A Survey of Rigid 3D Pointcloud Registration Algorithms

Ben Bellekens, Vincent Spruyt, Rafael Berkvens, and Maarten Weyn

CoSys-Lab, Faculty of Applied Engineering
University of Antwerp, Belgium
ben.bellekens@uantwerpen.be

***Abstract*— Geometric alignment of 3D pointclouds, obtained using a depth sensor such as a time-of-flight camera, is a challenging task with important applications in robotics and computer vision. Due to the recent advent of cheap depth sensing devices, many different 3D registration algorithms have been proposed in literature, focussing on different domains such as localization and mapping or image registration. In this survey paper, we review the state-of-the-art registration algorithms and discuss their common mathematical foundation. Starting from simple deterministic methods, such as Principal Component Analysis (PCA) and Singular Value Decomposition (SVD), more recently introduced approaches such as Iterative Closest Point (ICP) and its variants, are analyzed and compared. The main contribution of this paper therefore consists of an overview of registration algorithms that are of interest in the field of computer vision and robotics, for example Simultaneous Localization and Mapping.**

Keywords–3D pointcloud; PCL; 3D registration; rigid transformation; survey paper

I. INTRODUCTION

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 pointcloud data. On 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 yield an unstructured pointcloud. These pointclouds 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 [1], [2], [3]. In the latter case however, point clouds obtained at different time instances need to be aligned, a process which is often referred to as registration. Registration algorithms are able to estimate the ego-motion of the robot by calculating the transformation that optimally maps two pointclouds, each of which is subject to camera noise.

These registration algorithms can be classified coarsely into rigid and non-rigid approaches. Rigid approaches assume a rigid environment such that the transformation can be modeled using only 6 Degrees Of Freedom (DOF). Non-rigid methods on the other hand, are able to cope with articulated objects or soft bodies that change shape over time.

Registration algorithms are used in different fields and applications, such as 3D object scanning, 3D mapping, 3D localization and ego-motion estimation, human body detection. Most of these state-of-the-art applications employ either a simple Singular Value Decomposition (SVD) [4] or Principal Component Analysis (PCA) based registration, or use a more advance iterative scheme based on the Iterative Closest Point (ICP) algorithm [5]. Recently, many variants on the original ICP approach have been proposed, the most important of which are non-linear ICP [6], generalized ICP [7], and non-rigid ICP [8].

The choice for one of these algorithms 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. To our knowledge, a general discussion of each of the above methods is not available in literature. As a result it is difficult to compare these algorithms objectively. Therefore, in this paper we discuss the mathematical foundations that are common to the most widely used 3D registration algorithms, and we compare their strengths and weaknesses in different 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 different characteristic 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, augmented reality, 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.

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) [9], [10], [11], [12]. The reconstructed map is represented by a set of pointclouds which are aligned by means of registration and can be used for obstacle avoidance, map exploration, autonomous vehicle control, etc.[3], [13], [14]. Furthermore, depth information is often combined with a

 ISBN: 978-1-61208-356-8

traditional RGB camera [2], [15] in order to greatly facilitate real-world problems such as object detection in cluttered scenes, object tracking and object recognition [16].

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. [8], [17]. Similarly, non-rigid transformations are used to obtain a multi-modal representation of a scene, by combining MRI, CT, and PET volumes into a single 3D model [8].

III. Definitions

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

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

A. Least-Square 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 which 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-square optimization and is often used for fitting and regression problems.

Whereas a linear least-square problem can be solved analytically, this is often not the case for non-linear least-square 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 between corresponding points of two pointcloud datasets.

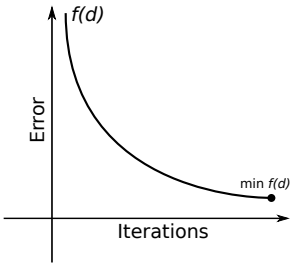

Figure 1. ICP Least square approach.

B. Homogeneous transformations

A homogeneous transformation in three dimensions is specified by a 4×4 affine transformation matrix [18]. This matrix is used to project each point in Cartesian space with respect to a specific viewpoint. In the following, let $\mathbf{v_1} = (x_1, y_1, z_1, 1)^\intercal$ be a point whose base is defined by viewpoint one and let $\mathbf{v_2} = (x_2, y_2, z_2, 1)^\intercal$ be a point whose base is defined by viewpoint two. Then it is possible to express $\mathbf{v_2}$ relative to the base of viewpoint one as $T\mathbf{v_1} = \mathbf{v_2}$, where T is an affine transformation matrix defined by (1). This is illustrated more clearly by Figure 2.

$$T = \begin{pmatrix} r_{1,1} & r_{1,2} & r_{1,3} & \mathbf{t}_{1,4} \\ r_{2,1} & r_{2,2} & r_{2,3} & \mathbf{t}_{2,4} \\ r_{3,1} & r_{3,2} & r_{3,3} & \mathbf{t}_{3,4} \\ a_{4,1} & a_{4,2} & a_{4,3} & a_{4,4} \end{pmatrix} \tag{1}$$

The transformation matrix shown by (1) represents an affine transformation if $a_{4,1} = a_{4,2} = a_{4,3} = 0$ and $a_{4,4} \neq 0$. Affine transformations are constructed with a 3×3 rotation matrix R and column vector $\mathbf{t}$ representing a translation.

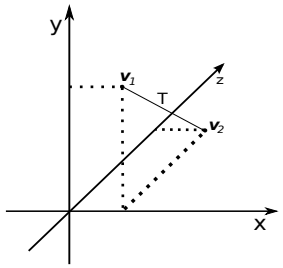

Figure 2. homogeneous transformation.

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 singular value decomposition are pairwise registration methods based on the covariance matrices and the cross correlation matrix of the pointclouds, 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 pointclouds.

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 [19]. 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.

 ISBN: 978-1-61208-356-8

Therefore, if the covariance matrix of two pointclouds 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. Pointcloud centering simply corresponds to subtracting the centroid coordinates from each of the point coordinates. The centroid of the pointcloud corresponds to the average coordinate and is thus obtained by dividing the sum of all point-coordinates by the number of points in the pointcloud.

Since registration based on PCA simply aligns the directions in which the pointclouds 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 pointcloud, and therefore represents the pointcloud's rotation. In the following, let A be the covariance matrix, let $\mathbf{v}$ be an eigenvector of this matrix, and let λ be the corresponding eigenvalue. The eigenvalues decomposition problem is then defined as:

$$A\mathbf{x} = \lambda\mathbf{x} \tag{2}$$

and further reduces to:

$$\mathbf{x}(A - \lambda I) = 0. \tag{3}$$

It is clear that (3) only has a non-zero solution if $A - \lambda I$ is singular, an consequently if its determinant equals zero:

$$\det(A - \lambda I) = 0 \tag{4}$$

The eigenvalues can simply be obtained by solving (4), whereas the corresponding eigenvectors are obtained by substituting the eigenvalues into (2).

Once the eigenvectors are known for each pointcloud, registration is achieved by aligning these vectors. In the following, let matrix T_t^y represent the transformation that would align the largest eigenvector of the target pointcloud t with the y-axis. Let matrix T_y^s represent the transformation that would align the largest eigenvector of the source pointcloud s with the y-axis. Then the final transformation matrix T_t^s that aligns the source pointcloud with the target pointcloud can be obtained easily, as illustrated by Figure 3.

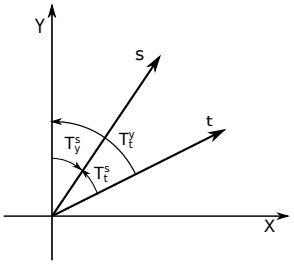

Figure 3. PCA alignment from source to target.

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

B. Singular Value Decomposition

PCA based registration simply aligns the directions of the largest variance of each pointcloud and therefore 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 pointcloud is approximately normally distributed.

However, if point correspondences between the two pointclouds 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-square problem that can be solved robustly using the SVD method [4].

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

$$M = USV^{\intercal} \tag{5}$$

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

$$R_t^s = UV^{\intercal} \tag{6}$$

and the translation from target pointcloud to source pointcloud is defined by:

$$\mathbf{t} = \mathbf{c_s} - R_t^s\mathbf{c_t} \tag{7}$$

C. Iterative Closest Point

Whereas the SVD algorithm directly solves the least-square problem, thereby assuming perfect data, Besl and Mc. Kay [5] 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 by Figure 4.

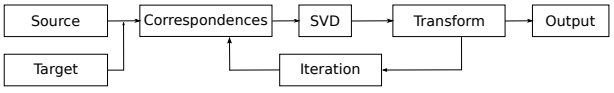

Figure 4. ICP overview scheme.

The input of the ICP algorithm consists of a source pointcloud and a target pointcloud. Point correspondences between these pointclouds are defined based on a nearest neighbor 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 pointclouds. After registration, 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.

 ISBN: 978-1-61208-356-8

1) ICP point-to-point: The ICP point-to-point algorithm was originally described in [1] and simply obtains point correspondences by searching for the nearest neighbor target point $\mathbf{q}_i$ of a point $\mathbf{p}_j$ in the source pointcloud. The nearest neighbor matching is defined in terms of the Euclidean distance metric:

$$\hat{i} = \arg\min_i \|\mathbf{p}_i - \mathbf{q}_j\|^2, \tag{8}$$

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

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

$$\hat{R}, \hat{\mathbf{t}} = \arg\min_{R,\mathbf{t}} \sum_{i=1}^{N} \|(R\mathbf{p}_i + \mathbf{t}) - \mathbf{q}_i\|^2 \tag{9}$$

ICP then iteratively solves (8) and (9) to improve upon the estimates of the previous iterations. This is illustrated by Figure 5, where surface s is aligned to surface t using n ICP iterations.

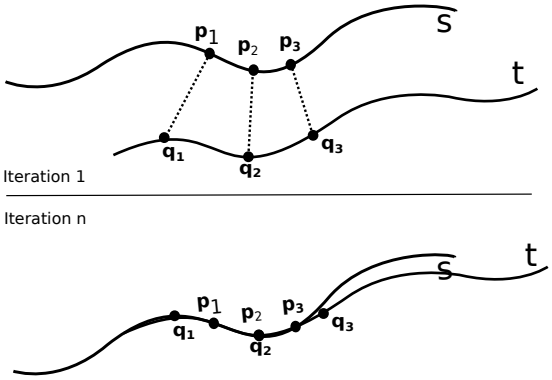

Figure 5. ICP alignment based on a point to point approach.

2) ICP point-to-surface: Due to the simplistic definition of point correspondences, the ICP point-to-point algorithm proposed by [20] is rather sensitive to outliers. Instead of directly finding the nearest neighbor to a source point $\mathbf{p}_j$ in the target pointcloud, one could take the local neighborhood of a correspondence candidate $\mathbf{q}_i$ into account to reduce the algorithm's sensitivity to noise.

The ICP point-to-surface algorithm assumes that the point clouds are locally linear, such that the local neighborhood of a point is co-planar. This local surface can then be defined by its normal vector $\mathbf{n}$, which is obtained as the smallest eigenvector of the covariance matrix of the points that surround correspondence candidate $\mathbf{q}_i$.

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 n:

$$\hat{R}, \hat{\mathbf{t}} = \arg\min_{\hat{R},\hat{\mathbf{t}}} \left(\sum_{i=1}^{N} \|((R\mathbf{p}_i + \mathbf{t}) - \mathbf{q}_i)\mathbf{n_i}\| \right) \tag{10}$$

This is illustrated more clearly by Figure 6.

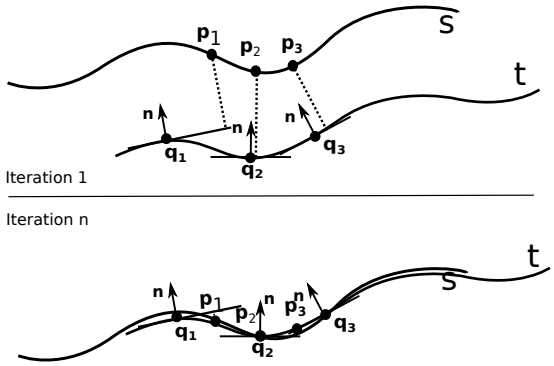

Figure 6. ICP alignment based on a point to surface approach.

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-square 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 (11). 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 behaves like an L1 cost function. Moreover, the Huber loss function is smooth and differentiable, allowing traditional numerical optimization methods to be used to efficiently traverse the search space.

$$e^2(n) = \begin{cases} n^2/2 & \text{if } |\mathrm{n}| \leq k \\ k|n| - n^2/2 & \text{if } |\mathrm{n}| > k \end{cases} \tag{11}$$

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

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

$$\hat{R}, \hat{\mathbf{t}} = \arg\min_{\hat{R},\hat{\mathbf{t}}} \sum_{i=1}^{N} e^2(n) \tag{12}$$

where

$$n = \|(R\mathbf{p} - \mathbf{t}) - \mathbf{q}\| \tag{13}$$

To obtain the optimal estimates $\hat{R}, \hat{\mathbf{t}}$ in (12), the Levenberg-Marquardt algorithm (LMA) [6] is used. The LMA method is an iterative procedure similar to the well known gradient descent and Gauss-Newton algorithms, that 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 pointcloud is taken from a known geometric surface instead of being obtained through noisy measurements. However,

 ISBN: 978-1-61208-356-8

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 pointcloud represents a discretized sample set of a known geometric surface model since offsets along the surface are only allowed in the target pointcloud.

To solve this, Segal *et al.*[7] proposed the Generalized ICP algorithm which performs plane-to-plane matching. They introduced a probabilistic interpretation of the minimization process such that structural information from both the source pointcloud and the target pointcloud 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 pointcloud is obtained from a known geometric surface, Segal *et al.* assume that both the source pointcloud $A = \{\mathbf{a}_i\}$ and the target pointcloud $B = \{\mathbf{b}_i\}$ consist of random samples from an underlying unknown pointcloud $\hat{A} = \{\hat{\mathbf{a}}_i\}$ and $\hat{B} = \{\hat{\mathbf{b}}_i\}$. For the underlying and unknown pointclouds $\hat{A}$ and $\hat{B}$, perfect correspondences exist, whereas this is not the case for the observed pointclouds A and B, since each point $\mathbf{a}_i$ and $\mathbf{b}_i$ is assumed to be sampled from a normal distribution such that $\mathbf{a}_i \sim \mathcal{N}(\hat{\mathbf{a}}_i, C_i^A)$ and $\mathbf{b}_i \sim \mathcal{N}(\hat{\mathbf{b}}_i, C_i^B)$. The covariance matrices C_i^A and C_i^B are unknown. If both pointclouds 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 $\hat{\mathbf{b}}_i = T\hat{\mathbf{a}}_i$. If T would be known, we could apply this transformation to the observed source pointcloud A, and define the error to be minimized as $d_i^T = \mathbf{b}_i - T\mathbf{a}_i$. Because both $\mathbf{a}_i$ and $\mathbf{b}_i$ are assumed to be drawn from independent normal distributions, d_i^T which is a linear combination of $\mathbf{a}_i$ and $\mathbf{b}_i$, is also drawn from a normal distribution:

$$d_i^T \sim \mathcal{N}(\hat{\mathbf{b}}_i - T\hat{\mathbf{a}}_i, C_i^B + TC_i^A T^\intercal) \quad (14)$$

$$= \mathcal{N}(0, C_i^B + TC_i^A T^\intercal) \quad (15)$$

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

$$\hat{T} = \arg\min_T \sum_i \log\left(p(d_i^T)\right)$$

$$= \arg\min_T \sum_i d_i^{T\intercal}(C_i^B + TC_i^A T^\intercal)^{-1} d_i^T \quad (16)$$

Segal *et al.* showed that both point-to-point and point-to-plane ICP are specific cases of (16), 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, (17) reduces to:

$$\hat{T} = \arg\min_T \sum_i d_i^{T\intercal} d_i^T$$

$$= \arg\min_T \sum_i \|d_i^T\|^2, \quad (17)$$

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 difference between a naive PCA based approach, a correspondences based SVD approach, and the ICP point-to-point registration approach. To allow for a fair comparison, we use the publicly available dataset proposed by Pomerlau *et al.* [21].

Figure 7 shows the matching error plotted against the number of iterations for the ICP point-to-point algorithm (dark-gray) without pre-alignment, and for the ICP point-to-point algorithm (light-gray) where the data has been pre-aligned using the SVD approach. In the latter case, a simple nearest-neighbor matching was used to define point correspondences, after which the SVD algorithm was used to solve the least-squares problem. This result clearly shows the importance of a rough initial alignment before applying the ICP algorithm.

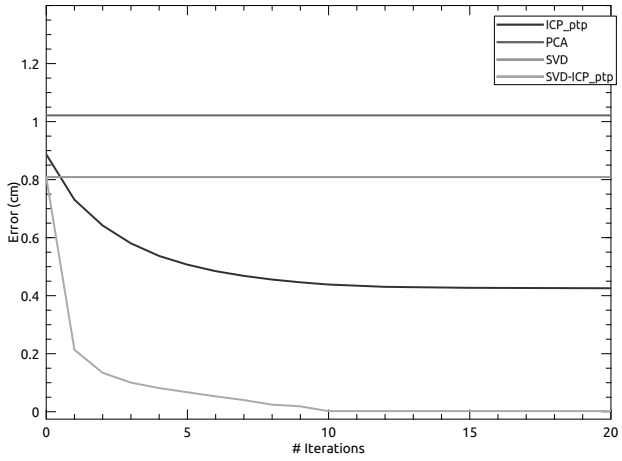

Figure 7. Comparison between PCA, SVD and general point to point ICP

Furthermore, figure 7 shows the results of a single SVD based least-squares iteration, and the results obtained using the PCA based registration approach. It is clear that the PCA based approach yields the largest matching error, due to the fact that it does not incorporate correspondence information, such that this method is highly sensitive to outliers.

On the other hand, a simple PCA or SVD based approach is extremely computational efficient, whereas the iterative ICP scheme is often too computationally expensive for real-time applications. However, Figure 7 shows that convergence can be reached quickly if a rough initial alignment is available.

Finally, it is important to note that result of the variants of ICP such as point-to-plane and plane-to-plane greatly depend on the input data. If the source pointcloud does not contain much noise, while the target pointcloud is mostly smooth and piece-wise planar, the point-to-plane algorithm outperforms

 ISBN: 978-1-61208-356-8

the traditional point-to-point method. On the other hand if the geometric structures in the scene are mostly quadratic or polynomial, the traditional ICP point-to-point algorithm yields better results. Similarly, if a lot of noise is observed in the source pointcloud, ICP plane-to-plane outperforms ICP point-to-plane.

VI. Conclusion

In this paper we provided an overview of six state-of-the-art rigid 3D registration algorithms commonly used in robotics and computer vision. We discussed the mathematical foundation that is common to each of these algorithms and showed that each of them represents different approaches to solve a common least-square optimization problem.

Furthermore, we used a publicly available dataset to compare the results of these algorithms and concluded that the results are extremely data dependent such that the choice for a specific algorithm should mainly depend on the application and input data.

References

[1] 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.

[2] 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.

[3] 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.

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

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

[6] S. Fantoni, U. Castellani, and A. Fusiello, "Accurate and Automatic Alignment of Range Surfaces," in 2012 Second International Conference on 3D Imaging, Modeling, Processing, Visualization & Transmission. IEEE, Oct. 2012, pp. 73–80.

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

[8] 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

[9] F. Endres, J. Hess, N. Engelhard, J. Sturm, D. Cremers, and W. Burgard, "An evaluation of the RGB-D SLAM system," in 2012 IEEE International Conference on Robotics and Automation, vol. 3, no. c. IEEE, May 2012, pp. 1691–1696.

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

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

[12] K. Berger, S. Meister, R. Nair, and D. Kondermann, "A state of the art report on kinect sensor setups in computer vision," ...and Depth Imaging. Sensors ..., 2013.

[13] J. Sprickerhof and A. Nüchter, "An Explicit Loop Closing Technique for 6D SLAM." ..., 2009, pp. 1–6.

[14] A. Huang and A. Bachrach, "Visual odometry and mapping for autonomous flight using an RGB-D camera," International ..., 2011, pp. 1–16.

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

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

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

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

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

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

[21] F. Pomerleau, S. Magnenat, F. Colas, M. Liu, and R. Siegwart, "Tracking a depth camera: Parameter exploration for fast ICP," in 2011 IEEE/RSJ International Conference on Intelligent Robots and Systems. IEEE, Sep. 2011, pp. 3824–3829.

 ISBN: 978-1-61208-356-8