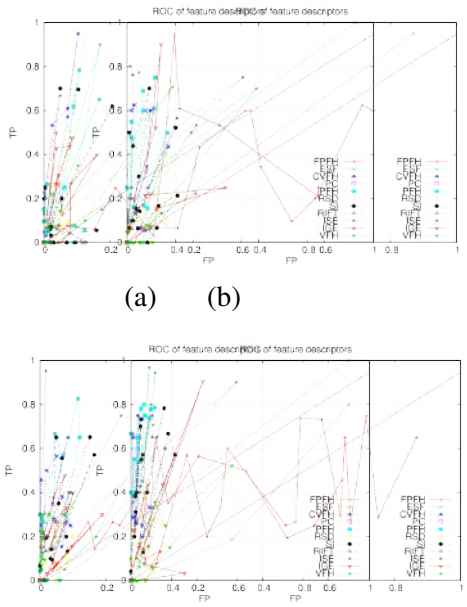
The principal curvature (PC) descriptor as its name suggest computes the curvature for an oriented pointcloud, and uses this information as descriptor. Beside the curvatures, it computes the magnitudes and directions for each point along the XYZ axes, and stores the largest and smallest values for these ones. As tuning parameter the maximum radii for neighbors search is considered.

3.4 Robustness Test

In our robustness tests we investigated the sensibility of the different feature descriptors with respect to common disturbances encountered during the object recognition pipelines (see Figure 3.2). For qualitative evaluation we considered the ROC curve, which incorporates the information from a confusion matrix, while for the quantitative comparison the AC_d metric was used defined as Fawcett (2006):

$$AC_d = 1 - \sqrt{W \cdot (1 - TP)^2 - (1 - W) \cdot Fp^2} \tag{3.3}$$

where TP and FP denote the true positives respectively the false positives and W is a weighting factor giving an application specific weighting to Tp in favor of FP.



(a) (b)

(c) (d)

Figure 3.3: 3D data corrupted with occlusion and contour noise (best viewed in color): (a) occlusion noise at the observation side, (b) occlusion noise at the template side, (c) contour noise at the observation side, (d) contour noise at the template

3.4.1 Test methodology

The test methodology followed in this paper is similar to the generic test procedures in the main literature with the focus on the recognition applications. This means that we divided our captured dataset into $1/3$ and $2/3$ ratio used for training and testing purposes. The training set we denote with *templates* while the testing objects we consider as *observations* of the template objects.

In the training phase we compute for each object each aforementioned descriptor, and store the descriptors in a FLANN Muja & Lowe (2009) data structure and

store it for later queries. This can be viewed also as an offline phase in a real life recognition application.

In the test phase each of computed descriptors for the considered object is searched in the FLANN database containing the specific template feature descriptors. The object is compared against each other object in the database and the one with the smallest metric defined in eq. (3.1) is considered to be a match. In this manner are determined the indicators for the recognition such as the T_p expressing the true positive rate and F_p the incorrect classification rate for the objects.

Another important aspect of the test procedure is that for each type of tested disturbances, the added artificial disturbance is considered both for the template and for the observation side in two separate phases, thus the total object-object comparison exceeds half million test cases.

For the nominal case, i.e. no added artificial noise, the best performance f meaning closest to the point (0,1) is achieved by the PFH and ISE descriptors, while the poorest is the FPFH in this case.

3.4.2 Occlusion test results

The occlusion test tackles one of the most common issues in the object recognition pipeline, the object occlusion problem. This practically means that the template is only partially visible, i.e. an occluded observation is available for the recognition. In our experimental setup we considered variable occlusion to be tested, by removing a radial zone of an object at a random position. The amount of removed points is given as a tuning parameter expressing the removed points as a percentage of pointcloud size of the template. The results are shown in Figure 3.3 part (a) and part (b) for the occlusion noise at observation and at the template side.

For the observation side occlusion test case the best behavior is obtained with the CVFH. ISE and SI type of descriptors, while the poorest results are given by VFH and FPFH. Also on the template side added disturbance the best robustness is achieved with the CVFH, which proves the efficient approach of the clustered

viewpoint approach for the occlusion handling.

3.4.3 Segmentation error test results

Another important source of error in the object recognition applications are emerging from the segmentation phase. The segmentation errors can be viewed also as contour noise around the object, which is present due to the inaccuracy in the segmentation process. In order to test this kind of error source, first we determined the boundary of the test object, and than we removed a tunable region for each point from the boundary. The results for this test case are shown in Figure 3.3 part (c) and part (d) for the observation occlusion noise and for the template side noise respectively. The best results are achieved with VFH and ISE respectively.

3.4.4 Sensor noise test results

The most common problem with the real data recognition setups is the sensor noise, i.e. measurement inaccuracies due to the physical sensor limitations. Usually the non-systematic part of the noise is approximated with a Gaussian probability distribution. In our case we used also this assumption, and we considered a system with variable added Gaussian noise covariance along the XYZ coordinates added independently.

This kind of disturbance affects drastically the performance of the CVFH and VFH at the observation side noise, while the SI is vulnerable for the template side Gaussian noise disturbances. Still in this case the ISE tolerates well the noise both on observation and template sides.

3.4.5 Subsampling test results

The last test case deals with the varying sampling type error. This kind of disturbance is present in the case of the time-of-flight cameras, as the same object

captured at different distances from the sensor has a varying sampling ratio. Thus this is a disturbance which is important to be analyzed for such type of sensors. In our test cases we computed different subsampling by applying a tunable sized voxel grid on the original data.

The majority of the feature descriptors are robust against this type of disturbance, however the ISE giving the best performance till now fails at this test phase. Also the FPFH get worse in this case, getting close to point (1,1) n ROC.

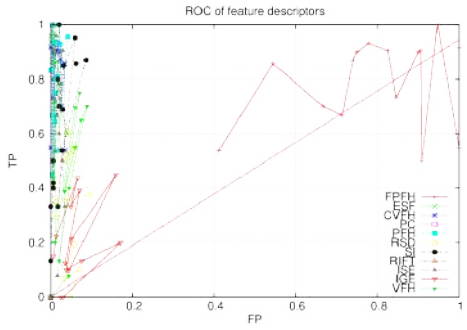


Figure 3.4: Feature descriptor test result for the nominal data (best viewed in color)

3.4.6 Quantitative results

In order to have a quantitative evaluation of the test bench we considered the AC_d metrics defined in eq. (3.3) for the analyzed descriptors. Each descriptor varies also with the test object considered for the recognition problem, hence in order to reduce the dimensionality of the evaluation, we considered the average values for the metrics over all test object for each descriptor. This gives a good performance indicator for the different test cases without loosing the important characteristics of the test outputs. The evaluation was done for both the observation and template side disturbances.

The results for the test cases shown also on the ROC curves are summarized in the Table 3.1 for the object side noise disturbances and in the Table 3.2 for the

Descr.	Nom.	Occ.	Cont.	Gauss	Sampl.
VFH	0.75	0.63	0.64	0.52	0.58
CVFH	0.61	0.61	0.57	0.51	0.59
PFH	0.82	0.56	0.71	0.68	0.71
FPFH	0.41	0.46	0.42	0.46	0.40
RSD	0.73	0.52	0.62	0.62	0.77
RIFT	0.59	0.55	0.53	0.51	0.49
PC	0.48	0.44	0.45	0.39	0.46
SI	0.83	0.54	0.67	0.74	0.83
ISE	0.87	0.59	0.67	0.87	0.63
IGE	0.64	0.57	0.65	0.62	0.60
ESF	0.83	0.57	0.59	0.80	0.74

Table 3.1: Observation side robustness tests containing the nominal, occlusion, contour, Gaussian and sampling noise test cases

template side case. The level of added noise in this test bench was $4cm$ for the contour noise removal band, 10% for the occlusion percentage of the original object, $5cm$ voxel grid for the sampling and the Gaussian noise with $0.03m$ covariance.

In average the best performance is achieved in this case with the ISE type feature descriptor and ESF for the object and template side respectively.

In average the best performance in these cases is achieved with the ISE type feature descriptor and ESF for the object and template side respectively, however the performances of the descriptor lowered considerably with the noise.

In terms of run-times, the best performance was achieved with FPFH, while computationally the most intensive ones were the ISE and IGE type of descriptors.

Descr.	Nom.	Occ.	Cont.	Gauss	Sampl.
VFH	0.71	0.65	0.64	0.52	0.62
CVFH	0.81	0.69	0.75	0.54	0.74
PFH	0.78	0.71	0.73	0.61	0.690
FPFH	0.88	0.79	0.77	0.74	0.63
RSD	0.73	0.64	0.63	0.63	0.69
RIFT	0.59	0.49	0.47	0.50	0.49
PC	0.59	0.51	0.49	0.55	0.48
SI	0.83	0.66	0.75	0.62	0.84
ISE	0.87	0.66	0.76	0.86	0.64
IGE	0.64	0.65	0.65	0.61	0.63
ESF	0.89	0.59	0.59	0.79	0.77

Table 3.2: Template side robustness tests containing the nominal, occlusion, contour, Gaussian and sampling noise test cases

3.5 CNN Based Object Recognition

3.5.1 Motivation

Our target application was an object recognition system based on 2D CNNs Krizhevsky *et al.* (2012) and 3D pose estimation. The combination of 2D state-of-the-art systems with 3D features carries great possibilities Lahoud & Ghanem (2017). Clearly, in order to use this technique, not only the RGB images are needed, but also their corresponding depth images. We implemented a system, that does not rely entirely on 2D or 3D features, rather combines them and extracts the most important segments. This system is able to provide good results even when the available computational resources are limited. Since our main goal was to use it on mobile robots, the processing power cannot be compared to a computer's GPU, such as in case of dedicated servers.

We also tested how do different neural network frameworks perform when the same, lightweight dataset is used. The comparison highlights the advantages and drawbacks of each, regarding the domain of object recognition for mobile robot applica-

tions.

3.5.2 Problem description

This system helps a mobile robot to recognize custom objects in its environment. Since the capabilities of such a robot are limited, the solution needs to be constructed in such a way that it is supported by a modest hardware. An example for such a hardware is a Raspberry Pi 3 model B with a Intel® Movidius™ Neural Compute Stick. The stick supports Caffe and TensorFlow, and thanks to its reduced size the usage is comfortable even for lightweight systems, such as mobile robots.
In order to get started, a basic knowledge is essential about the functionality of neural networks, convolution, image processing techniques and mobile robot programming. However, the learning curve for these non-parametric estimation techniques is rather steep, i.e. in relatively short time custom solution can be achieved in developing custom object recognition tasks.

3.5.3 Related work

The concept of neural networks and deep learning gains more territory each year in computer vision and robotics, thus there is a rich literature on frameworks related to this technology. Placing 2D bounding boxes around detected objects is a common feature for most detection systems. In this paper we focus on three major frameworks, namely Darknet Redmon (2013–2016), Caffe Jia *et al.* (2014) and TensorFlow Abadi *et al.* (2015).
Convolutional Neural Networks are specific feed-forward networks, that perform extremely well on visual recognition tasks. Since object recognition belongs to this field, it is convenient to take advantage of this characteristic. The most representative networks that are worth to mention are Region-based Convolutional Neural Networks (R-CNN) Girshick *et al.* (2013), Fast R-CNN Girshick (2015), You Only Look Once (YOLO) Redmon *et al.* (2015), Faster R-CNN Ren *et al.* (2015) and Region-based Fully Convolutional Networks (R-FCN) Dai *et al.* (2016). All of the

above require a powerful GPU, at least for the training phase. In order to have a decent, preferably real-time performance, it is recommended to use GPU during the deployment process as well.
Using the Pascal VOC 2007 dataset, the performances are compared in Table 3.3.

Method	**mAP**	**Rate**
YOLO	63.4%	45 fps
YOLOv2	76.8%	67 fps
R-CNN	58.5%	3 fps
Fast R-CNN	70.0%	-
Faster R-CNN (VGG-16)	73.2%	5 fps
R-FCN (ResNet50)	77.4%	8-11 fps
R-FCN (ResNet101)	79.5%	5-8 fps

Table 3.3: Performance comparison of neural networks for object recognition based on mean average precision (mAP) and frames per second (fps).

Darknet Redmon (2013–2016) is an open source neural network framework, and it uses YOLO, a state-of-the-art object detection system. Its performance depends heavily on the machine's GPU, but under optimal circumstances it performs real-time, meaning 40-90 fps. It makes use of a single neural network to the whole image, then divides into multiple regions and predicts the locations, using bounding boxes and probabilities. The predictions are influenced by the environment of the detected object. Since it uses only one network evaluation, its speed exceeds multiple times the speed of R-CNNs or even Fast R-CNNs. Recently they introduced YOLOv2Redmon & Farhadi (2016), which provides some additional improvements over the former variant.

Caffe Jia *et al.* (2014) is a deep learning framework developed by Berkeley AI Research. It mainly focuses on image processing, and it is not designed for other deep learning application such as text or sound recognition. The previously mentioned networks, namely R-CNN, Fast R-CNN and Faster R-CNN can all be integrated and effectively used with Caffe.

TensorFlow Abadi *et al.* (2015) is probably the most widely used open source deep learning framework. It was developed by researchers and engineers working on the Google Brain Team within Google's Machine Intelligence research organization. It is a flexible and adaptable system, that can be useful on many fields of interest.

3.6 Object recognition results

3.6.1 2D

The first objective is the preparation of a sufficiently rich dataset, meaning that it contains satisfactory amount of data, in this case images. The resolution of the pictures should be adequate, neither too high, nor too low, and it is essential that the overall quality is acceptable.

Usually a preliminary processing and filtering is recommended in order to emphasize the contour of the objects and to reduce the images' size. Labels need to be

assigned to each class, so the identity of the detectable objects from the images can be ”learned”. The dataset is divided in two parts: a bigger one for training and a smaller for validation. The structure of the neural network is then determined. Since CNNs achieved the best performance in this domain, their usage is highly recommended. The desired amount of convolutional, pooling and fully-connected layers, respectively the activation functions need to be specified.
The network treats the images as matrices filled with pixel values. During the training process filters are applied on the images, that are updated after each iteration. As the error decreases, the filters become more and more accurate. Once the training process is finished, the filters can be used on new images or even on a camera stream.

3.6.2 3D

3D state-of-the-art detection methods have an increased runtime and need way more processing power than their 2D counterparts. Using point clouds acquired from RGB-D sensors, semantic segmentation and training is a costly process, wearing down even powerful GPUs.

A major challenge in the 3D object detection part relies in the part segmentation: the separation of the background and region of interest Militaru *et al.* (2016b). For this a useful technique is the use of the 2D bounding boxes as hints in the 3D point cloud segmentation part, thus speeding up and making more robust the recognition pipeline.

This approach assumes that the color and depth cameras are relatively calibrated. In our case we used a Kinect like camera, for which this constraint is satisfied. According to our previous investigations on the feature 3D based object recognition robustness Tamas & Jensen (2014) we chose the viewpoint feature histogram (VFH) based variant for its speed and robustness against sampling noise. The disadvantage of this approach, i.e. the sensitivity to the boundary segmentation was compensated with the use of pre-segmentation based on the 2D bounding boxes

determined by the 2D detection algorithm.

3.6.3 Fused 2D-3D approach

Test conditions: The tests were conducted on a Dell laptop, with Nvidia GeForce GTX 1060 6GB graphics card and Intel® Core™ i7-7700HQ CPU @ 2.80GHz x 8 processor, using Ubuntu 16.04 LTS, 64-bit OS, CUDA 9.0, cuDNN 7.0.5, TensorFLow 1.6.0, Caffe 1.0.
Our experiments began by the comparison of three frameworks, and several networks. In order to achieve the best accuracy, the same custom dataset was used for each of them. This dataset was created by us, using a Kinect camera and it contains 4 object classes, including pictures about objects from several angles.
In the case of Darknet, the modified YOLOv2 network was used. For this, the data preparation takes more time, since the images must be labeled. This label contains the class number, and the position of the object relative to the image. This ensures not only the recognition, but also the right placement of the bounding boxes during detection. These bounding boxes are weighted by the predicted probabilities. The configuration files include the structure of the network and the parameters that need to be set. The training executed until the average loss was stagnating around 0.08. After 30.000 iterations, the model reached its present form. Figure 3.5 presents the visual results from the recognition process. CPU is adequate only if pictures are used in order to test the model, but for real-time performance CUDA is recommended.

For Caffe, the labels for each class were assigned in the program. The network used is a modified Alexnet Krizhevsky *et al.* (2012). The process started with histogram equalization, resizing, division in two parts then storage in LMDB database. With the help of already implemented Caffe tools, the mean image of the training data can be generated. The structure of the network is defined in a prototxt file, specific to Caffe. The solver, responsible for model optimization, contains the base learning rate, the learning rate policy, step size, maximum number of iterations and

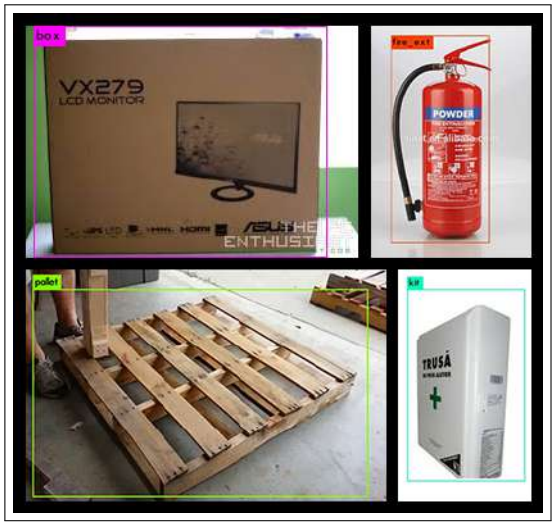


Figure 3.5: Examples of detection of the objects of interest, after applying the trained model, using Darknet Redmon (2013–2016) and the modified YOLOv2 Redmon & Farhadi (2016) network.

more advanced parameters, that can be chosen individually. The switch between CPU and GPU works seamlessly, but the process used with CPU is almost ineffective. During training with GPU, the loss quickly stabilized around 0.38. Figure 3.6 shows the evolution of the accuracy and loss after the first 10000 iterations.

TensorFlow was the only framework that showed a decent performance even when only CPU was used. In order to have more common points for comparison, the network is a modified Alexnet in this case too. With respect to the length of the code, this solution included the largest amount of it. The labels were assigned in the program and the configuration of the parameters does not require a separate script. Since the entire purpose of TensorFlow is the usage of computational graphs, it can be executed much more efficiently than simply in Python. Once the training process reached the maximum given number of iterations it stopped, and the average loss stabilized around 0.39. The test shows the results in Figure 3.7 after 10000 itera-

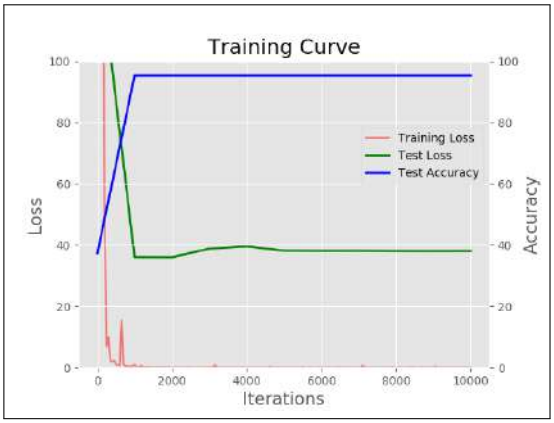


Figure 3.6: The evolution of the training parameters during the first 10000 iterations. Red - training loss, green - test loss, blue - test accuracy (best viewed in color).

tions.

The results for each method were exported, compared and the final comparison can be found in Table 3.4. It presents the performance recorded on the test set. Each model was used on the same dataset, and the precision is recorded for each class separately, then the average is calculated. The results are based on a small test set, that contains a total of 40 images, 10 for each object class. Each percentage shows the number of correct detections over the total number of images for each object class separately.

It is important to mention, that the final conclusion cannot be drawn by taking into consideration only these results. The created code can be modified, and the network structure can be changed in order to achieve a better performance. This comparison puts accent on computer vision and CNNs respectively. Darknet and YOLO were created only for this purpose, thus professionally optimized, including out-of-the-box features. Caffe has some useful built-in tools as well, but since our dataset is a custom one and consists of a limited number of pictures, its accuracy is poorer. The situation is similar to TensorFlow, where a bigger dataset would result

Figure 3.7: The correct and predicted labels of nine, randomly chosen objects from the list. True - real label, Pred - predicted label.

Class	Darknet	Caffe	TensorFlow
Box	60%	80%	80%
Fire extinguisher	100%	40%	40%
Pallet	60%	80%	70%
First-aid kit	90%	70%	60%
Total	**77.5%**	**67.5%**	**62.5%**

Table 3.4: The results of the final comparison, for each class separately and the average of detection rate.

in better precision. Also, for the latter everything needs to be written manually. Without a high-level API, the code becomes unnecessarily long and repetitive. On the other hand it is the most flexible and versatile variant. This fact was proved to be convincing enough to continue the research in order to find a more efficient network structure. Using Keras Chollet *et al.* (2015) with TensorFlow backend, the problem of the long-drawn code is eliminated. We created additional datasets consisting of a higher number of images in order to have a better insight of its influence on the performance. During the process, multiple factors were taken into consideration, and their influence on the results was recorded. The tested activation functions are the three most used ones, namely ”relu”, ”sigmoid” and ”tanh”. The number of layers varies between 2 and 4, and the number of neurons on some layers goes up to 256. The convolutional kernels are the size of 3x3. The most successful combinations are listed in Table 3.5. Since one layer is not sufficient, this possibility is not included in the table. For the sake of simplicity and time management, we used only two object classes in order to find the best architecture. The training phase lasted for 30 epochs and the fully connected layer consists of 500 neurons in each case. Multiple combinations were eliminated, because the model did not converge properly.

The most efficient networks were then tested on the original dataset of four object classes, and on another dataset consisting of 20 different objects. The results of

Layer 1		Layer 2		Layer 3		Layer 4		**Results**		
Size	Act f.	Size	Act f.	Size	Act f.	Size	Act f.	Cl 1	Cl 2	Avg
2	relu	2	relu	-	-	-	-	92%	94%	93%
2	sigmoid	2	sigmoid	-	-	-	-	85%	57%	71%
2	tanh	2	tanh	-	-	-	-	91%	88%	89.5%
2	relu	4	relu	-	-	-	-	95%	94%	94.5%
2	tanh	4	tanh	-	-	-	-	100%	75%	87.5%
4	relu	4	relu	-	-	-	-	99%	77%	88%
4	relu	8	relu	-	-	-	-	99%	82%	90.5%
4	tanh	8	tanh	-	-	-	-	99%	69%	84%
8	relu	8	relu	-	-	-	-	99%	91%	95%
8	tanh	8	tanh	-	-	-	-	99%	83%	91%
8	relu	16	relu	-	-	-	-	97%	95%	**96%**
16	relu	16	relu	-	-	-	-	98%	94%	**96%**
16	tanh	16	tanh	-	-	-	-	96%	88%	92%
16	relu	32	relu	-	-	-	-	99%	89%	94%
8	relu	8	relu	8	relu	-	-	98%	83%	90.5%
8	tanh	8	tanh	8	tanh	-	-	95%	96%	95.5%
8	relu	8	relu	16	relu	-	-	96%	95%	95.5%
8	tanh	8	tanh	16	tanh	-	-	99%	86%	92.5%
8	relu	16	relu	32	relu	-	-	93%	98%	95.5%
8	tanh	16	tanh	32	tanh	-	-	99%	85%	92%
16	relu	16	relu	16	relu	-	-	98%	95%	**96.5%**
16	tanh	16	tanh	16	tanh	-	-	100%	78%	89%
32	relu	32	relu	32	relu	-	-	98%	88%	93%
32	tanh	32	tanh	32	tanh	-	-	99%	70%	84.5%
32	relu	32	relu	64	relu	-	-	96%	98%	**97%**
32	relu	32	relu	64	relu	-	-	100%	79%	89.5%
32	tanh	64	tanh	32	tanh	-	-	97%	90%	93.5%
2	relu	4	relu	8	relu	16	relu	96%	93%	94.5%
2	tanh	4	tanh	8	tanh	16	tanh	100%	88%	94%
32	relu	64	relu	128	relu	256	relu	96%	94%	95%
32	tanh	64	tanh	128	tanh	256	tanh	97%	95%	**96%**

Table 3.5: The most successful network architectures and their accuracy for two object classes.

Layer 1		Layer 2		Layer 3		Layer 4		4 classes		20 classes	
Nr	Act f.	Nr	Act f.	Size	Act f.	Size	Act f.	30 ep	100 ep	30 ep	100 ep
32	relu	32	relu	64	relu	-	-	77.5%	75%	46.5%	48.1%
16	relu	16	relu	16	relu	-	-	72.5%	75%	46.25%	47.1%
8	relu	16	relu	-	-	-	-	62.5%	85%	44.8%	45.1%
6	relu	16	relu	-	-	-	-	80%	72.5%	46.45%	44.7%
32	tanh	64	tanh	128	tanh	256	tanh	35%	64%	52.55%	53.75%

Table 3.6: The results of the multi-class implementation for the most efficient networks taken from Table 3.5, tested on the original dataset (4 classes) and on a more extended one (20 classes).

both tests are shown in Table 3.6. In the case of the original dataset, the results show a powerful increase in comparison to the previously presented Alexnet structure. However, when more objects are added, a significant decrease in precision can be observed. This can happen due to multiple facts regarding the relevance of the training and test datasets, the significantly higher number of object classes or other disturbing effects. As a potential deployment use-case, the model was also tested on the Intel® Movidius™ Neural Compute Stick. It is a small scale device, that supports Caffe and TensorFlow and allows the tuning and deploying of CNNs on low-power appliances. The stick comes with a SDK, which offers an API, tools and examples as well. It creates an internal compiled format based on the desired input, which can be used later on the deployment device. The API provides software for connection establishment with the stick, loading the previously created graph and running detection on it. Its performance was proved to be satisfactory for such a small scale device. It is able even for real-time detection with a low power consumption/size constrained applications such as in case of autonomous rovers.

3.7 Conclusions

In this chapter the problem of 3D feature descriptor robustness was analysed for radiometric data. The main aim was to give both a qualitative and quantitative evaluation of the existing feature descriptors for the 3D domain on this type of specific data, and to test against a large variety of disturbances including the occlusion, segmentation error, sensor noise and sub-sampling cases too. The result were presented using the ROC curves and the AC_d metric for the different test benches. Beside this, the fused 2D-3D detector pipeline based on deep networks was presented and analysed for its reliability. In the second part of this chapter we summarized our lessons learned different approaches from three main frameworks in the current state of the art in 3D object recognition. The main focus was on the implementation of 2D and 3D object recognition techniques using a Kinect like depth camera for mobile robots in indoor environment. A demo code and video showing the results of this investigation is available on the website of the authors.

Chapter 4

2D-3D Heterogeneous Data Based Pose Estimation

This chapter introduces the patch based external camera pose estimation method. In the first part is presented the central perspective camera case with some application examples. Next, the case of the catadioptric camera is considered where the spectral and radiometric information fusion is highlighted with real life examples. This chapter is based on the papers Tamas & Kato (2013a); Tamas *et al.* (2014).

4.1 Taxonomy and Challenges of Camera External Pose Estimation

One of the most challenging issue in robotic perception applications is the fusion of information from several different sources. Today the majority of the platforms include range (2D or 3D sonar/lidar) and camera (color/infrared) sensors that are usually work independently, although the information from different sources can be used in a complementary way.

In order to fuse the data coming from these independent devices, the measured data has to be transformed into a common coordinate frame. This is achieved by

either extrinsic or intrinsic-extrinsic calibration, depending on whether the internal camera parameters are available or not. In case of the extrinsic parameter estimation for a range-camera sensor pair the 3D rigid translation between the two coordinate systems is determined.

There is a considerable research effort invested in autonomous car driving projects both on academic and industrial side. While for the specific scenarios such as highways there are already a number of successful applications, this topic is still generally not solved for complex environments such as the urban ones Geiger *et al.* (2014); Lin *et al.* (2013). The recent developments in the autonomous driving in urban environment using a great variety of close-to-market sensors including different cameras put into the focus the need for information fusion emerging from these sensors Furgale *et al.* (2013).

While internal calibration can be solved in a controlled environment, using special calibration patterns, pose estimation must rely on the actual images taken in a real environment. There are popular methods dealing with point correspondence estimation such as Scaramuzza *et al.* (2006b) or other fiducial marker images suggested in Kannala & Brandt (2006), which may be cumbersome to use in real life situations. This is especially true in a multimodal setting, when omnidirectional images need to be combined with other non-conventional sensors like lidar scans providing only range data. The Lidar-omnidirectional camera calibration problem was analyzed from different perspectives: in Scaramuzza *et al.* (2007), the calibration is performed in natural scenes, however the point correspondences between the 2D-3D images are selected in a semi-supervised manner. Mirzaei *et al.* (2012) tackles calibration as an observability problem using a (planar) fiducial marker as calibration pattern. In Pandey *et al.* (2012), a fully automatic method is proposed based on mutual information (MI) between the intensity information from the depth sensor and the omnidirectional camera. Also based on MI, Taylor & Nieto (2012) performs the calibration using particle filtering. However, these methods require a range data with recorded intensity values, which is not always possible.

Instead of establishing point matches or relying on artificial markers or recorded

intensity values, we propose a pose estimation algorithm which works on segmented planar patches. Since segmentation is required anyway in many real-life image analysis tasks, such regions may be available or straightforward to detect. The main advantage of the proposed method is the use of regions instead of point correspondence and a generic problem formulation which allows to treat several types of cameras in the same framework. We reformulate pose estimation as a shape alignment problem, which can be efficiently solved in a similar way as in Domokos *et al.* (2012). The method has been quantitatively evaluated on a large synthetic dataset and it proved to be robust and efficient in real-life situations.

This chapter introduces a novel region based calibration framework for non-conventional 2D cameras and 3D lidar sensors emerging from different sources in a single step automatic manner. The main novelty is the formulation of the calibration problem as a general 2D-3D non-linear registration problem which works without special targets or established point matches. The registration is accomplished by solving system of nonlinear equations based on the idea of Domokos *et al.* (2012). However, the equations are constructed in a different way here due to the different dimensionality of the lidar and camera coordinate frames as well as the different camera models. The concept was proven to be viable on a large scale synthetic dataset as well as on real data using perspective and dioptric cameras too.

4.2 Perspective Camera Case

4.2.1 Related work

The need for registration between heterogeneous sensor data is common to multiple research fields including aerial remote sensing Mastin *et al.* (2009); Mishra & Zhang (2012), medical images processing Pluim *et al.* (2003) or mobile robotic applications Geiger *et al.* (2012); Pandey *et al.* (2012); Scaramuzza *et al.* (2007); Taylor & Nieto (2012). Different approaches are tackling the non-trivial 2D-3D registration problem: point or line correspondence finding between the two domains

Mastin *et al.* (2009); Stamos & Allen (2001), intensity image based correlation Pandey *et al.* (2012), use of specific artificial land-mark Alismail *et al.* (2012); Gong *et al.* (2013); Herrera C *et al.* (2012); Krause & Evert (2012); Naroditsky *et al.* (2011); Unnikrishnan & Hebert (2005); Zhang *et al.* (2008) or mutual information extraction and parameter optimizationTaylor & Nieto (2012); Viola & Wells (1997); Williams *et al.* (2004).

The extrinsic calibration of 3D lidar and low resolution color camera was first addressed in Unnikrishnan & Hebert (2005) which generalized the algorithm proposed in Zhang (2004). This method is based on manual point feature selection from both domains and it assumes a valid camera intrinsic model for calibration. A similar manual point feature correspondence based approach is proposed in Scaramuzza *et al.* (2007).

There are also extensions to the simultaneous intrinsic-extrinsic calibration presented in the work Mirzaei *et al.* (2012) which used the intensity information from lidar to find correspondences between the 2D-3D domains. Other works are based on the fusion of IMU or GPS information in the process of 2D-3D calibration Nunez *et al.* (2009), mainly in initialization phase of the calibration Taylor & Nieto (2012).

Recently there has been an increasing interest in various calibration problem setups ranging from high-resolution spatial data registration Mastin *et al.* (2009) to low-resolution, high frame rate depth commercial cameras such as Kinect Herrera C *et al.* (2012); Zhang & Zhang (2011), or in the online calibration during different measurements in time such as in case of a traveling mobile robot Levinson & Thrun (2013); Pandey *et al.* (2012).

4.3 Region-based calibration framework

Consider a lidar camera with a 3D coordinate system having its origin **O** in the center of laser sensor rotation, x and y axes pointing to the right and down, respectively, while z is pointing away from the sensor. Setting the world coordinate system to the lidar's coordinate system, we can always express a 3D lidar point **X** with its

homogeneous world coordinates $\mathbf{X} = (X_1, X_2, X_3, 1)^T$. The perspective camera sees the same world point $\mathbf{X}$ as a homogeneous point $\mathbf{x} = (x_1, x_2, 1)^T$ in the image plain obtained by a perspective projection $\mathbf{P}$:

$$\mathbf{x} = \mathbf{P}\mathbf{X} \tag{4.1}$$

where $\mathbf{P}$ is the 3×4 camera matrix, which can be factored into the well known $\mathbf{P} = \mathbf{K}\mathbf{R}[\mathbf{I}|\mathbf{t}]$ form, where $\mathbf{I}$ is the identity matrix, $\mathbf{K}$ is the 3×3 upper triangular *calibration* matrix containing the camera intrinsic parameters:

$$\mathbf{K} = \begin{pmatrix} f_x & & o_x \\ & f_y & o_y \\ & & 1 \end{pmatrix}, \tag{4.2}$$

A classical solution of the calibration problem is to establish a set of 2D-3D point matches using a special calibration target Alismail *et al.* (2012); Herrera C *et al.* (2012); Mirzaei *et al.* (2012); Naroditsky *et al.* (2011); Unnikrishnan & Hebert (2005), and then solve for $\mathbf{P}$ via a system of equation based on (4.1) or the minimization of some error function. When a calibration target is not available, then solutions typically assume that the lidar points contain also the laser reflectivity value (interpreted as a gray-value), which can be used for intensity-based matching or registration Mastin *et al.* (2009); Naroditsky *et al.* (2011); Pandey *et al.* (2012); Scaramuzza *et al.* (2007).

However, in many practical applications (e.g infield mobile robot), it is not possible to use a calibration target and most lidar sensors will only record depth information. Furthermore, lidar and camera images might be taken at different times and they need to be fused later based solely on the image content. Therefore the question naturally arises: what can be done when neither a special target nor point correspondences are available? Herein, we propose a solution for such challenging situations. In particular, we will show that by identifying a single planar region both in the lidar and camera image, the extrinsic calibration can be solved. When two

such non-coplanar regions are available then the full calibration can be solved. Of course, these are just the necessary minimal configurations. The more such regions are available, a more stable calibration is obtained.

Our solution is based on the 2D shape registration approach of Domokos Domokos *et al.* (2012), where the alignment of non-linear shape deformations are recovered via the solution of a special system of equations. Here, however, the calibration problem yields a 2D-3D registration problem, which requires a different technique to construct the system of equations: Since correspondences are not available, 4.1 cannot be used directly. However, individual point matches can be integrated out yielding the following integral equation:

$$\int_{\mathcal{D}} \mathbf{x} d\mathbf{x} = \int_{\mathbf{P}\mathcal{F}} \mathbf{z} d\mathbf{z}, \tag{4.3}$$

where $\mathcal{D}$ corresponds to the region visible in the *camera* image and $\mathbf{P}\mathcal{F}$ is the image of the *lidar region* projected by the camera matrix $\mathbf{P}$. The above equation corresponds to a system of 2 equations only, which is clearly not sufficient to solve for all parameters of the camera matrix $\mathbf{P}$. Therefor we adopt the general mechanism proposed in Domokos *et al.* (2012) to construct new equations. Indeed, (4.1) remains valid when a function $\omega : \mathbb{R}^2 \to \mathbb{R}$ is acting on both sides of the equation

$$\omega(\mathbf{x}) = \omega(\mathbf{P}\mathbf{X}), \tag{4.4}$$

and the integral equation of (4.3) becomes

$$\int_{\mathcal{D}} \omega(\mathbf{x}) d\mathbf{x} = \int_{\mathbf{P}\mathcal{F}} \omega(\mathbf{z}) d\mathbf{z}. \tag{4.5}$$

Adopting a set of nonlinear functions $\{\omega_i\}_{i=1}^{\ell}$, each ω_i generates a new equation yielding a system of ℓ independent equations. Hence we are able to generate sufficiently many equations. The parameters of the camera matrix $\mathbf{P}$ are then simply obtained as the solution of the nonlinear system of equations (4.5). In practice, an overdetermined system is constructed, which is then solved by minimizing the

algebraic error in the *least squares sense* via a standard *Levenberg-Marquardt* algorithm.

Note that computing the integral on the right hand side of (4.5) involves the actual execution of the camera projection $\mathbf{P}$ on $\mathcal{F}$, which might be computationally unfavorable. However, choosing power functions for ω_i:

$$\omega_i(\mathbf{x}) = x_1^{n_i} x_2^{m_i}, \quad n_i \leq 3 \text{ and } m_i \leq 3 \tag{4.6}$$

and using a triangular mesh representation $\mathcal{F}^{\triangle}$ of the lidar region $\mathcal{F}$, we can adopt an efficient computational scheme. First, let us note that this particular choice of ω_i yields the 2D geometric moments of the projected lidar region $\mathbf{P}\mathcal{F}$. Furthermore, due to the triangular mesh representation of $\mathcal{F}$, we can rewrite the integral adopting ω_i from (4.16) as

$$\int_{\mathcal{D}} x_1^{n_i} x_2^{m_i} d\mathbf{x} \quad = \quad \int_{\mathbf{P}\mathcal{F}} z_1^{n_i} z_2^{m_i} d\mathbf{z} \quad \approx \quad \sum_{\forall \triangle \in \mathcal{F}^{\triangle}} \int_{\triangle} z_1^{n_i} z_2^{m_i} d\mathbf{z}. \tag{4.7}$$

The latter approximation is due to the approximation of $\mathcal{F}$ by the discrete mesh $\mathcal{F}^{\triangle}$. The integrals over the triangles are various geometric moments which can be computed using e.g. the following formula for $x^p y^q$ Best (1964):

$$2 \sum_{k=0}^{p} \sum_{l=0}^{q} \frac{(-1)^{k+l} \binom{p}{k} \binom{q}{l} \nu_{kl}}{k+l+2} x_0^{p-k} y_0^{q-l} \tag{4.8}$$

where

$$\nu_{kl} = \sum_{i=0}^{k} \sum_{j=0}^{l} \frac{\binom{k}{i} \binom{l}{j}}{k-i+l-j+1} (x_0 - x_1)^i (x_1 - x_2)^{k-i} (y_0 - y_1)^j (y_1 - y_2)^{l-j} \tag{4.9}$$

with the notation x_i and y_i, $i = 0 \ldots 2$ for the vertices of the triangles.

The summary of the numerical implementation of the proposed method is presented in Algorithm 4.1. Note that normalization is critical to ensure a numerically

stable solution (see Domokos *et al.* (2012) for details). For extrinsic calibration, $\mathbf{K}$ is known a priori, while for intrinsic-extrinsic calibration, it is initialized with $f_x = f_y = 800$ (corresponding to a normal 65 degree field of view) and $\mathbf{o}$ is set to the center of the image.

Algorithm 4.1 The proposed calibration algorithm

Input: 3D point cloud and 2D binary image representing the same region, and the calibration matrix $\mathbf{K}$

Output: Parameters of the camera matrix $\mathbf{P}$

1: Normalize 3D points into the unit cube and 2D point into the unit square centered in the origin.
2: Triangulate the region represented by the 3D point cloud.
3: Construct the system of equations of (4.7) with the polynomial ω_i functions of (4.16).
4: Initialize the camera matrix as $\mathbf{P} = \mathbf{K}[\mathbf{I}|\mathbf{0}]$.
5: Solve the nonlinear system of equation in (4.7) using the Levenberg-Marquardt algorithm
6: Unnormalize the solution.

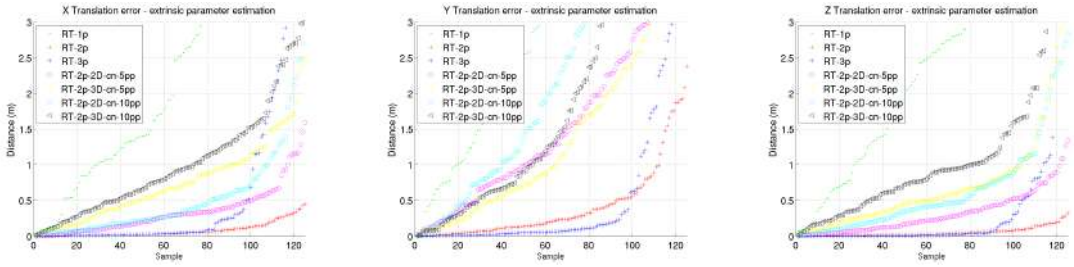


Figure 4.1: Extrinsic calibration results (from left-to-right) the translation errors along X, Y and Z axis.

4.4 Evaluation on synthetic data

For the quantitative evaluation of the proposed method, we generated a benchmark set containing 29 different shapes and their transformed versions, a total number of 2D-3D data pairs exceeding 2500 samples divided into different test setups. The 3D-2D image pairs were generated in the following way: The 3D image were generated by placing 2D planes in the 3D Euclidean space and the shapes were placed on these planes, whose size was normalized into a 1m^3 cube, with a random initial rotation in the range of $-\frac{\pi}{16},\ldots,\frac{\pi}{16}$ and a depth of 10m. For images with 2 and 3 planes, a similar procedure was applied and the placement of planes relative to each other was initially perpendicular followed by a random translation of $5-15$m and rotation of $-\frac{\pi}{16},\ldots,\frac{\pi}{16}$. The initial placement of the 3D planes was considered by taking into account common urban structural properties, in which environment the real data experiments were performed.

The synthetic 2D images of the 3D data were generated with a camera being rotated in range of $-\frac{\pi}{4},\ldots,\frac{\pi}{4}$ and the random displacement of $2-10$m. The camera calibration matrix $\mathbf{K}$ used for projection had a random focal length f_x, f_y in the range of $400-1600$ with a 5% difference between f_x and f_y, and the principal point $\mathbf{o}$ was set to the center of the 1024×768 image plane with an added 5% random variation. Thus random 2D projections were obtained with the so defined random camera matrix $\mathbf{P}=\mathbf{KR}[\mathbf{I}|\mathbf{t}]$.

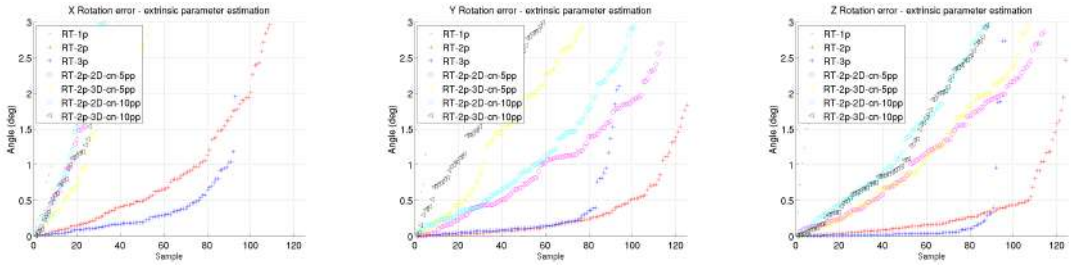


Figure 4.2: Extrinsic calibration results, including (from left-to-right) the rotation errors around X, Y and Z axis.

In practice, the planar regions used for calibration are segmented out from the lidar and camera images. In either case, we cannot produce perfect shapes, therefore robustness against segmentation errors was also evaluated on simulated data: we randomly added or removed squares uniformly around the boundary of the shapes, both in 3D and 2D, yielding an error around the contour of 5% and 10% of the original shape. Using these corrupted images, we tested the robustness with respect to 3D errors using the 3D contour noise corrupted images and corresponding noise-less 2D projections as well as robustness with respect to 2D image errors using 2D contour noise corrupted images and their noise-less 3D images.

The algorithm was implemented in Matlab and all test cases were run on a standard quad-core PC. Calibration errors were characterized in terms of the percentage of non-overlapping area of the reference and registered images (denoted by δ), the Frobenius norm of the difference between the found and the true camera matrices, as well as differences in the external parameters.

4.4.1 Extrinsic parameter estimation

In this test case the results for extrinsic parameter estimation using a virtual camera with known $\mathbf{K}$ is presented. The plots contain information about the test cases with 1, 2, and 3 planes used for calibration, as well as the robustness test results with corrupted 3D and 2D regions.

In Fig. 4.1 the result for translation error between the camera and lidar coordinate frames is presented. The translation error is not the same on the different axes: the largest is on the Y axis, as in this direction the resolution of the camera was lower than along the X axis. The best results were achieved with the 3 plane set as this is the most constrained setup for the camera-lidar position. For the majority of the cases the error was less than 3cm for a range of 10m. Segmentation errors increase these errors, but it is robust enough up to 10% error level.

In Fig. 4.2 the result for rotation error between the camera and lidar coordinate frames is presented. The results are not the same for the different axes. The most

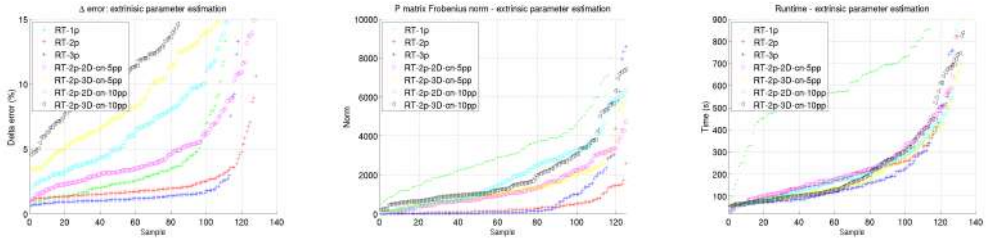


Figure 4.3: Synthetic data calibration results for extrinsic parameter estimation, including the δ error, the Frobenius norm and the runtime

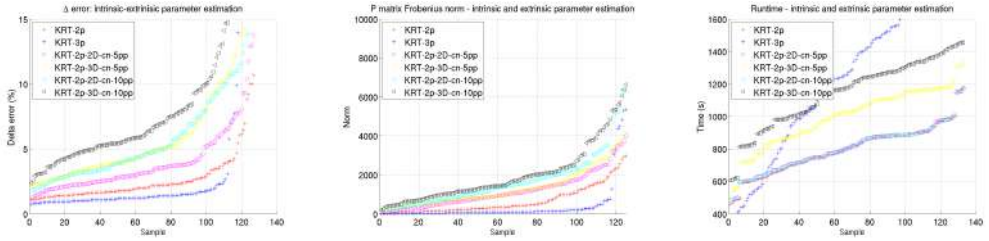


Figure 4.4: Intrinsic-extrinsic calibration results (from left-to-right) the δ error, the Frobenius norm and the runtime.

robustness is manifested along the Z axis rotation: this is not surprising as this axis is perpendicular to the observation plane, hence rotation causes only a minimal distortion on the image. Other axes are more affected by error, but for more than 70% of the cases the error was lower than 0.5 degrees on each axis, which is quite good for a 45 degrees rotation range.

The registration error δ and the runtime together with the Frobenius norm of the projection matrix is shown in Fig. 4.3. The contour noise on the 3D template, which can be interpreted as depth noise, causes a larger error than segmentation errors in the 2D camera image. This is visible in the δ error plot as well as in the Frobenius norm. The runtime for the majority of the cases is less than 2min.

Figure 4.5: Calibration example with real data of an outdoor environment (left-right): 3D data with the selected region (yellow); color 2D data with the corresponding region (green); color information overlaid on 3D data using the extrinsic parameter estimates

4.5 Real data experiments

Next, we present some calibration results on real data. The planar regions are simple rectangular regions segmented in the 2D and 3D domains. The lidar and camera images are taken from different viewpoints.

Color images were taken by a standard camera of 1024×768 resolution with prior calibration and radial distortion removal. The IR image is taken by an industrial Flir camera of 240×240 resolution with unknown intrinsic parameters. The images were captured at night time in order to reduce the effect of solar heat on the building facades.

It is important to highlight that the conventional internal calibration for such a camera is not trivial, calibration of IR and depth data is rather cumbersome, often special setups are needed both for intrinsic and extrinsic calibration Borrmann *et al.* (2012). Thus we applied the intrinsic-extrinsic calibration for the IR experiment.

4.5.1 Comparison using public datasets

For the performance evaluation of the proposed method a calibration on the public KITTI dataset (`http://www.cvlibs.net/datasets/kitti/`) was considered, which contained also the ground truth for the camera-lidar devices. In Fig. 4.5, the extrinsic calibration of a color camera with known $\mathbf{K}$ matrix and sparse 3D

Tf.	t_x	t_y	t_z	roll	pitch	yaw
Prop.	0.011	0.029	0.38	1.4	1.9	1.5
Norm. Taylor & Nieto (2012)	0.014	0.036	0.41	1.5	2.3	1.6
Int. Taylor & Nieto (2012)	0.007	0.026	0.18	0.8	1.2	0.9

Table 4.1: Comparative results with the proposed method (Prop), normal based MI(Norm)Taylor & Nieto (2012) and intensity based MI (Int)Taylor & Nieto (2012).

lidar data from the drive $nr = 5$ is shown.

In order to evaluate the accuracy of the registration, the transformation parameters were compared against the ground truth values. Also the calibration test was performed using the mutual information extraction described in Taylor & Nieto (2012) for 3D data with intensity and normal information. The calibration results are summarized in the Table 4.1. The results of the proposed method (Prop.) proved to be sensitive to the segmentation accuracy, nevertheless the registration both visually and numerically was accurate in the range of few millimeters translation and around 1 degree rotation. The mutual information based method with lidar intensity (Int.) data gave smaller absolute errors but this method using only depth data (Norm.) became quite sensitive to local minimals and the final calibration error was larger. Also the runtime GPU implementation of the mutual information method is with an order of magnitude slower than the CPU implementation of the proposed algorithm.

4.6 Catadioptric Camera Case

4.6.1 Omnidirectional camera model

A unified model for central omnidirectional cameras was proposed by Geyer and Daniilidis Geyer & Daniilidis (2000), which represents central panoramic cameras as a projection onto the surface of a unit sphere. This formalism has been

adopted and models for the internal projection function have been proposed by Micusik Mičušík (2004); Mičušík & Pajdla (2004) and subsequently by Scaramuzza Scaramuzza *et al.* (2006a) who derived a general polynomial form of the internal projection valid for any type of omnidirectional camera. In this work, we will use the latter representation. Let us first see the relationship between a point $\mathbf{x}$ in the omnidirectional image $\mathcal{I}$ and its representation on the unit sphere $\mathcal{S}$ (see Fig. 4.6). Note that only the half sphere on the image plane side is actually used, as the other half is not visible from image points. Following Scaramuzza *et al.* (2006a,b), we assume that the camera coordinate system is in $\mathcal{S}$, the origin (which is also the center of the sphere) is the projection center of the camera and the z axis is the optical axis of the camera which intersects the image plane in the *principal point*. To represent the nonlinear (but symmetric) distortion of central omnidirectional optics, Scaramuzza *et al.* (2006a,b) places a surface g between the image plane and the unit sphere $\mathcal{S}$, which is rotationally symmetric around z. The details of the derivation of g can be found in Scaramuzza *et al.* (2006a,b). Herein, as suggested by Scaramuzza *et al.* (2006b), we will use a fourth order polynomial $g(\|\mathbf{x}\|) = a_0 + a_2\|\mathbf{x}\|^2 + a_3\|\mathbf{x}\|^3 + a_4\|\mathbf{x}\|^4$ which has 4 parameters representing the internal parameters (a_0, a_2, a_3, a_4) of the camera (only 4 parameters as a_1 is always 0 Scaramuzza *et al.* (2006b)). The bijective mapping $\Phi : \mathcal{I} \rightarrow \mathcal{S}$ is composed of 1) lifting the image point $\mathbf{x} \in \mathcal{I}$ onto the g surface by an orthographic projection

$$\mathbf{x}_g = \begin{bmatrix} \mathbf{x} \\ a_0 + a_2\|\mathbf{x}\|^2 + a_3\|\mathbf{x}\|^3 + a_4\|\mathbf{x}\|^4 \end{bmatrix} \tag{4.10}$$

and then 2) centrally projecting the lifted point $\mathbf{x}_g$ onto the surface of the unit sphere $\mathcal{S}$:

$$\mathbf{x}_\mathcal{S} = \Phi(\mathbf{x}) = \frac{\mathbf{x}_g}{\|\mathbf{x}_g\|} \tag{4.11}$$

Thus the omnidirectional camera projection is fully described by means of unit vectors $\mathbf{x}_\mathcal{S}$ in the half space of $\mathbb{R}^3$.

Let us see now how a 3D world point $\mathbf{X} \in \mathbb{R}^3$ is projected onto $\mathcal{S}$. This is

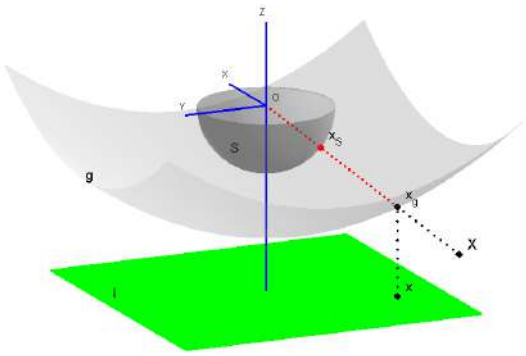


Figure 4.6: Omnidirectional camera model

basically a traditional central projection onto $\mathcal{S}$ taking into account the extrinsic pose parameters, rotation $\mathbf{R}$ and translation $\mathbf{t}$, acting between the camera (represented by $\mathcal{S}$) and world coordinate system. Thus for a world point $\mathbf{X}$ and its image $\mathbf{x}$ in the omnidirectional camera, the following holds on the surface of $\mathcal{S}$:

$$\Phi(\mathbf{x}) = \mathbf{x}_{\mathcal{S}} = \Psi(\mathbf{X}) = \frac{\mathbf{RX}+\mathbf{t}}{\|\mathbf{RX}+\mathbf{t}\|} \tag{4.12}$$

4.7 Pose Estimation

Consider a Lidar camera with a 3D coordinate system having its origin in the center of laser sensor rotation, x and y axes pointing to the right and down, respectively, while z is pointing away from the sensor. Setting the world coordinate system to the Lidar's coordinate system, we can relate a 3D Lidar point $\mathbf{X}$ with its image $\mathbf{x}$ in the omnidirectional camera using (4.12). In practical applications, like robot navigation or data fusion, the omnidirectional camera is usually calibrated (*i.e.* its intrinsic parameters (a_0, a_2, a_3, a_4) are known) and the relative pose $(\mathbf{R}, \mathbf{t})$ has to be estimated. Inspired by Tamas & Kato (2013b), we will reformulate pose estimation as a 2D-3D shape alignment problem. Our solution is based on the correspondence-less 2D shape registration approach of Domokos *et al.* Domokos *et al.* (2012), where

non-linear shape deformations are recovered via the solution of a nonlinear system of equations. This method was successfully applied for a number of registration problems in different domains such as volume Santa & Kato (2013) or medical Mitra *et al.* (2012) image registration. In our case, however, the registration has to be done on the spherical surface $\mathcal{S}$, which requires a completely different way to construct equations.

Any corresponding $(\mathbf{X}, \mathbf{x})$ Lidar-omni point pair satisfies (4.12). Thus a classical solution of the pose estimation problem is to establish a set of such point matches using *e.g.* a special calibration target or, if lidar points contain also the laser reflectivity value, by standard intensity-based point matching, and solve for $(\mathbf{R}, \mathbf{t})$. However, we are interested in solution *without* a calibration target or correspondences because in many practical applications (*e.g.* infield mobile robot), it is not possible to use a calibration target and most lidar sensors will only record depth information. Furthermore, lidar and camera images might be taken at different times and they need to be fused later based solely on the image content.

We will show that by identifying a single planar region both in the lidar and omni camera image, the extrinsic calibration can be solved. Since correspondences are not available, (4.12) cannot be used directly. However, individual point matches can be integrated out yielding the following integral equation on the sphere $\mathcal{S}$:

$$\iint_{\mathcal{D}_\mathcal{S}} \mathbf{x}_\mathcal{S} \, \mathrm{d}\mathcal{D}_\mathcal{S} = \iint_{\mathcal{F}_\mathcal{S}} \mathbf{z}_\mathcal{S} \, \mathrm{d}\mathcal{F}_\mathcal{S} \tag{4.13}$$

$\mathcal{D}_\mathcal{S}$ and $\mathcal{F}_\mathcal{S}$ denote the surface patches on $\mathcal{S}$ corresponding to the omni and lidar planar regions $\mathcal{D}$ and $\mathcal{F}$, respectively. The above equation corresponds to a system of 2 equations, because a point on the surface $\mathcal{S}$ has only 2 independent components. However, we have 6 pose parameters (3 rotation angles and 3 translation components). To construct a new set of equations, we adopt the general mechanism from Domokos *et al.* (2012) and apply a function $\omega : \mathbb{R}^3 \to \mathbb{R}$ to both sides of the

equation, yielding

$$\iint_{\mathcal{D}_S} \omega(\mathbf{x}_S)\, d\mathcal{D}_S = \iint_{\mathcal{F}_S} \omega(\mathbf{z}_S)\, d\mathcal{F}_S \tag{4.14}$$

To get an explicit formula for the above integrals, the surface patches $\mathcal{D}_S$ and $\mathcal{F}_S$ can be naturally parametrized via Φ and Ψ over the planar regions $\mathcal{D}$ and $\mathcal{F}$. Without loss of generality, we can assume that the third coordinate of $\mathbf{X} \in \mathcal{F}$ is 0, hence $\mathcal{D} \subset \mathbb{R}^2$, $\mathcal{F} \subset \mathbb{R}^2$; and $\forall \mathbf{x}_S \in \mathcal{D}_S : \mathbf{x}_S = \Phi(\mathbf{x}), \mathbf{x} \in \mathcal{D}$ as well as $\forall \mathbf{z}_S \in \mathcal{F}_S : \mathbf{z}_S = \Psi(\mathbf{X}), \mathbf{X} \in \mathcal{F}$ yielding the following form of (4.14):

$$\iint_{\mathcal{D}} \omega(\Phi(\mathbf{x})) \left\| \frac{\partial \Phi}{\partial x_1} \times \frac{\partial \Phi}{\partial x_2} \right\| dx_1\, dx_2 = \iint_{\mathcal{F}} \omega(\Psi(\mathbf{X})) \left\| \frac{\partial \Psi}{\partial X_1} \times \frac{\partial \Psi}{\partial X_2} \right\| dX_1\, dX_2 \tag{4.15}$$

where the magnitude of the cross product of the partial derivatives is known as the surface element. Adopting a set of nonlinear functions $\{\omega_i\}_{i=1}^{\ell}$, each ω_i generates a new equation yielding a system of ℓ independent equations. Although arbitrary ω_i functions could be used, power functions are computationally favorable Domokos *et al.* (2012):

$$\omega_i(\mathbf{x}_S) \;=\; x_1^{l_i} x_2^{m_i} x_3^{n_i}, \text{with } 0 \;\leq\; l_i, m_i, n_i \;\leq\; 2 \text{ and } l_i + m_i + n_i \;\leq\; 3 \tag{4.16}$$

Hence we are able to construct an overdetermined system of 15 equations, which can be solved in the *least squares sense* via a standard *Levenberg-Marquardt* algorithm. The solution directly provides the pose parameters of the omni camera. To guarantee an optimal solution, initialization is also important. In our case, a good initialization ensures that the surface patches $\mathcal{D}_S$ and $\mathcal{F}_S$ overlap as much as possible. This is achieved by computing the centroids of the surface patches $\mathcal{D}_S$ and $\mathcal{F}_S$ respectively, and initializing $\mathbf{R}$ as the rotation between them. Translation of the planar region $\mathcal{F}$ will cause a scaling of $\mathcal{F}_S$ on the spherical surface. Hence an initial $\mathbf{t}$ is determined by translating $\mathcal{F}$ along the axis going through the centroid of $\mathcal{F}_S$ such that the area of $\mathcal{F}_S$ becomes approximately equal to that of $\mathcal{D}_S$. The summary

of the proposed algorithm with the projection on the unity sphere is presented in Algorithm 4.2.

Algorithm 4.2 The proposed calibration algorithm.

Input: 3D point cloud and 2D omnidirectional binary image representing the same region, and the g coefficients

Output: External Parameters of the camera as $\mathbf{R}$ and $\mathbf{t}$

1: Back-project the 2D image onto the unity sphere.
2: Back-project the 3D template onto the unit sphere.
3: Initialize the rotation matrix $\mathbf{R}$ from the centroids of the shapes on sphere.
4: Construct the system of equations of (4.13) with the polynomial ω_i functions.
5: Solve the set of nonlinear system of equation in (4.15) using the Levenberg-Marquardt algorithm

4.8 Evaluation on synthetic data

For the quantitative evaluation of the proposed method, we generated a benchmark set using 30 different shapes as 3D planar regions and their omnidirectional images taken by a virtual camera, a total of 150 2D-3D data pairs. The synthetic omni images were generated with a virtual camera being rotated in the range of $(-40^\circ \cdots +40^\circ)$ and randomly translated in the range of $(0 \cdots 200)$, equivalent to $(0 \cdots 1.4)$ meters, if we consider the template size as a $5m \times 5m$ rectangle in 3D space.

The algorithm was implemented in Matlab and all experiments were run on a standard quad-core PC. Calibration errors were characterized in terms of the percentage of non-overlapping area of the reference 3D shape and the backprojected omni image (denoted by δ), as well as the error in each of the estimated pose parameters.

In practice, the planar regions used for calibration are segmented out from the lidar and omni images. In either case, we cannot produce perfect shapes, therefore robustness against segmentation errors was also evaluated on simulated data (see

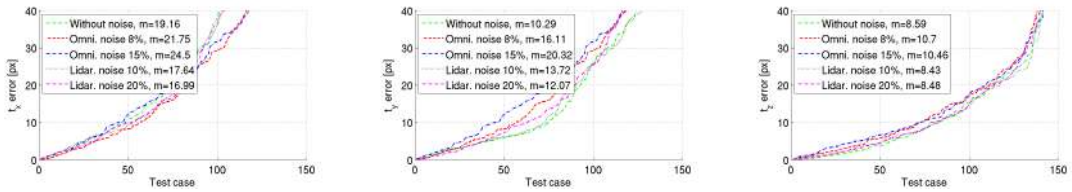


Figure 4.7: Translation errors along the x, y, and z axis. m denotes median error, *Omni. noise* and *Lidar. noise* stand for contour error on the omni and 3D regions, respectively. (best viewed in color)

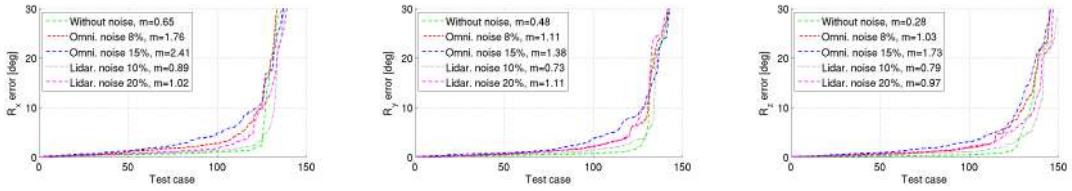


Figure 4.8: Rotation errors along the x, y, and z axis. m denotes median error, *Omni. noise* and *Lidar. noise* stand for contour error on the omni and 3D regions, respectively (best viewed in color)

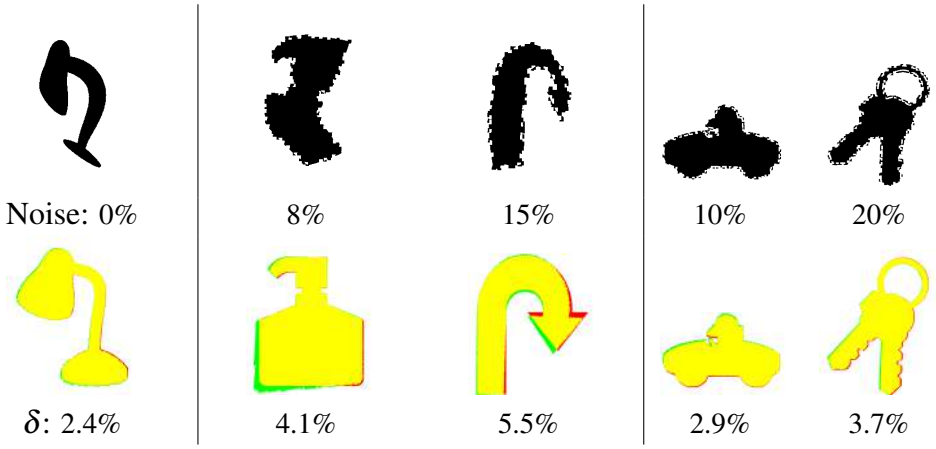


Figure 4.9: Synthetic contour noise examples. First column contains a result without noise, the next two show contour noise on the omnidirectional image, while the last two on the 3D planar region. The second row shows the δ error and the backprojected shapes overlayed in green and red colors.(best viewed in color)

samples in Fig. 4.9): we randomly added or removed squares uniformly around the boundary of the shapes, both in the omni images and on the 3D planar regions, yielding an error around the contour of 5% -20% of the original shape.

Quantitative comparisons in terms of the various error plots are shown in Fig. 4.7, Fig. 4.8, (each testcase is sorted independently in a best-to-worst sense). Note that our method is quite robust against segmentation errors up to 15% error level.

4.9 Experimental validation

For the experimental validation of the proposed algorithm two different omnidirectional camera setups are shown. In order to fuse the omnidirectional camera data and the lidar scan both the internal and external parameters are needed. The internal parameters of the omnidirectional camera were determined using the toolbox of Scaramuzza *et al.* (2006b). This internal calibration needs to be performed only once for a camera, and the as output of this calibration process the four polynomial

parameters of the camera was considered. The external parameters are computed using the proposed algorithm.

4.9.1 Urban data registration results

The input data with the segmented regions as well as the results are shown in Fig. 4.10 for a catadioptric-lidar and dioptric-lidar camera pairs respectively. As test data was acquired in an urban environment, for segmentation common regions like windows or doors could be used both in 2D and 3D data Goron *et al.* (2010b).

The omnidirectional images were captured with a commercial SLR camera with a catadioptric lens and a omnidirectional mirror respectively. For the 3D urban scene a custom lidar was used to acquire data similar to the one described in Tamas & Majdik (2012b) with an angular resolution up to half degree and a depth accuracy of $1cm$. After the raw data acquisition the segmentation was performed in both domains. For the 3D data after determining the parametric equation and boundaries of the selected region, this was sampled with a uniform sampling in order to get a homogeneous set of pointcloud. This initial set of point was transformed with R_0 and t_0 into the $Z = 0$ plane, and the transformed point coordinates represent the input for the right hand side of the (4.15) denoted with $\mathbf{X}$. The $\mathbf{x}$ variable of the left hand side of equation (4.15) is fed with the point coordinates of the segmented omnidirectional image.

Once the output R_a and t_a is obtained from Algorithm 4.2, the final transformation acting between the lidar and omni camera can be computed as a composite rigid transformation using R_a, t_a and R_0, t_0 respectively. The final computed transformation was used to fuse the depth and color data by reprojecting the pointcloud on the image plane using the internal and external camera parameters, and thus obtaining the color for each point in the 3D pointcloud. The method proved to be robust against segmentation errors, but a sufficiently large overlap between the regions is required for better results.

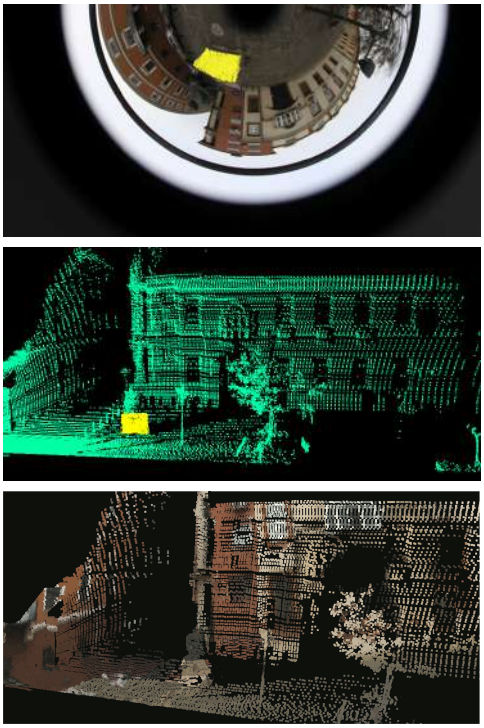


Figure 4.10: Catadioptric and lidar images with segmented area marked in yellow, and the fused images after pose estimation. (best viewed in color)

4.10 Summary and Outlook

A nonlinear explicit correspondence-less calibration method was proposed in this work. The calibration is based on the 3D-2D registration of a common lidar-camera region. The proposed method uses minimal information (only depth data and shape of regions) and is general enough to be used in a great variety of applications. It has been tested on a large synthetic dataset. The algorithm was also validated in real life experiments with different cameras and with both extrinsic and intrinsic-extrinsic calibration experiments. In this paper a new method for pose estimation of non-conventional cameras is proposed. The method is based on a correspondence-less registration technique, which allows reliable estimation of extrinsic camera param-

eters. The algorithm was quantitatively evaluated on a large synthetic data set and proved to be robust on real data as well.

Chapter 5

Relevant Applications to 3D Position Estimation

This chapter presents some indoor and outdoor applications related to the perception of the environment. The theoretical background for these applications is detailed in Chapter while the information related to the depth processing is based on Chapter .

In the first part of this chapter are introduces a navigation application with a UAV in a structured indoor environment. Further on, the motion planning based on parsed 3D environment information is presented for a single and dual arm robot. In the last part of this chapter the details for the position estimation in a augmented reality based application are presented Blaga & Tamas (2018); Militaru *et al.* (2016a); Páll *et al.* (2015).

5.1 UAV Navigation in Structured Environments

5.1.1 Introduction

Unmanned aerial vehicles (UAV) are being increasingly investigated not only for military uses, but also for civilian applications. Famous recent examples are UAV

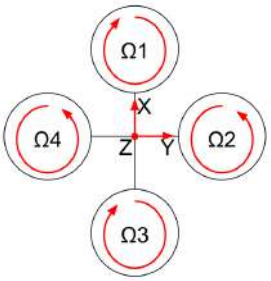


Figure 5.1: Simple quadcopter schematics, where Ω_1, Ω_2, Ω_3, and Ω_4 are the propellers rotational speed.

applications for package delivery services, including those by online giant Amazon[1] and in the United Arab Emirates (Pavlik, 2014), or Deutsche Bahn's exploration of UAVs to control graffiti spraying[2]. More classical applications have long been considered, such as search and rescue (Beard *et al.*, 2013; Tomic *et al.*, 2012) or monitoring and fertilizer spraying in agriculture (Odido & Madara, 2013). Motivated by this recent interest, we focus here on the automation of a low-cost quadcopters such that they can perform tasks in structured environments without human interaction. We employ the AR.Drone, a lightweight UAV widely used in robotic research and education (Benini *et al.*, 2013; Krajník *et al.*, 2011; Stephane *et al.*, 2012).

5.1.2 Quadcopter Structure and Control

Quadcopters are classified as rotorcraft aerial vehicles. These mobile robots are frequently used in research because of their high maneuverability and agility.

The quadcopter's frame can be shaped as an **x** or **+** and the latter is presented in Figure 5.1. A brushless direct current motor is placed at each end of the configuration. The motors are rotating the fixed pitch propellers through a gear reduction system in order to generate lift. If all the motors are spinning with the same speed, $\Omega_1 = \Omega_2 = \Omega_3 = \Omega_4$, and the lift up force is equal with the weight of the quad-

[1] Amazon testing drones for deliveries, BBC News 2013
[2] German railways to test anti-graffiti drones. BBC News 2013

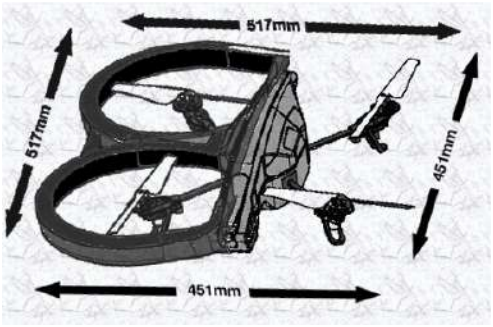


Figure 5.2: Parrot AR.Drone schematics with and without indoor hull[1].

copter, then the drone is hovering in the air. This motor speed is called hovering speed Ω_h. Four basic movements can be defined in order to control the flight of the quadcopter.

5.1.3 Quadcopter Hardware and Software

The chosen quadcopter is the Parrot AR.Drone 2.0, see Figure 5.2 for a schematic. It is a low-cost but well-equipped drone suitable for fast development of research applications (Krajník *et al.*, 2011). The drone has a 1 GHz 32 bit ARM Cortex A8 processor with dedicated video processor at 800 MHz. A Linux operating system is running on the on board micro-controller. The AR.Drone is equipped with an IMU composed of a 3 axis gyroscope with 2000 °/second precision, a 3 axis accelerometer with $\pm$ 50 mg precision, and a 3 axis magnetometer with 6° precision. Moreover, it is supplied with two cameras, one at the front of the quadcopter having a wide angle lens, 92° diagonal and streaming 720p signal with 30 fps. The second camera is placed on the bottom, facing down. For lower altitude measurements an ultrasound sensor is used, and for greater altitude a pressure sensor with $\pm$ 10 Pa precision is employed.

Parrot designed the AR.Drone also for hobby and smart-phone users. Therefore,

[1] image based on: http://ardrone2.parrot.com/ardrone-2/specifications/

the drone has a WiFi module for communication with mobile devices. The stabilization and simple flight control (roll, pitch, yaw, and altitude) is already implemented on the micro-controller and it is not recommended to make changes in the supplied software. The quadcopter can stream one camera feed at a time together with all the rest of sensor data.

5.1.4 Methodological and Theoretical Background

In this section, we present the methods taken from the literature that we employ in this research project, with their theoretical background. Our goal is to fly the drone autonomously in corridor-like environments by using only the sensors on the AR.Drone.

In the mobile robotics field, the automation of vehicles needs to solve the localization problem. In order to decide which type of localization should be used, the available sensors must be known. The AR.Drone is not necessarily supplied with global positioning sensors (GPS) and even if it is, the GPS can not be used indoor. Hence, we chose vision-based position tracking based on feature detection, which is presented in Section 5.1.5. This method will supply information about the relative position of the drone and the target.

In particular, the desired flight direction is represented by the vanishing point, a 2D point on the image. The position of the detected point is going to be tracked, because it changes in a dynamic fashion due to the drone motion, and it is affected by noise on the images. The chosen tracking methods are probabilistic state estimator filters, detailed in Section 5.1.6. These methods support sensor fusion, meaning that information from different sensors is combined. In our case, we fuse visual information with IMU measurements, using to enhance the precision of the estimation.

5.1.5 Feature detection

The AR.Drone has two color cameras, one at the bottom facing down and the other at the front, looking ahead. For the experiments we use the front camera.

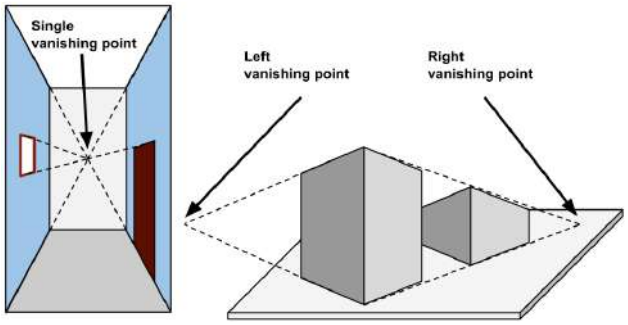


Figure 5.3: Single VP (left) detected in a corridor and multiple VPs (right) detected when buildings are present.

The localization procedure should be general in the sense that no artificial changes, such as landmarks should be made to the environment in order to track the position of the quadcopter. Hence, the perspective view's characteristics are analyzed on the 2D image. One of the basic features of a 2D projection of 3D space is the appearance of vanishing points (VPs). The interior or the outdoor view of a building has many parallel lines and most of these lines are edges of buildings. Parallel lines projected on an image can converge in perspective view to a single point, called VP, see Figure 5.3. The flight direction of the quadcopter is determined by this 2D point on the image.

In order to obtain the pixel coordinates of the VP, first the image is pre-processed, next the edges of the objects are drawn out, and finally the lines are extracted based on the edges.

Pre-processing In order to process any image, we must first calibrate the camera. Distortions can appear on the image due to the imperfections of the camera and its lens. Our project is based on gray-scale images processing. Therefore we are not focusing on the color calibration of the camera, and only spatial distortions are handled. The calibration parameters are applied on each frame. The calibration process has two parts: intrinsic and extrinsic camera parameters.

The general equation which describes the perspective projection of a 3D point

Figure 5.4: Camera calibration, on the left the barrel distorted image, on the right the calibrated image.

on a 2D plane is:

$$s \cdot \begin{bmatrix} u \\ v \\ 1 \end{bmatrix} = \vec{K}[\vec{R}|\vec{t}] \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix} ; \text{and } \vec{K} = \begin{bmatrix} f_x & \gamma & c_x \\ 0 & f_y & c_y \\ 0 & 0 & 1 \end{bmatrix}$$

where $(u,\ v)$ are the projected point coordinates, s is the skew factor, and $\vec{K}$ is a camera matrix with the intrinsic parameters, focal length f_x, f_y and the optical centers c_x, c_y. These five parameters must be defined for a specific resolution of the camera. $[\vec{R}|\vec{t}]$ is a joint rotational-translational matrix with the extrinsic parameters, and $(X,\ Y,\ Z)$ are the 3D coordinates of the point in the world frame.

The spatial distortion is corrected with the extrinsic parameters. Barrel distortion (Zhang, 2000) appears on the front camera, because the camera is equipped with a wide angle lens. The distortion appears when an image is mapped around a sphere, as shown on Figure 5.4. The straight lines are curved on the distorted image (left) which is unacceptable for line-detection-based position tracking. The barrel effect has a radial distortion that can be corrected with:

$$\begin{aligned} x_c^b &= x_u(1 + k_1 r^2 + k_2 r^4 + k_3 r^6) \\ y_c^b &= y_u(1 + k_1 r^2 + k_2 r^4 + k_3 r^6) \end{aligned} \tag{5.1}$$

where (x_u, y_u) are the coordinates of a pixel point on the distorted image, $r = x_u^2 + y_u^2$,

and (x_c^b, y_c^b) is the radial corrected position.

The tangential distortion appears when the lens is not perfectly parallel to the sensor in the digital camera. The correction of this error is expressed as:

$$\begin{aligned} x_c^t &= x_c^b + [2p_1 x_u y_u + p_2(r^2 + 2x_u^2)] \\ y_c^t &= y_c^b + [p_1(r^2 + 2y_u^2) + 2p_2 x_u y_u] \end{aligned} \tag{5.2}$$

where (x_c^t, y_c^t) is the tangential corrected position.

The extrinsic distortion coefficients appear in (5.1) and (5.2), where k_n is the n^{th} radial distortion coefficient and p_n is the n^{th} tangential distortion coefficient.

After obtaining the calibrated image some low-level intermediate pre-processing can be applied. In particular, we will use:

- smoothing, which eliminates noise and small fluctuations on images, it is equivalent with a low-pass filter in the frequency domain, and its drawback is blurring the sharp edges;
- gradient operators, which sharpen the image and act like a high-pass filter in the frequency domain.

Edge detection is based on gradient operators and it is considered to be a pre-processing step. An edge on a 2D image can bee seen as a strong change of intensity between the two surfaces i.e. as a first derivative maximum or minimum. For easier detection, a gradient filter is applied on a gray-scale image. As this is a high-pass filters, gradient operator also increases the noise level on the image. We choose the Canny algorithm, it is a multi-step edge detector. First, it smooths the image with a Gaussian filter, and then finds the intensity gradient of the image by using the Sobel method. The Sobel algorithm is a first order edge detector, and it performs a 2D spatial gradient on the image.

Line detection Sharp and long edges can be considered as lines on a gray-scale image. The most frequently used line detector is the Hough Transformation (HT)

(Kiryati *et al.*, 1991). This algorithm performs a grouping of edge points to obtain long lines from the output of the edge detector algorithm. The preliminary edge detection may be imperfect such as presenting missing points or spatial deviation from the ideal line. The HT method is based on the parametric representation of a line: $\rho = x\cos\theta + y\sin\theta$ where ρ is the perpendicular distance from the origin to the line and θ is the angle between the horizontal axis and the perpendicular line to the line to be detected. Both variables are normalized such that $\rho \geq 0$ and $\theta \in [0,\ 2\pi)$.

The family of lines, $L_{x_0 y_0}$, going through a given point (x_0, y_0) can be written as a set of pairs of $(\rho_\theta,\ \theta)$. Based on our previous normalization, $L_{x_0 y_0}$ can be represented as a sinusoid for the function $f_\theta(\theta) = \rho_\theta$. If two sinusoidal curves for $L_{x_a y_a}$ and $L_{x_b y_b}$ are intersecting each other at a point $(\rho_{\theta_{ab}},\ \theta_{ab})$, then the two points $(x_a\ y_a)$ and $(x_b\ y_b)$ are on the line defined with the parameters, $\rho_{\theta_{ab}}$ and θ_{ab}. The algorithm searches for intersections of sinusoidal curves. If the number of curves in the intersection is more than a given threshold, then the pair (ρ_θ, θ) is considered to be a line on the image.

The Probabilistic Hough Transform (PHT) is an improvement of HT (Stephens, 1991). The algorithm takes a random subset of points for line detection, therefore the detection is performed faster.

5.1.6 Feature tracking

The true position of a moving object cannot be based only on observation, because measurement errors can appear even with the most advanced sensors. Hence, the position should be estimated based not only on the measurements, but also taking in consideration other factors such as the characteristics of the sensor, its performance and the dynamics of the moving object. Position tracking of a moving object with a camera is a well studied problem (Peterfreund, 1999; Schulz *et al.*, 2001). In our case the estimators are used to reduce the noise in the VP detection and to track the position of the VP with higher confidence.

The estimation process can be enhanced by sensor fusion, meaning that different types of data are combined in order to have a more precise estimation of the current state. The 2D coordinates of the VP on the video feed is one measurement data and the other is the yaw orientation angle of the drone w.r.t. the body frame obtained from the IMU.

5.1.7 Experimental part

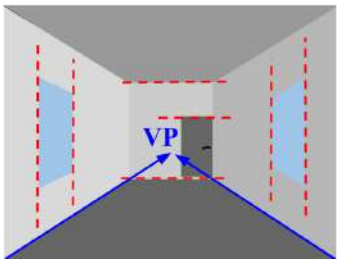
The corridor following problem was described in (Bills *et al.*, 2011; Gomez-Balderas *et al.*, 2013). A quadcopter should be able to fly through a corridor-like environment by using in this problem only visual perception and the IMU, without using any additional tags or changes in the environment. We extended the solution (Lange *et al.*, 2012) with different filtering algorithms and a simple control strategy, in order to make the quadcopter autonomous not only in indoor but also in outdoor corridor-like environments.

5.1.8 Perspective UAV vision

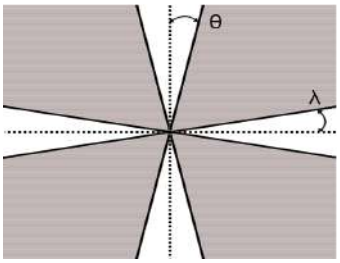
Earlier, we proposed to use the vanishing point (VP) in order to detect the direction in which the quadcopter should fly. The VP is, by definition, the intersection of parallel lines viewed in perspective. On each frame the edges of the objects are detected as lines.

Image processing Our image processing algorithm finds the VP, which indicates the end of the corridor. The input is the undistorted image and the output is the measurement of the horizontal location of the VP. The image processing algorithm has three steps: edge detection, line detection, and filtering.

For edge detection we used the Canny (Canny, 1986) algorithm, see Section 5.1.5, which is known to have good detection, good localization and minimal response. The detection is considered to be good because the probability is low for



(a) The red dashed lines are not useful to find the direction of the corridor(left) or the direction of the railway (right).



(b) The recognized lines with orientation angles θ and λ less than 10° are neglected in the VP detection procedure.

Figure 5.5: Illustration of VP and VP filtering.

Figure 5.6: VP detection in corridors: darker gray rectangle is the VP observation, and the lighter gray rectangle is the estimate of the VP, with the white lines are the detected edges, see Section 5.1.9.

both marking false edges and failing to mark real ones. Moreover, the good localization criterion means that the marked edge is close to the center of the real edge. The minimum response property means that an edge should be marked only once, while image noise should not return false detection. The input of the Canny algorithm is a grayscale version of the original image and the output image will contain only the detected edges.

The next phase handles the extraction of long edges by using the PHT algorithm presented in Section 5.1.5. These lines are filtered by their orientation angle, because not all the edges on an image are useful to determine the flight direction, as shown in Figure 5.5a by the red dashed lines. We considered the almost horizontal and vertical lines as noise (not useful for VP detection) in the line detection phase.

Hence, all the detected lines having angles in $0 \pm \lambda$, $\pi \pm \lambda$ or $\frac{\pi}{2} \pm \theta$ and $-\frac{\pi}{2} \pm \theta$ are neglected, as presented in Figure 5.5b where the tuning parameters, λ and θ are less or equal with 10°. Furthermore, we divided the image plane into a 23×33 grid and searched for the cell which contains the most intersections obtained from the filtered set of lines. This cell, also shown in Figure 5.6, has high probability to contain the VP. The center of the cell is the observed VP, and will be used as an input for the *Estimator* node.

The PHT threshold refers to the minimum number of detected intersections, see Section 5.1.5 for details. The minimum line length is the threshold for the number of points that can form a line. The maximum line gap is the only parameter that has to be changed for the two different environments: indoor and outdoor. This parameter limits the possible gap between two lines, for which they can be considered as one line.

Ground truth detection It is essential to have a ground truth, in order to evaluate and compare different approaches and tuning setups. In our case, we must know where exactly is the end of the hall on each frame. For this reason, we decided to place an easily distinguishable object down the hall. The Ground truth detection is not part of our automated flight method. We are using it only in offline mode, to process the logged flight data and evaluate the algorithm.

The tag is composed of tree colored stripes with the same size: the middle one is yellow and the two on the sides are orange. First, the calibrated image goes trough a noise reduction process because of the different brightness values for the same color, while preserving edges. In the next phase we identify the orange and yellow regions with their properties: orientation, centroid, contour, and area. Then we search for the regions of orange, yellow, and orange color ”code” as mentioned above. The centroid of the yellow rectangle is considered to be the location of the end of the corridor.

5.1.9 VP tracking

The observed VP coordinate is processed by the *Estimator* node and the output is the estimated horizontal position of the VP. We need position estimation to filter out faulty VP detection caused by noise on the image and to perform fusion. We are using the three motion models, see Section 5.1.6 to describe the horizontal movement of the VP measurement. The linear Kalman filter is implemented with the linear constant velocity model. The motion of the VP is highly nonlinear, therefore we implemented nonlinear filters: the extended Kalman filter, the unscented Kalman filter, and the particle filter. The nonlinear filters were tested with the linear and the two nonlinear models, specifically the constant velocity model extended with the yaw orientation angle and the constant acceleration model.

5.1.10 Control

After having the estimated location of a mobile robot, it should perform its task, in our case the quadcopter should fly along the desired path. The motion of the robot must 1be regulated with a controller.

We chose a simple strategy,

$$u_k = \begin{cases} 0 & \text{, if} \quad e_k < |tr| \\ \text{sign}(e_k) \cdot |u_{max}| & \text{,} \quad \text{other case} \end{cases}$$

where, u_k is the control signal at time k, e_k is the error, and u_{max} is the maximum value of the control signal and tr is a threshold value.

We aim to control the horizontal position of the VP on the image, which consequently will generate the desired trajectory and our control strategy is based on the switching method (Liberzon, 2003). We are switching between 2 controllers, one with higher priority and one with lower priority. The high priority controller regulates the orientation of the drone and it is called **yaw control**. The low priority controller regulates the lateral translational position adn it is called **flight control**.

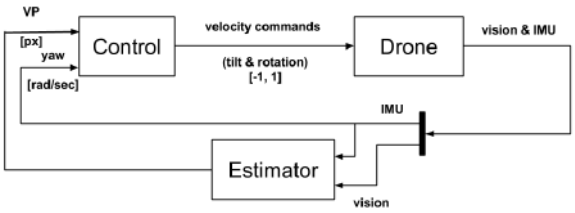


Figure 5.7: The control loop is composed of the quadcopter as the system , the estimator, and the controller.

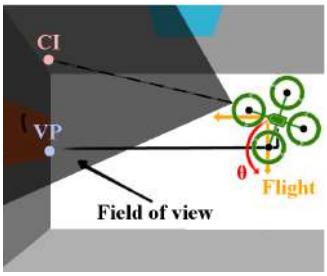


Figure 5.8: Drone controlled with the switching method. First, the drone corrects the θ angle (red arrow) while hovering. Next, the roll angular velocity is controlled, which produces a lateral translation, in function of the distance between the vanishing point (VP) and the center of the image (CI), while the drone flies forward with a constant velocity. The trajectory (black line) approximately follows the middle of the hallway in this way.

The block diagram of the control loop is shown on Figure 5.7. The practical functionality of the controller is presented in Figure 5.8, where the quadcopter is moved from the center of the hall and its yaw angle is not aligned with the reference angle.

Yaw control

We defined the initial position of the drone, where the front camera is looking down the hall and the drone is positioned on the middle of the corridor. In the initialization stage, the rotational angle yaw around the **Z** axis, θ is fixed and considered to be the reference signal for the controller. In regular mode, the θ control holds the angle within a small range around this reference by using a maximum gain

$u_\theta = 0.02$. This is a prerequirement for the flight control sage. The measurement of θ is obtained form the navigational data. The yaw angle can suffer from drift, but it maintained fine in our experiments

Flight control

The flight controller generates the translational velocity commands, which are sent to the drone. The lateral movement is controlled with a maximum gain $u_{pitch} = 0.015$, while the forward movement is constant, $u_{roll} = 0.065$. Moreover, the controller is responsible with landing the UAV in case the image processing algorithm dose not find the end of the corridor. The drone is landed if on 20 consecutive frames no VP can be observed. This situation can happen in case the drone is at the end of the hall (facing the wall, so 0 lines detected), but this is also a safety measurement when the drone looses sight of the end of the hallway.

5.1.11 Indoor and outdoor results

In this section we compare the VP detection and estimation performance between the indoor and outdoor parameter setup in the detection phase of the algorithm. The experiments were carried out in an outdoor hallway-like environment where we compared both indoor and outdoor parameter settings. We are not investigating the performance of disturbance rejection due to different illumination and wind speed. We only aim to reduce the noise in the VP position caused by the increased texture complexity outdoor compared to an indoor scene.

The increase of the precision in estimation from the LKF to the UKF can be found in Table 5.1, where the linear simplified model was used, and the values represent the error calculated from the difference between the ground truth and the estimation, t_R is the elapsed time while one estimation is done, std is standard variation. The results are showing an increase of estimation performance for the UKF, but also reveals longer execution time weakness compared to the LKF.

Table 5.1: VP estimation performance errors with LKF and UKF, in case of linear model and outdoor flight with in- and out-door detection parameters.

	Indoor setup		Outdoor setup	
	LKF	UKF	LKF	UKF
std [px]	22.34	6.29	19.40	8.52
mean [px]	-34.73	-2.26	-11.22	2.65
mode [px]	-76.52	-17.97	-55.49	-16.43
median [px]	-30.17	-1.31	18.83	6.79
t_R [ms]	0.19	0.48	0.19	0.49

5.2 Single Robot Arm Path Planning

5.2.1 Introduction

An emerging field within robotics is that of service robots especially the ones giving the opportunity to cooperate with humans. The human-robot interaction in the current state is mainly representative in the indoor environment, where the interaction with other objects is must. The object in this context can be defined as a physically separable union which is independent from its environment, such that it can be moved separately from the rest of its surroundings. Two main directions are characterising the human-robot interaction with respect to object handling: the object perception and object manipulation. The first part is mainly focusing on the object separation, i.e. segmentation while the second part is dealing with the grasping and planning in the 3D space.

The object segmentation from its surrounding is not always a trivial task: several research work from the computer vision domain are focusing on this problem from decades. A common approach for solving this problem is the passive segmentation, the camera parameters relative to the object are not modified. However such an approach may have several drawbacks, especially when one has to deal with complex scenes, such as the human environment in general Holz *et al.* (2014). Also a com-

Figure 5.9: The 7DoF robot arm model with the 3D depth sensor mounted on the top of it

mon problem for the segmentation and scene interpretation tasks is related to the computational speed and accuracy of the object boundaries. A possible solution to overcome these limitation is the *active perception* approach which allows change in the camera parameters and even the interaction with the scene in order to gain more reliable information about this Bajcsy (1988a).

More generally, the concept of active perception can be found also in the human reasoning: one might move an object or move around an object in order to gain more information about this Herbst & Fox (2014). In case that one has a movable camera, i.e. a camera mounted on a robotic arm can take advantage of this extra degree of freedom, and can use it to gain getter representation about the surrounding objects, as this might be often needed. Nor the humans nor the surrounding scene cannot be expected to perform by itself a movement which helps the perception system to gain extra information, thus this has to be done by the robot itself.

These ideas motivate the introduction and usage of the active perception concept for the human-robot interaction Sankaran *et al.* (2013). Having the possibility to choose the camera external parameters for the perception system, i.e. move around the camera being mounted on a robotic arm, this leads to the ability of the robot to choose the way in which reasons about the working space.

In this paper we describe the method that we developed for an eye-in-hand mo-

bile manipulator development containing a 7 degree-of-freedom (DoF) robotic arm used for object handling. The main contribution of this work was the development of kinematic setup for this device as well as the object detection and handling in the 3D working space.

In the first part of the paper we briefly describe the state of the art methods used for object detection and handling in indoor environment with mobile robotic manipulators. Then we describe the details of the setup that we used in order to develop the necessary real life experimental setup. Further on we show the details both for the object detection as well as the planning approach for the object handling problem. Finally, the paper is concluded with the results of these investigations.

5.2.2 Problem description

The problem of object handling in the human-robot interaction context is far from being trivial, several aspects needs to be considered in order to have a solution to this problem Holz *et al.* (2014). Different approaches arise from either the camera mounting position with respect to the robot arm (i.e. on-board or off-board), the degree of clutter in the scene, or the planning approach (deterministic vs. probabilistic). A short overview of the related work in the main literature is presented in the next section.

5.2.3 3D active sensing

The scene perception using active sensing approach is not a new concept, it dates back at least to the 80's Bajcsy (1988a). Both the work in the 2D and in the 3D part has been extended especially in the mapping domain, where the map extension problem is referred as frontier based mapping Herbst *et al.* (2014).

In our approach we used 3D information sensing, so the in the next part we focus on the related literature. Also in the part of the research an important question is related to the best view selection from the scene, which is done for example by Potthast et al. Potthast & Sukhatme (2014), by searching each depth points back

projected ray how much contribution has in terms of information gain. An extension to a single point planning approach is used in Atanasov *et al.* (2013), where multiple position plans are computed in a single step, however the viewpoint selection and the path planning are done offline.

For dense feature matching a good overview can be found in Herbst & Fox (2014). In this paper the authors deal with a semi-complete scene map for which further information is gather in order to enhance its completeness. A similar eye-in-hand depth range sensor approach is presented in Monica *et al.* (2016): a dense map is generated with a projected light sensor and in the mean time a reasoning about the scene object is performed. This approach is similar to the one presented in the current paper too: an eye-in-hand 3D sensor for scene interpretation and object handling.

5.2.3.1 Planning with 7DoF arm

Object manipulation and grasping with robotic arm has a long history in the robotics community. There was a paradigm shift during the time from the deterministic way of planning such as the A^* towards the probabilistic planning methods in the main literature such as the Rapidly-Exploring Random Trees. A good overview of these techniques can be found in the book of LaValle LaValle (2006).

A similar problem to the one discussed in this paper can be found in the works Hirano *et al.* (2005) focusing on planning issues in a cluttered scene and Zollner *et al.* (2004) showing demonstrations for a dual-arm planning problem. Our work is based on the concepts described in Kanter (2012), which was developed within Robot Operating Systems (ROS) using the out-of-box planning algorithms such as the Open Motion Planning Library (OMPL) presented in Sucan *et al.* (2012). Some other variants for planning algorithms can be found also in Hauser (2013).

5.2.4 Arm motion planning

Starting with a 7 DoF arm the development of the the forward and inverse kinematics is not trivial any more, thus approximate or sample based solutions such as the Rapidly–Exploring Random Tree (RRT) are often used LaValle (2006). This is why we adopted such a solution for our setup too.

5.2.4.1 Low level interface

The low level libraries for the Cyton arm is based on a action-server architecture, which communicates with the serial chained servo motors from the joints and is interfaced to the computer via the USB port.

At the motor driver level, we reused the already available ROS Dynamixel package, which could be customised in a flexible way in order to communicate with each joint individually by assigning an identifier (ID) to each motor in part. In the next level of control we used these references to address the physical joints of the arm Fekete (2015).

5.2.4.2 Building the kinematics model

For the construction of the kinematics model of the robot arm we used two steps. The first one was building the physical structure based on the joint-link description language specific to the ROS environment. Next based on this description we configured the planning module together with the low level drivers in order to have a fully functional model for the motion planner.

For the first part of this task, i.e. the construction of the physical description we based our work on the existing models parameters from the commercial robot software package as well as the predefined universal robotics description from (URDF) for this type of arm presented in Kanter (2012). The later model contains two different module: one for the arm with rotational joint controls and one for the gripper with prismatic joint. The module controls are based on the individual joint level controllers and state feedback nodes for each node.

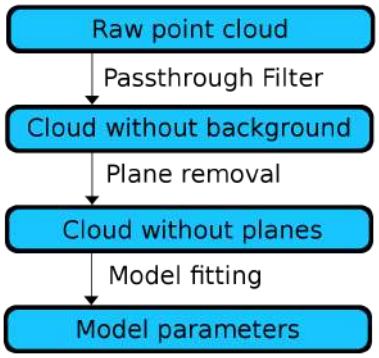


Figure 5.10: The processing pipeline for the plane extraction algorithm

5.2.5 Depth information preprocessing

Having a point cloud from the camera, we need to search for our object in it. The searched object can be described either as a mathematical model (cylinder, sphere) or as another point cloud. In both cases, we desired to retrieve from the scene only the points that belong to the object. Depending on the description of the object and the restrictions about the scene contents, the following algorithms can be considered.

5.2.6 Plane extraction

The first scene we considered was a bottle standing on a tall box in the shape of a cuboid. The bottle had a cylindrical shape and the box can be seen as a set of planes. If we eliminate the points that belong to the box, we can obtain the points belonging to the cylinder.

The box was inside a room and the camera sensed other objects, such as walls, chairs etc. that were behind the box. The arm is supposed to grab the bottle, so we can ignore the points that are outside of the arm's working space. In order to remove the background, we used a PCL pass through filter that keeps only the points that have their z coordinate between $0.5m$ and $1m$.

After the filtering, the point cloud contained only our region of interest: the box and the bottle. From this, we could extract all the planes. Then most of the remaining points were belonging to the bottle. For this purpose, we used one of the planar segmentation algorithms provided by PCL. Providing a point cloud, the algorithm detects a group of points that belong to the same plane. By repeatedly applying the algorithm, we can remove all the planes in the scene.

The next step is finding the cylinder's parameters (the cylinder axis and radius). This requires fitting the remaining points to the mathematical model of a cylinder. We used the PCL cylinder segmentation algorithm, which is similar to the planar segmentation described above. The RANSAC was chosen for the search method, the distance threshold (maximum distance between the points and the cylinder model) was set to $2cm$. We configured the searched cylinder's radius to be between $1cm - 5cm$ (the bottle's radius was about $2.5cm$).

The algorithm works the same if there is a sphere instead of the cylinder. The only difference is the fitting step, in which we use the PCL sphere segmentation, with the following parameters: RANSAC search method, the distance threshold of $2cm$ and the radius between $5cm - 10cm$ (the sphere had a radius of about $7cm$). This algorithm has best results when the scene contains planes and, by removing them, most of remaining points belong to our object.

5.2.7 Region growing segmentation

This algorithm can be used to detect the object in a more complex scene, where there are many objects, which do not necessarily have a plane surface. Our program is designed to search for a model (cylinder or sphere) in this environment.

The Point Cloud Libray (PCL) region growing segmentation algorithm divides the point cloud into disjoint sets of points which belong to the same smooth surface, called clusters. In other words, it separates the smooth surfaces from the scene.

But what is considered a smooth surface? To answer this question, the algorithm uses the concept of surface normals. Consider that a surface represents the

boundaries of a 3D object. The surface normal is defined for each point as the line perpendicular on the surface in that point. A line is perpendicular on the surface in a point if it is perpendicular on the plane that is tangent to the surface in that point. Therefore, the problem is reduced to estimating the tangent plane of the point cloud in each of its points. In order to find the tangent plane in a point, the PCL normal estimation algorithm searches for the neighboring points and finds the plane that minimizes the sum of distances from the points to that plane. The plane normal can be computed easily from the plane's coefficients.

A surface is considered to be smooth if, for each pair of two neighboring points, the tangent planes are similar, so their normals have a similar orientation. The "similarity" of normals is measured using the angle between their direction vectors.

But the angle between normals may not be enough to determine if the surface is smooth or not. Considering the following situation: a cylinder standing vertically on a plane. We would expect that the plane's points' normals to be vertical, and the cylinder's points' normals to be horizontal. But that does not happen in our case. The normals at each point are approximated using the neighbors of that point. The points of the plane that are very close to the cylinder will have their normals influenced by the points of the bottle. And the points of the cylinder that are close to the plane will have their normals influenced by the points of the plane. Therefore, the plane's points' normals will gradually become more and more horizontal as we get closer to the cylinder. Hence, if the algorithm was based only on angles between normals, it could consider the area between the two surfaces to be smooth, and therefore the cylinder and the plane would belong to the same smooth surface, which is not true.

In order to divide the scene into smooth surfaces, the algorithm uses the region growing approach. Starting with a region containing only one point, called the seed, the alogithm expands the region by adding neighboring points that satisfy the smoothness constraints. When the region cannot be extended anymore (all the neighbors would violate the smoothness constraints), the cluster is complete, and the algorithm starts again with a different seed, until there are no more points in the

cloud that have not been processed.

Having the groups of points that form smooth objects, we iterated through them, trying to fit each cluster to the cylindrical model. If most of the points of a cluster were fitted to the model, then we consider the object to be found.

5.2.7.1 Correspondence grouping detection

Another approach for object retrieval is the recent correspondence grouping one suggested in Aldoma *et al.* (2013). This method aims for detecting similarities between two or more objects. In this case we will use it to find a certain 3-D object (model) in a scene by grouping the correspondences in clusters, from which the pose will be extracted.

The descriptors for the keypoints are used to accurately determine the correspondences between points and hopefully find the match of the model in the scene, and must be robust to noise, resolution variation, translation and rotation. The algorithm that we used was SHOT (Signatures of Histograms of Orientations), which proved to be robust enough for the experimental part.

In the last step the correspondence grouping algorithm classifies all the correspondences that are thought to belong to the model into clusters, and rejecting the ones that are not. For this we have used the Hough algorithm which relies on a Hough Voting process, that outputs the rotation matrix and the translation vector. This aims at gathering evidence about the existence of the initial object in the plane by voting if enough feature correspondences are present.

Unfortunately this method seems to run for several tens of seconds which is far larger the the most unfavorable one from the two other methods, so we decided not using it for our real life experiments.

5.2.8 Experimental validation

In the first stage of our investigations we used also simulated robot arm, in order to test the object handling algorithms. In the next steps we performed out real life

experiment on a Cyton Gamma R2 seven degree of freedom robot arm and using an Asus Xtion Pro depth camera in a typical indoor environment. In the first part of the real life experiments we had to determine the external camera parameters relative to the gripper fingers of the robot arm.

5.2.8.1 Hand-eye camera calibration

The camera calibration in general has a broad interest in the computer vision and robotics community, several approaches existing for the intrinsic and extrinsic parameter estimation problems including the specific hand-eye calibration problem dating back already at the 80'sTsai & Lenz (1988).

In our approach to the calibration of the depth camera mounted on the arm and the gripper frame is based on a relative calibration to a fixed frame in space, as this is shown in Figrure 5.11. The main ideas is to use a calibration pattern as an external reference with an internally calibrated depth camera. The position of this checkboard and fixed frame with respect to the depth camera can be determined with standard methods from the computer vision field taking into account the physical size of the pattern.

Denoting with ${}_{C}\mathbf{T}^{F}$ the transformation from the camera to the fixed frame this can be used later on in computation of the ${}_{G}\mathbf{T}^{C}$ (gripper-camera transform) in a closed kinematic chain. The ${}_{B}\mathbf{T}^{G}$ transformation from the base frame to the gripper frame can be determined using the kinematic model of the robot. Further more, by moving the robot arm to at least four fixed corners of the calibration pattern, the position of the base with respect to the fixed frame ${}_{B}\mathbf{T}^{F}$ can be determined using a singular value decomposition (SVD) approach. Having all these transformations in the kinematic chain, the searched gripper camera transformation can be obtained using:

$$ {}_{G}\mathbf{T}^{C} = \left({}_{B}\mathbf{T}^{C}\right)^{-1} \times {}_{B}\mathbf{T}^{G} \tag{5.3} $$

By computing this transformation between the gripper and depth camera coor-

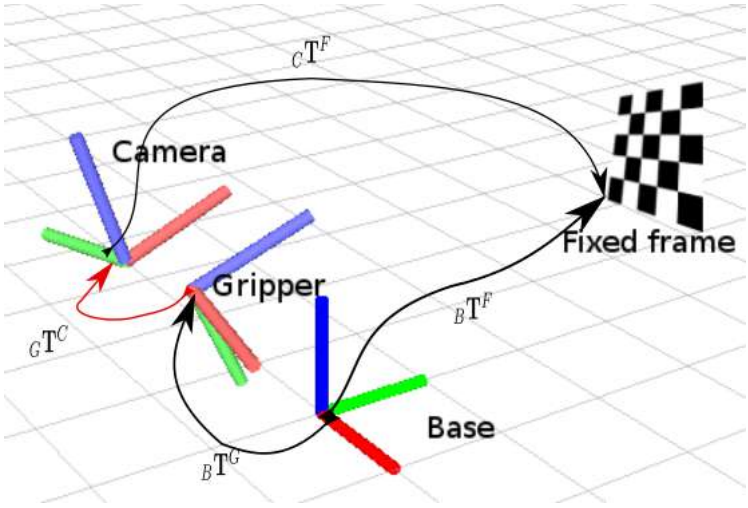


Figure 5.11: The 7DoF robot arm model.

Table 5.2: Detection rate for the scene objects with a single and multiple cylindrical and spherical object in the scene

	Cyl.1	Cyl.2	Sph.	Cyl.1 all	Sph. all
Plane ext.	5/5	4/5	5/5	4/5	5/5
Reg. grow	3/5	2/5	5/5	4/5	5/5

dinate frames the camera can be integrated in the planning process easily.

5.2.8.2 Scene object retrieval

Further on we present the results of our investigations related to the object retrieval from the working space using the 3D perception pipeline described earlier.

In the Tables 5.2 we summarized the results of our test cases for the detection rate of the suggested approaches from five different viewpoints about the same scene. The first columns contain the results from the scene with only one object of interest, while the last two columns show the results for the retrieval focusing on a specific type of object in a multi-object scenario.

Table 5.3: Standard deviation of the object centre used during the recognition from several viewpoints

	Cyl.1	Cyl.2	Sph.	Cyl.1 all	Sph. all
Plane ext.	0.03	0.43	0.01	0.48	0.55
Reg. grow	0.42	0.03	0.49	0.02	0.41

Finally, the Table 5.3 summarizes the results from the standard deviation for the detected object's geometric center using different viewpoints about the same scene by moving the arm around the objects.

As one can observe the region growing method tends to be more robust, although the runtime of this is slightly larger than the plane extraction variant for the same scenes.

5.2.8.3 Planning benchmark

In order to test the performance of the planning algorithms on the arm, ten points from the configuration space have been chosen. The test have been carried out for the same points in two scenarios one unconstrained which contained no obstacles and one constrained in which an obstacle has been placed.

In the Table 5.4 we summarized the runtime performance analysis for our experimental setup using a standard laptop. As one could expect, the planning time with a constrained working space is greater than the one without collision objects in the scene for the sampling based algorithms. The most affected variants are in this case the RRT and its derivatives. The planning times for each algorithm are less then 5 seconds, which in this case is the minimum for the robot arm displacement in general.

Table 5.4: Planning algorithms average runtimes in seconds for the 7 DoF arm

	RRT	RRT–Connect	KPIECE	PRM	EST
Unconst.	2.18	4.13	2.48	2.24	2.13
Constr.	3.15	4.96	2.77	2.45	2.567

5.3 Dual-Arm Cobot Path Planning

5.3.1 Introduction

Recent advances in the industrial domain (for example Industry 4.0) have opened up a multitude of opportunities for robots, as they are expected to perform the tasks either on their own or along other humans, as coworkers Militaru *et al.* (2016a, 2017); Páll *et al.* (2016); Tamas & Baboly (2017). In such cases, most of the time it is expected from robots to adapt to the environment and to take decisions on their own. In practice however, sensor and motor inaccuracies, noise–affected data, algorithm misbehaviours, improper classifications or object occlusions are just a few of the challenges that the robot has to deal with in the decision making process.

A very useful framework to handle sensing uncertainty in particular is active perception, which closes the loop between the sensing and control modules of the robot by taking control actions with the explicit purpose of obtaining more informative data from the sensors.

In this paper the following active perception problem is considered. A robot is working in a factory and has the task of sorting objects of different shapes that travel on a conveyor belt. The classification of the objects is done using a sensor the robot is equipped with, and due to sensor inaccuracies, multiple scans are required for a proper classification. The robot has a set of poses (viewpoints) from which it scans the conveyor belt. It starts from an initial viewpoint and can move between viewpoints in order to gain more information about the objects.

Our main contribution is a solution of the sorting task in the framework of partially observable Markov decision processes (POMDPs). The step of modeling the

problem as a POMDP is essential, and we describe it in detail. Object classes are uncertain so a probability distribution (belief state) over them is maintained, which is updated using the observations. At each step, the robot has the option of either changing the viewpoint to make a new observation, or to decide on the class of the object at the end of the belt, sort it, and advance the belt. A sequence of these actions is planned over a long horizon so as to maximize a cumulative sum of rewards assessing the quality of sorting decisions, using a state-of-the-art planner called DESPOT Somani *et al.* (2013).

A key feature of the method is that it allows the robot to observe multiple positions from the conveyor belt at the same time, from each scan. This allows us to accumulate and propagate information between positions when the conveyor belt advances, so that the robot already starts with a good estimate of the new class at the end of the belt. This reduces the total number of steps needed to sort the objects, while still maintaining a high classification accuracy.

In addition to simulation results, we present also real experiments with a Baxter robot equipped with an ASUS Xtion 3D sensor mounted on one of its wrists. The objects being sorted are light bulbs of different shapes, and their classification is carried out the data coming from the sensor in the form of point clouds. For both simulation and on the physical robot, we report two sets of experimental results. In the first case the robot observes only a single object from the conveyor belt without propagating the belief states, while in the second case the observations are extended to two positions and the propagation strategy is adopted.

Related work: References Aloimonos *et al.* (1988); Bajcsy (1988b) provide early approaches toward active perception. Since then, the field has grown tremendously, as the availability and range of applications of robotic platforms and sensors has increased.

In active perception, there are two ways to perform detection: passive and active. In the former, the observation viewpoint is changed, leaving the state of the objects unaltered, while in the latter, the objects are actively manipulated in order to improve classification. References Atanasov *et al.* (2015); Patten *et al.* (2016) are

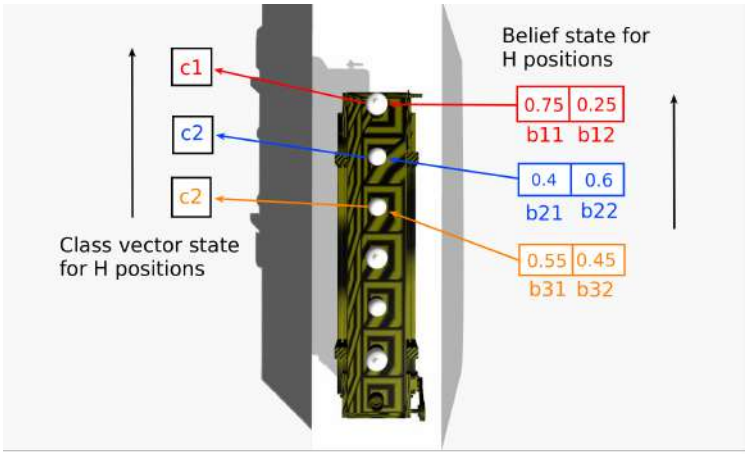


Figure 5.12: Belief state and class vector state before propagation

examples of the former, while Aleotti *et al.* (2014) provides a good example for the latter case. The method proposed in this paper uses the passive approach.

The meaning of this second function is the following. As mentioned, the robot is concerned only with a limited number, H, of positions from the conveyor belt. A motion action leaves the classes unaltered (first branch). However, when a decision action is executed the first object is removed from the conveyor belt, its class is eliminated from the vector of classes and each of remaining values is shifted with one position (second branch), except the last one which is sampled from a uniform distribution (third branch); see also Figures 5.12 and 5.13. Note that in reality, the class will be given by the true, subsequent incoming object, but since the POMDP transition function is time-invariant, this cannot be encoded and we use a uniform distribution over the classes instead.

5.3.2 Active perception pipeline

In this section, the components of the active perception pipeline are presented. The active perception pipeline has two main modules: a detection module, having multiple submodules, and the planning module. The detection submodule is composed of

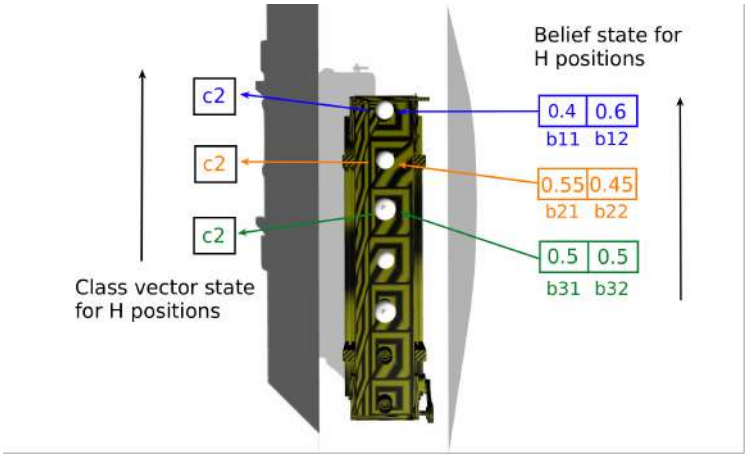


Figure 5.13: Belief state and class vector state after propagation

the acquisition, preprocessing and classification submodules. Several components of the pipeline are similar to the ones presented in Mezei & Tamas (2016), what is different is the classification algorithm and module used, and most importantly, the presence of a planning module in the pipeline.

The robot performs 3D scans of the conveyor belt in order to detect the objects that are being transported, which for our particular application are light bulbs of different shapes. The acquired data is in form of point clouds Rusu & Cousins (2011d) and tasks such as saving, loading, are handled by the acqusition submodule.

The classification module receives as input a prepared point cloud, corresponding to a candidate light bulb and has the role of classifying it. A prior training step is necessary, in which the uniformly sampled point clouds containing the light bulb classes of interest for the application are used. The Viewpoint Feature Histogram (VFH) Rusu *et al.* (2010c) provides descriptors containing information about the shape of the cloud and also viewpoint information. During training, a kd-tree that is built from the clouds taken from each observation point correspond to each class of light bulb. The classification becomes a nearestneighbour search problem, in which for a candidate cloud a list of the trained clouds is returned sorted by the shortest distance to the candidate cloud.

The crucial component of the pipeline is the planning module, which has the role of finding a good sequence of observations, which improves the likelihood of a proper sorting for a candidate light bulb. The planning module solves the POMDP problem using the DESPOT Somani *et al.* (2013) algorithm, using an online approach that interleaves the planning and plan execution stages. The belief state is updated with the results coming from the detection module. The planning module returns an action, either of the motion or of the decision type, which is processed and further transmitted to the motion planning and execution modules specific to the robotic platform that is used.

5.3.3 Results

The Baxter research robot developed by Rethink Robotics has been used for both simulated and real experiments. An ASUS Xtion 3D sensor was mounted on one of the robot's wrist, while a conveyor belt, transporting different models of light bulbs, was placed in front of the robot. The motion planning tasks for the arms were carried out by solvers specific to the robot platform. The functionalities of PCL (Point Cloud Library) were used to construct the modules that handle all aspects regarding the point clouds acquired by the sensor. The active perception pipeline was developed in C++ and Python and was made to be compatible with ROS (Robot Operating System), while Gazebo was used for simulation purposes.

In each type of experiment an observation probability distribution was computed experimentally beforehand for every vertex of the graph. From each vertex, several scans were performed for each true class of light bulb. After preprocessing each scan, the segmented bulbs were classified using the presented method with the results being recorded in tables, and the observation probability distribution was computed as the fraction of the experiments in which each class (correct or incorrect) was observed. Table 5.5 exemplifies the observation probability distribution for the elongated object (true class), when observing the end of the belt, for a few representative viewpoints. It shows how likely is to observe the respective class

when looking at it and how likely is to misclassify it as one of the other classes, from different viewpoints. In the extended case, Table 5.6 shows how likely is to observe the different combinations of bulbs on the last two positions of the belt, from the same viewpoints as in Table 5.5.

Pr(o) / Point	elongated	livarno	mushroom	standard
64	0.6	0.2	0	0.2
43	0.5	0.3	0.1	0.1
87	0.8	0.1	0.1	0

Table 5.5: Observation probability distribution for a single bulb when the underlying object is of the elongated class

Pr(o) / Pt	el p1	liv p1	mshr p1	strd p1	el p2	liv p2	mshr p2	strd p2
64	0.3	0.6	0.1	0	0.4	0.1	0.4	0.1
43	0.1	0.8	0	0.1	0.5	0.2	0.2	0.1
87	0.2	0.7	0	0.1	0.2	0.2	0.6	0

Table 5.6: Probability distribution when observing two positions, when a livarno bulb is on the first position and an elongated bulb is on the second one

5.3.4 Results on the real robot

An illustration of the real setup and experiment is given in Figure 5.14 are shown respectively the quality of the classification and the number of required steps. A video of the robot can be found at `http://community.clujit.ro/display/TEAM/Active+preception`

Figure 5.14: A real robot performing the sorting of light bulbs

5.4 Augemented Reality Based Monitoring

5.4.1 Introduction

The main challenges of the current manufacturing era are related to the high industrial performance requirements with relatively fast production life cycles and severe environmental constraints. The source of these problems relates to the loose vertical integration of digital trends within a manufacturing system with the natural demand for flexibility. One of the key components of the manufacturing execution systems relates to the visualization problem, incorporating the use of augmented/virtual reality devices for industrial workers Perzylo *et al.* (2016).

The augmented reality term, according to the definition in Milgram *et al.* (1994), refers to the addition of virtual objects to real world scenes in order to extend the capabilities of scene visualisation. In order to achieve this, one of the main require-

ments of an AR system is to align (spatially register) the objects of the real scene with the virtual ones. Besides the entertainment market, AR in tourism, real estate, marketing and remote maintenance is also playing an important role Palmarini *et al.* (2018). Even though in the last 50 years the AR technologies emerged continuously, they are still in development for the industrial application field. This trend is changing fast in the Industry 4.0 context Weyer *et al.* (2015), mainly due to the remote assistance and maintenance applications. The digital twin concept integrates also well from the new industrial trends into the AR framework: this can be directly imported as visualisation modules for the applications. Thus, they are getting an important role in the chain of manufacturing execution systems (MES).

Several advantages of the AR systems were proved recently to be of major impact in the MES systems, including safety, precision, and learning curve of the operators. From the safety side, according to the International Labor Organization, every minute a work accident is happening, thus every step in the direction of reducing this trend is of major impact on the human workforce. As the AR has a positive impact on the safety in MES, this can reduce the rate of accidents Tatic & Tesic (2017).

In terms of Key Performance Indicators (KPIs), the increased precision and reduced assembly time with AR for safety critical aerospace industry was already proved in the 90's Mizell (2001). Thus, AR applications are favourable also for optimising the production indicators on a large scale MES Wang *et al.* (2018).

On the training side of the operators, a recent study Radkowski *et al.* (2015) highlights the fact that, with the introduction of the AR systems, the learning curve got enhanced for the remote operators, and first-time fix rates increased by 90%. For the assembly or remote assisted training applications, the impact of introducing AR solutions into MES has one major benefit: suggestive 3D visualisation is appropriate for the human perception. An example of such a visualisation is visible in Figure 5.15, containing a partially assembled light bulb in the workspace of a collaborative robot, augmented with the final view of the same object.

The widespread of the AR solutions in the MES systems is expected to have an

overall positive impact. In this paper, the authors highlight some recent AR trends within the MES, as well as describing use cases with different AR devices/technologies. These were successfully integrated into customer specific applications.

5.4.2 Application overview

Our aim was improving day-to-day manufacturing operations by developing human-robot collaboration systems with augmented reality. One of the main goals is the 3D vision enhancement of industrial operators with the help of head-mounted display (HMD) equipment, due to being an eye level device type which eases instantaneous perception of the scenario, mixed with AR.

Having used augmented reality, virtual elements for the assembly sequence of the final product can be visualised before beginning the actual production, as it can be seen in Figure 5.15. For the design, the assembly model of the product may be decomposed in a characteristic hierarchy with definite assembly sequences. For the production, the industrial worker can obtain relevant visual information with the use of these sequences. Those may be adapted according to the current stage of the assembly. The overall development of the product can considerably prosper using this system, resulting to a significant decrease in lead time (initiation - execution latency), cost reduction and quality improvement Caudell & Mizell (1992).

Figure 5.15: Preview of final product with augmented visualisation Blaga & Tamas (2018)

When talking about visualising a 3D assembly scenario, the first phase is synchronising in real-time the real with the virtual in the shortest time possible. We need to take into account the constant need for space calibration so that the real and virtual scene are kept aligned in space for the final stage of one, stable system Martín-Gutiérrez & Ginters (2013). For this reason, one of the methods that can be used to identify essential elements and their position is the marker method. With this, HMD calibration can be done as well. Furthermore, the human-robot collaboration setting needs to be tuned by calculating the structural connection between the robotic space and device to be operated on. Thus, accurate links between the position and orientation of involved equipments in the scenario are obtained.

To develop an AR application having robot collaboration, an important step is accounting for the disparity between real points of reference for the involved technologies that determine their respective locations. In these circumstances, the robot may get imprecise data regarding the operator's place. In turn, the operator can receive misleading information related to the robotic workspace. To straighten this out, the correct turning transformations need to be computed, along with calculating the position of virtual items in the workspace of the robot. It is worth taking into account human-robot collaboration safety matters, where attention and alertness of humans is unsatisfactory.

Implementing AR applications require dedicated software and hardware. Specialised equipment like HMDs and proper trackers must be attained Nee *et al.* (2012). When considering hardware, an appropriate wearable device is required so that is favourable for the entire system. Characteristics such as size, weight and portability are considered when talking about manufacturing tasks, which needs to assist the industrial worker all day. In order to avoid severely reducing data that can be visualised at once, a convenient field of view may be also chosen Syberfeldt *et al.* (2017).

There are many potential use cases in manufacturing scenarios: production, training, maintenance. In this paper, the focus is on tasks involving step by step operations, risk and safety management, as well as refining design and visualisa-

tion. Completing tasks in time and error reducement are desirable outcomes when using AR.

The main problem to be addressed is a stable connection and transmission of information between AR, with HoloLens, and the robotic workspace, using Robot Operating System (ROS). The current pose of the device and the pose of the marker identified using marker detection is sent to ROS. This data is transformed in ROS messages and then used as frames for the transform (tf) tree. ROS also transmits back the pose of the right end-effector, used for the assembly scenario. This information is needed in order to accurately augment the next element to be assembled for the finalising product, as it may be seen in Figure 5.15.

First of all, the camera of the HoloLens needs to be calibrated so that the finest tracking accuracy is obtained, specifically when glancing straight into a flat marker. For this type of device, lens distortion in the video image that is displayed may be removed as well. In order to accomplish this, the tool ARToolKit ART (2018) was combined with camera calibration from OpenCV ope (2018). Parameters of the camera may be found using explicit calibration algorithms Blaga & Tamas (2018).

With ARToolKit and a see-through device, in this case HoloLens, virtual items can be superimposed in the real setting. Computing positions of virtual elements in the 3D space is done by identifying and tracking square markers in the real space. Camera view video is captured by the camera and sent to the software which searches for fiducial markers in each frame of the video. If any with the pattern is identified, the software then calculates the square position, as well as the orientation of the pattern, considering the camera. After this data is known, a virtual model with relative calibration is drawn, as it can be seen in in Figure 5.16.

For the ARToolKit with HoloLens integration, a Universal Windows Platform (UWP) wrapper was used. It also contained the marker recognition and tracking algorithms. Knowing the camera parameters and the pattern of the marker, the accuracy of the parameters can be verified. A custom virtual item can be visualised. The marker recognition and tracking procedures found in the kit may determine the position of the object.

The HoloLens application is communicating with the robotic workspace through a common protocol with the Rosbridge package. This yields an interface between ROS and JSON, in particular for non-ROS applications, as one can visualise in Figure 5.17. In order to send data from HoloLens to ROS, custom C# classes were created so that they can simulate the message type that needs to be received by the ROS service. For this, Hololens' current position and orientation is send, as well as the transform of the initial marker for the pattern that is tracked. To receive this, the service callback gets the data and, with the help of a ROS publisher, it publishes the information on the /tf topic. The transform between the robot's right end-effector and the HoloLens' initial position is computed, resulting in the element of the robot transform with respect to the device's coordinate system. This transform is the response for the HoloLens application. The transforms can be seen in Figure 5.18.

HoloLens development system using Unity3D has a left-handed coordinate system. ROS works with a right-handed coordinate system. In this situation, the conversions between left-handed to right-handed and back need to be computed.

In order to finalise the conversions, further calculations needs to be done so that

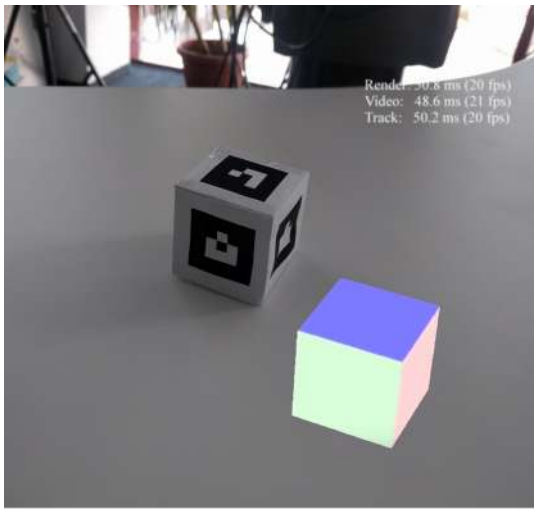


Figure 5.16: Visualisation of the real object and the AR marker during calibration Blaga & Tamas (2018)

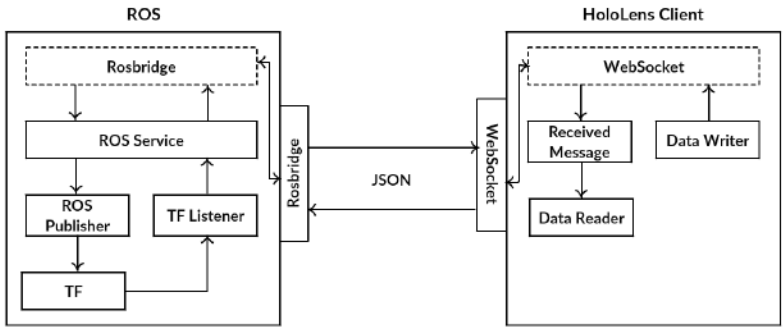


Figure 5.17: Rosbridge communication between cobot and AR device Blaga & Tamas (2018)

the HoloLens' system is represented in the robotic coordinate system. This can be written as:

$$T_H^G = T_H^X \cdot T_X^G$$

To convert from ROS to HoloLens, the inverse computations need to be addressed. This is defined by transform multiplication:

$$T_G^H = T_Z^G \cdot T_Z^H$$

The extra step needs to be done here as well, where the right-handed HoloLens is computed to the left-handed classic one, the same inversion type as mentioned above.

The pose estimation of the Hololens HMD in the coordinate system of the robot is done by estimating the spatial relation among them. This was achieved by atteching the camera to the robot arm, and recording several arm-camera data pairs and from this estimating the cross calibration between them kth (2018). For this to be feasible, the Hololens' position considering its start position is already known (device having motion tracking), the start position does not change in relation to the base of the robot. The end-effector pose is also prior computed, with respect to the base. In order to find the unknowns, multiple samples are used. These are gathered

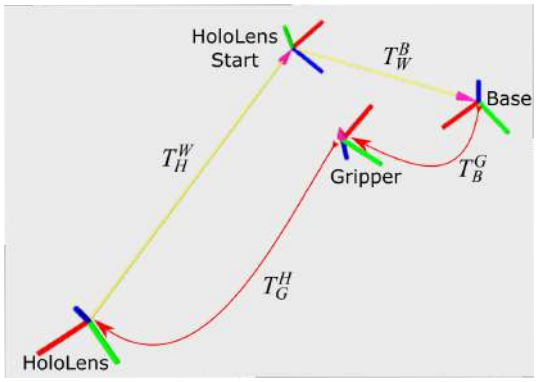


Figure 5.18: Frames used in the application Blaga & Tamas (2018)

by having the end-effector move in a variety of positions and angles, managing to cover a large range.

Having a multitude of samples that were measured, relative calibration can be greatly improved, in such a way that the virtual element may be precisely spatially aligned with the real items.

5.4.3 AR and robot external calibration

For the path visualisation application (section 5.4.4.1), a different calibration algorithm was used, as this is particular to each robot used in the experiments.

In case of the Google Tango AR device, the self-localisation was not performed accurately enough, so to re-calibration was needed before each experiment. In order to speed-up the external calibration procedure, a straightforward method was chosen, described below.

Both devices (AR and mobile robot) run motion tracking algorithms, each one having its own coordinate system. To create a cross link between the two coordinate systems, the AR device is placed on top of the robot, in a predefined position. Then, the computation of a rigid homogeneous calibration transform is done, as follows:

$$T_{ARW}^{RW} = T_{ARW}^{AR} \cdot T_{AR}^{R} \cdot T_{R}^{RW}$$

where *RW* stands for Robot World coordinate frame, *R* is the Robot, *AR* is the AR Device and *ARW* is the AR World (see Figure 5.19c). T_{ARW}^{AR} is the pose of AR device, obtained from motion tracking. T_{AR}^{R} is constant, measured manually. T_{R}^{RW} is inverse of robot pose, from motion tracking.

The calibration procedure is the following:

1. User places AR device on top of the mobile robot, in the predefined position. Then, send a signal to the *Calibration* module in order to compute the calibration transform.

2. User takes AR device and points the camera towards robot.

3. When the mobile robot receives a goal, it plans a path from its current position to the desired one. Each waypoint in the path is converted into AR World coordinate frame, then sent to the AR device.

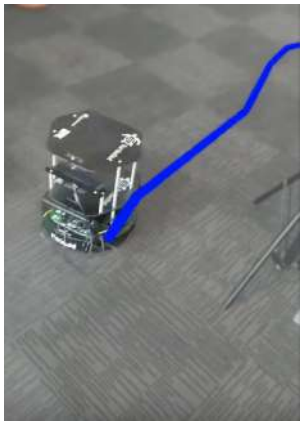
Using this method, a transform from *Robot World* to *AR World* is obtained, with absolute precision up to 10cm in the robot coordinate frame system, the radius of the robot being a half meter.

5.4.4 Experimental results

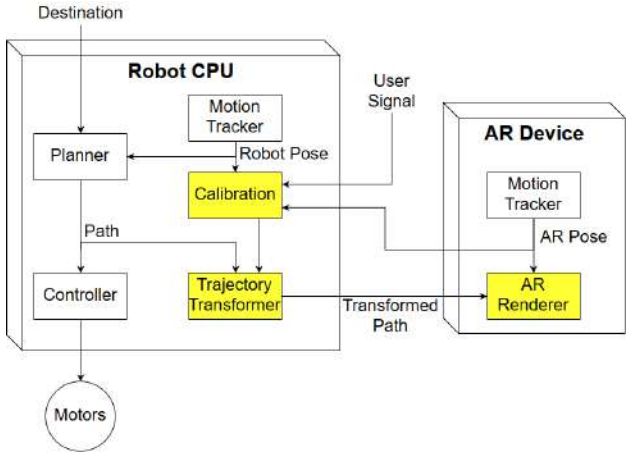
5.4.4.1 Path visualisation application

The main goal is to improve the safety of workers and automated guided vehicle (AGV). In this section, an application for visually inspecting robot's path is described. As seen in Figure 5.19a, the application draws AGV's computed path over live images of the environment. The user checks if the drawn path intersects any obstacles and, in that case, stops the AGV.

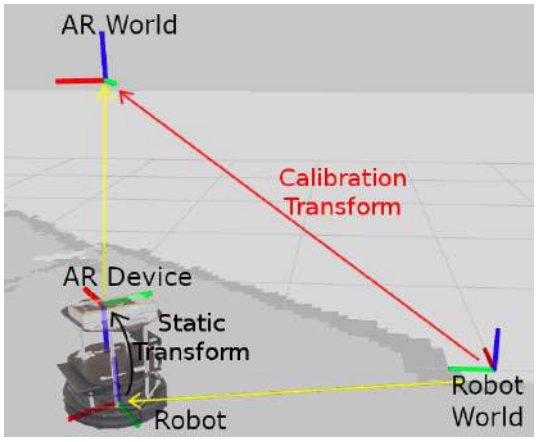
In this implementation, existing technologies are reused. We use TurtleBot as AGV, which has a navigation stack available in ROS Foote (2018). This includes motion tracking, planning and executing paths. Regarding the AR device, Google



(a) AR device image



(b) Architecture



(c) Calibration frames

Figure 5.19: Path visualisation

Tango Marder-Eppstein (2016) technology powered device is chosen, having motion tracking and augmented reality technologies out-of-the-box. The two devices communicate with wireless network using ROS framework, with additional Android support for the AR device.

Figure 5.19b shows the main components of the application, highlighting our

contribution. That is an external calibration algorithm, described in Section 5.4.3, as well as a rendering procedure. Experiments were performed in order to determine the quality of AR visualisation.

The application has multiple steps:

1. Calibrate device (see Section 5.4.3)
2. AR device overlays transformed path on live images. More specifically, each pair of consecutive way-points is converted into a cylinder. The pose of a cylinder C having its endpoints P_1 and P_2 can be computed as follows (see Figure 5.20):
   - Position: $C = \frac{P_1+P_2}{2}$
   - Orientation: the half-way quaternion between $\overline{OZ}$ (default cylinder orientation) and $\overline{P_1P_2}$
   - Compute $\overline{rot_{xyz}} = \overline{OZ} \times \overline{dir}$
   - Compute $rot_w = 1 + \overline{OZ} \bullet \overline{dir}$
   - Normalize rot
3. In case of danger, the user may stop the robot. While the AGV is moving, it repeatedly recomputes the trajectory. Therefore, the displayed path is updated on the AR device screen, providing real-time visualisation as this can be seen in in Figure 5.19a.

In this case, the planner produces a large number of waypoints, with small distance between them. Therefore, the AR device has a low frame rate. To overcome the issue, the Douglas-Peucker algorithm is employed Douglas & Peucker (1973), which reduces the number of points on a path without modifying its shape.

In the experiments, the distance between AGV real position and the one shown on AR device is manually measured. The initial error (measured right after calibration) is, on average, 0.1m. After the robot travels 10m, the accumulated mean

(a) Translation (b) Rotation

Figure 5.20: Rendering cylinder between two points

error is 0.25m (smaller than robot diameter). Hence the AGV can be used in a safe manner for path planning and visualisation applications.

5.4.4.2 Robot state visualisation

At certain moments it is of great importance to know what the robot coworker is doing. Thus, by visualising certain variables, parameters that make up its state, a deeper understanding of its actions can be gained. The human coworker is offered a set of tools, capable of performing diagnostic, safety stop and restart operations, with the main goal of enhancing safety level and maintenance parameters of the manufacturing process.

The developed application is to be used on a Microsoft HoloLens device and exposes data from the ROS environment existent on the robot. The communication of the two entities, robot and HoloLens, is facilitated by the rosbridge package, the entities exchanging messages in JSON format.

The application is composed of three panels: connectivity, advanced and extended options panel. At first, only the connectivity panel is shown, as its name suggests the facilitation of the connection procedures with the ROS environment present on the Baxter robot. If the connection is successful the advanced panel will pop up. The advanced panel contains functionality for sending/receiving information from the ROS environment. It also contains indicators relevant to the activities performed by the robot. The human user can query information about the running nodes and published topics, as well as mandatory nodes. The global state

of the robot can be inferred from the information contained in this panel. There are three indicators that will change their colour in red or green, depending on the sub-modules that reflect the robot's activity state.

In case of malfunction, the advanced options panel will allow the user to perform safety stop, restart actions, etc. More precisely, the human can check the running state of a certain node, it can start a chosen node or it may stop different nodes that are not running properly.

Figure 5.21: Performing a status check using the monitoring tool

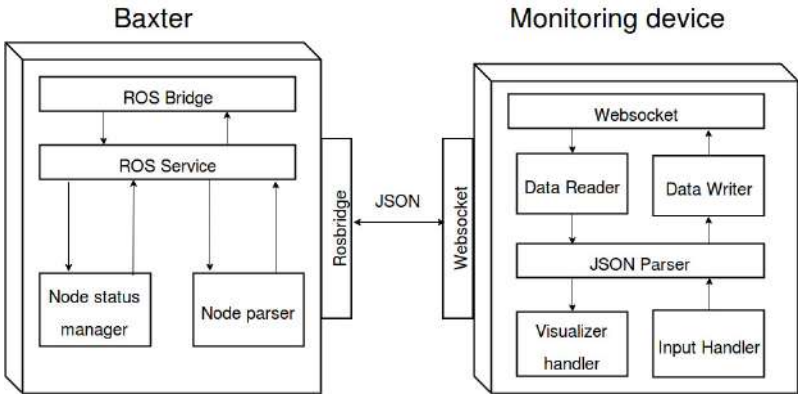


Figure 5.22: The components of the monitoring tool

More details regarding this application can be seen in the video [1].

[1] https://youtu.be/3AR9hE6JKh0

For this application we relied on the default external calibration of the Hololens device, which was sufficiently precise for visualisation purposes. The only custom setup is related to the predefined position of the smart glass in the startup phase: this is placed in a predefined position in the workspace of the robot, thus having an instant cross calibration with the frame of the robot.

5.4.4.3 Digital twin visualisation

The use of digital twin got widespread with the appearance of the low cost 3D printing solutions. The real product that needs to be assembled is virtually cloned using CAD modelling. This procedure can precisely simulate the physical traits, the virtual model being considered a digital twin for the real one. This model was divided into several components so that a definite ranking may be determined so that a convenient sequence is organised, as it can be seen in Figure 5.23.

Figure 5.23: CAD model with augmented part (blue)

In order to execute the extrinsic calibration of the HoloLens camera, different algorithms were used so that calibration error is minimal. Using the provided camera-robot calibration package, when starting the application, the user had to change the pose of the HoloLens during the detection of the marker positioned on the right end-effector of the robot, as well as moving the arm into a range of poses. This was determined to be inefficient due to being a burden on the user and not

task optimal for a single worker. Additionally, if marker detection fails in just one iteration, the end result would be poor. By attaching the device to the end-effector, the first task of the day is considerably improved, so that the calibration is better approachable, as seen in Figure 5.24.

Figure 5.24: Calibration procedure with Baxter and HoloLens

The final application contains a light bulb's CAD model. The robot's right end-effector holds the socket of the light bulb, whilst matching marker is fixed on the arm. After the HoloLens application is started, the extrinsic calibration needs to be executed. Proper calibration gives precise synchronisation between the real and virtual elements, resulting in fitting spatial alignment. More than one iteration may be needed for this. After being finished, the industrial user may visualise the virtual components of the scene in their correct place, with respect to the coordinate system of the HoloLens.

Some experiments were done to validate the calibration between HoloLens and Baxter robot. In order to be able to properly repeat the tests, they were done in three phases. In the first place, the right end-effector holds the device fixed, while the normal procedure of the calibration is executed and its position taking into account the gripper was documented. After, the HoloLens was fixed to the left end-effector, recording its' position again. After, the virtual component's transformation of the end-effector was compared with the real one, computing the degree in which the virtual element is overlapping its physical correspondence.

For the ARCore system, experiments were done as well, but in a fairly different fashion for some points. At the start of the application, after detecting and getting

into the tracking regime, the virtual element is superimposed onto the real scenario. Thus, the degree of overlapping for the virtual unit over the real one can be computed, similar as above.

After repeating the individual procedures several times, the data that was obtained was processed using two ways: the mean and the standard deviation. Firstly, each of the samples and the mean for each component related to the device's position with respect to the end-effectors was used in order to compute their corresponding Eucledian distances. The resulting errors for each end-effector were used to determine the data's mean and standard deviation. As it may be visualised in Table 5.7, for the right end-effector the measurements are considerably lower than the left one due to the calibration being done with the first. Next, the virtual element superposition on the real one was calculated taking into account the pixel number of the virtual which overlap the real component, converted into percentages. The final data may be seen in Table 5.8. For the HoloLens system, a $\approx 58.83\%$ mean is determined, corresponding with the overlapping percentage, with $\approx 11.84\%$ standard deviation. For the ARCore Application, the determined mean is $\approx 78.32\%$, with $\approx 9.87\%$ standard deviation.

Table 5.7: The output from the calibration process Blaga & Tamas (2018)

End-Effector	**Error Mean (cm)**	**Error Standard Deviation (cm)**
Right	0.47	0.19
Left	0.95	0.45

Table 5.8: The augmented and physical object overlapping

	Mean (px)	**Standard Deviation (px)**
Blaga & Tamas (2018) Overlap - HoloLens [%]	58.83	11.84
Overlap - ARCore [%]	78.32	9.87

After, the real assembly objects can be used simultaneously with the running ap-

plication. At a particular moment, the industrial operator may visualise the matching element that needs to be assembled and receive data about assembly information, as well as its proper position and orientation. For this, the robot's right end-effector holds a part of the assembly, whilst the succeeding part may be displayed using HoloLens. By having the response data given by ROS, being the pose of the right end-effector with respect to the coordinate system of the device, the virtual object's pose is computed, taking into consideration the current pose of the physical element as well. The overall application algorithm may be seen in Figure 5.25.

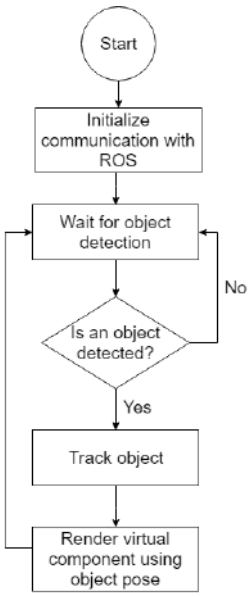


Figure 5.25: ARCore application algorithm

A modest alteration to the above mentioned scenario is that the robot may not hold any part of the assembly, the real objects being positioned in only near the robotic workspace, so that the end-effectors of the robot can reach the product and still be able to aid the industrial operator in the assembly task. This application may be visualised in Figure 5.27b.

For this particular scenario, an AR-capable mobile device was used, having integrated the ARCore SDK in the application.

When the application launches, the transformation between the right end-effector of the robot and the initial pose of the ARCore application is searched in order for the robot component transform to be in the right coordinate system. Then, the data is exchanged with ARCore with the help of this service. The overall architecture may be visualised in Figure 5.26.

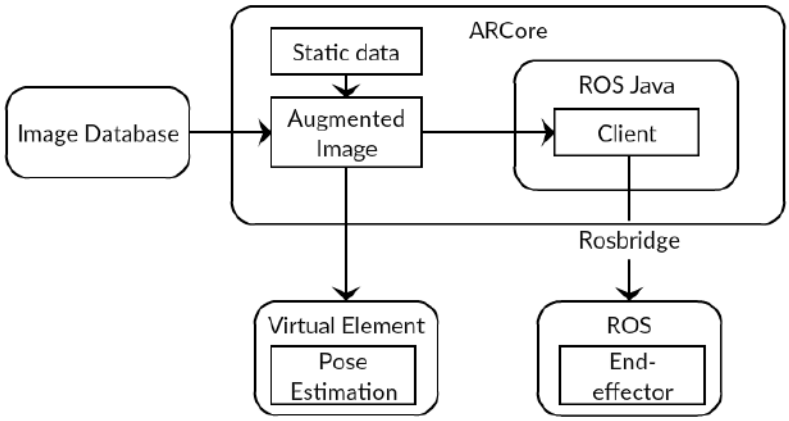
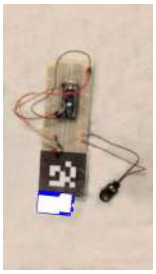


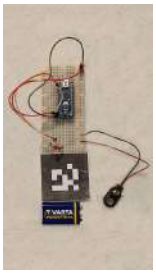
Figure 5.26: ARCore application architecture

To display the data regarding the currently tracked item, the industrial worker only needs to tap on the device's screen. This tap is converted from 2D to 3D in order to compare its position to the 3D pose of the object. Then the requested information may be visualised in a dialog. Several application snapshots may be seen in Figure 5.27.

A comparison between the AR technologies used, HoloLens, ARCore and Google Tango, was constructed in Table 5.9.



(a) Task augmentation snapshot



(b) Application outcome

Figure 5.27: ARCore application

	HoloLens	ARCore	Google Tango	Moverio BT300
Release Date	2016, March	2018, March	2014, June	2017, January
Development Status	Stable	Stable	Discontinued	Stable
Device Type	Head-mounted	Hand-held	Hand-held	Head-mounted
Cost	$3,000	$280+	$500	$700
Documentation	Vast	Limited	Withdrawn	Updated
Depth Sensor Type	✓	×	✓	×
Localisation mode	Motion tracking	IMU + vodo	IMU + vodo	Vodo
ROS Integration	Rosbridge	Rosjava	Rosjava	Rosjava

Table 5.9: AR technologies comparison

5.5 Summary

In this chapter we showed an approach for the autonomous flight control of a quadcopter in hallway-like environments. We used low-level image processing enhanced with probabilistic filters which supports data fusion, while using a low-cost quadcopter.

Next we presented a custom solution for the object manipulation in an indoor environment using a 7 DoF commercial robot with a 3D depth camera mounted close to its gripper. We presented our solution to the scene parsing as well as planning related problems for the object manipulation using different approaches.

Further on the task of sorting objects transported using a conveyor belt was handled using an active perception approach. The proposed pipeline uses 3D data to

classify the objects and computes the actions using a planner, after describing the problem as a POMDP. The functionality of the pipeline was tested both in simulation and on a real Baxter robot equipped with an Asus 3D sensor. A key point of the approach is that it propagates information across multiple positions of interest from the conveyor belt, thus reducing the total number of steps needed for sorting.

Beside this practical examples were presented for different experimental setups in the AR domain relying on pose estimation. The common parts in these setups were related to the external calibration of these devices with respect to other cameras, e.g. mounted on different types of robots. The experiments are easy to reproduce following the detailed description of these setups as well using the code shared by the authors.

FUTURE PLANS

Advising and management activities

Past

During my research I had to chance to manage a great number of stundents both from Romania and abroad. I have been involved as PhD coadvisor both at UT-Cluj and at Szeged University, Hungary. The thesis defense of Robert Frohlich from Szeged University took place in 2019 and the the on-going coadvised thesis of Mezey-Ady Danie at the UTCluj is expected to be finished in 2021. Robert's work is related more to 5 while with Daniel we cooperate more on the topics presented in 5. Beside this, I had the occasion to supervise BSc and MSc studnets at Bern University during my research stay in 2014 who were involved in projects related to 5.

I have successfully obtained funding for my research from a number of national and international founds as PI:

- Two UEFISCDI national projects: Bridge grant type project BG39-2016 and Innovation Cheques CI125-2018.
- One H2020 consortium workpackage project within the ROSIN call in 2019.
- One industrial project with the Vitirover company from France in 2017.

Beside this I have been involved in a number of national and international projects as participant:

- Sciex type postdoc project in Switzerland (acceptance rate less than 5%).
- 4D postdoc project in the period of 2010-2013.
- COST EU project COSCH in the period 2014-2017.
- OTKA founded project in the period 2018-2021.
- UEFISCDI-TE type projects from 2012 and 2016.
- H2020 RIA type project launched in 2020.

Both as PI and project member these projects contribute to the development of the management skills required in a national and multi-national project setup.

As scientific&project meeting expertise, I must highlight COST local meeting organized in 2015, the special session of AQTR organized in 2016 as well as the role of main KEPAF conference organizer in 2017.

Future

In the future I will reuse the managing and advising capabilities earned during the last years of activities especially in the projects where I was acting as PI. I expect a clear contribution in the bridging activities between the academic–industrial sector, as this requires a specific way of approaching the problems. I have experienced in a number of industrial projects, that the results of my research in the robotics and control engineering can be validated in a number of innovative projects.

More specifically I expect to have clear contribution on the transfer learning for the end-to-end 3D perception systems in robotics applications from the leading innovative industries. Also the MES related coworkers are in the focus of the future research activities with the industry. Beside this we are expecting an important demand from the agricultural robotics sector as well in terms of applied research inquires towards our team.

For the advising activities I expect that I will cooperate with a number of BSc, MSc and PhD students. I expect to have coadvised topics from both from the academic and industrial field. Currently we expect to have at least 2-3 positions in the field of robotics engineering with applied research orientation.

In order to maintain the balance between the fundamental and applied research I expect to take part of both types of projects. Thus I intend to enlarge my networking both in the academic and industrial fields.

Research activities

In this part I will give a short overview of the past–current research activities (post doctroal period) and I will highlight the near future and long term plans.

Past and current focus

In the post-doctoral period I focused more on the robotics domain and within this domain the 3D perception related challenges. The state of the art for the depth perception grow exponentially after the appearance of the low-cost depth cameras such as Kinect for the consumer market.

Before having access to such a camera we constructed our own 3D sensor and we followed the early stage development of the 3D data processing and robotics tools such as PCL or ROS. This had an impact on our way of doing research and we are still using and reusing the tools withing these frameworks developed in the last 10 years.

Beside the classical depth perception, we made experiments with heterogeneous data, i.e. stereo depth, fused 2D-3D data which has a major interest even today in the robotics community. These are fundamental building blocks for the object recognition and pose estimation algorithms used heavily in this domain.

Recently, we managed to close the loop between the perception and discrete action space for robots using probabilistic approaches. This is still an ongoing work which has to be extended in order to make it feasible in real life applications as well.

Near future goals

For the near future goals I will focus on the transfer learning techniques for the robotics environment perception and mapping applications. This will hopefully enable us to have end-to-end pipelines in the robot perception-action space.

As a first step we are focusing on the following topics

3D object localization The focus is on an end-to-end learning framework delivering a flexible, pointcloud sparseness independent way of localizing objects in the surrounding of a robot. The proposed framework makes an explicit segmentation and object recognition in the same time with a sufficiently high rate that even for dynamics scenes (e.g. with people) can be adopted.

active perception The main aim of the research is to have an optimal view point estimate for a moving camera base, e.g. a robot arm with a camera mounted on it. This is essential in the coworking setup of an industrial robot in order to cope with the uncertainties due to the human worker and the un-modeled dynamics around the robot.

active mapping The main aim of the topic is to construct a versatile map representation for navigating agent which can learn a predefined set of objects of interests. The representation of the information is essential in this case, thus different novel map representation techniques are planned to be used.

Long term goals

For the long term goals I am seeking service based recognition approaches for the mobile robotics domain. This has both theoretical and practical challenges which have to be solved in order to earn a flexible, reusable environment perception module.

In order to develop such a complex perception module, several components have to reused/integrated from the aforementioned short term research goals. The desired

end-to-end module will highly depend on the architectural/fundamental developments of the AI based perception techniques.

In long term I am also planning to attract funding for this research both from academic and industrial funding agencies in order to develop and maintain a research group in the broad field of robotics.

Teaching activities

Past

My teaching activity so far was involved in Romanian academic system beside the invited talks given at different summer shools/semniar abroad (especially in Germany, Hungary and Switzerland). At the UTCluj side I have been involved in two main courses: Hydro-pneumatic control equipment and Manufacturing Execution Systems. For the later one, the integration of the perception and robotics research domain proved to be fruitful.

I actively take part in the university promoting for the high school, doing from 2 years a volunteer course in the field of robotics in high-schools. This helps to have a better understanding among the young students of the engineering and indirectly promoting the STEM values for them. In long term this means better motivated students at the university and a larger number of young scientists for the next years.

Future

In the future I would like to extend the my teaching activities based on my industrial experience and network connections. As in the engineering field is essential the applied scientific approach of a problem, I will focus on both the theoretical and practical aspects of the teaching curricula.

Based on the already established network with international companies such as

Bosch, Fest , Baumann or Accenture we plan to extend the laboratory works using the latest equipment and technologies from the control engineering and robotics field.

In the future I also plan to get more involved in the promotion of the STEM curricula for the high school students showing them the real benefits of the engineering.

Acknowledment

The work presented here reflects the contribution of a number of collaborators to whom I am grateful. Special credits must be taken to the BSc, MSc and PhD students included but not limited to Robert Frohlich, Mezei-Ady Daniel, Cristian Militaru, Elod Pall, Andreea Blaga, Goron Lucian, Alexander Bulezyuk and Orsoly Fulop who contributed to the majority of the publications from which appeared in a number of conferences and journal papers.

I had a long term cooperation with Zoltan Kato from Szeged who contributed to the theoretical investigations presented in chapter 5 and with Lucian Busoniu with whom we cooperated in topics presented in chapter 5 beside my PhD advisor Gheoghe Lazea who played an important role in the promotion of the postdoc projects from my activity. Beside the aforementioned names, I was influenced directly or indirectly in my scientific life by a large number of people. And of course I need to thank to my family supporting me both during my stays abroad and in Cluj.

For the project founding I must acknowledge the support from the UEFISCDI in a number of different projects as well as from the Sciex, MTA Bolyai foundations, POSTDRU and the OTKA funding schemes.

Finally, as the most of the current work is based on our already published papers, I am grateful for their publishing services managing to reach our papers the scientific community. Explicitly I recognize that the copyright materials for the already published work remain with the respective publisher.

References

(2018). ARToolKit. 126

(2018). Kth camera robot calibration. 128

(2018). *The OpenCV Reference Manual*. Itseez, 2nd edn. 126

ABADI, M., AGARWAL, A., BARHAM, P., BREVDO, E., CHEN, Z., CITRO, C., CORRADO, G.S., DAVIS, A., DEAN, J., DEVIN, M., GHEMAWAT, S., GOODFELLOW, I., HARP, A., IRVING, G., ISARD, M., JIA, Y., JOZEFOWICZ, R., KAISER, L., KUDLUR, M., LEVENBERG, J., MANÉ, D., MONGA, R., MOORE, S., MURRAY, D., OLAH, C., SCHUSTER, M., SHLENS, J., STEINER, B., SUTSKEVER, I., TALWAR, K., TUCKER, P., VANHOUCKE, V., VASUDEVAN, V., VIÉGAS, F., VINYALS, O., WARDEN, P., WATTENBERG, M., WICKE, M., YU, Y. & ZHENG, X. (2015). TensorFlow: Large-scale machine learning on heterogeneous systems. Software available from tensorflow.org. 55, 57

ALDOMA, A., VINCZE, M., BLODOW, N., GOSSOW, D., GEDIKLI, S., RUSU, R. & BRADSKI, G. (2011). Cad-model recognition and 6dof pose estimation using 3d cues. In *Computer Vision Workshops (ICCV Workshops), 2011 IEEE International Conference on*, 585–592. 47

ALDOMA, A., MARTON, Z.C., TOMBARI, F., WOHLKINGER, W., POTTHAST, C., ZEISL, B., RUSU, R.B., GEDIKLI, S. & VINCZE, M. (2012). Tutorial: Point

cloud library: Three-dimensional object recognition and 6 dof pose estimation. *IEEE Robot. Automat. Mag.*, **19**, 80–91. 43

ALDOMA, A., TOMBARI, F., PRANKL, J., RICHTSFELD, A., DI STEFANO, L. & VINCZE, M. (2013). Multimodal cue integration through hypotheses verification for rgb-d object recognition and 6dof pose estimation. In *Robotics and Automation (ICRA), 2013 IEEE International Conference on*, 2104–2111, IEEE. 112

ALEOTTI, J., LODI RIZZINI, D. & CASELLI, S. (2014). Perception and Grasping of Object Parts from Active Robot Exploration. *Journal of Intelligent and Robotic Systems: Theory and Applications*, **76**, 401–425. 118

ALI, H., SHAFAIT, F., GIANNAKIDOU, E., VAKALI, A., FIGUEROA, N., VARVADOUKAS, T. & MAVRIDIS, N. (2014). Contextual object category recognition for RGB-D scene labeling. *Robotics and Autonomous Systems*, **62**, 241–256. 39

ALISMAIL, H.S., BAKER, L.D. & BROWNING, B. (2012). Automatic calibration of a range sensor and camera system. In *Second Joint 3DIM/3DPVT Conference: 3D Imaging, Modeling, Processing, Visualization and Transmission*, 286–292, IEEE, Zurich, Switzerland. 70, 71

ALOIMONOS, J., WEISS, I. & BANDOPADHAY, A. (1988). Active vision. *International Journal of Computer Vision*, **1**, 333 – 356. 117

ATANASOV, N., SANKARAN, B., LE NY, J., KOLETSCHKA, T., PAPPAS, G.J. & DANIILIDIS, K. (2013). In *Robotics and Automation (ICRA), 2013 IEEE International Conference on*, IEEE. 107

ATANASOV, N., LE NY, J., DANIILIDIS, K. & PAPPAS, G.J. (2015). Decentralized active information acquisition: Theory and application to multi-robot SLAM. *2015 IEEE International Conference on Robotics and Automation (ICRA)*, 4775–4782. 117

BAJCSY, R. (1988a). Active perception. In *Proc IEEE, 76:996–1005*. 105, 106

BAJCSY, R. (1988b). Active perception. *The Proceedings of the IEEE*, **76**, 966–1005. 117

BAY, H., ESS, A., TUYTELAARS, T. & VAN GOOL, L. (2008). Speeded-Up Robust Features (SURF). *Computer Vision and Image Understanding (CVIU)*, **110**, 346–359. 13, 16

BEARD, C., CHEN, Z.Q., KUMAR, V., LEE, Y., LEON-SALAS, W.D. & RAO, P. (2013). SAVEUS: Saving Victims in Earthquakes through Unified Systems. *IJCNDS*, **10**, 402–420. 91

BELHEDI, A., BOURGEOIS, S., GAY-BELLILE, V., SAYD, P., BARTOLI, A. & HAMROUNI, K. (2012). Non-parametric depth calibration of a tof camera. In *Image Processing (ICIP), 2012 19th IEEE International Conference on*, 549–552. 40

BENINI, A., MANCINI, A. & LONGHI, S. (2013). An IMU/UWB/Vision-based Extended Kalman Filter for Mini-UAV Localization in Indoor Environment using 802.15. 4a Wireless Sensor Network. *Journal of Intelligent & Robotic Systems*, **70**, 461–476. 91

BENTLEY, J.L. (1975). Multidimensional Binary Search Trees Used for Associative Searching. *Communication of ACM*, **18**, 509–517. 22

BESL, P.J. & MCKAY, H.D. (1992). A Method for Registration of 3-D Shapes. *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, **14**, 239–256. 20, 23, 24

BEST, G.C. (1964). Helpful formulas for integrating polynomials in three dimensions (in Technical Notes and Short Papers). *International Journal of Mathematics and Computer Science*, **18**, 310–312. 73

BILLS, C., CHEN, J. & SAXENA, A. (2011). Autonomous MAV flight in indoor environments using single image perspective clues. In *IEEE International Conference on Robotics and Automation*, 5776–5783, IEEE. 98

BLAGA, A. & TAMAS, L. (2018). Augmented reality for digital manufacturing. In *2018 26th Mediterranean Conference on Control and Automation (MED)*, 173–178, IEEE. viii, x, 90, 124, 126, 127, 128, 129, 137

BORRMANN, D., AFZAL, H., ELSEBERG, J. & NÜCHTER, A. (2012). Thermal 3d modeling of indoor environments for saving energy. In *IEEE/RSJ International Conference on Intelligent Robots and Systems, IROS 2012, Vilamoura, Algarve, Portugal, October 7-12*, 4538–4539, IEEE. 78

CANNY, J. (1986). A computational approach to edge detection. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 679–698. 98

CAUDELL, T.P. & MIZELL, D.W. (1992). Augmented reality: An application of heads-up display technology to manual manufacturing processes. In *System Sciences, 1992. Proceedings of the Twenty-Fifth Hawaii International Conference on*, vol. 2, 659–669, IEEE. 124

CENSI, A. (2007). An accurate closed-form estimate of icp's covariance. In *ICRA*, 3167–3172, IEEE. 26

CHOLLET, F. *et al.* (2015). Keras. `https://keras.io`. 63

DAI, J., LI, Y., HE, K. & SUN, J. (2016). R-FCN: object detection via region-based fully convolutional networks. *CoRR*, **abs/1605.06409**. 55

DOMOKOS, C., NEMETH, J. & KATO, Z. (2012). Nonlinear shape registration without correspondences. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, **34**, 943–958. 4, 69, 72, 74, 81, 82, 83

DOUGLAS, D.H. & PEUCKER, T.K. (1973). Algorithms for the reduction of the number of points required to represent a digitized line or its caricature. *Cartographica: The International Journal for Geographic Information and Geovisualization*, **10**, 112–122. 132

ELSEBERG, J., BORRMANN, D. & NÜCHTER, A. (2012). 6dof semi-rigid slam for mobile scanning. In *IROS*, 1865–1870, IEEE. 26

FAWCETT, T. (2006). An introduction to roc analysis. *Pattern Recogn. Lett.*, **27**, 861–874. 40, 48

FEKETE, A. (2015). Robot grasping and manipulation of objects. Tech. rep. 108

FISCHLER, M.A. & BOLLES, R.C. (1981). Random Sample Consensus: A Paradigm for Model Fitting with Applications to Image Analysis and Automated Cartography. *Communications of the ACM*, **24**, 381–395. 9, 30, 34

FOOTE, T. (2018). Turtlebot navigation stack. 130

FREEDMAN, S.A.M.M.A.Y., B. (2010). Depth mapping using projected patterns. 18

FULOP, A.O. & TAMAS, L. (2018). Lessons learned from lightweight cnn based object recognition for mobile robots. In *2018 IEEE International Conference on Automation, Quality and Testing, Robotics (AQTR)*, 1–5, IEEE. 39

FURGALE, P.T., SCHWESINGER, U., RUFLI, M., DERENDARZ, W., GRIMMETT, H., MUEHLFELLNER, P., WONNEBERGER, S., TIMPNER, J., ROTTMANN, S., LI, B., SCHMIDT, B., NGUYEN, T.N., CARDARELLI, E., CATTANI, S., BRUNING, S., HORSTMANN, S., STELLMACHER, M., MIELENZ, H., KÖSER, K., BEERMANN, M., HANE, C., HENG, L., LEE, G.H., FRAUNDORFER, F., ISER, R., TRIEBEL, R., POSNER, I., NEWMAN, P., WOLF, L.C., POLLEFEYS, M., BROSIG, S., EFFERTZ, J., PRADALIER, C. & SIEGWART, R. (2013). Toward automated driving in cities using close-to-market sensors: An overview of the

V-Charge Project. In *Intelligent Vehicles Symposium*, 809–816, Gold Coast City, Australia. 4, 68

GEIGER, A., MOOSMANN, F., CAR, O. & SCHUSTER, B. (2012). Automatic camera and range sensor calibration using a single shot. In *International Conference on Robotics and Automation*, 3936–3943, IEEE. 69

GEIGER, A., LAUER, M., WOJEK, C., STILLER, C. & URTASUN, R. (2014). 3D Traffic Scene Understanding From Movable Platforms. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, **36**, 1012–1025. 4, 68

GEYER, C. & DANIILIDIS, K. (2000). A unifying theory for central panoramic systems. In *European Conference on Computer Vision*, 445–462, Dublin, Ireland. 79

GIRSHICK, R.B. (2015). Fast R-CNN. *CoRR*, **abs/1504.08083**. 55

GIRSHICK, R.B., DONAHUE, J., DARRELL, T. & MALIK, J. (2013). Rich feature hierarchies for accurate object detection and semantic segmentation. *CoRR*, **abs/1311.2524**. 55

GOMEZ-BALDERAS, J.E., FLORES, G., CARRILLO, L.G. & LOZANO, R. (2013). Tracking a ground moving target with a quadrotor using switching control. *Journal of Intelligent & Robotic Systems*, **70**, 65–78. 98

GONG, X., LIN, Y. & LIU, J. (2013). Extrinsic calibration of a 3d lidar and a camera using a trihedron. *Optics and Lasers in Engineering*, **51**, 394 – 401. 70

GORON, L.C., TAMAS, L., RETI, I. & LAZEA, G. (2010a). 3D Laser Scanning System and 3D Segmentation of Urban Scenes. In *Proceedings of the 17th IEEE International Conference on Automation, Quality and Testing, Robotics (AQTR)*, Cluj-Napoca, Romania. 12

GORON, L.C., TAMAS, L., RETI, I. & LAZEA, G. (2010b). 3D laser scanning system and 3D segmentation of urban scenes. In *Automation Quality and Testing Robotics (AQTR), 2010 IEEE International Conference*, vol. 1, 1–5, IEEE. 13, 87

HANE, C., ZACH, C., COHEN, A., ANGST, R. & POLLEFEYS, M. (2013). Joint 3D Scene Reconstruction and Class Segmentation. *2013 IEEE Conference on Computer Vision and Pattern Recognition*, 97–104. 42

HAUSER, K. (2013). Recognition, prediction, and planning for assisted teleoperation of freeform tasks. *Autonomous Robots*, **35**, 241–254. 107

HERBST, E. & FOX, D. (2014). Active object segmentation for mobile robots. Tech. rep. 105, 107

HERBST, E., HENRY, P. & FOX, D. (2014). Toward online 3-d object segmentation and mapping. In *ICRA*, 3193–3200. 106

HERRERA C, D., KANNALA, J. & HEIKKILA, J. (2012). Joint depth and color camera calibration with distortion correction. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, **34**, 1–8. 70, 71

HERVIER, T., BONNABEL, S. & GOULETTE, F. (2012). Accurate 3d maps from depth images and motion sensors via nonlinear kalman filtering. In *2012 IEEE/RSJ International Conference on Intelligent Robots and Systems, IROS 2012, Vilamoura, Algarve, Portugal, October 7-12*, 5291–5297. 9, 11, 26

HIRANO, Y., KITAHAMA, K. & YOSHIZAWA, S. (2005). Image-based object recognition and dexterous hand/arm motion planning using rrts for grasping in cluttered scene. In *2005 IEEE/RSJ International Conference on Intelligent Robots and Systems, Edmonton, Alberta, Canada, August 2-6, 2005*, 2041–2046. 107

HOLZ, D., NIEUWENHUISEN, M., DROESCHEL, D., STÜCKLER, J., BERNER, A., LI, J., KLEIN, R. & BEHNKE, S. (2014). Active recognition and manipulation for mobile robot bin picking. In F. Röhrbein, G. Veiga & C. Natale, eds., *Gearing up and accelerating cross-fertilization between academic and industrial robotics research in Europe:*, vol. 94 of *Springer Tracts in Advanced Robotics*, 133–153, Springer International Publishing. 104, 106

HUITL, R., SCHROTH, G., HILSENBECK, S., SCHWEIGER, F. & STEINBACH, E. (2012). TUMindoor: An extensive image and point cloud dataset for visual indoor localization and mapping. In *Proc. of the International Conference on Image Processing*, Orlando, FL, USA. 26

IP, C.Y., LAPADAT, D., SIEGER, L. & REGLI, W.C. (2002). Using shape distributions to compare solid models. In *Proceedings of the Seventh ACM Symposium on Solid Modeling and Applications*, SMA '02, 273–280, ACM, New York, NY, USA. 47

JIA, Y., SHELHAMER, E., DONAHUE, J., KARAYEV, S., LONG, J., GIRSHICK, R., GUADARRAMA, S. & DARRELL, T. (2014). Caffe: Convolutional architecture for fast feature embedding. *arXiv preprint arXiv:1408.5093*. 55, 57

JOHNSON, A.E. & HEBERT, M. (1999). Using spin images for efficient object recognition in cluttered 3D scenes. *IEEE Trans. Pattern Anal. Mach. Intell.*, **21**, 433–449. 45

KANNALA, J. & BRANDT, S.S. (2006). A Generic Camera Model and Calibration Method for Conventional, Wide-Angle, and Fish-Eye Lenses. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, **28**, 1335–1340. 68

KANTER, G. (2012). Robot manipulator control using stereo visual feedback. Tech. rep. 107, 108

KASINSKI, A. & SKRZYPCZYNSKI, P. (2001). Perception network for the team of indoor mobile robots: concept, architecture, implementation. *Engineering Applications of Artificial Intelligence*, **14**, 125–137. 9

KAUSHIK, R., XIAO, J., MORRIS, W. & ZHU, Z. (2009). 3d laser scan registration of dual-robot system using vision. In *IEEE/RSJ International Conference on Intelligent Robots and Systems*, IROS'09, 4148–4153. 7, 9, 14

KHOSHELHAM, K. & ELBERINK, S.O. (2012). Accuracy and resolution of kinect depth data for indoor mapping applications. *Sensors*, **12**, 1437–1454. 11

KIRYATI, N., ELDAR, Y. & BRUCKSTEIN, A.M. (1991). A probabilistic Hough transform. *Pattern recognition*, **24**, 303–316. 97

KRAJNÍK, T., VONÁSEK, V., FIŠER, D. & FAIGL, J. (2011). AR-drone as a platform for robotic research and education. In *Research and Education in Robotics-EUROBOT*, 172–186, Springer. 91, 92

KRAUSE, S. & EVERT, R. (2012). Remission based improvement of extrinsic parameter calibration of camera and laser scanner. In *Control Automation Robotics Vision (ICARCV), 2012 12th International Conference on*, 829–834. 70

KRIZHEVSKY, A., SUTSKEVER, I. & HINTON, G.E. (2012). Imagenet classification with deep convolutional neural networks. In *Proceedings of the 25th International Conference on Neural Information Processing Systems - Volume 1*, NIPS'12, 1097–1105, Curran Associates Inc., USA. 3, 54, 59

KÜMMERLE, R., STEDER, B., DORNHEGE, C., RUHNKE, M., GRISETTI, G., STACHNISS, C. & KLEINER, A. (2009). On measuring the accuracy of slam algorithms. *Auton. Robots*, **27**, 387–407. 26

LAHOUD, J. & GHANEM, B. (2017). 2d-driven 3d object detection in rgb-d images. In *2017 IEEE International Conference on Computer Vision (ICCV)*, vol. 00, 4632–4640. 3, 54

LANGE, S., SÜNDERHAUF, N., NEUBERT, P., DREWS, S. & PROTZEL, P. (2012). Autonomous corridor flight of a UAV using a low-cost and light-weight RGB-D camera. In *Advances in Autonomous Mini Robots*, 183–192, Springer. 98

LAVALLE, S.M. (2006). *Planning Algorithms*. Cambridge University Press, New York, NY, USA. 107, 108

LAZEBNIK, S., SCHMID, C. & PONCE, J. (2005). A sparse texture representation using local affine regions. *IEEE Trans. Pattern Anal. Mach. Intell.*, **27**, 1265–1278. 44, 45

LEVINSON, J. & THRUN, S. (2013). Automatic online calibration of cameras and lasers. In *Proceedings of Robotics: Science and Systems*, Berlin, Germany. 70

LIBERZON, D. (2003). *Switching in systems and control*. Springer. 101

LIN, D., FIDLER, S. & URTASUN, R. (2013). Holistic Scene Understanding for 3D Object Detection with RGBD Cameras. In *International Conference on Computer Vision, Sydney, Australia*, 1417–1424, IEEE Computer Society. 4, 68

LOWE, D.G. (2004a). Distinctive Image Features from Scale-Invariant Keypoints. *International Journal of Computer Vision*, **60**, 91–110. 13, 16

LOWE, D.G. (2004b). Distinctive image features from scale-invariant keypoints. *Int. J. Comput. Vision*, **60**, 91–110. 44

MAGNUSSON, M., DUCKETT, T. & LILIENTHAL, A.J. (2007). Scan Registration for Autonomous Mining Vehicles Using 3D-NDT. *Journal of Field Robotics*, **24**, 803–827. 7

MAGNUSSON, M., NÜCHTER, A., LÖRKEN, C., LILIENTHAL, A.J. & HERTZBERG, J. (2009). Evaluation of 3d registration reliability and speed - a comparison of icp and ndt. In *ICRA*, 3907–3912, IEEE. 9, 11, 26

MAJDIK, A., POPA, M., TAMAS, L., SZOKE, I. & LAZEA, G. (2010). New Approach in Solving the Kidnapped Robot Problem. In *The 41st International Symposium on Robotics*. 15

MARDER-EPPSTEIN, E. (2016). Project tango. In *ACM SIGGRAPH 2016 Real-Time Live!*, 25, ACM. 131

MARTÍN-GUTIÉRREZ, J. & GINTERS, E., eds. (2013). *2013 International Conference on Virtual and Augmented Reality in Education, VARE 2013, 7-9 November 2013, Puerto de la Cruz, Tenerife, Spain*, vol. 25 of *Procedia Computer Science*, Elsevier. 125

MARTON, Z.C., PANGERCIC, D., BLODOW, N., KLEINEHELLEFORT, J. & BEETZ, M. (2010). General 3D Modelling of Novel Objects from a Single View. In *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 3700–3705, Taipei, Taiwan. 34

MARTON, Z.C., PANGERCIC, D., BLODOW, N. & BEETZ, M. (2011). Combined 2D-3D categorization and classification for multimodal perception systems. *Int. J. Rob. Res.*, **30**, 1378–1402. 47

MASTIN, A., KEPNER, J. & III, J.W.F. (2009). Automatic registration of lidar and optical images of urban scenes. In *IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 2639–2646, IEEE, Miami, Florida, USA. 69, 70, 71

MAY, S., DROESCHEL, D., FUCHS, S., HOLZ, D. & NÜCHTER, A. (2009). Robust 3d-mapping with time-of-flight cameras. In *IROS*, 1673–1678, IEEE. 9

MEZEI, A.D. & TAMAS, L. (2016). Active perception for object manipulation. In *Intelligent Computer Communication and Processing (ICCP), 2016 IEEE 12th International Conference on*, 269–274, IEEE. 119

MIČUŠÍK, B. (2004). *Two-View Geometry of Omnidirectional Cameras*. Phd thesis, Department of Cybernetics, Faculty of Electrical Engineering, Czech Technical University, Prague, Czech Republic. 80

MIČUŠÍK, B. & PAJDLA, T. (2004). Para-catadioptric Camera Auto-calibration from Epipolar Geometry. In *Asian Conference on Computer Vision*, vol. 2, 748–753, Seoul, Korea South. 80

MILGRAM, P., TAKEMURA, H., UTSUMI, A. & KISHINO, F. (1994). Augmented reality: A class of displays on the reality-virtuality continuum. 282–292. 122

MILITARU, C., MEZEI, A.D. & TAMAS, L. (2016a). Object handling in cluttered indoor environment with a mobile manipulator. In *2016 IEEE International Conference on Automation, Quality and Testing, Robotics (AQTR)*, 1–6, IEEE. 90, 116

MILITARU, C., MEZEI, D. & TAMAS, L. (2016b). Object handling in cluttered indoor environment with a mobile manipulator. In *AQTR 2016: International Conference on Automation, Quality and Testing, Robotics*. 58

MILITARU, C., MEZEI, A.D. & TAMAS, L. (2017). Lessons learned from a cobot integration into MES. In *ICRA - Recent Advances in Dynamics for Industrial Applications Workshop*, Singapore. 116

MIRZAEI, F.M., KOTTAS, D.G. & ROUMELIOTIS, S.I. (2012). 3d lidar-camera intrinsic and extrinsic calibration: Identifiability and analytical least-squares-based initialization. *The International Journal of Robotics Research*, **31**, 452–467. 68, 70, 71

MISHRA, R. & ZHANG, Y. (2012). A review of optical imagery and airborne lidar data registration methods. *The Open Remote Sensing Journal*, **5**, 54–63. 69

MITRA, J., KATO, Z., MARTI, R., OLIVER, A., LLADÓ, X., SIDIBÉ, D., GHOSE, S., VILANOVA, J.C., COMET, J. & MERIAUDEAU, F. (2012). A Spline-based

Non-linear Diffeomorphism for Multimodal Prostate Registration. *Medical Image Analysis*, **16**, 1259–1279. 82

MIZELL, D. (2001). Boeing's wire bundle assembly project. *Fundamentals of Wearable Computers and Augmented Reality*, 447–467. 123

MONICA, R., ALEOTTI, J. & CASELLI, S. (2016). A kinfu based approach for robot spatial attention and view planning. *Robotics and Autonomous Systems*, **75**, 627–640. 107

MORISSET, B., RUSU, R.B., SUNDARESAN, A., HAUSER, K., AGRAWAL, M., LATOMBE, J.C. & BEETZ, M. (2009). Leaving Flatland: Toward Real-Time 3D Navigation. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, 3786–3793. 10, 11, 23, 45, 46

MUJA, M. & LOWE, D.G. (2009). Fast approximate nearest neighbors with automatic algorithm configuration. In *International Conference on Computer Vision Theory and Application VISSAPP'09)*, 331–340, INSTICC Press. 49

MURILLO, A.C., KOŠECKÁ, J., GUERRERO, J.J. & SAGÜÉS, C. (2008). Visual Door Detection Integrating Appearance and Shape Cues. *Robotics and Autonomous Systems*, **56**, 512–521. 10

NAGATANI, K., MATSUZAWA, T. & YOSHIDA, K. (2009). Scan-point planning and 3-d map building for a 3-d laser range scanner in an outdoor environment. In A. Howard, K. Iagnemma & A. Kelly, eds., *FSR*, vol. 62 of *Springer Tracts in Advanced Robotics*, 207–217, Springer. 11

NARODITSKY, O., IV, E.P. & DANIILIDIS, K. (2011). Automatic alignment of a camera with a line scan lidar system. In *International Conference on Robotics and Automation*, 3429–3434, IEEE, Shanghai, China. 70, 71

NEE, A., ONG, S., CHRYSSOLOURIS, G. & MOURTZIS, D. (2012). Augmented reality applications in design and manufacturing. *CIRP Annals*, **61**, 657–679. 125

NÜCHTER, A. & HERTZBERG, J. (2008). Towards Semantic Maps for Mobile Robots. *Robotics and Autonomous Systems*, **56**, 915–926. 9

NUECHTER, A. (2009). *3D Robotic Mapping*. Springer Tracts in Advanced Robotics (STAR), Springer. 20

NUNEZ, P., DREWS, P., ROCHA, R. & DIAS, J. (2009). Data fusion calibration for a 3d laser range finder and a camera using inertial data. In *European Conference on Mobile Robots*, 31—36, Dubrovnik, Croatia. 70

ODIDO, M.D. & MADARA, D. (2013). Emerging Technologies: Use of Unmanned Aerial Systems in the Realisation of Vision 2030 Goals in the Counties. *International Journal of Applied*, **3**. 91

OHNO, K., TADOKORO, S., NAGATANI, K., KOYANAGI, E. & YOSHIDA, T. (2010). Trials of 3-D Map Construction Using the Tele-operated Tracked Vehicle Kenaf at Disaster City. In *International Conference on Robotics and Automation (ICRA)*, 2864–2870, Anchorage, AK, USA. 11

PÁLL, E., TAMÁS, L. & BUŞONIU, L. (2015). Vision-based quadcopter navigation in structured environments. In *Handling Uncertainty and Networked Structure in Robot Control*, 265–290, Springer, Cham. 90

PÁLL, E., TAMÁS, L. & BUŞONIU, L. (2016). Analysis and a home assistance application of online aems2 planning. In *Intelligent Robots and Systems (IROS), 2016 IEEE/RSJ International Conference on*, 5013–5019, IEEE. 116

PALMARINI, R., ERKOYUNCU, J.A., ROY, R. & TORABMOSTAEDI, H. (2018). A systematic review of augmented reality applications in maintenance. *Robotics and Computer-Integrated Manufacturing*, **49**, 215 – 228. 123

PANDEY, G., MCBRIDE, J.R., SAVARESE, S. & EUSTICE, R.M. (2012). Automatic targetless extrinsic calibration of a 3d lidar and camera by maximizing mu-

tual information. In *Proceedings of the AAAI National Conference on Artificial Intelligence*, 2053–2059, Toronto, Canada. 68, 69, 70, 71

PAPADAKIS, P. (2013). Terrain traversability analysis methods for unmanned ground vehicles: A survey. *Engineering Applications of Artificial Intelligence*, **26**, 1373–1385. 10

PATHAK, K., BIRK, A., VASKEVICIUS, N. & POPPINGA, J. (2010a). Fast registration based on noisy planes with unknown correspondences for 3D mapping. *IEEE Transactions on Robotics*, **26**, 424–441. 24

PATHAK, K., BIRK, A., VASKEVICIUS, N. & POPPINGA, J. (2010b). Fast Registration Based on Noisy Planes with Unknown Correspondencesfor 3D Mapping. *IEEE Transactions on Robotics*, **26**, 424–441. 9, 20, 22

PATTEN, T., ZILLICH, M., FITCH, R., VINCZE, M. & SUKKARIEH, S. (2016). Viewpoint Evaluation for Online 3-D Active Object Classification. *IEEE Robotics and Automation Letters*, **1**, 73–81. 117

PAVLIK, J.V. (2014). Transformation: Examining the Implications of Emerging Technology for Journalism, Media and Society. *Athens Journal of Mass Media and Communications*. 91

PERZYLO, A., SOMANI, N., PROFANTER, S., KESSLER, I., RICKERT, M. & KNOLL, A. (2016). Intuitive instruction of industrial robots: Semantic process descriptions for small lot production. In *2016 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2293–2300, IEEE. 122

PETERFREUND, N. (1999). Robust tracking of position and velocity with Kalman snakes. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, **21**, 564–569. 97

PLUIM, J., MAINTZ, J. & VIERGEVER, M. (2003). Mutual-information-based registration of medical images: a survey. *Medical Imaging, IEEE Transactions on*, **22**, 986–1004. 69

POTTHAST, C. & SUKHATME, G.S. (2014). A probabilistic framework for next best view estimation in a cluttered environment. *J. Visual Communication and Image Representation*, **25**, 148–164. 106

RADKOWSKI, R., HERREMA, J. & OLIVER, J. (2015). Augmented reality-based manual assembly support with visual features for different degrees of difficulty. *International Journal of Human–Computer Interaction*, **31**, 337–349. 123

REDMON, J. (2013–2016). Darknet: Open source neural networks in c. `http://pjreddie.com/darknet/`. vi, 55, 57, 60

REDMON, J. & FARHADI, A. (2016). Yolo9000: Better, faster, stronger. *arXiv preprint arXiv:1612.08242*. vi, 57, 60

REDMON, J., DIVVALA, S.K., GIRSHICK, R.B. & FARHADI, A. (2015). You only look once: Unified, real-time object detection. *CoRR*, **abs/1506.02640**. 55

REN, S., HE, K., GIRSHICK, R.B. & SUN, J. (2015). Faster R-CNN: towards real-time object detection with region proposal networks. *CoRR*, **abs/1506.01497**. 55

RUSU, R. & COUSINS, S. (2011a). 3d is here: Point cloud library (pcl). In *Robotics and Automation (ICRA), 2011 IEEE International Conference on*, 1 –4. 40

RUSU, R.B. (2009). *Semantic 3D Object Maps for Everyday Manipulation in Human Living Environments*. PhD Thesis. 9

RUSU, R.B. & COUSINS, S. (2011b). 3D is here: Point Cloud Library (PCL). In *Proceedings of the Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, Shanghai, China. 9

RUSU, R.B. & COUSINS, S. (2011c). 3D is here: Point Cloud Library (PCL). In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, 1–4. 19

RUSU, R.B. & COUSINS, S. (2011d). 3D is here: Point cloud library (PCL). In *In Robotics and Automation (ICRA), 2011 IEEE International Conference on*, 1–4, IEEE. 119

RUSU, R.B., MARTON, Z.C., BLODOW, N., DOLHA, M. & BEETZ, M. (2008). Towards 3D Point Cloud Based Object Maps for Household Environments. *Robotics and Autonomous Systems*, **56**, 927–941. 9, 31, 32, 34

RUSU, R.B., BRADSKI, G., THIBAUX, R. & HSU, J. (2010a). Fast 3D Recognition and Pose Using the Viewpoint Feature Histogram. In *Proceedings of the 23rd IEEE International Conference on Intelligent Robots and Systems (IROS)*, Taipei, Taiwan. 9

RUSU, R.B., BRADSKI, G., THIBAUX, R. & HSU, J. (2010b). Fast 3D Recognition and Pose Using the Viewpoint Feature Histogram. In *Proceedings of the 23rd IEEE International Conference on Intelligent Robots and Systems (IROS)*, Taipei, Taiwan. 46

RUSU, R.B., BRADSKI, G., THIBAUX, R. & HSU, J. (2010c). Fast 3d recognition and pose using the viewpoint feature histogram. *2010 IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2155–2162. 119

SALVI, J., MATABOSCH, C., FOFI, D. & FOREST, J. (2007). A Review of Recent Range Image Registration Methods with Accuracy Evaluation. *Image and Vision Computing*, **25**, 578–596. 21

SANKARAN, B., ATANASOV, N., LE NY, J., KOLETSCHKA, T., PAPPAS, G. & DANIILIDIS, K. (2013). Hypothesis testing framework for active object detection. In *IEEE International Conference on Robotics and Automation (ICRA)*. 105

SANTA, Z. & KATO, Z. (2013). Correspondence-Less Non-rigid Registration of Triangular Surface Meshes. In *Conference on Computer Vision and Pattern Recognition*, 2275–2282, IEEE Computer Society, Washington, DC, USA. 82

SCARAMUZZA, D., MARTINELLI, A. & SIEGWART, R. (2006a). A Flexible Technique for Accurate Omnidirectional Camera Calibration and Structure from Motion. In *International Conference on Computer Vision Systems*, 45–51, Washington, USA. 80

SCARAMUZZA, D., MARTINELLI, A. & SIEGWART, R. (2006b). A Toolbox for Easily Calibrating Omnidirectional Cameras. In *International Conference on Intelligent Robots*, 5695–5701, Bejing. 68, 80, 86

SCARAMUZZA, D., HARATI, A. & SIEGWART, R. (2007). Extrinsic self calibration of a camera and a 3d laser range finder from natural scenes. In *IEEE International Conference on Intelligent Robots and Systems*, 4164–4169, IEEE/RSJ, IEEE, San Diego, USA. 68, 69, 70, 71

SCHARSTEIN, D. & SZELISKI, R. (2002). A Taxonomy and Evaluation of Dense Two-Frame Stereo Correspondence Algorithms. *International Journal of Computer Vision*, **47**, 7–42. 3, 7

SCHULZ, D., BURGARD, W., FOX, D. & CREMERS, A.B. (2001). Tracking multiple moving targets with a mobile robot using particle filters and statistical data association. In *IEEE International Conference on Robotics and Automation*, vol. 2, 1665–1670, IEEE. 97

SHAMS, R., KENNEDY, R.A., SADEGHI, P. & HARTLEY, R.I. (2007). Gradient intensity-based registration of multi-modal images of the brain. In *IEEE 11th International Conference on Computer Vision, ICCV 2007, Rio de Janeiro, Brazil, October 14-20, 2007*, 1–8, IEEE. 45

SHARP, G.C., LEE, S.W. & WEHE, D.K. (2002). ICP Registration Using Invariant Features. *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, **24**, 90–102. 21

SILVA, L.C.D., MORIKAWA, C. & PETRA, I.M. (2012). State of the art of smart homes. *Engineering Applications of Artificial Intelligence*, **25**, 1313–1321. 10

SOMANI, A., YE, N., HSU, D. & LEE, W.S. (2013). Despot: Online pomdp planning with regularization. In *Advances in Neural Information Processing Systems 26*, vol. 2, 1772–1780. 117, 120

STAMOS, I. & ALLEN, P.K. (2001). Automatic registration of 2-d with 3-d imagery in urban environments. In *International Conference on Computer Vision*, 731–737. 70

STEDER, B., RUSU, R.B., KONOLIGE, K. & BURGARD, W. (2010). NARF: 3D Range Image Features for Object Recognition. In *Workshop at the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, Taipei, Taiwan. 13

STEDER, B., RUSU, R.B. & KONOLIGE, K. (2011). Point Feature Extraction on 3D Range Scans Taking into Account Object Boundaries. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, 2601–2608. 13

STEPHANE, P., NICOLAS, B., PIERRE, E. & FREDERIC, D.H. (2012). AR.Drone Developer Guide. 91

STEPHENS, R.S. (1991). Probabilistic approach to the Hough transform. *Image and vision computing*, **9**, 66–71. 97

STRASDAT, H. (2012). *Local Accuracy and Global Consistency for Efficient Visual SLAM*. PhD Thesis. 9, 26

SUCAN, I.A., MOLL, M. & KAVRAKI, L.E. (2012). The Open Motion Planning Library. *IEEE Robotics & Automation Magazine*, **19**, 72–82. 107

SURMANN, H., LINGEMANN, K., NUCHTER, A. & HERTZBERG, J. (2001). A 3D Laser Range Finder for Autonomous Mobile Robots. 153–158, Seoul, South Korea. 20

SYBERFELDT, A., DANIELSSON, O.M. & GUSTAVSSON, P. (2017). Augmented reality smart glasses in the smart factory: Product evaluation guidelines and review of available products. *IEEE Access*, **5**, 9118–9130. 125

TAMAS, L. & BABOLY, L. (2017). Industry 4.0 – mes vertical integration use-case with a cobot. In *ICRA - IC3 Workshop*, Singapore. 116

TAMAS, L. & GORON, L.C. (2012). 3D Map Building with Mobile Robots. In *Proceedings of the 20th Mediterranean Conference on Control and Automation (MED)*, Barcelona, Spain. 9, 11, 12

TAMAS, L. & GORON, L.C. (2014). 3d semantic interpretation for robot perception inside office environments. *Engineering Applications of Artificial Intelligence*, **32**, 76–87. 6, 30, 33, 35

TAMAS, L. & JENSEN, B. (2014). Robustness analysis of 3d feature descriptors for object recognition using a time-of-flight camera. In *22nd Mediterranean Conference on Control and Automation*, 1020–1025, IEEE. 39, 58

TAMAS, L. & KATO, Z. (2013a). Targetless Calibration of a Lidar - Perspective Camera Pair. In *International Conference on Computer Vision, Bigdata3dcv Workshops*, 668–675, Sydney, Australia. 4, 67

TAMAS, L. & KATO, Z. (2013b). Targetless Calibration of a Lidar - Perspective Camera Pair. In *International Conference on Computer Vision, Bigdata3dcv Workshops*, 668–675, Sydney, Australia. 81

TAMAS, L. & MAJDIK, A. (2012a). Heterogeneous Feature Based Correspondence Estimation. In *Proceedings of the IEEE International Conference on Multisensor Fusion and Integration for Intelligent Systems (MFI)*, 89–94, Hamburg, Germany. 9

TAMAS, L. & MAJDIK, A. (2012b). Heterogeneous feature based correspondence estimation. In *IEEE Conference on Multisensor Fusion and Integration for Intelligent Systems*, 89–94, IEEE, IEEE, Munich, Germany. 25, 87

TAMAS, L., FROHLICH, R. & KATO, Z. (2014). Relative pose estimation and fusion of omnidirectional and lidar cameras. In *ECCV Workshop on Computer Vision for Road Scene Understanding and Autonomous Driving (ECCV-CVRSUAD)*, Lecture Notes in Computer Science, 1–12, Zurich, Switzerland. 4, 67

TATIC, D. & TESIC, B. (2017). The application of augmented reality technologies for the improvement of occupational safety in an industrial environment. *Computers in Industry*, **85**, 1 – 10. 123

TAYLOR, Z. & NIETO, J. (2012). A mutual information approach to automatic calibration of camera and lidar in natural environments. In *Australian Conference on Robotics and Automation*, 3–5, Wellington, Australia. x, 68, 69, 70, 79

TOLA, E. & LEPETIT, V. (2010). Daisy: An efficient dense descriptor applied to wide baseline stereo. *PAMI*, **32**. 16

TOLA, E., LEPETIT, V. & FUA, P. (2008). A fast local descriptor for dense matching. In *Conference on Computer Vision and Pattern Recognition*, Alaska, USA. 16

TOMIC, T., SCHMID, K., LUTZ, P., DOMEL, A., KASSECKER, M., MAIR, E., GRIXA, I., RUESS, F., SUPPA, M. & BURSCHKA, D. (2012). Toward a Fully Autonomous UAV: Research Platform for Indoor and Outdoor Urban Search and Rescue. *Robotics Automation Magazine, IEEE*, **19**, 46–56. 91

TSAI, R.Y. & LENZ, R.K. (1988). Real time versatile robotics hand/eye calibration using 3d machine vision. In *Robotics and Automation, 1988. Proceedings., 1988 IEEE International Conference on*, 554–561, IEEE. 113

TUTTAS, S. & STILLA, U. (2011). Window Detection in Sparse Point Clouds using Indoor Points. In *ISPRS Conference: Photogrammetric Image Analysis (PIA)*, vol. XXXVIII-3/W22, Munich, Germany. 10, 34

UNNIKRISHNAN, R. & HEBERT, M. (2005). Fast extrinsic calibration of a laser rangefinder to a camera. Tech. rep., Carnegie Mellon University. 70, 71

VIOLA, P. & WELLS, W.M., III (1997). Alignment by maximization of mutual information. *International Journal of Computer Vision*, **24**, 137–154. 70

WAHL, E., HILLENBRAND, U. & HIRZINGER, G. (2003). Surflet-pair-relation histograms: a statistical 3d-shape representation for rapid classification. In *3-D Digital Imaging and Modeling, 2003. 3DIM 2003. Proceedings. Fourth International Conference on*, 474–481. 45

WANG, X., ONG, S. & NEE, A. (2018). A comprehensive survey of ubiquitous manufacturing research. *International Journal of Production Research*, **56**, 604–628. 123

WEYER, S., SCHMITT, M., OHMER, M. & GORECKY, D. (2015). Towards industry 4.0-standardization as the crucial challenge for highly modular, multi-vendor production systems. *IFAC-PapersOnLine*, **48**, 579–584. 123

WILLIAMS, N., LOW, K.L., HANTAK, C., POLLEFEYS, M. & LASTRA, A. (2004). Automatic image alignment for 3d environment modeling. In *Computer Graphics and Image Processing, 2004. Proceedings. 17th Brazilian Symposium on*, 388–395. 70

WOHLKINGER, W. & VINCZE, M. (2011). Ensemble of shape functions for 3d object classification. In *Robotics and Biomimetics (ROBIO), 2011 IEEE International Conference on*, 2987–2992. 47

WULF, O. & WAGNER, B. (2003). Fast 3D-scanning methods for laser measurement systems. In *Proceedings of the International Conference on Control Systems and Computer Science (CSCS)*, 183–188, IEEE Control Society, IEEE, Bucharest, Romania. 12

ZHANG, A., HU, S., CHEN, Y., LIU, H., YANG, F. & LIU, J. (2008). Fast Continuous 360 Degree Color 3D Laser Scanner. In *The International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences*, vol. 36, 409–415, ISPRS, Beijing, China. 7, 70

ZHANG, C. & ZHANG, Z. (2011). Calibration between depth and color sensors for commodity depth cameras. In *Proceedings of the 2011 IEEE International Conference on Multimedia and Expo*, 1–6, IEEE, Barcelona, Catalonia, Spain. 70

ZHANG, Q. (2004). Extrinsic calibration of a camera and laser range finder. In *International Conference on Intelligent Robots and Systems*, 2301 – 2306, IEEE, Sendai, Japan. 70

ZHANG, Z. (1992). *Iterative Point Matching for Registration of Free-Form Curves*. Technical Report No. RR-1658. 23

ZHANG, Z. (2000). A flexible new technique for camera calibration. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, **22**, 1330–1334. 95

ZOLLNER, R., ASFOUR, T. & DILLMANN, R. (2004). Programming by demonstration: Dual-arm manipulation tasks for humanoid robots. In *In Proceedings of the International Conference on Intelligent Robots and Systems*. 107