A Benchmark Survey of Rigid 3D Point Cloud Registration Algorithms

Ben Bellekens*, Vincent Spruyt*, Rafael Berkvens*, Rudi Penne†, and Maarten Weyn*
*CoSys-Lab, Faculty of Applied Engineering
University of Antwerp, Belgium
Email: {ben.bellekens, rafael.berkvens, maarten.weyn}@uantwerpen.be and v.spruyt@ieee.org
†Op3Mech, Faculty of Applied Engineering, Dept. of Mathematics
University of Antwerp, Belgium
Email: rudi.penne@uantwerpen.be

***Abstract*—Advanced user interface sensors are able to observe the environment in three dimensions with the use of specific optical techniques such as time-of-flight, structured light or stereo vision. Due to the success of modern sensors, which are able to fuse depth and color information of the environment, a new focus on different domains appears. This survey studies different state-of-the-art registration algorithms, which are able to determine the motion between two corresponding 3D point clouds. This survey starts from a mathematical field of view by explaining two deterministic methods, namely Principle Component Analysis (PCA) and Singular Value Decomposition (SVD), towards more iteratively methods such as Iterative Closest Point (ICP) and its variants. We compare the performance of the different algorithms to their precision and robustness based on a real world dataset. The main contribution of this survey consists of the performance benchmark that is based on a real world dataset, which includes 3D point clouds of a Microsoft Kinect camera, and a mathematical overview of different registration methods, which are commonly used in applications such as simultaneous localization and mapping, and 3D-scanning. The outcome of our benchmark concludes that the ICP point-to-surface method is the most precise algorithm. Beside the precision, the result for the robustness we can conclude that a combination of applying a ICP point-to-point method after an SVD method gives the minimum error.**

Keywords–3D point cloud; PCL; Kinect camera; 3D Fine registration; rigid transformation; survey paper; Robustness; Precision; SLAM

I. INTRODUCTION

This article, which is an extended version of the conference paper [1], contains new results that defines the robustness and the precision of the different registration algorithms.

With the advent of inexpensive depth sensing devices, robotics, computer vision and ambient application technology research has shifted from 2D imaging and Laser Imaging Detection And Ranging (LIDAR) scanning towards real-time reconstruction of the environment based on 3D point cloud data. On the one hand, there are structured light based sensors such as the Microsoft Kinect and Asus Xtion sensor, which generate a structured point cloud, sampled on a regular grid, and on the other hand, there are many time-of-flight based sensors such as the Softkinetic Depthsense camera, which yield an unstructured point cloud. These point clouds can either be used directly to detect and recognize objects in the environment where ambient technology is been used, or can be integrated over time to completely reconstruct a 3D map of the camera's surroundings [2], [3], [4]. However, in the latter case, point clouds obtained at different time instances need to be aligned, a process that is often referred to as registration. Registration algorithms are able to estimate the ego-motion of a robot by calculating the transformation that optimally maps two point clouds, each of which is subject to camera noise.

Registration algorithms can be classified coarsely into rigid and non-rigid approaches. Rigid approaches assume a fixed rigid environment such that a homogeneous transformation can be modelled using only 6 Degrees Of Freedom (DOF). On the other hand, non-rigid methods are able to cope with articulated objects or soft bodies that change shape over time. Additionally, registration algorithms can be classified into coarse and fine approaches. Coarse registration approaches compute an initial geometric alignment whereas fine registration approaches compute a transformation that can register two point clouds precisely. A combination of coarse and fine registration algorithms is often used in applications to reduce the number of iterations while an optimal alignment still occurs.

Registration algorithms are used in different fields and applications, such as 3D object scanning, 3D mapping, 3D localization and ego-motion estimation or human body detection. Most of these state-of-the-art applications employ either a simple Singular Value Decomposition (SVD) [5] or Principal Component Analysis (PCA) based registration, or use a more advanced iterative scheme based on the Iterative Closest Point (ICP) algorithm [6]. Recently, many variants on the original ICP approach have been proposed, the most important of which are non-linear ICP [7], and generalized ICP [8]. These are explained and discussed in this publication.

To our knowledge, a general discussion of each of the above methods that are applied in a real world scenario where environment data is been acquired with a 3D sensor is not available in literature. Salvi *et al.* presented a survey article, which gives an overall view of coarse and fine registration methods that are able to register range based images [9]. But they presented a performance comparison based on synthetic data and real data that was recorded by a laser scanner.

The choice of an algorithm generally depends on several important characteristics such as accuracy, computational complexity, and convergence rate, each of which depends on the application of interest. Moreover, the characteristics of most registration algorithms heavily depend on the data used, and thus on the environment itself. As a result, it is difficult to compare these algorithms data independently. Therefore, in this paper we discuss the mathematical foundations that are common to the most widely used 3D registration algorithms, and we compare their robustness and precision in a real world situations.

This paper is outlined as follows: Section II briefly discusses several important application domains of 3D registration algorithms. In Section III, rigid registration is formulated as a least square optimization problem. Section IV explains the most important rigid registrations algorithms, which are PCA, SVD, ICP point-to-point, ICP point-to-surface, ICP non-linear and Generalized ICP. Finally, Section V provides a discussion of the precision and the robustness of each of these methods in a real world setting. Section VI concludes the paper.

II. Application Domains

Important application domains of both rigid and non-rigid registration methodologies are robotics, healthcare, astrophotography, and more. In these applications, the common goal is to determine the position or pose of an object with respect to a given viewpoint. Whereas rigid transformations are defined by 6 DOF, non-rigid transformations allow a higher number of DOF in order to cope with non-linear or partial stretching or shrinking of the object [10]. Following subsections will give an overview of the robotic applications and healthcare applications where 3D rigid registration methods are being applied.

A. Robotics

Since the introduction of inexpensive depth sensors such as the Microsoft Kinect camera, great progress has been made in the robotic domain towards Simultaneous Localization And Mapping (SLAM) [11], [12], [13], [14]. The reconstructed 3D occupancy grid map is represented by a set of point clouds, which are aligned by means of registration and can be used for techniques such as obstacle avoidance, map exploration and autonomous vehicle control [4], [15], [16]. Furthermore, depth information is often combined with a traditional RGB camera [3], [17] in order to greatly facilitate real-world problems such as object detection in cluttered scenes, object tracking and object recognition [18]. The main goal in robotic applications is to develop a robust, precise and accurate algorithm that can execute almost at real-time. In order to reach this goal much research is nowadays focusing toward graphical processing unit (GPU) and multicore processing, which enables the execution of many computation task during one timeslot on multiple processing cores [19], [20].

B. Healthcare

Typical applications of non-rigid registration algorithms can be found in healthcare, where a soft-body model often needs to be aligned accurately with a set of 3D measurements. Applications are cancer-tissue detections, hole detection, artefact recognition, etc. [10], [21]. Similarly, non-rigid transformations are used to obtain a multi-modal representation of a scene, by combining magnetic resonance imaging (MRI), computer tomography (CT), and positron emission tomography PET volumes into a single 3D model [10].

III. Definitions

In this section, we briefly introduce the least-squares optimization problem and discuss the concept of homogeneous transformations since these form the basis of 3D registration algorithms.

Rigid registration can be approached by defining a cost function that represents the current error, which indicates how well two point clouds overlap. This cost function is then minimized using common optimization techniques. If the distance between corresponding points in each 3D point cloud needs to be minimized, this can be simplified to a linear least-squares minimization problem by representing each point using homogeneous coordinates.

A. Homogeneous transformations

A homogeneous transformation in three dimensions is specified by a 4×4 projective transformation matrix [22]. This matrix is used to project each point in Cartesian space with respect to a specific viewpoint. Since we use (moving) rigid orthonormal reference frames, we can restrict our considerations to rigid transformations. In the following, let $\tilde{\mathbf{v_1}} = (x_1, y_1, z_1, 1)^\intercal$ be standard homogeneous coordinates of a point in an orthonormal base defined by viewpoint one, and let $\tilde{\mathbf{v_2}} = (x_2, y_2, z_2, 1)^\intercal$ be standard homogeneous coordinates of the same point in an orthonormal base defined by viewpoint two. Then it is possible to express $\tilde{\mathbf{v_2}}$ relative to the base of viewpoint one as $T\tilde{\mathbf{v_1}} = \tilde{\mathbf{v_2}}$, where T is a Euclidean transformation matrix defined by (1).

$$T = \begin{pmatrix} r_{1,1} & r_{1,2} & r_{1,3} & t_1 \\ r_{2,1} & r_{2,2} & r_{2,3} & t_2 \\ r_{3,1} & r_{3,2} & r_{3,3} & t_3 \\ 0 & 0 & 0 & 1 \end{pmatrix} \tag{1}$$

The transformation matrix shown by (1) consists of a 3×3 rotation matrix (2),

$$R = \begin{pmatrix} r_{1,1} & r_{1,2} & r_{1,3} \\ r_{2,1} & r_{2,2} & r_{2,3} \\ r_{3,1} & r_{3,2} & r_{3,3} \end{pmatrix} \tag{2}$$

and the column vector $\vec{t} = (\vec{t_1}, \vec{t_2}, \vec{t_3})^\intercal$ representing a translation. Because the nine entries of the rotation matrix can be generated by three parameters (e.g., the Euler angles), we conclude that a rigid transformation has six DOF.

B. Least-Squares Minimization

A rigid transformation is defined by only 6 DOF, whereas many noisy observations, i.e., point coordinates, are available. Therefore, the number of parameters of any cost function for this problem is much smaller than the number of equations, resulting in an ill-posed problem that does not have an exact solution. A well known technique to obtain an acceptable solution in such case, is to minimize the square of the residual error. This approach is called least-squares optimization and is often used for fitting and regression problems.

Whereas a linear least-squares problem can be solved analytically, this is often not the case for non-linear least-squares optimization problems. In this case, an iterative approach can be used by iteratively exploring the search space of all possible solutions in the direction of the gradient vector of the cost function. This is illustrated by Figure 1, where the cost function $f(d)$ of the ICP registration algorithm is minimized iteratively. The cost function in this case represents the sum of the squared Euclidean distances, defined by the rotation and the translation, between all corresponding points of two point cloud viewpoints.

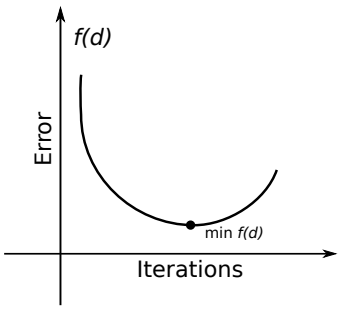

Figure 1. ICP Least square approach.

IV. Registration Algorithms

Both rigid and non-rigid registration algorithms can be further categorized into pairwise registration algorithms and multiview registration methods. Pairwise registration algorithms calculate a rigid transformation between two subsequent point clouds while the multi-view registration process takes multiple point clouds into account to correct for the accumulated drift that is introduced by pairwise registration methods.

In the next sections, we discuss five widely used rigid registration algorithms. Each of these methods tries to estimate the optimal rigid transformation that maps a source point cloud on a target point cloud. Both PCA alignment and SVD are pairwise registration methods based on the covariance matrices and the cross correlation matrix of the point clouds, while the ICP algorithm and its variants are based on iteratively minimizing a cost function that is based on an estimate of point correspondences between the point clouds. The selected correspondences will determine the quality of how the final transformation fits the source point cloud to the target point cloud.

A. Principal Component Analysis

PCA is often used in classification and compression techniques to project data on a new orthonormal basis in the direction of the largest variance [23]. The direction of the largest variance corresponds to the largest eigenvector of the covariance matrix of the data, whereas the magnitude of this variance is defined by the corresponding eigenvalue.

Therefore, if the covariance matrix of two point clouds differs from the identity matrix, a rough registration can be obtained by simply aligning the eigenvectors of their covariance matrices. This alignment is obtained as follows.

First, the two point clouds are centered such that the origins of their original bases coincide. Point cloud centering simply corresponds to subtracting the centroid coordinates from each of the point coordinates. The centroid of the point cloud corresponds to the average coordinate and is thus obtained by dividing the sum of all point-coordinates by the number of points in the point cloud.

Since registration based on PCA simply aligns the directions in which the point clouds vary the most, the second step consists of calculating the covariance matrix of each point cloud. The covariance matrix is an orthogonal 3×3 matrix, the diagonal values of which represent the variances while the off-diagonal values represent the covariances.

Third, the eigenvectors of both covariance matrices are calculated. The largest eigenvector is a vector in the direction of the largest variance of the 3D point cloud and, therefore, it represents the point cloud's orientation. In the following, let A be the covariance matrix, let $\vec{v}$ be an eigenvector of this matrix, and let λ be the corresponding eigenvalue. The eigenvalues decomposition problem is then defined as:

$$A\tilde{\mathbf{x}} = \lambda\tilde{\mathbf{x}} \tag{3}$$

and further reduces to:

$$\tilde{\mathbf{x}}(A - \lambda I) = 0. \tag{4}$$

It is clear that (4) only has a non-zero solution if $A - \lambda I$ is singular and, consequently, if its determinant equals zero:

$$\det(A - \lambda I) = 0 \tag{5}$$

The eigenvalues can simply be obtained by solving (5), whereas the corresponding eigenvectors are obtained by substituting the eigenvalues into (3).

Once the eigenvectors are known for each point cloud, registration is achieved by aligning these vectors. In the following, let matrix T_t^y represent the transformation that would align the largest eigenvector t of the target point cloud with the y-axis. Let matrix T_y^s represent the transformation that would align the largest eigenvector s of the source point cloud with the y-axis. Then the final transformation matrix T_t^s that aligns the source point cloud with the target point cloud can be obtained easily, as illustrated by Figure 2.

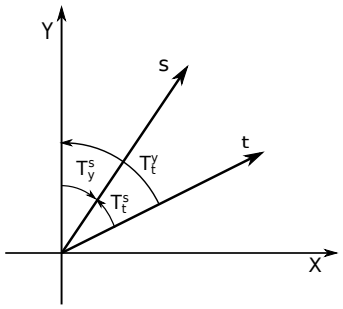

Figure 2. PCA alignment from source to target.

Finally, the centroid of the target data is added to each of the transformed coordinates to translate the aligned point cloud, such that its center corresponds to the center of the target point cloud.

B. Singular Value Decomposition

PCA based registration simply aligns the directions of the largest variance of each point cloud and, therefore, it does not minimize the Euclidean distance between corresponding points of the datasets. Consequently, this approach is very sensitive to outliers and only works well if each point cloud is approximately normally distributed.

However, if point correspondences between the two point clouds are available, a more robust approach would be to directly minimize the sum of the Euclidean distances between these points. This corresponds to a linear least-squares problem that can be solved robustly using the SVD method [5].

Based on the point correspondences, the cross correlation matrix M between the two centered point clouds can be calculated, after which the eigenvalue decomposition is obtained as follows:

$$M = USV^{\intercal} \tag{6}$$

The optimal solution to the least-squares problem is then defined by rotation matrix R as:

$$R_t^s = UV^{\intercal} \tag{7}$$

and the translation from target point cloud to source point cloud is defined by:

$$\tilde{\mathbf{t}} = \tilde{\mathbf{c}_\mathbf{s}} - R_t^s \tilde{\mathbf{c}_\mathbf{t}} \tag{8}$$

C. Iterative Closest Point

Whereas the SVD algorithm directly solves the least-squares problem, thereby assuming perfect data, Besl and Mc. Kay [6] introduced a method that iteratively disregards outliers in order to improve upon the previous estimate of the rotation and translation parameters. Their method is called 'ICP' and is illustrated conceptually in Figure 3.

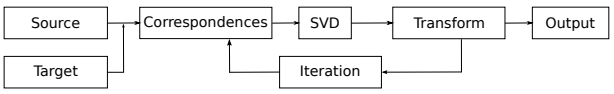

Figure 3. ICP overview scheme.

The input of the ICP algorithm consists of a source point cloud and a target point cloud. Point correspondences between these point clouds are defined based on a nearest neighbour approach or a more elaborate scheme using geometrical features or color information. SVD, as explained in the previous section, is used to obtain an initial estimate of the affine transformation matrix that aligns both point clouds. After transformation, this whole process is repeated by removing outliers and redefining the point correspondences.

Two widely used ICP variants are the ICP point-to-point and the ICP point-to-surface algorithms. These approaches only differ in their definition of point correspondences and are described in more detail in the next sections.

1) ICP point-to-point: The ICP point-to-point algorithm was originally described in [2] and simply obtains point correspondences by searching for the nearest neighbour target point $\vec{q_i}$ of a point $\vec{p_j}$ in the source point cloud. The nearest neighbour matching is defined in terms of the Euclidean distance metric:

$$\hat{i} = \arg\min_i \|\vec{p_i} - \vec{q_j}\|^2, \tag{9}$$

where $i \in [0, 1, ..., N]$, and N represents the number of points in the target point cloud.

Similar to the SVD approach discussed in Section IV-B, the rotation R and translation $\vec{t}$ parameters are estimated by minimizing the squared distance between these corresponding pairs:

$$\hat{R}, \hat{\vec{t}} = \arg\min_{R,\vec{t}} \sum_{i=1}^{N} \|(R\vec{p_i} + \vec{t}) - \vec{q_i}\|^2 \tag{10}$$

ICP then iteratively solves (9) and (10) to improve upon the estimates of the previous iterations. This is illustrated by Figure 4, where surface s is aligned to surface t after n ICP iterations.

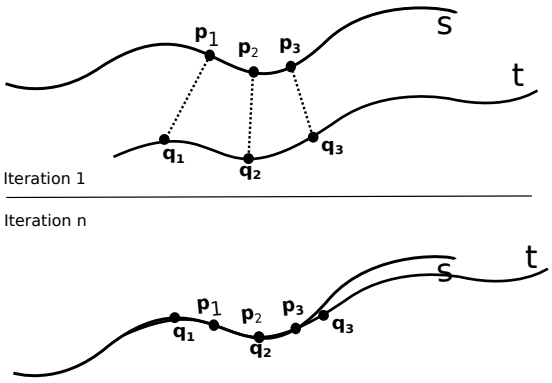

Figure 4. ICP alignment based on a point to point approach.

2) ICP point-to-surface: Due to the simplistic definition of point correspondences, the ICP point-to-point algorithm proposed by [24] is rather sensitive to outliers. Instead of directly finding the nearest neighbour to a source point $\vec{p_j}$ in the target point cloud, one could take the local neighbourhood of a correspondence candidate $\vec{q_i}$ into account to reduce the algorithm's sensitivity to noise.

The ICP point-to-surface algorithm assumes that the local neighbourhood of a point in a point cloud is co-planar. This local surface can then be defined by its normal vector $\vec{n}$, which is obtained as the smallest eigenvector of the covariance matrix of the points that surround correspondence candidate $\vec{q_i}$.

Instead of directly minimizing the Euclidean distance between corresponding points, we can then minimize the scalar projection of this distance onto the planar surface defined by the normal vector $\vec{n}$:

$$\hat{R}, \hat{\vec{t}} = \arg\min_{\hat{R},\hat{\vec{t}}} \left(\sum_{i=1}^{N} \|((R\vec{p_i} + \vec{t}) - \vec{q_i})\vec{n_i}\| \right) \tag{11}$$

This is illustrated more clearly in Figure 5.

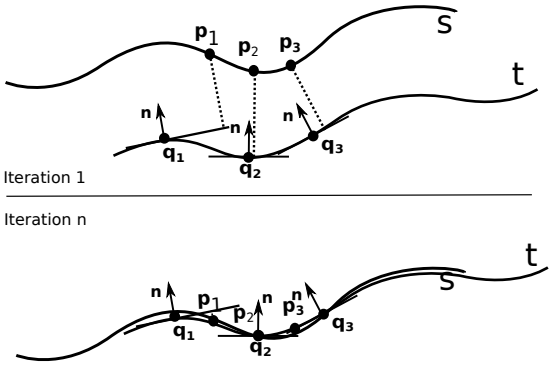

Figure 5. ICP alignment based on a point to surface approach.

3) ICP non-linear: Both the point-to-point and point-to-surface ICP approaches defined a differentiable, convex, squared cost function, resulting in a simple linear least-squares optimization problem, known as a L2-optimization, that can be solved numerically using SVD. However, L2-optimization is known to be highly sensitive to outliers because the residuals are squared. An approach that solves this problem is known as L1-optimization, where the sum of the absolute value of the residuals is minimized instead of the square. However, the L1 cost function is non-differentiable at the origin, which makes it difficult to obtain the optimal solution.

As a compromise between L1 and L2 optimization, the so called Huber loss function can be used as shown by (12). The Huber loss function is quadratic for small values and thus behaves like an L2 problem in these cases. For large values, however, the loss function becomes linear and, therefore, it behaves like an L1 cost function. As Figure 6 shows differentiation between the Huber-Loss function by the green curve, the blue curve shows the L2 quadratic function. Moreover, the Huber loss function is smooth and differentiable, allowing traditional numerical optimization methods to be used to efficiently traverse the search space.

$$e(n) = \begin{cases} n^2/2 & \text{if } |n| \leq k \\ k|n| - k^2/2 & \text{if } |n| > k \end{cases} \quad (12)$$

where k is an empirically defined threshold and n is the distance measure.

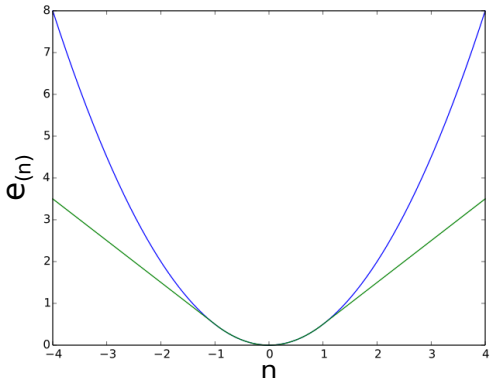

Figure 6. Huber Loss function.

The ICP non-linear algorithm uses the Huber loss function instead of a naive squared loss function to reduce the influence of outliers:

$$\hat{R}, \vec{\hat{t}} = \arg\min_{\hat{R}, \vec{t}} \sum_{i=1}^{N} e^2(n) \quad (13)$$

where

$$n = \|(R\vec{p} - \vec{t}) - \vec{q}\| \quad (14)$$

To obtain the optimal estimates $\hat{R}, \vec{\hat{t}}$ in (13), the Levenberg-Marquardt algorithm (LMA) [7] is used. The LMA method is an iterative procedure similar to the well known gradient descent and Gauss-Newton algorithms, which can quickly find a local minimum in non-linear functions.

4) Generalized ICP: A major disadvantage of the traditional point-to-point ICP algorithm, is that it assumes that the source point cloud is taken from a known geometric surface instead of being obtained through noisy measurements. However, due to discretization errors it is usually impossible to obtain a perfect point-to-point matching even after full convergence of the algorithm. The point-to-surface ICP algorithm relaxes this constraint by allowing point offsets along the surface, in order to cope with discretization differences. However, this approach still assumes that the source point cloud represents a discretized sample set of a known geometric surface model since offsets along the surface are only allowed in the target point cloud.

To solve this, Segal *et al.* [8] proposed the Generalized ICP (GICP) algorithm that performs plane-to-plane matching. They introduced a probabilistic interpretation of the minimization process such that structural information from both the source point cloud and the target point cloud can be incorporated easily in the optimization algorithm. Moreover, they showed that the traditional point-to-point and point-to-surface ICP algorithms are merely special cases of the Generalized ICP framework.

Instead of assuming that the source point cloud is obtained from a known geometric surface, Segal *et al.* assume that both the source point cloud $A = \{\vec{a}_i\}$ and the target point cloud $B = \{\vec{b}_i\}$ consist of random samples from an underlying unknown point cloud $\hat{A} = \{\hat{\vec{a}}_i\}$ and $\hat{B} = \{\hat{\vec{b}}_i\}$. For the underlying and unknown point clouds $\hat{A}$ and $\hat{B}$, perfect correspondences exist, whereas this is not the case for the observed point clouds A and B, since each point $\vec{a}_i$ and $\vec{b}_i$ is assumed to be sampled from a normal distribution such that $\vec{a}_i \sim \mathcal{N}(\hat{\vec{a}}_i, C_i^A)$ and $\vec{b}_i \sim \mathcal{N}(\hat{\vec{b}}_i, C_i^B)$. The covariance matrices C_i^A and C_i^B are unknown. If both point clouds would consist of deterministic samples from known geometric models, then both covariance matrices would be zero such that then $A = \hat{A}$ and $B = \hat{B}$.

In the following, let T be the affine transformation matrix that defines the mapping from $\hat{A}$ to $\hat{B}$ such that $\hat{\vec{b}}_i = T\hat{\vec{a}}_i$. If T would be known, we could apply this transformation to the observed source point cloud A, and define the error to be minimized as $d_i^T = \vec{b}_i - T\vec{a}_i$. Because both $\vec{a}_i$ and $\vec{b}_i$ are assumed to be drawn from independent normal distributions d_i^T, which is a linear combination of $\vec{a}_i$ and $\vec{b}_i$, is also drawn from a normal distribution:

$$d_i^T \sim \mathcal{N}(\hat{\vec{b}}_i - T\hat{\vec{a}}_i, C_i^B + TC_i^A T^\intercal) \quad (15)$$
$$= \mathcal{N}(0, C_i^B + TC_i^A T^\intercal) \quad (16)$$

The optimal transformation matrix $\hat{T}$ is then the transformation that minimizes the negative log-likelihood of the observed errors d_i:

$$\hat{T} = \arg\min_T \sum_i \log\left(p(d_i^T)\right)$$
$$= \arg\min_T \sum_i d_i^{T\intercal} (C_i^B + TC_i^A T^\intercal)^{-1} d_i^T \quad (17)$$

Segal *et al.* showed that both point-to-point and point-to-plane ICP are specific cases of (17), only differing in their choice of covariance matrices C_i^A and C_i^B; If the source point

cloud is assumed to be obtained from a known geometric surface, $C_i^A = 0$. Furthermore, if points in the target point cloud are allowed three degrees of freedom, then $C_i^B = I$. In this case, (18) reduces to:

$$\hat{T} = \arg\min_T \sum_i d_i^{T\intercal} d_i^T = \arg\min_T \sum_i \|d_i^T\|^2, \quad (18)$$

which indeed is exactly the optimization problem that is solved by the traditional point-to-point ICP algorithm. Similarly, C_i^A and C_i^B can be chosen such that obtaining the maximum likelihood estimator corresponds to minimizing the point-to-plane or the plane-to-plane distances between both point clouds.

V. Results & Discussion

In this section, we illustrate the performance of the different registration methods that are based on an iteratively approach. In order to illustrate the performance we tested the precision and the robustness of the different methods. The robustness factor of an algorithm will explain how well an algorithm performs during a period of time on different input parameters. Besides the robustness, the precision factor will clarify how well an algorithm performs on the same input parameter. The results for precision and robustness are based on a set of 3D point clouds that are included in a dataset. All results are generated using the Robot Operating System (ROS) and the Point cloud Library (PCL) [25], [26]. Furthermore, the execution processes of the different methods are calculated by an Asus Zenbook UX32VD, core i7-3517U in combination with 10 GB of RAM-memory.

A. Dataset

The dataset that we used to benchmark the performance is built by a Pioneer-3dx robot and consists of a laser scanner, odometry hardware and 3D point cloud data. The Pioneer-3dx robot is a commonly used robot for academic and research purposes. See Figure 7 for the robot used to build this dataset. To ensure that all sensor measurements have a time-stamp and transformation with respect to the center of the robot, we have used the ROS.

On one hand, ROS is used as a tool to record all sensor measurement including the timestamps, while on the other hand, ROS is used as a platform to schedule the different 3D point clouds based on their timestamps. To reduce the size of the dataset, we decreased the number of point clouds per second. Figure 8 visualizes the dataset by means of an occupancy grid map and a travelled path.

The occupancy grid map is the result of a Rao-Blackwellized particle filter SLAM algorithm with a Bayesian probability distribution [27]. The implementation that we used utilizes the laser range scanner and odometry data to generate an occupancy grid map. However, the location updates are performed by the algorithm are not used to recalculate the travelled path, resulting in a periodically erratic trajectory. To obtain a smooth trajectory, we used the occupancy grid map calculated by the SLAM algorithm to perform adaptive Monte Carlo localization [28]. Because we knew the initial position of the robot, the algorithm did not have to perform global localization, but simply had to track the robot during the complete run. This ensures that location corrections are applied incrementally, resulting in the smooth trajectory. Thus, after the SLAM method has calculated an occupancy grid map, the trajectory was calculated by an Adaptive Monte Carlo Localization approach.

Figure 7. The mobile Pioneer-3dx robot with a mounted Microsoft Kinect Camera, Laser scanner and Sonar sensor.

Figure 8. Occupancy grid map from SLAM approach and the smoothed travelled path

B. Robustness

To measure the robustness of the rigid 3D point cloud registration algorithms, we applied them at various times on different corresponding point clouds and recorded their error and computation time. By averaging over these data points, we obtain information about the robustness of a specific algorithm. We want to compute the robustness of the different rigid registration algorithms so that we can analyze, which algorithm performs best in a real world scenario. We focused on a scenario of mapping an indoor environment to generate a 3D model in which all spatial objects are visible and correctly aligned. We could iterate over all point clouds in our database thanks to the timestamps and playback mechanisms in ROS. Figure 9 shows us a one dimensional axis with vertical marker. Each of these markers represent a 3D point cloud, which was taken at a certain time with respect to the start pose or the beginning of the dataset.

For each set of two point clouds, the fitness score, the averaged and normalized error after registration and alignment between the two point clouds, and the computation time of each algorithm is computed to measure the robustness of the

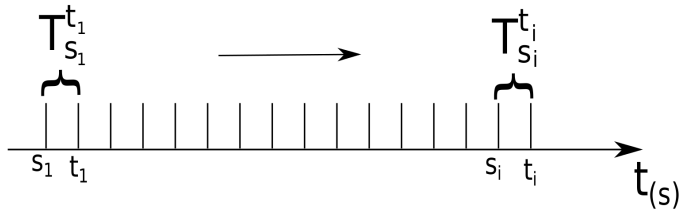

Figure 9. The benchmark robustness scheme includes a set of two 3D point clouds. Each set contains a source point cloud S_i, a target point cloud t_i, and a transformation. Every point cloud is indicated as an individual marker on the time line t.

different algorithms. In this case, there were 165 sets of point cloud pairs or 330 single point clouds.

Figure 10 compares the number of iterations to the averaged fitness score after geometric alignment for each iterative registration process. The result of this correlation can be seen on the green curves, which all converge towards a minimum at 40 iterations. Within this dataset the average of the ICP point-to-point algorithm reaches the lowest minimum in comparison to the other ICP variants. As already stated in the introduction an ICP approach is often used after a coarse registration that can lead to lower minimum. As can be seen in Figure 10, the lowest error value at 40 iterations is SVD_ ICP. This means that a coarse SVD registration has been applied onto the point cloud pair after which an ICP point-to-point is applied. Secondly, the figure shows the computation time for each algorithm at a specific number of iterations. GICP has the worst computation time while ICP point-to-point has the fastest computation time. The reason why ICP point-to-surface is slower than ICP point-to-point is mainly due to the surface normal vector computation. This normal vector computation time could be decreased if the number of nearest neighbour points that should be included onto the surface, is lower. This will change the behaviour, so it will gradually perform more like an ICP point-to-point approach.

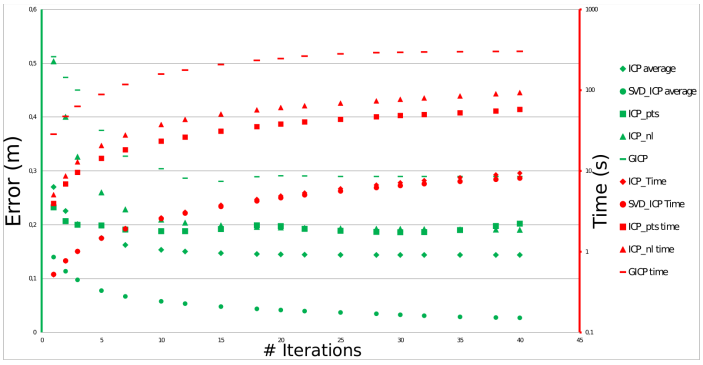

Figure 10. This figure show the comparison between the number of registration iteration and the time logarithmic in red and the comparison between the number of iterations and the fitness score in green for the average of ICP point-to-point (ICP), SVD applied before ICP (SVD_ICP), ICP point-to-surface (ICP_pts), ICP non-linear (ICP_nl) and Generalized ICP (GICP)

The previous paragraph stated the robustness as the average fitness score after alignment while this paraghraph will define the robustness by the sum of the average and the distance of one variance. Thus, the robustness factor is not only the average of each registration method, measured on a set of different 3D point clouds, but it also depending on the variance of the averaged fitness score or how far the fitness score will change over time. This result are visualized in Figure 11. On this graph, the number of iterations is shown on the x-axis and the sum of the average with the distance of one variance onto the y-axis. The robustness of the ICP point-to-surface method is very good due to the constant behavior during the entire dataset. This behavior is normal because the number of new surfaces will not decrease over time whilst two point clouds are being registered. In contrast to the previous method, the robustness of the other ICP approaches will go from worst in the beginning to better at the end due to the many changes in correspondences while registering two point clouds. When applying a coarse registration before an ICP approach the robustness will be much better at convergence than using all other stated methods.

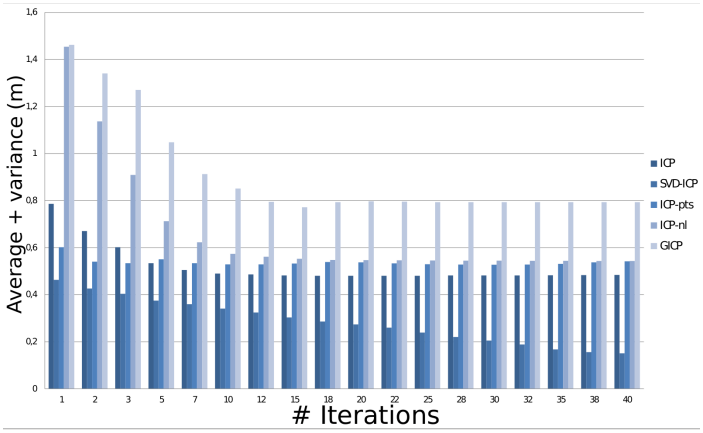

Figure 11. The x-axis represents the number of registration iterations and the y-axis represents the sum of the average and the variance for ICP point-to-point (ICP), SVD applied before ICP (SVD_ICP), ICP point-to-surface (ICP_pts), ICP non-linear (ICP_nl) and Generalized ICP (GICP)

C. Precision

To illustrate the behaviour of the results of the different stated registration algorithms during a certain period of time, the robustness was computed. In order to analyze the precision of the different registration algorithms, the rotation and translation part of the transformation matrix after alignment will be discussed separately. The precision of the different algorithms will illustrate how well they perform on the same two point clouds but with different correspondences. Figure 12 expands the flow that is used to compute the precision of the stated registration algorithms. Depending on the number of precision iterations, more or less subsamples will be computed. Thus, each subsample of the source point cloud will be registered with the target point cloud, which results in a series of alignment transformations that are compliant with the lowest fitness score at 40 iterations. Furthermore, the list of transformations will be divided into a list of rotation matrices and a list of translation matrices. In order to compare the different rotations independently, the 3×3 rotation matrices had to be converted into Euler angles. This means that each rotation around the x, y, and z axes can be represented by yaw, pitch and roll.

The different subsamples of the source point cloud has less points than the initial source point cloud and they are created on set of random indices, which are based on the indices of the source point cloud.

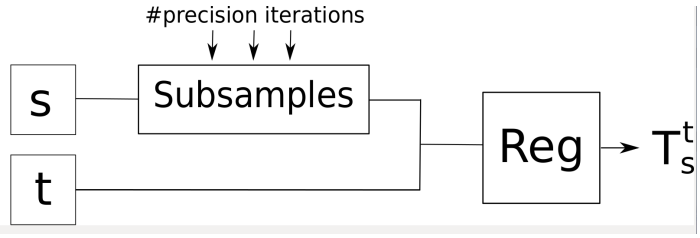

Figure 12. The benchmark precision scheme

To illustrate the precision of the translation part, we computed the average translation of the x, y, and z direction. Since the points in the source point cloud are randomly selected on each precision iteration, different correspondences between the source and target point cloud are observed, leading to a slightly different transformation. The standard deviation of this translation that is calculated for each x, y and z element in the transformation matrix, gives us the precision and is shown in the following three figures, 13, 14, and 15.

The variance of the x-translation can be observed in Figure 13. As can be seen, the value of the variance of PCA is zero. This is because of the different steps PCA undergoes to achieve an affine transformation. The variance on the centroid position of the source point cloud will not change a lot if a few points a missing. ICP point-to-surface has a lower variance in x-translation than ICP point-to-point due to surface normal estimation. The advantage of the surface estimation makes the ICP point-to-surface approach more precise due to low changes of surfaces. In comparison to the results of the robustness is the variance of applying an SVD approach before an ICP point-to-point method worse than without a coarse registration approach. Solving the problem by a non-linear cost-function, such as a Huber-Loss function, will result in the worst precision. These benchmark results are only applicable for indoor environmental data, that is retrieved with a Microsoft Kinect Camera.

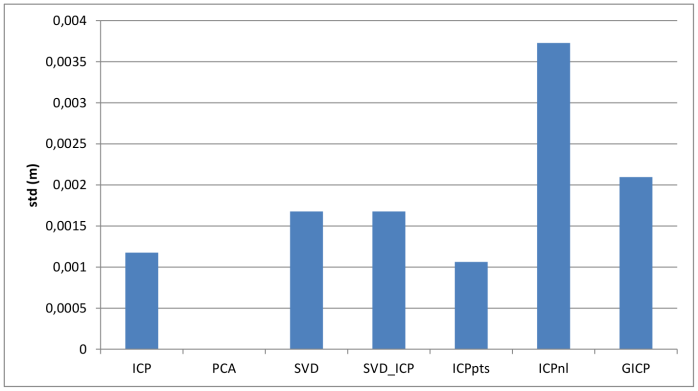

Figure 13. The x-axis represents the different methods and the y-axis represents the variance of the precision test in the x direction

When observing the variance of the translation in the y direction, a remarkable result for the non-linear approach can be seen in Figure 14. These results are much worse than in the x direction. This could be the result of setting the number of ICP iterations too low. As for the non-linear approach it is important to choose this number of iterations correctly because of the different minimization cost-function. In order to ensure a fair competition between the different algorithms we set the number of ICP iteration fixed to 40. As can be seen in Figure 11, each algorithm has reached a global minimum at 40 iterations. The other approaches have a similar result for the y direction as for the x direction.

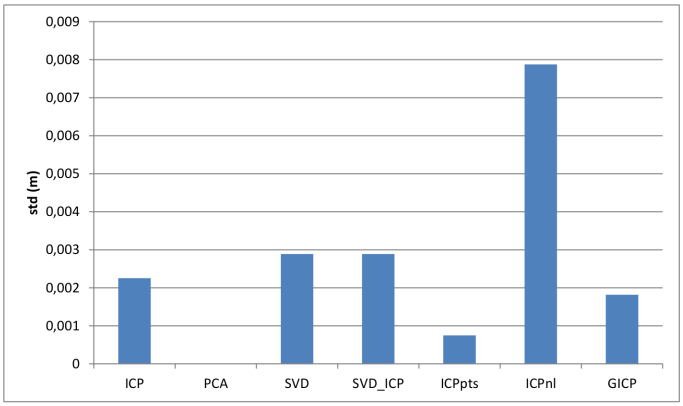

Figure 14. The x-axis represents the different methods and the y-axis represents the variance of the precision test in the y direction

The precision benchmark for the z direction gives better results than the x and y direction. Unlike the x and y directions we expected that the z direction, which represents the depth measurement, will give worse result due to noisy point clouds. The result of the variance of the z direction conclude that the precision of PCA is zero in all directions. This is because PCA translates the centered source point cloud against the centroid of the target point cloud and secondly, because PCA will not optimize the result. Thus, we can conclude that ICP point-to-surface has the best precision for the translation part.

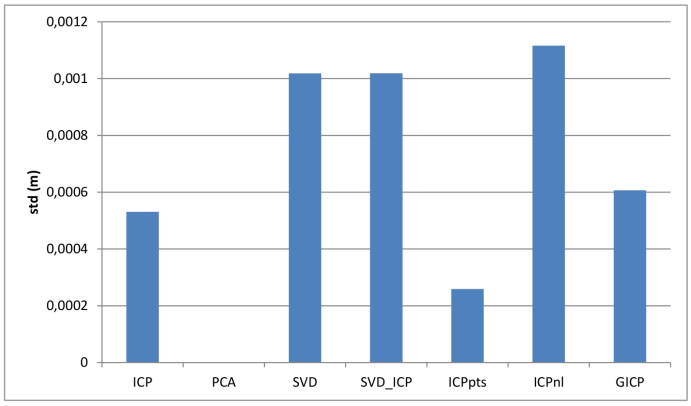

Figure 15. The x-axis represents the different methods and the y-axis represents the variance of the precision test in the z direction

The following figures shows the results of the precision for rotational part of the transformation. The 3×3 rotation matrix has been converted to Euler angles, in which each rotation is represented independently from each other by yaw, pitch

and roll. First, Figure 16 gives more insight to the variance of the different registration methods for the yaw rotation. The figure shows a remarkable difference for the PCA approach. This is because PCA observe the whole point cloud through the correlation between the different points by using the covariance matrix, while the ICP and SVD approaches will look for point correspondences. The variance in yaw direction is large due to the different subsamples, which will create point clouds where the density can change a lot in the direction of the smallest eigenvalue. This means that the probability of changing the direction of the largest eigenvector is large and thus the yaw rotation has a lower precision than the correspondence based approaches.

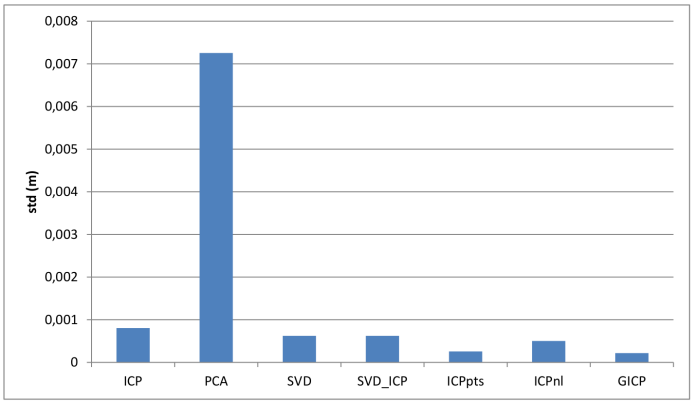

Figure 16. The x-axis represents the different methods and the y-axis represents the variance of the precision test in the yaw direction

To illustrate the precision of the transformation matrices after aligning with the different registration methods in the pitch direction. This result can be seen in Figure 17. The PCA method will perform more precisely in the pitch direction then the yaw direction. Secondly, the ICP point-to-surface approach will give the best results due to normal vector extension, which is a good parameter that is not changing a lot in the different subsamples of the source point cloud. Additionally, the variance of the method where the ICP approach is applied after a SVD is worse than the ICP point-to-point and the ICP point-to-surface methods.

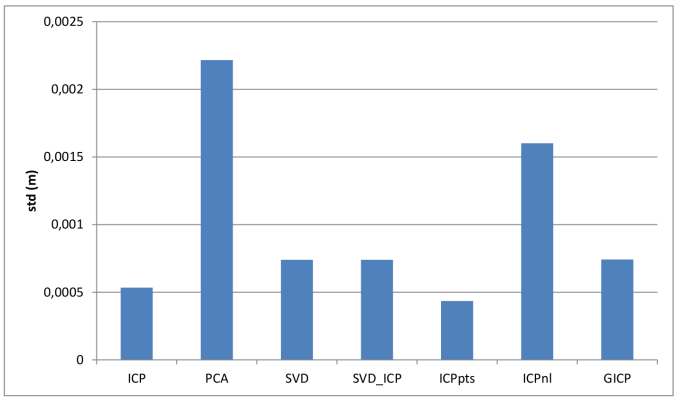

Figure 17. The x-axis represents the different methods and the y-axis represents the variance of the precision test in the pitch direction

The variances of the roll rotations are visualized in Figure 18. The algorithm that performs best is the ICP point-to-surface approach. Additionally, GICP performs better than ICP point-to-point method, the difference between these algorithm are negligible.

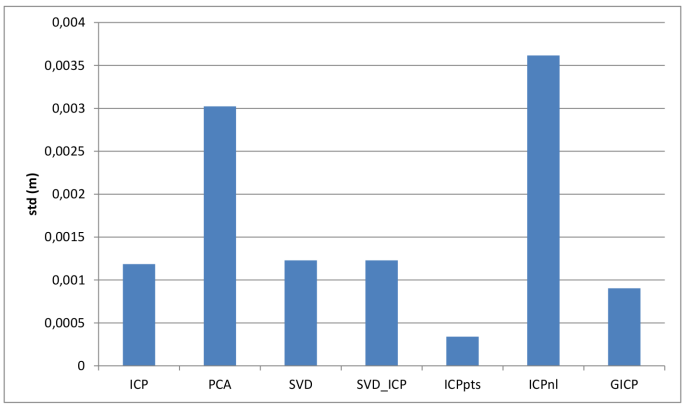

Figure 18. The x-axis represents the different methods and the y-axis represents the variance of the precision test in the roll direction

The different visualizations show that the result of the ICP point-to-surface method is the most rotation precise registration method. Followed by the GICP that has the best precision in yaw direction and the third in pitch. Due to the fact that the yaw direction is more valuable than the pitch, GICP is the second most precise algorithm based on rotational part of the transformation. The reason why yaw is more valuable than pitch is specific for this case where we want the most precise algorithm for a mobile robot SLAM application where the yaw rotation can change a lot in comparison with the pitch rotation. The ICP point-to-point algorithm results in the third most precise algorithm. This result is based on the rotational part of the transformation.

VI. Conclusion

This survey paper provides an overview of six different rigid 3D registration methods commonly used in robotics and computer vision. We discussed the mathematical foundations that are common to each of these algorithms and showed that each of them represents different approaches to solve a common least-squares optimization problem.

Finally, we compared the different methods with a critical view on their performance on a dataset, that was created with a Pioneer-3DX robot and a Microsoft Kinect Camera. To illustrate the performance, we quantified the robustness and the precision of the different registration methods. As result for the robustness we can conclude for this dataset that a combination of applying a ICP point-to-point method after an SVD method gives the minimum error based on 165 different point cloud pairs. On the other hand, the ICP point-to-surface is the most precise algorithm based on the rotational and translational part of the transformation after applying the precision benchmark test of this paper.

References

[1] B. Bellekens, V. Spruyt, and M. Weyn, “A Survey of Rigid 3D Pointcloud Registration Algorithms,” in AMBIENT 2014, The Fourth International Conference on Ambient Computing, Applications, Services and Technologies., 2014, pp. 8–13.

[2] R. B. Rusu, "Semantic 3D Object Maps for Everyday Manipulation in Human Living Environments," KI - Künstliche Intelligenz, vol. 24, no. 4, Aug. 2010, pp. 345–348. [Online]. Available: http://link.springer.com/10.1007/s13218-010-0059-6

[3] R. A. Newcombe, A. J. Davison, S. Izadi, P. Kohli, O. Hilliges, J. Shotton, D. Molyneaux, S. Hodges, D. Kim, and A. Fitzgibbon, "KinectFusion: Real-time dense surface mapping and tracking," in 2011 10th IEEE International Symposium on Mixed and Augmented Reality. IEEE, Oct. 2011, pp. 127–136.

[4] C. Kerl, J. Sturm, and D. Cremers, "Dense visual SLAM for RGB-D cameras," in 2013 IEEE/RSJ International Conference on Intelligent Robots and Systems. IEEE, Nov. 2013, pp. 2100–2106.

[5] S. Marden and J. Guivant, "Improving the Performance of ICP for Real-Time Applications using an Approximate Nearest Neighbour Search," 2012, pp. 3–5.

[6] P. J. Besl and N. D. McKay, "Method for registration of 3-D shapes," P. S. Schenker, Ed., Apr. 1992, pp. 586–606.

[7] S. Fantoni, U. Castellani, and A. Fusiello, "Accurate and automatic alignment of range surfaces," in Proceedings - 2nd Joint 3DIM/3DPVT Conference: 3D Imaging, Modeling, Processing, Visualization and Transmission, 3DIMPVT 2012. IEEE, Oct. 2012, pp. 73–80.

[8] A. V. Segal, D. Haehnel, and S. Thrun, "Generalized-ICP," in Proceedings of Robotics: Science and Systems, Seattle, 2009, p. 8.

[9] J. Salvi, C. Matabosch, D. Fofi, and J. Forest, "A review of recent range image registration methods with accuracy evaluation," Image and Vision Computing, vol. 25, 2007, pp. 578–596.

[10] D. Rueckert, L. I. Sonoda, C. Hayes, D. L. Hill, M. O. Leach, and D. J. Hawkes, "Nonrigid registration using free-form deformations: application to breast MR images." IEEE Transactions on Medical Imaging, vol. 18, no. 8, Aug. 1999, pp. 712–21. [Online]. Available: http://www.ncbi.nlm.nih.gov/pubmed/10534053

[11] F. Endres, J. Hess, N. Engelhard, J. Sturm, D. Cremers, and W. Burgard, "An evaluation of the RGB-D SLAM system," Robotics and Automation (ICRA), 2012 IEEE International Conference, vol. 3, no. c, May 2012, pp. 1691–1696. [Online]. Available: http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=6225199

[12] J. Aulinas, Y. Petillot, J. Salvi, and X. Lladó, "The SLAM problem: a survey." CCIA, 2008, pp. 363—-371.

[13] P. F. I. N. D. E. Carrera, "MADRID RGB-D SLAM Author : Jorge García Bueno," no. October, 2011.

[14] K. Berger, S. Meister, R. Nair, and D. Kondermann, "A state of the art report on kinect sensor setups in computer vision," in Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics), vol. 8200 LNCS, 2013, pp. 257–272.

[15] J. Sprickerhof and A. Nüchter, "An Explicit Loop Closing Technique for 6D SLAM." ECMR, 2009, pp. 1–6. [Online]. Available: http://plum.eecs.jacobs-university.de/download/ecmr2009.pdf

[16] A. S. Huang and A. Bachrach, "Visual odometry and mapping for autonomous flight using an RGB-D camera," International Symposium on Robotics Research (ISRR), 2011, pp. 1–16.

[17] M. Ruhnke, L. Bo, D. Fox, and W. Burgard, "Compact RGBD Surface Models Based on Sparse Coding." AAAI, 2013.

[18] S. Savarese, "3D generic object categorization, localization and pose estimation," in 2007 IEEE 11th International Conference on Computer Vision. IEEE, 2007, pp. 1–8.

[19] R. Shams, P. Sadeghi, R. Kennedy, and R. Hartley, "A survey of medical image registration on multicore and the GPU," pp. 50–60, 2010. [Online]. Available: http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=5438962

[20] D. Lee, H. Kim, and H. Myung, "Image feature-based real-time RGB-D 3D SLAM with GPU acceleration," Journal of Institute of Control, Robotics and Systems, vol. 19, no. Urai, 2013, pp. 457–461.

[21] W. R. Crum, "Non-rigid image registration: theory and practice," British Journal of Radiology, vol. 77, no. suppl_2, Dec. 2004, pp. S140–S153.

[22] J. Kay, "Introduction to Homogeneous Transformations & Robot Kinematics," Rowan University Computer Science Department, no. January, 2005, pp. 1–25.

[23] B. Draper, W. Yambor, and J. Beveridge, "Analyzing pca-based face recognition algorithms: Eigenvector selection and distance measures," Empirical Evaluation Methods in Computer Vision, Singapore, 2002, pp. 1–14.

[24] K. Low, "Linear least-squares optimization for point-to-plane icp surface registration," Tech. Rep. February, 2004.

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

[26] "ROS," 2015. [Online]. Available: http://www.ros.org/

[27] G. Grisetti, C. Stachniss, and W. Burgard, "Improved Techniques for Grid Mapping," Robotics, IEEE Transactions on Robotics, pp. 1–12.

[28] D. Fox, W. Burgard, F. Dellaert, and S. Thrun, "Monte Carlo Localization: Efficient Position Estimation for Mobile Robots," Aaai-99, no. Handschin 1970, 1999, pp. 343–349. [Online]. Available: http://dl.acm.org/citation.cfm?id=315322