Engineering Applications of Artificial Intelligence 32 (2014) 76–87

Contents lists available at ScienceDirect

Engineering Applications of Artificial Intelligence

journal homepage: www.elsevier.com/locate/engappai

3D semantic interpretation for robot perception inside office environments

Levente Tamas [a,*], Lucian Cosmin Goron [b]

[a] Robotics Research Group, Department of Automation, Technical University of Cluj-Napoca (UTCLUJ), Cluj-Napoca, Romania
[b] Computer Vision Group, Computer Aided Medical Procedures and Augmented Reality, Technical University of Munich (TUM), Munich, Germany

ARTICLE INFO

Article history:
Received 21 May 2013
Received in revised form
3 November 2013
Accepted 2 March 2014

Keywords:
Robotics
Perception
Segmentation
Registration
Indoor mapping

ABSTRACT

Making sense out of human indoor environments is an essential feature for robots. The paper at hand presents a system for semantic interpretation of our surrounding indoor environments such as offices and kitchens. The perception and the interpretation of the measured data are essential tasks for any intelligent system. There are different techniques for processing 3D point clouds. The majority of them include acquisition, iterative registration, segmentation, or classification stages. We describe a generic pipeline for indoor data processing and semantic information extraction. The proposed pipeline is validated using several data sets collected using different 3D sensing devices.

1. Introduction

The 3D perception of the surrounding environment is still an important research field for both the industrial and the 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, and 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 and 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, and 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 paper 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

* Corresponding author. Tel.: +40 264 401586; fax: +40 264 599893.
E-mail addresses: levente.tamas@aut.utcluj.ro (L. Tamas), goron@in.tum.de (L. Cosmin Goron).
URLS: http://rrg.utcluj.ro/ (L. Tamas), http://ias.cs.tum.edu/people/goron (L. Cosmin Goron).

http://dx.doi.org/10.1016/j.engappai.2014.03.001
0952-1976/

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 paper are as follows:

- 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 for 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. Related work

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

A good example for creating the surrounding environment is given in Kasinski and 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, and hospitals – the application presented in Kasinski and Skrzypczynski (2001) is closer to business and enterprise.

Several research works 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 (Magnusson et al., 2009; Hervier et al., 2012) without closuring the loop during the mapping.

In Nüchter and 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 and Bolles, 1981) algorithm. Further identification of indoor components is obtained using a rather complex recognition pipeline.

The authors in Rusu et al. (2008) presented an approach similar to the one described in this paper in Section 4. 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.

With regard to door detection, some very interesting work has been done, such as Murillo et al. (2008) where authors use computer vision to classify pixels using a somehow smaller set of classes than 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 and 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, which is 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, and 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 and service robots, which can interact autonomously with the environment (Papadakis, 2013).

3. 3D scan registration

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 the orientation of the mapping robot are 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 and 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).

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 the 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 and Elberink, 2012).

In our experiments the data sets were recorded at normal light conditions for spaces ranging between a few cm and 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 and Goron, 2012). A single scan takes up to 60 s 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 1 cm. 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., 2010).

3.2. 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 and McKay, 1992; Nuechter, 2009; Pathak et al., 2010). 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 independently 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:

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

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 a 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 and 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.

3.3. 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, 2004) 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 paper 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 Table 1.

Table 1
Feature descriptor comparison.

Data set	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 0.1 K. 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.

3.4. 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, and sample consensus based one (Pathak et al., 2010).

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.

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

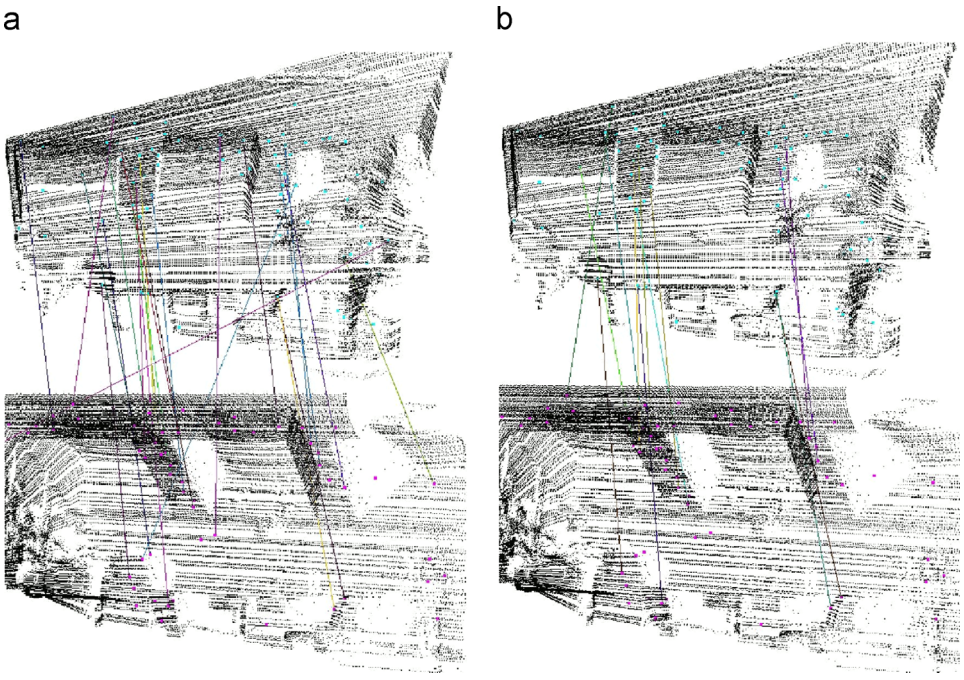

Fig. 1. Initial correspondences (a) and filtered correspondences (b).

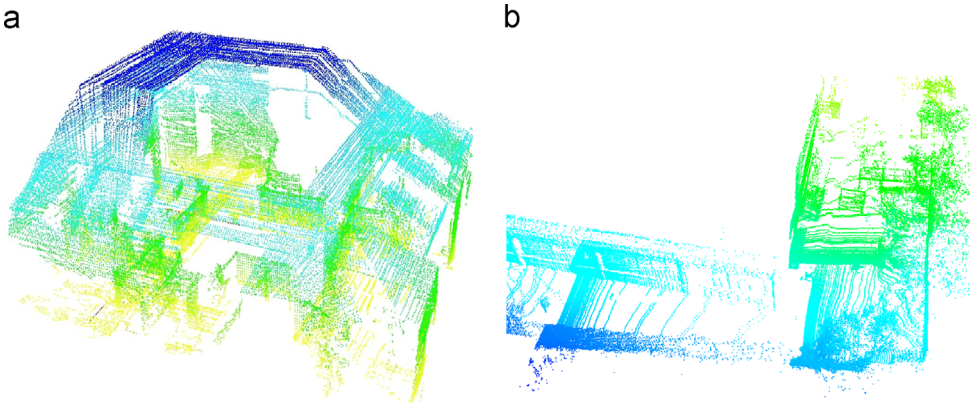

Fig. 2. Example of registered indoor map (a) and outdoor registered map (b).

The comparison of the unfiltered and filtered set of correspondences is shown in Fig. 1 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 depends on the parameters used as a threshold for the sample consensus rejection.

The complete ICP-based algorithm can be found in the works (Besl and McKay, 1992; Zhang, 1992), therefore only a short overview is given, with emphasis on the additional descriptor information for the points. The *ICP* with initial alignment is described in Algorithm 1 in the Appendix. It has two input point clouds, P_s for the source and P_t for the target. Steps 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 the outdoor environment is presented in Fig. 2. 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 1 m.

3.5. Performance evaluation

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 (Kümmerle et al., 2009; Strasdat, 2012; Huitl et al., 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. (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 considered as a least-square estimator which has the Cramer–Rao bound. This bound defined as $N = \mathrm{cov}(\hat{x}) = [I(x)]^{-1}$ where $\hat{x}$ represents the estimate of x and the $I(x)$ denotes the Fisher information matrix. The later matrix can be rewritten as

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

where σ denotes the sensor noise and A_i is the skew matrix composed of the point coordinates of the ith 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 data set 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 as $\sigma = 1$ cm. 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) were considered.

The summary of the evaluation run for different pairs of scans from the public data set is summarized in Table 2 for the NARF, FPFH and the combined set of keypoints used for initialization.

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 gives 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-axis on which the largest errors were measured as well.

Table 2
Evaluation of the registration on public data set.

Alg./transfrom error[cm]	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

4. 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 a PR2 (*Personal Robot 2*) robot are used for achieving this task. In the following, a comprehensive overview of the proposed method is given.

4.1. 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, look into Table 3.

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.

4.2. Principal axes estimation

In this subsection, a technique is described for estimating the principal axes of indoor point cloud data described in detail in Algorithm 2. First, planes are fitted using the RANSAC (Fischler and Bolles, 1981) algorithm, by segmenting the inliers of each plane out of the point cloud. For an example of the final fitted planes, look at Fig. 3(a). 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

Table 3
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

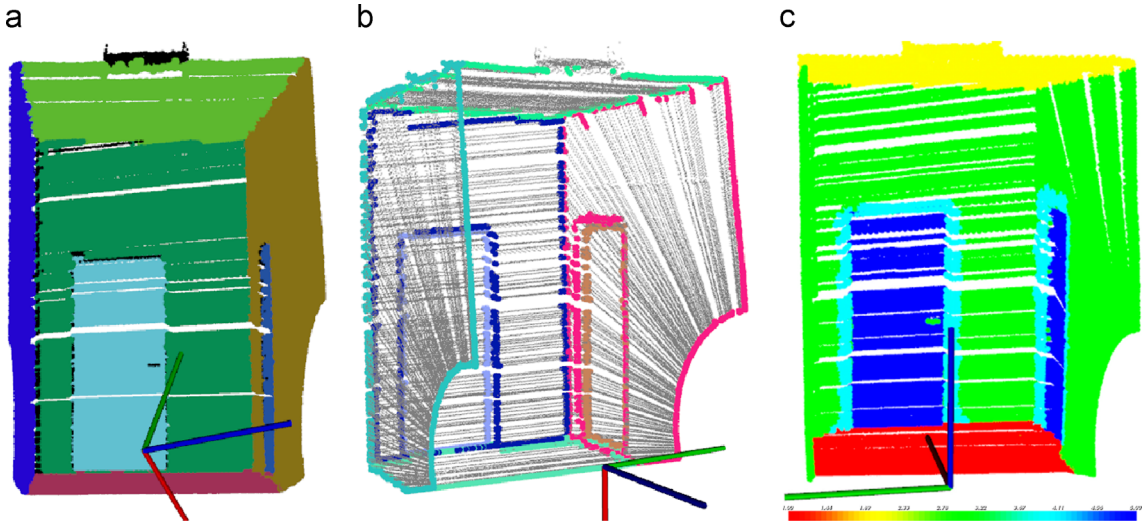

Fig. 3. Significant stages for the interpretation of indoor environments, where axes *X*, *Y*, and *Z*, are colored in *red*, *green*, and *blue*, respectively: (a) segmentation of planar surfaces, where it can be observed that all major planes are segmented correctly, including the two doors; the coordinate system is not correct, thus having the axes randomly tilted in 3*D* 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. (For interpretation of the references to color in this figure caption, the reader is referred to the web version of this paper.)

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 is most probably incorrect, as it can be seen in Fig. 3(b). 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 are needed, since the cross product between the two mentioned normals returns the third axis.

4.3. Quadrilateral detection

A strategy is detailed here for detecting quadrilateral-like shapes and for explaining 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 data sets on which the experiments were performed makes the qualitative comparison difficult, as in the original work only a single scenario is analyzed while in this paper more than 10 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[2] (*Point Cloud Library*) project. For visualizing the boundaries take a look at Fig. 3(b), 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.

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. For each iteration, a new line segment is checked and if it satisfies certain conditions, it is added to a quadrilateral configuration as described in Algorithm 3. 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 is 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

[2] http://pointclouds.org/.

to be analyzed. For a better understanding consider Algorithm 4. 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 – see Fig. 3 – 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 are 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 Fig. 3(c), by using a pass through filter along the *Z*-axis.

4.4. Detection of furniture

Our method is based not only on similar concept proposed in Tuttas and Stilla (2011) but also incorporates constrains on the position and size of the detected windows. Although windows and pictures may have the same silhouette, the way that they lay in the supporting plane (i.e. the windows are embedded in the plane, while the pictures usually tumble out from the plane) makes them distinguishable.

The number of furniture classified in the work (Rusu et al., 2008) is lower than that in the current work, the focus being in a statistical analysis of the proposed algorithms. The limitation of the current approach is 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 and 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 Fig. 4.

After extracting the primary planes, they are classified into floor and ceiling 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 Algorithm 5. 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

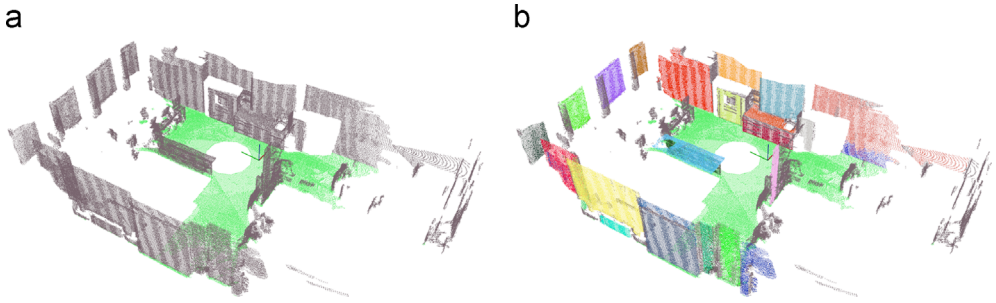

Fig. 4. Segmentation result of human indoor environment, visualized without ceiling: (a) 3D data set with highlighted floor; (b) detection of vertical and horizontal surfaces.

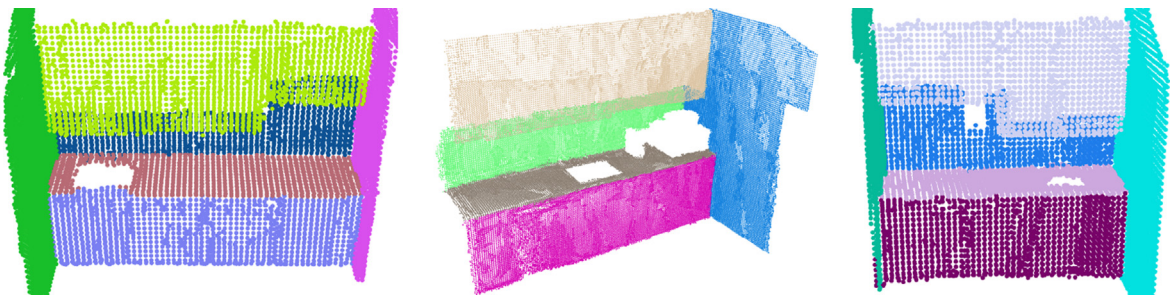

Fig. 5. Detection of important furniture pieces, inside kitchen areas.

Table 4
Detection results presented by individual labels.

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

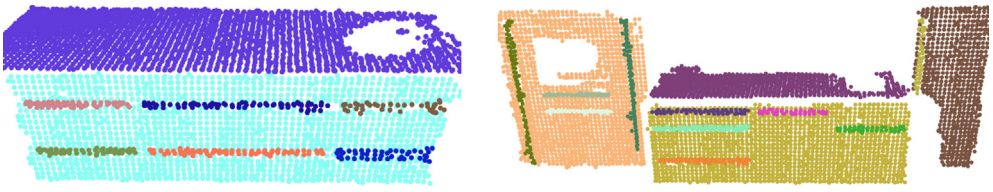

Fig. 6. Obtaining the handles of furniture, which can be used in grasping applications.

cloud for speed considerations and to simplify the computation of patch 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 the fixtures in typical kitchen environments. The results of this process can be visualized in Fig. 5.

4.5. Experimental results

The proposed system was tested on several point clouds, recorded inside various office and kitchen environments. Table 4 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 remaining 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 4. Detection of indoor components was achieved by identifying planar patches and line segments in the above-mentioned data sets. The floors and ceilings 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 to their physical sizes (Fig. 6).

5. Conclusions

This work covers the necessary steps for a 3D environment perception in different environments for automatic reasoning in case of autonomous agents. As the first step of the perception, the data acquisition and preprocessing part is presented including the scan registration. For the registration algorithms a robust keypoint feature based variant was proposed which proved to provide reliable results for sparse and cluttered environments.

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*. Also described, is a technique for estimating the principal axes while dealing with 3D indoor data sets. From experimental observations, the presented approach is relatively robust to noise and easy to compute. The method was tested on different data sets with promising results.

This system can be useful in different areas of the robotic research. One would be the interpretation of registered clouds taken from human indoor environments. Also, it can be useful to know where basic components, e.g. doors or windows, are situated for certain environment mapping algorithms.

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. heating radiators and furniture pieces such as round windows and paintings. And for long-term the perception can be improved by incorporating vision into the process, thus contributing to the overall robustness. Using the Hough transform instead of the RANSAC algorithm is also taken into consideration. Despite its lack of randomness, using the Hough transform ensures

the best possible fitted shapes, each time the labeling pipeline is executed.

Acknowledgments

This paper was supported by the project "Development and support of multidisciplinary postdoctoral programs in major technical areas of national strategy of Research – Development – Innovation" 4D-POSTDOC, contract no. POSDRU/89/1.5/S/52603, project co-funded by the European Social Fund through Sectoral Operational Program Human Resources Development 2007–2013, and by the German Academic Exchange Service (or DAAD – Deutscher Akademischer Austauschdienst) and by SCIEX NMS-CH project number 12.239.

Appendix A. Pseudo-code algorithms

Algorithm 1. ICP with initial alignment.

Input: P_s, P_t
1: F_s=ComputeFPFH (P_s);
2: F_t=ComputeFPFH (P_t);
3: (t^*, A_f) = InitialAlignment (F_s, F_t);
4: **while** $(error_{diff} < \epsilon) \vee (iter < iter_{max})$ **do**
5: A_d=GetClosestPoints (t^*, P_s, P_t);
6: $t^* = \arg\min\left(\frac{1}{|A_d|}\sum_{j \in A_d} w_j |t(p_s) - p_t|^2\right)$;
7: **end while**
Output: t^*

Algorithm 2. Estimating principal axes.

Input: *cloud* // Data of complete room.
Input: $axes = \emptyset$ // Empty set of vectors.

1: $bool \leftarrow 0, bool_1 \leftarrow 0, bool_2 \leftarrow 0, bool_3 \leftarrow 0$ // Set boolean variables.
2: **repeat** //
3: $[plane, inliers] = detectPlaneInCloud(cloud)$ // Fit plane to data set.
4: **if** $sizeOf(inliers) \geq minPlaneInliers$ **then** // Check number of inliers.
5: **if** $bool_1 = 0$ **then** // Most significant plane.
6: $plane_1 \leftarrow plane$ // Keep detected plane.
7: $axes \leftarrow normal_1$ // Add plane normal to axes.
8: $bool_1 \leftarrow 1$ // Update bool variable.
9: **else** //
10: **if** $bool_2 = 0$ **then** //
11: **if** $plane_1 \perp plane$ **then** // Check if perpendicular.
12: $plane_2 \leftarrow plane$ //
13: $axes \leftarrow normal_2$ //
14: $bool_2 \leftarrow 1$ //
15: **end if** //
16: **else** //
17: **if** $bool_3 = 0$ **then** //
18: **if** $plane_2 \perp plane$ **then** //
19: $plane_3 \leftarrow plane$ //
20: $axes \leftarrow normal_3$ //
21: $bool_3 \leftarrow 1$ //
22: **end if** //
23: **end if** //
24: **end if** //
25: **end if** //
26: $extractInliersFromCloud(cloud, inliers)$ // Remove inliers from data.
27: **end if** //
28: **if** $(bool_1 = 0) \wedge (bool_2 = 0) \wedge (bool_3 = 0)$ **then** // Three axes found,
29: $bool \leftarrow 0$ // stop fitting planes.
30: **end if** //
31: **until** $(sizeOf(cloud) \geq minPlaneInliers) \wedge (bool = 1)$ // Stopping conditions.

Output: $axes = \{normal_1, normal_2, normal_3\}$ // Initial guess of axes.

Algorithm 3. Fitting lines to boundaries of planes.

Input: $Planes = \{pp_1, pp_2, \ldots, pp_m\}$ // Detected plane inliers.
Input: $Lines = \emptyset$ // Empty set of lines, and
Input: $Inliers = \emptyset$ // corresponding inliers.

1: **for** k **to** $sizeOf(Planes)$ **do** //
2: $bp_k = getBoundaryPoints(pp_k)$ // Get boundaries of plane.
3: **repeat** //
4: $[lm_k, ip_k] = fitLineToCloud(bp_k)$ // Fit line to boundaries.
5: $Clusters = detectClustersFromPoints(ip_k)$ // Get clusters of inliers.
6: $cp_k = passLargestCluster(Clusters)$ // The biggest cluster.
7: **if** $sizeOf(cp_k) \geq minSizeOfInliers$ **then** // Check number of inliers.
8: $lm_k \rightarrow Lines$ // Add found line to set,
9: $ip_k \rightarrow Inliers$ // save its inliers also.
10: $extractInliersFromCloud(bp_k, cp_k)$ // Remove inliers from data.
11: **end if** //
12: **until** $sizeOf(bp_k) \geq minSizeOfInliers$ // Stopping conditions.
13: **end for** //

Output: $Lines = \{lm_1, lm_2, \ldots, lm_n\}$ // Set of line models,
Output: $Inliers = \{ip_1, ip_2, \ldots, ip_n\}$ // and their inlier points.

pp – points of planes; `which were detected along the axes`
bp – boundary points; `found on the edges of planes`
lm – line models; `fitted to the boundary points of planes`
ip – inlier points; `of the previously found line models`
cp – cluster points; `extracted from each of the line's inliers`

Algorithm 4. Estimating quadrilateral-like shapes.

Input: n // Number of fitted lines.
Input: $H = \{line_1, line_2, \ldots, line_n\}$ // Set of line models.
Input: $G = \emptyset$ // Empty quadrilateral set.

1: **repeat** //
2: $line = H(1)$ // Start with first line.
3: $quad = \emptyset$ // Initialize empty quad.
4: **repeat** //
5: **if** $quad = = \emptyset$ **then** // First edge of quad.
6: add *line* to *quad* // Add line to quad.
7: delete *line* from H // Remove line from set.
8: update n // Decrease number of lines.
9: $line = H(1)$ // Restart with first line.
10: **else** //
11: **if** $condition(a) \wedge condition(b)$ **then** // Check the two conditions,
12: add *line* to *quad* // mentioned above.
13: delete *line* from H //
14: update n //
15: $line = H(1)$ //
16: **else** //
17: $line = H(indexOf(line)+1)$ // Continue with next line.
18: **end if** //
19: **end if** //
20: **until** $(indexOf(line) \leq n) \wedge (sizeOf(quad) < 4)$ // Stopping conditions.
21: **if** $sizeOf(quad) > 2$ **then** // More than two lines,
22: add *quad* to G // add quadrilateral to set.
23: **end if** //
24: **until** $H \neq \emptyset$ // No more lines in set.

Output: $G = \{quad_1, quad_2, \ldots, quad_m\}$ // Detected quadrilaterals.

Algorithm 5. Detecting furniture surfaces and fixtures.

Input: *cloud* // Input data set.
Input: $axes = \{X, Y, Z\}$ // Orthogonal axes of room.
Input: $surfaces = \emptyset$ // Empty set of surfaces,

Input: $handles = \emptyset$

FURNITURE SURFACES

for i **to** $sizeOf(axes)$ **do**
  repeat
    $[plane\_model, plane\_inliers] = detectPlaneAlongAxis(cloud, axes[i])$
    if $sizeOf(plane\_inliers) \geq minPlaneInliers$ **then**
      $extractPointsFromCloud(cloud, plane\_inliers)$
      $plane\_clusters = getClustersFromPoints(plane\_inliers)$
      for j **to** $sizeOf(plane\_clusters)$ **do**
        if $sizeOf(plane\_clusters[j]) \geq minPlaneCluster$ **then**
          $surfaces \leftarrow plane\_clusters[j]$
        end if
      end for
    end if
  until $sizeOf(cloud) \geq minPlaneInliers$
end for

FURNITURE FIXTURES

for i **to** $sizeOf(surfaces)$ **do**
  if $(surfaces[i] \perp floor) \wedge (heightOf(surfaces[i]) > 2\ [\mathrm{m}])$ **then**
    $points = getPointsOnSurface(surfaces[i])$
    $fixtures = getClustersFromPoints(points)$
    for j **to** $sizeOf(fixtures)$ **do**
      repeat
        $[line\_model, line\_inliers] = fitLineToCloud(fixtures[j])$
        if $(sizeOf(line\_inliers) \geq minLineInliers)$ **then**
          $extractPointsFromCloud(fixtures[j], line\_inliers)$
          $line\_clusters = getClustersFromPoints(line\_inliers)$
          for k **to** $sizeOf(line\_clusters)$ **do**
            if $sizeOf(line\_clusters[k]) \geq minLineCluster$ **then**
              $handles \leftarrow line\_clusters[k]$
            end if
          end for
        end if
      until $sizeOf(fixtures[j]) \geq minLineInliers$
    end for
  end if
end for

Output: $surfaces = \{surface_1, \ldots, surface_n\}$
// Set of detected surfaces,
Output: $handles = \{handle_1, \ldots, handle_m\}$
// and estimated handles.

// and furniture fixtures.

References

Bay, H., Ess, A., Tuytelaars, T., Van Gool, L., 2008. Speeded-up robust features (SURF). Comput. Vis. Image Underst. 110 (3), 346–359.

Bentley, J.L., 1975. Multidimensional binary search trees used for associative searching. Commun. ACM 18 (9), 509–517.

Besl, P.J., McKay, H.D., 1992. A method for registration of 3-D shapes. IEEE Trans. Pattern Anal. Mach. Intell. 14 (2), 239–256.

Censi, A., 2007. An accurate closed-form estimate of ICP's covariance. In: 2007 IEEE International Conference on Robotics and Automation. Roma, Italy, April 10–14, pp. 3167–3172.

Elseberg, J., Borrmann, D., Nüchter, A., 2012. 6dof semi-rigid slam for mobile scanning. In: IEEE/RSJ International Conference on Intelligent Robots and Systems. Vilamoura, Algarve, October 7–12, pp. 1865–1870.

Fischler, M.A., Bolles, R.C., 1981. Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography. Commun. ACM 24 (6), 381–395.

Goron, L.C., Tamas, L., Reti, I., Lazea, G., 2010. 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.

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, pp. 5291–5297.

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: Proceedings of the International Conference on Image Processing. Orlando, FL, USA.

Kasinski, A., Skrzypczynski, P., 2001. Perception network for the team of indoor mobile robots: concept, architecture, implementation. Eng. Appl. Artif. Intell. 14 (2), 125–137.

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, pp. 4148–4153.

Khoshelham, K., Elberink, S.O., 2012. Accuracy and resolution of kinect depth data for indoor mapping applications. Sensors 12 (2), 1437–1454.

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 (4), 387–407.

Lowe, D.G., 2004. Distinctive image features from scale-invariant keypoints. Int. J. Comput. Vis. 60, 91–110.

Magnusson, M., Duckett, T., Lilienthal, A.J., 2007. Scan registration for autonomous mining vehicles using 3D-NDT. J. Field Robot. 24 (10), 803–827.

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, IEEE, pp. 3907–3912.

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), Taipei, Taiwan, pp. 3700–3705.

May, S., Droeschel, D., Fuchs, S., Holz, D., Nüchter, A., 2009. Robust 3d-mapping with time-of-flight cameras. In: 2009 IEEE/RSJ International Conference on Intelligent Robots and Systems. St. Louis, MO, USA, October 11–15, pp. 1673–1678.

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), pp. 3786–3793.

Murillo, A.C., Košecká, J., Guerrero, J.J., Sagüés, C., 2008. Visual door detection integrating appearance and shape cues. Robot. Auton. Syst. 56, 512–521.

Nagatani, K., Matsuzawa, T., Yoshida, K., 2009. Scan-point planning and 3-d map building for a 3-d laser range scanner in an out door environment. In: Howard, A., Iagnemma,K., Kelly, A. (Eds.), Field and Service Robotics, Results of the 7th International Conference, FSR 2009, Cambridge, Massachusetts, USA, 14–16, July 2009, Springer Tracts in Advanced Robotics, vol. 62. Springer, pp. 207–217.

Nüchter, A., Hertzberg, J., 2008. Towards semantic maps for mobile robots. Robot. Auton. Syst. 56, 915–926.

Nuechter, A., 2009. 3D Robotic Mapping, Springer Tracts in Advanced Robotics (STAR). Springer, Berlin Heidelberg.

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), Anchorage, AK, USA, pp. 2864–2870.

Papadakis, P., 2013. Terrain traversability analysis methods for unmanned ground vehicles: a survey. Eng. Appl. Artif. Intell. 26 (4), 1373–1385.

Pathak, K., Birk, A., Vaskevicius, N., Poppinga, J., 2010. Fast registration based on noisy planes with unknown correspondences for 3D mapping. IEEE Trans. Robot. 26, 424–441.

Rusu, R.B., 2009. Semantic 3D Object Maps for Everyday Manipulation in Human Living Environments (Ph.D. thesis).

Rusu, R.B., Cousins, S., 2011. 3D is here: Point Cloud Library (PCL). In: Proceedings of the IEEE International Conference on Robotics and Automation (ICRA), Shanghai, China.

Rusu, R.B., Marton, Z.-C., Blodow, N., Dolha, M., Beetz, M., 2008. Towards 3D point cloud based object maps for household environments. Robot. Auton. Syst. 56, 927–941.

Rusu, R.B., Bradski, G., Thibaux, R., Hsu, J., 2010. 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.

Salvi, J., Matabosch, C., Fofi, D., Forest, J., 2007. A review of recent range image registration methods with accuracy evaluation. Image Vis. Comput. 25, 578–596.

Scharstein, D., Szeliski, R., 2002. A taxonomy and evaluation of dense two-frame stereo correspondence algorithms. Int. J. Comput. Vis. 47 (1–3), 7–42.

Sharp, G.C., Lee, S.W., Wehe, D.K., 2002. ICP registration using invariant features. IEEE Trans. Pattern Anal. Mach. Intell. 24 (1), 90–102.

Silva, L.C.D., Morikawa, C., Petra, I.M., 2012. State of the art of smart homes. Eng. Appl. Artif. Intell. 25 (7), 1313–1321.

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.

Steder, B., Rusu, R.B., Konolige, K., 2011. Point feature extraction on 3D range scans taking into account object boundaries. In: Proceeding of the IEEE International Conference on Robotics and Automation. ICRA 2011, Shanghai, China, May 9–13, pp. 2601–2608.

Strasdat, H., 2012. Local Accuracy and Global Consistency for Efficient Visual SLAM (Ph.D. thesis).

Surmann, H., Lingemann, K., Nuchter, A., Hertzberg, J., 2001. A 3D Laser Range Finder for Autonomous Mobile Robots, Seoul, South Korea, pp. 153–158.

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.

Tamas, L., Majdik, A., 2012. Heterogeneous feature based correspondence estimation. In: Proceedings of the IEEE International Conference on Multisensor Fusion and Integration for Intelligent Systems (MFI), Hamburg, Germany, pp. 89–94.

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.

Zhang, Z., 1992. Iterative Point Matching for Registration of Free-form Curves. Technical Report No. RR-1658.

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, pp. 409–415.