Adv. Appl. Clifford Algebras (2018) 28:50

0188-7009/020001-16
published online May 3, 2018
https://doi.org/10.1007/s00006-018-0864-9

Advances in Applied Clifford Algebras

A Curvature-Based Descriptor for Point Cloud Alignment Using Conformal Geometric Algebra

Adam Leon Kleppe* and Olav Egeland

Abstract. This paper presents a descriptor for course alignment of point clouds using conformal geometric algebra. The method is based on selecting keypoints depending on shape factors to identify distinct features of the object represented by the point cloud, and a descriptor is then calculated for each keypoint by fitting two spheres that describe the local curvature. The method for estimating the point correspondences is to a larger extent based on geometric arguments than the method of Kleppe et al. (IEEE Trans Autom Sci Eng, 2017), which results in improved performance. The accuracy of the curvature-based descriptor is validated in experiments, and is shown to compare favorably to state-of-the-art methods in an experiment on course alignment of industrial parts to be assembled with robots.

Keywords. Keypoint descriptor, Conformal geometric algebra, Initial alignment, Point clouds.

1. Introduction

The 3D–3D registration problem [16] is well-established in computer vision, and is still an active field of research. The problem involves two sub-problems: Calculating the displacement between two point clouds, and estimating the point correspondences between the point clouds [4]. A large number of methods have been proposed to solve the registration problem in 3D [4,24]. These methods can be classified as either coarse or fine registration methods. Both course and fine registration may have to be applied in order to find a globally optimal solution. Then coarse registration is used to find the initial alignment, which is improved with a fine registration method.

*Corresponding author.

Most of the fine registration methods, including the iterative closest point (ICP) [2,5,19], can be described as expectation–maximization methods [16]. These methods alternate between the estimation of the displacement and the estimation of point correspondences. A known restriction with expectation–maximization algorithms is that they converge to locally optimal solutions. To ensure convergence to the global optimum, the algorithm either can be expanded to include global optimization techniques, such as Go-ICP [28] or Sparse ICP [3], or good initial conditions can be computed with course registration methods, such as [20,21,23].

Coarse registration methods can be further divided into global and local approaches [4], where a global approach will estimate the displacement between two point clouds based on global parameters like the centroid to find the translation, and global principal component analysis to find the orientation. A local approach will be based on the identification of keypoints in the point cloud, that have certain characteristic features, and then characterize the keypoints with a descriptor. Then keypoint correspondences in two point clouds can be established by comparing the the descriptors of the keypoints. This approach is used in point signatures [6], spin images [14] and point feature histograms [20,21].

In [15] we proposed a course registration method where keypoints were selected based on the local surface around each point. A descriptor was then calculated for each keypoint by fitting two spheres to the surface around the keypoint. The descriptors from two point clouds were then compared to estimate the point correspondence between the keypoints based on least-squares optimization. The displacement of the two point clouds was then found based on the point correspondences, and this gave the initial alignment.

In this paper we propose an improvement to the descriptor in [15] where a new and improved descriptor is used, based on a geometric approach. The main improvement of the method is that more keypoint correspondences are found, which improves the estimation of the displacement. The method is tested in experiments which demonstrates the the high accuracy of the proposed method in comparison to existing methods for 3D–3D registration.

This paper is is organized as follows: Sect. 2 presents the methods and notation used in the paper, including the required background in conformal geometric algebra. Section 3 describes the method proposed in [15] as well as the improvement proposed in this paper. Section 4 presents the experimental, which demonstrates the successful application of the method. The result from these experiments are then presented and discussed, and lastly the conclusion is found in Sect. 5.

2. Preliminaries

2.1. 3D–3D Registration Problem

Consider the point cloud $\boldsymbol{X} = \{\boldsymbol{x}_i\}$, $i = 1, \ldots, n_x$ of observed point positions $\boldsymbol{x}_i \in \mathbb{R}^3$, and the point cloud $\boldsymbol{Y} = \{\boldsymbol{y}_j\}$, $j = 1, \ldots, n_y$ of model point positions $\boldsymbol{y}_i \in \mathbb{R}^3$, where the model points are assumed to be computed from a CAD

model of an object, while the observation points are assumed to be computed from a 3D camera. The 3D–3D registration problem is then to minimize the error function

$$E(\boldsymbol{R},\boldsymbol{t}) = \sum_{i=1}^{n_x} \|\boldsymbol{y}_{j^*} - \boldsymbol{R}\boldsymbol{x}_i - \boldsymbol{t}\|^2 \tag{2.1}$$

with respect to $\boldsymbol{R} \in SO(3)$ and $\boldsymbol{t} \in \mathbb{R}^3$, where $\boldsymbol{y}_{j^*}$ is the optimal point corresponding to the data point $\boldsymbol{x}_i$.

In the ICP method, the point $\boldsymbol{y}_{j^*}$ corresponding to the data point $\boldsymbol{x}_i$ is found from

$$j^* = \underset{j}{\operatorname{argmin}} \|\boldsymbol{y}_j - \boldsymbol{R}\boldsymbol{x}_i - \boldsymbol{t}\| \tag{2.2}$$

The solution is then found by iteration. At each step the correspondence is found from the minimization of (2.2) for the current estimate of the pose, and then the estimate of the pose $\boldsymbol{R},\boldsymbol{t}$ is found by minimizing (2.1) for the current estimate of the correspondence. This minimization will require that the initial guess for the pose and the correspondence is sufficiently close to the optimal solutions.

Initial alignment can be performed with a local approach using descriptors for points in two point clouds, and then to find the initial pose by matching the two point clouds based on these descriptors. Such descriptors can be calculated for all points in the point cloud [20], or for selected keypoints.

2.2. Conformal Geometric Algebra

Conformal geometric algebra [9,13] extends the Euclidean space $\mathbb{R}^3$ with basis vectors given by the orthogonal unit vectors e_1, e_2, e_3 to the 5 dimensional space $\mathbb{R}^{4,1}$ with basis $\{e_1, e_2, e_3, e_0, e_\infty\}$ where $e_0^2 = e_\infty^2 = 0$ and $e_0 \cdot e_\infty = -1$.

Consider a Euclidean point $\boldsymbol{p} = p_1e_1 + p_2e_2 + p_3e_3 \in \mathbb{R}^3$, which can be written as the column vector $[\boldsymbol{p}] = [p_1, p_2, p_3]^{\mathrm{T}}$. This point can be represented by the conformal point $\boldsymbol{P} \in \mathbb{R}^{4,1}$ defined by $\boldsymbol{P} = \boldsymbol{p} + \frac{1}{2}\boldsymbol{p}^2e_\infty + e_0$ where $\boldsymbol{P}$ has the property that $\boldsymbol{P}_1 \cdot \boldsymbol{P}_2 = -\frac{1}{2}||\boldsymbol{p}_1 - \boldsymbol{p}_2||$. The conformal point $\boldsymbol{P}$ can also be written as the column vector

$$[\boldsymbol{P}] = \begin{bmatrix} [\boldsymbol{p}] \\ \frac{1}{2}\boldsymbol{p}^2 \\ 1 \end{bmatrix} \tag{2.3}$$

A plane is given by $\boldsymbol{\Pi} = \boldsymbol{n} + \beta e_\infty$ where $\boldsymbol{n} \in \mathbb{R}^3$ is the unit normal of the plane, and $\beta \in \mathbb{R}$ is the distance from the origin to the plane. This is referred to as the IPNS representation of a plane in [13], while it is called a dual plane in [9]. The $\boldsymbol{\Pi}$ can be written as the column vector

$$[\boldsymbol{\Pi}] = \begin{bmatrix} [\boldsymbol{n}] \\ \beta \\ 0 \end{bmatrix} \tag{2.4}$$

A sphere $\boldsymbol{S}$ is given in the IPNS representation as $\boldsymbol{S} = \boldsymbol{P}_c - \frac{1}{2}r^2e_\infty$ where $\boldsymbol{P}_c = \boldsymbol{p}_c + \frac{1}{2}\boldsymbol{p}_c^2e_\infty + e_0$ is the center point and r is the radius of the

sphere. The vector representation of $\boldsymbol{S}$ is

$$[\boldsymbol{S}] = \begin{bmatrix} [\boldsymbol{p}_c] \\ \frac{1}{2}(\boldsymbol{p}_c^2 - r^2) \\ 1 \end{bmatrix} \tag{2.5}$$

The radius r of a given sphere $\boldsymbol{S}$ could be found using $\boldsymbol{S}^2 = \boldsymbol{S} \cdot \boldsymbol{S} = r^2$. Let δ be the distance from the sphere $\boldsymbol{S}$ to the point $\boldsymbol{p}$. The distance $d = r + \delta$ from the center of the sphere $\boldsymbol{S}$ to a point $\boldsymbol{p}$ can be found by

$$d^2 = \boldsymbol{S} \cdot \boldsymbol{S} - 2\boldsymbol{S} \cdot \boldsymbol{P} \tag{2.6}$$

which is verified by the calculation $\boldsymbol{S} \cdot \boldsymbol{S} - 2\boldsymbol{S} \cdot \boldsymbol{P} = (\boldsymbol{p} - \boldsymbol{p}_c)^2$. It is seen that

$$2\boldsymbol{S} \cdot \boldsymbol{P} = r^2 - (r^2 + 2\delta r + \delta^2) = -2\delta r \left(1 + \frac{2\delta}{r}\right) \tag{2.7}$$

It follows that if $2\delta/r \ll 1$, then the distance δ from the sphere $\boldsymbol{S}$ to the point $\boldsymbol{p}$ can be approximated by

$$\frac{(\boldsymbol{S} \cdot \boldsymbol{P})^2}{\boldsymbol{S}^2} = \delta^2 \left(1 + \frac{2\delta}{r}\right)^2 \approx \delta^2 \tag{2.8}$$

2.3. Principal Component Analysis

2.3.1. Local Reference Frame. Principal component analysis (PCA) can be used to construct a local reference frame around a selected point $\boldsymbol{p}$ in a point cloud [27]. To generate a local reference frame for the point $\boldsymbol{p}_i$ in the point cloud $\boldsymbol{X}$, a neighbourhood

$$\boldsymbol{N}_i = \{\boldsymbol{p}_j : -2\boldsymbol{P}_j \cdot \boldsymbol{P}_i \leq r^2\}, \quad \boldsymbol{P}_j \in \boldsymbol{X} \tag{2.9}$$

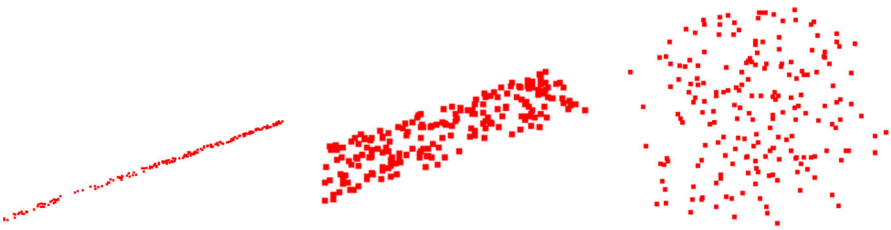
of points are selected within the radius r of $\boldsymbol{p}_i$, where r is a user-defined variable which must be chosen based on the application. A covariance matrix

$$\mathbf{C}_{\boldsymbol{p}_i} = \sum_{\boldsymbol{p}_k \in \boldsymbol{N}_i} ([\boldsymbol{p}_k] - [\bar{\boldsymbol{p}}_i])([\boldsymbol{p}_k] - [\bar{\boldsymbol{p}}_i])^{\mathrm{T}} \tag{2.10}$$

is calculated for all the points in $\boldsymbol{N}_i$, where $[\bar{\boldsymbol{p}}_i] = \frac{1}{n}\sum_{\boldsymbol{p}_k \in \boldsymbol{N}_i} [\boldsymbol{p}_k]$, and n is the number of points in $\boldsymbol{N}_i$. A local reference frame at $\boldsymbol{p}_i$ can then be defined by the eigenvectors $\boldsymbol{v}_j$ for $j = 1, 2, 3$ of $\mathbf{C}_{\boldsymbol{p}_i}$.

2.3.2. Shape Factors. Principal component analysis can also be used to define the shape factors [1,18] around the point $\boldsymbol{p}_i$. These shape factors define the shape of the neighbouring points $\boldsymbol{N}_i$ around $\boldsymbol{p}_i$, and are found from the eigenvalues $\lambda_1, \lambda_2, \lambda_3$ of $\mathbf{C}_{\boldsymbol{p}_i}$, where $\lambda_1 \geq \lambda_2 \geq \lambda_3 \geq 0$.

The shape of the neighbourhood $\boldsymbol{N}_i$ can be characterized by the eigenvalues of $\mathbf{C}_{\boldsymbol{p}_i}$. Three special cases are of special interest. The first case is $\lambda_1 > 0$ and $\lambda_2 = \lambda_3 = 0$, which means that the points of the neighbourhood are in a linear shape where the points are on a line in the direction of the eigenvector $\mathbf{v}_1$. The second case is when $\lambda_1 = \lambda_2 > 0$ and $\lambda_3 = 0$, which occurs when the points of the neighbourhood are in a planar shape with the points on the plane spanned from the eigenvectors $\mathbf{v}_1$ and $\mathbf{v}_2$. The third case is when $\lambda_1 = \lambda_2 = \lambda_3$, which means that the points of the neighbourhood



(A) **(B)** **(C)**

FIGURE 1. Examples of point clouds with different shape factors. **a** Point cloud with a shape factor $C_l \approx 1$. **b** Point cloud with a shape factor $C_p \approx 1$. **c** Point cloud with a shape factor $C_s \approx 1$

form a sphere or a voluminous form. Examples of the three cases are shown in Fig. 1.

To identify points $\boldsymbol{p}_i$ where the neighbourhood $\boldsymbol{N}_i$ will be according to one of these three cases, three shape factor parameters are define by [1,18]

$$C_l = \frac{\lambda_1 - \lambda_2}{\lambda_1 + \lambda_2 + \lambda_3} \tag{2.11}$$

$$C_p = \frac{2(\lambda_2 - \lambda_3)}{\lambda_1 + \lambda_2 + \lambda_3} \tag{2.12}$$

$$C_s = \frac{3\lambda_3}{\lambda_1 + \lambda_2 + \lambda_3} \tag{2.13}$$

It is noted that $C_l + C_p + C_s = 1$. It is seen that if $C_l = 1$, then the neighbourhood forms a linear shape; if $C_p = 1$, the neighbourhood has a planar form; and if $C_s = 1$ the neighbourhood has a spherical or voluminous form.

2.4. Sphere Fitting

In conformal geometric algebra, a sphere $\boldsymbol{S}$ can be fitted to a point cloud $\boldsymbol{p}_i$, $i = 1, \ldots, n$ as proposed in [8] by minimizing the objective function

$$f(\boldsymbol{S}) = \sum_{i=1}^{n} \frac{(\boldsymbol{S} \cdot \boldsymbol{P}_i)^2}{\boldsymbol{S}^2} \tag{2.14}$$

This method is an extension to n-spheres in conformal geometric algebra of the 2D Pratt fit [17] for circles in a homogeneous description of the plane. The condition for a minimum of $f(\boldsymbol{S})$ is $(\sum_{i=1}^{n} \boldsymbol{P}_i(\boldsymbol{P}_i \cdot \boldsymbol{S})) \wedge \boldsymbol{S} = 0$. In matrix form, the condition is that $[\boldsymbol{S}]$ is the eigenvector $\mathbf{v}_*$ that corresponds to the smallest non-negative eigenvalue λ_* of the matrix

$$\mathbf{G} = \sum_{i=1}^{n} ([\boldsymbol{P}_i][\boldsymbol{P}_i]^{\mathrm{T}})\mathbf{M} \tag{2.15}$$

where

$$\mathbf{M} = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & -1 \\ 0 & 0 & 0 & -1 & 0 \end{bmatrix} \tag{2.16}$$

is due to the metric properties of the conformal geometric null basis. The sphere is then given by $[\boldsymbol{S}] = \alpha \mathbf{v}_*$, where α is a scalar that is selected to scale the expression for the sphere.

3. Method

The method proposed in [15] generates a set of descriptors based on the following steps: first a set of keypoints are found in both the model point cloud $\boldsymbol{X}$ and the observation point cloud $\boldsymbol{Y}$, $\boldsymbol{X}_{\text{keypoints}}$ and $\boldsymbol{Y}_{\text{keypoints}}$ respectively. This is done by generating a local reference frame at each point using PCA, and selecting the points which have a unique geometry based on their shape factors. A descriptor is then generated at each keypoint in $\boldsymbol{X}_{\text{keypoints}}$ and $\boldsymbol{Y}_{\text{keypoints}}$, and these descriptors are used to find the point correspondences between the keypoints in $\boldsymbol{X}$ and $\boldsymbol{Y}$. Finally, the pose is estimated using the point correspondences from the keypoint matching. This section gives a brief overview of the algorithm as well as describing the new improvements.

3.1. Selection of Keypoints

Instead of solving the 3D-registration problem for all points, it is more beneficial to select only a small set of points. This is to reduce both the complexity and the execution time. These points are called keypoints. One of the way to select these keypoints is by analyzing the local surface area around each point. If the local area is unique compared to the global surface, that point can be selected as a keypoint. To do this, we find the eigenvalues, λ_1, λ_2 and λ_3, and eigenvectors, $\mathbf{v}_1$, $\mathbf{v}_2$ and $\mathbf{v}_3$, from the principal component analysis at each point $\boldsymbol{p}_i$ in Sect. 2.3. From these eigenvalues we generate the shape factors for $\boldsymbol{p}_i$ and use these as a selection criteria for selecting keypoints:

$$|\boldsymbol{N}_i| \geq n_{\min} \quad \text{and} \quad (C_l \geq \delta_l \quad \text{or} \quad C_p \geq \delta_p \text{ or } C_s \geq \delta_s) \tag{3.1}$$

where $n_{\min}$, δ_l, δ_p and δ_s are user-specified keypoint parameters. The parameter $n_{\min}$ defines the minimum of neighbouring points that is required. δ_l, δ_p and δ_s has to be specified based on the shape of the point cloud.

After selecting the points using (3.1), we can define the set of keypoints $\boldsymbol{X}_{\text{keypoints}}$. This is again performed for the points in $\boldsymbol{Y}$ generating the keypoint set $\boldsymbol{Y}_{\text{keypoints}}$.

3.2. Generation of Descriptors

As proposed in [15], two spheres are generated to represent the curvature of the surface at each keypoint. This is done based on the the sphere fitting method presented in Sect. 2.4. First it is noted that the sphere fitting method

of can be modified by the introduction of weighting factors for the points, so that the objective function becomes

$$f(\boldsymbol{S}) = \sum_{i=1}^{n} w_i \frac{(\boldsymbol{S} \cdot \boldsymbol{P}_i)^2}{\boldsymbol{S}^2} \tag{3.2}$$

This has the same effect as modifying the input point data of the minimization problem. The condition for optimality becomes $(\sum_{i=1}^{n} w_i \boldsymbol{P}_i(\boldsymbol{P}_i \cdot \boldsymbol{S})) \wedge \boldsymbol{S} = 0$, which in matrix form is written

$$\mathbf{G} = \sum_{i=1}^{n} w_i([\boldsymbol{P}_i][\boldsymbol{P}_i]^{\mathrm{T}})\mathbf{M} \tag{3.3}$$

For each keypoint $\boldsymbol{p}_i$, weighting is introduced by defining the two planes

$$\boldsymbol{\Pi}_{i1} = \boldsymbol{v}_2 + \boldsymbol{p}_i \cdot \boldsymbol{v}_2 e_\infty, \quad \boldsymbol{\Pi}_{i2} = \boldsymbol{v}_1 + \boldsymbol{p}_i \cdot \boldsymbol{v}_1 e_\infty \tag{3.4}$$

where $\boldsymbol{\Pi}_{i1}$ is the plane spanned by $\mathbf{v}_1$ and $\mathbf{v}_3$, and $\boldsymbol{\Pi}_{i2}$ is the plane spanned by $\mathbf{v}_2$ and $\mathbf{v}_3$. The two planes are used so that the curvature in the direction of $\mathbf{v}_1$ and $\mathbf{v}_2$ can be estimated from

$$\mathbf{G}_{\boldsymbol{P}_{i1}} = \sum_{i=0}^{n} e^{-\gamma|\boldsymbol{P}_i \cdot \boldsymbol{\Pi}_{i1}|}([\boldsymbol{P}_i][\boldsymbol{P}_i]^{\mathrm{T}})\mathbf{M} \tag{3.5}$$

$$\mathbf{G}_{\boldsymbol{P}_{i2}} = \sum_{i=0}^{n} e^{-\gamma|\boldsymbol{P}_i \cdot \boldsymbol{\Pi}_{i2}|}([\boldsymbol{P}_i][\boldsymbol{P}_i]^{\mathrm{T}})\mathbf{M} \tag{3.6}$$

where $|\boldsymbol{P}_i \cdot \boldsymbol{\Pi}_{ij}|$ is the distance from the point $\boldsymbol{p}_i$ to the plane $\boldsymbol{\Pi}_{ij}$, and $\gamma \in \mathbb{R}$ is a weighting parameter. It is seen that the weighting factor in $\mathbf{G}_{\boldsymbol{P}_{ij}}$ is unity when the point $\boldsymbol{p}_i$ is on the plane $\boldsymbol{\Pi}_{ij}$, and that the weight decreases when the distance from the plane to the point increases.

The two spheres

$$[\boldsymbol{S}_{i1}] = \alpha_{i1}\mathbf{v}_{i1*}, \quad [\boldsymbol{S}_{i2}] = \alpha_{i2}\mathbf{v}_{i2*} \tag{3.7}$$

are then found from the eigenvector $\mathbf{v}_{ij*}$ corresponding to the smallest positive eigenvalue λ_{ij*} of the matrix $\mathbf{G}_{\boldsymbol{P}_{ij}}$ for $j = 1, 2$. These two spheres are used to calculate the descriptor of $\boldsymbol{p}_i$. This is then repeated for all points in $\boldsymbol{X}_{\text{keypoints}}$ and $\boldsymbol{Y}_{\text{keypoints}}$. An example of the two spheres is shown in Fig. 3.

3.3. Descriptor for Point Correspondence Estimation

In [15] the descriptor for keypoint $\boldsymbol{p}_i$ was given by $\{\boldsymbol{S}_{i1}, \boldsymbol{S}_{i2}\}$. The point correspondence between $\boldsymbol{X}_{\text{keypoints}}$ and $\boldsymbol{Y}_{\text{keypoints}}$ was established by minimizing the function $g(\boldsymbol{p}_k, \boldsymbol{p}_l)$ with respect to l for each k, where

$$\begin{aligned} g(\boldsymbol{p}_k, \boldsymbol{p}_l) = {} & (r_{k1} - r_{l1})^2 + (d_{k1} - d_{l1})^2 \\ & + (r_{k2} - r_{l2})^2 + (d_{k2} - d_{l2})^2 \end{aligned} \tag{3.8}$$

where r_{k1}, r_{k2}, r_{l1} and r_{l2} are the radii of $\boldsymbol{S}_{k1}$, $\boldsymbol{S}_{k2}$, $\boldsymbol{S}_{l1}$ and $\boldsymbol{S}_{l2}$ respectively, and d_{k1} and d_{k2} are the distances between $\boldsymbol{S}_{k1}$ and $\boldsymbol{S}_{k2}$ and $\boldsymbol{p}_k$, and d_{l1} and d_{l2} are the distances between $\boldsymbol{S}_{l1}$ and $\boldsymbol{S}_{l2}$ and $\boldsymbol{p}_l$.

It was experienced that when the method of [15] was used, it was a problem that when a keypoint is on a surface that is planar or close to planar, the radius of the sphere estimate will be large. Therefore, small variations

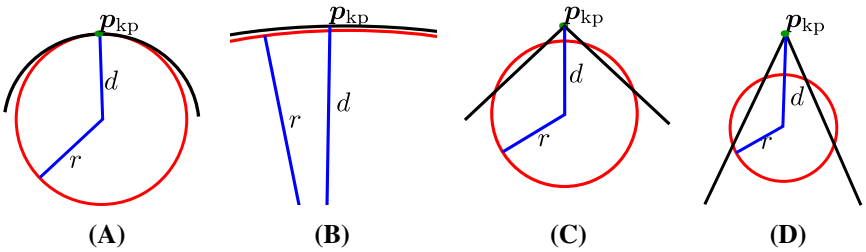


FIGURE 2. A sample of resulting sphere estimates on different surfaces. The different values of r and d indicates what the different shapes are a Sphere estimate of **a** spherical surface $r \approx d$. **b** Sphere estimate of a planar surface $r \approx d \approx \infty$. **c** Sphere estimate of an edge $r < d$. **d** Sphere estimate of a point $r \ll d$

in the point cloud due to noise, had a big impact on the estimate. As a consequence, the value of $(r_{k1} - r_{l1})^2$ and $(r_{k2} - r_{l2})^2$ may be large even if $\boldsymbol{p}_k$ and $\boldsymbol{p}_l$ are corresponding points.

To improve this, we propose a new method. An important factor with this descriptor is the relationship between d and r, as given by the distance $\delta = d - r$ from the point to the sphere, as d is the distance between the keypoint and the center of the sphere, and r is the radius of the sphere. The reason is that if $r \approx d$, then the curvature is spherical or planar, while if $r < d$ the surface has a more angled form, as seen in Fig. 2. The use of δ also cancels out the problem with unstable radii, since r and d are approximately the same length.

We therefore define the descriptor $\boldsymbol{F}_i$ as

$$\boldsymbol{F}_i = (\boldsymbol{S}_{i1} + \boldsymbol{S}_{i2}) \cdot \boldsymbol{P}_i = \frac{1}{2}(\delta_1^2 + \delta_2^2) \tag{3.9}$$

where δ_1 is the distance from the point $\boldsymbol{p}_i$ to sphere $\boldsymbol{S}_{i1}$, and δ_2 is the distance from the point $\boldsymbol{p}_i$ to sphere $\boldsymbol{S}_2$ which encapsulates the relationship between the radius and distance for $\boldsymbol{S}_{i1}$ and $\boldsymbol{S}_{i2}$. To find the point correspondence we solve the minimization problem

$$\min_l g(\boldsymbol{F}_k, \boldsymbol{F}_l), \quad \forall\ \boldsymbol{F}_k \in \boldsymbol{X}_{\text{keypoints}}, \boldsymbol{F}_l \in \boldsymbol{Y}_{\text{keypoints}} \tag{3.10}$$

for each k where

$$g(\boldsymbol{F}_k, \boldsymbol{F}_l) = (\boldsymbol{F}_k - \boldsymbol{F}_l)^2 = \frac{1}{2}(\delta_{k1}^2 - \delta_{l1}^2 + \delta_{k2}^2 - \delta_{l2}^2)^2 \tag{3.11}$$

This point correspondence method is more robust against noise, and therefore also achieves a more accurate point correspondence, which was verified in the experiments.

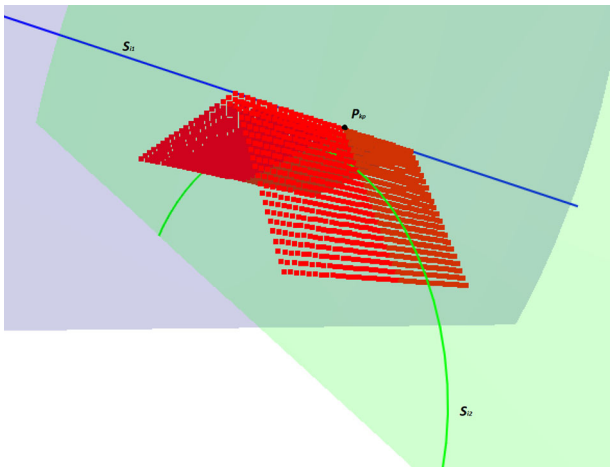


FIGURE 3. An example of a descriptor. The spheres are shown as circles in the figure to make it easier to view. The blue circle of $\boldsymbol{S}_1$ lies on the $\mathbf{v}_1$-$\mathbf{v}_3$ plane (light blue), while the green circle of $\boldsymbol{S}_2$ lies on the $\mathbf{v}_2$-$\mathbf{v}_3$ plane (light green). Note that the green sphere does not intersect with the keypoint $\boldsymbol{p}_{\text{kp}}$ (color figure online)

4. Experiments

There were two conducted experiments. The first compares the method proposed in [15] with the improved version, and the second experiment compares the improved method with state-of-the-art methods in an industrial application.

The first experiment was done with a total of 99 point clouds. Each point cloud was compared to itself, where the first point cloud was positioned at the origin, while the second point cloud was given a known displacement and each point in the point cloud was subjected to Gaussian noise. Since each point cloud were compared to themselves, the true point correspondence is known. This made it possible to compare which of the two versions were the most accurate.

In the second experiment, the improved method was compared with a selection of state-of-the-art methods in where 3D CAD models were compared to a point cloud captured by a 3D-camera. The methods that were compared were the curvature-based and the improved curvature-based method, fast point feature histograms (FPFH) [20], point-pair features (PPF) [10], Signature of Histogram of OrienTation (SHOT) [27] and 3D shape context (3DSC) [11].

4.1. Setup

4.1.1. Hardware. The computer that was used was a desktop computer with an Intel Core i7 7700k Sky Lake at 4.2 GHz with 32GB 2666 MHz DDR4 and

TABLE 1. Results from Experiment 1

	Exact (%)	5 mm (%)	10 mm (%)	20 mm (%)
Curvature-based descriptor	1.25	10.2	32.3	57.5
Improved descriptor	5.15	26.2	47.2	66.3

The results show the average results over all 99 point clouds. The exact column describe the amount of correct point correspondences were made, while the columns labelled with a distance describes how many of the point correspondences had an error below that distance

a EVGA GeForce GTX 1080 Founders Edition graphics card. The computer was running Ubuntu 16.04 LTS.

The point clouds that were taken with a 3D camera, were taken using the Zivid 3D camera provided by ZividLabs [25]. The Zivid 3D camera outputs 2.3 Mpixel RGBD image, with a field of view of 425×267 mm at a distance of 0.6 m with a depth resolution of 0.1 mm at the same distance.

4.1.2. Implementation. All the methods were implemented in the same manner as that described in [15].

The Curvature-based descriptor and the improved version proposed in this paper, were implemented using the versor library [7] together with the Eigen library [12].

FPFH, PPF, SHOT and 3DSC were implemented using the PCL library [22], and was implemented using the sample codes that were provided on their websites, or other supporting websites. The parameters were chosen to be similar to the proposed method to the extent it was possible. The point cloud were first down-sampled using a voxel grid of 1 mm, followed by the normal estimation algorithm. After each method had generated a descriptor, a RANSAC algorithm was performed with 1000 iterations and an inlier threshold of 5 mm.

4.2. Experiment 1

In the first experiment, the point correspondence method from [15] was compared to the method proposed in this paper. The experiment was set up so that each point cloud was compared to itself. A point cloud was set up at the origin, the same point cloud was placed with a known displacement from the origin and with a known noise applied. These two point clouds are then compared. The two methods then generated descriptors for both point clouds and estimated the point correspondences between the two. Since the two point clouds that were compared were the same point cloud, it was possible to know the true point correspondence. Table 1 shows the results of the experiment, and Fig. 4 shows a sample of the point clouds that were used.

The results in Table 1 clearly shows an improvement in the point correspondence accuracy. Not only does the amount of successes improve, but the

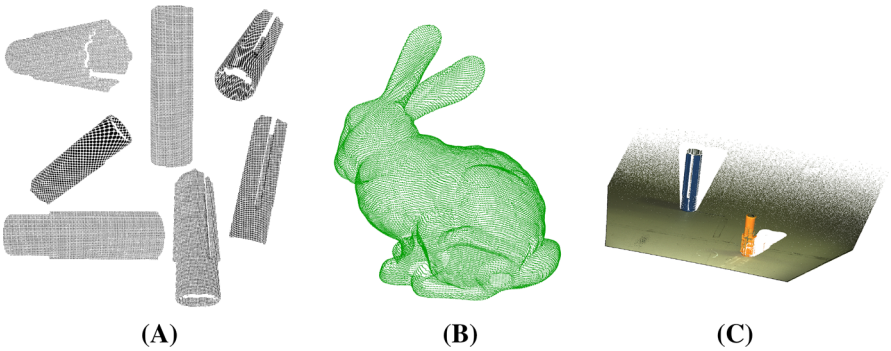


FIGURE 4. A sample of the point clouds that were used in the experiments. **a** Multiple point clouds of a 3D CAD model, from multiple views. **b** 3D point cloud of the Stanford Bunny. **c** A scene taken with the Zivid camera

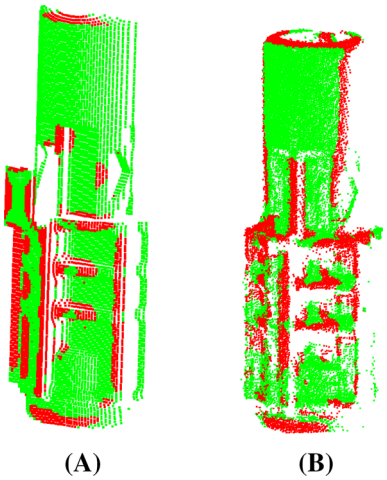


FIGURE 5. Point cloud sampling of the same object from a CAD model and a Zivid 3D camera. The red points are the sampled keypoints. **a** Point cloud sampled from a CAD model. **b** Point cloud sampled from Zivid Camera (color figure online)

overall accuracy of the point correspondences also improve. This improvement shows that the improved method for point correspondences is superior to the one proposed in [15].

4.3. Experiment 2

In the second experiment, both the curvature-based descriptor, the improved version and several state-of-the-art methods were used in a realistic industrial application. A scene, shown in Fig. 4c, where several items were placed on a table. The objects were extracted using [26], which generated a set of point

TABLE 2. Results from Experiment 2

Method	RMS	Execution time (ms)	Comment
CBD	3.8643	21487476	
Improved CBD	3.2355	21448246	
FPFH	3.3755	2390431	Point clouds were flipped upside down
PPF	10.3367	56926563	Estimate were too far away
SHOT	3.5237	30411992	Got a better RMS value on a different point cloud
3DSC	3.3462	163224032	

The RMS shows the overall distance error between the two compared point clouds. The comment describes some of the remarks that was done with the visual inspection of the results

clouds. These point clouds were compared to point clouds generated from CAD models, where the point clouds were sampled from the CAD model from different angles. A sample is shown in Figs. 4a and 5. The point clouds were then compared, in order to estimate the position of the object in the scene.

The methods that were used were the curvature-based descriptor, the improved version, FPFH, SHOT, PPF and 3DSC. The results from the experiments are shown in Table 2. Since the two point clouds are not the same, it is not possible to know the correct point correspondence, nor the correct displacement, so to measure the accuracy of the methods, a root-mean square calculation was done, where the distance between all the points in the CAD point cloud and the closest point in the scene point cloud was measured. This measurement is not sufficient, so a visual comparison was made and the execution time was recorded.

Since the CAD model was sampled from different angles, there were 42 point clouds for each of the two objects on the table. This paper does not focus on recognition, so in order to pick the point cloud which best fits the point cloud from the camera, the match that got the lowest RMS was selected as the best fit. This is not a very accurate method for recognition, but it was sufficient for the experiment. One point cloud was chosen to be the best match, and it was selected through visual inspection.

It is shown in Table 2 that the curvature-based descriptor and the improved version has a better estimate than the other methods, as seen in Fig. 6. The result from the FPFH method had the point clouds flipped upside down, resulting in an angular error of about π. The PPF method failed on the position estimation and had no overlap between the two point clouds. When comparing the CAD point clouds with the scene point cloud, SHOT got lower RMS values on other point clouds than the chosen point cloud.

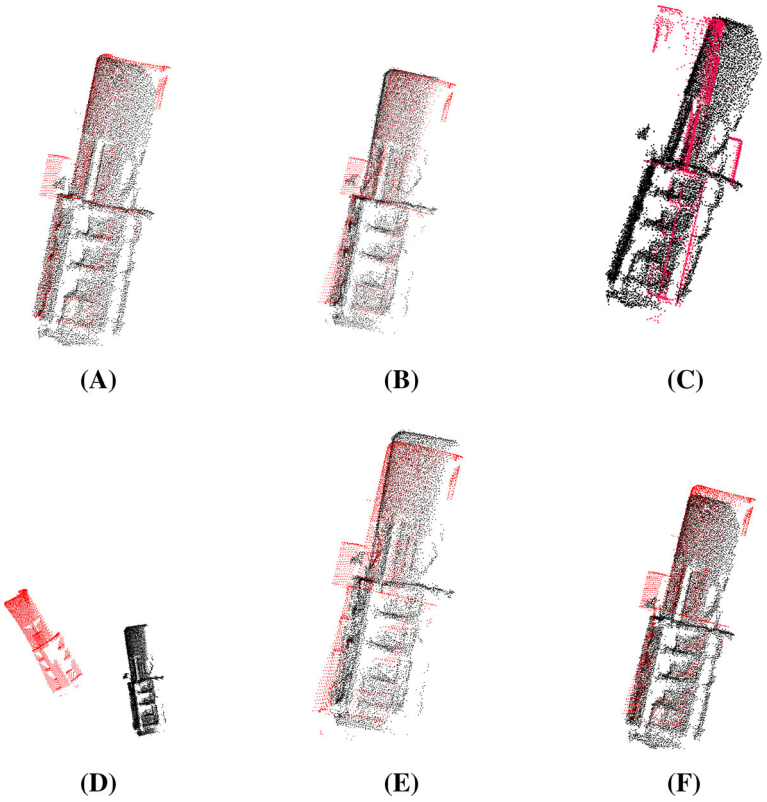


FIGURE 6. A sample of the resulting position estimation with the different descriptor methods. **a** Results from the improved curvature-based descriptor. **b** Results from the curvature-based descriptor. **c** Results from FPFH. **d** Results from PPF. **e** Results from SHOT. **f** Results from 3DSC

This experiments show that the curvature-based descriptor is more accurate on industrial applications, and that the improved version is a further improvement of the curvature-based descriptor descriptor. It is also worth noting that the execution time shown in Table 2 is a little misleading. The curvature-based descriptor is not optimized in regards to execution time, while the methods in the PCL library are to some degree, which means that the execution time of the curvature-based descriptor can be improved.

5. Conclusion

A descriptor for course alignment of point clouds using conformal geometric algebra has been presented in this paper. The method is based on selecting keypoints depending on shape factors, and a descriptor is then calculated for each keypoint by fitting two spheres that describe the local curvature. The method for estimating the point correspondences is an improvement from the

method presented in [15] by a factor of 4.12 in regards to exact matching, 2.57 within 5 mm, 1.46 within 10 mm and 1.15 within 20 mm. The experiments show that the accuracy of the method is an improvement from [15], and has shown to be more accurate than FPFH, PPF, SHOT and 3DSC.

References

[1] Alexander, A.L., Hasan, K., Kindlmann, G., Parker, D.L., Tsuruda, J.S.: A geometric analysis of diffusion tensor measurements of the human brain. Magn. Reson. Med. **44**(2), 283–291 (2000)

[2] Besl, P., McKay, N.: A Method for Registration of 3-D Shapes. IEEE Trans Pattern Anal Mach Intell **14**(2), 239–256 (1992)

[3] Bouaziz, S., Tagliasacchi, A., Pauly, M.: Sparse iterative closest point. Comput. Gr. Forum **32**(5), 113–123 (2013)

[4] Castellani, U., Bartoli, A.: 3D shape registration. 3D Imaging. Anal. Appl. **9781447140**, 221–264 (2014)

[5] Chen, Y., Medioni, G.: Object modeling by registration of multiple range images. In: Proceedingsof the 1991 IEEE International Conference on Robotics and Automation, vol. 10(3), pp. 2724–2729 (1991)

[6] Chua, C.S., Jarvis, R.: Point signatures: a new representation for 3D object recognition. Int. J. Comput. Vis. **25**(1), 63–85 (1997)

[7] Colapinto, P.: Versor: Spatial computing with conformal geometric algebra. Master's thesis, University of California at Santa Barbara (2011). http://versor.mat.ucsb.edu

[8] Dorst, L.: Total least squares fitting of k-spheres in n-D Euclidean space using an (n+2)-D isometric representation. J. Math. Imaging Vis. **50**(3), 214–234 (2014)

[9] Dorst, L., Fontijne, D., Mann, S.: Geometric Algebra for Computer Science: An Object-Oriented Approach to Geometry. Morgan Kaufmann Publishers Inc., San Francisco (2009)

[10] Drost, B., Ulrich, M., Navab, N., Ilic, S.: Model globally, match locally: Efficient and robust 3D object recognition. In: Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition, pp. 998–1005 (2010)

[11] Frome, A., Huber, D., Kolluri, R., Bülow, T., Malik, J.: recognizing objects in range data using regional point descriptors. In: European Conference on Computer Vision, vol. 3023, pp. 224–237 (2004)

[12] Guennebaud, G., Jacob, B., et al.: Eigen v3. http://eigen.tuxfamily.org (2010). Accessed 13 Nov 2017

[13] Hildenbrand, D.: Foundations of Geometric Algebra Computing. Geometry and Computing, vol. 8. Springer, Berlin (2013)

[14] Johnson, A.E., Hebert, M.: Using spin images for efficient object recognition in cluttered 3D scenes. IEEE Trans. Pattern Anal. Mach. Intell. **21**(5), 433–449 (1999)

[15] Kleppe, A., Tingelstad, L., Egeland, O.: Course alignment for model fitting of point clouds using a curvature-based descriptor. IEEE Trans. Autom. Sci. Eng. (2017) (**accepted**)

[16] Li, H., Hartley, R.: The 3D–3D registration problem revisited. In: Proceedings of the IEEE International Conference on Computer Vision (2007)

[17] Pratt, V.: Direct least-squares fitting of algebraic surfaces. ACM SIGGRAPH Comput. Gr. **21**(4), 145–152 (1987)

[18] Ritter, M., Benger, W., Cosenza, B., Pullman, K., Moritsch, H., Leimer, W.: Visual data mining using the point distribution tensor. In: ICONS, pp. 10–13 (2012)

[19] Rusinkiewicz, S., Levoy, M.: Efficient variants of the ICP algorithm. In: Proceedings of International Conference on 3-D Digital Imaging and Modeling, 3DIM, pp. 45–152 (2001)

[20] Rusu, R.B., Blodow, N., Beetz, M.: Fast point feature histograms (FPFH) for 3D registration. In: IEEE International Conference on Robotics and Automation, pp. 3212–3217 (2009)

[21] Rusu, R.B., Bradski, G., Thibaux, R., Hsu, J.: Fast 3D recognition and pose using the viewpoint feature histogram. In: Intelligent Robots and Systems (IROS), pp. 2155–2162 (2010)

[22] Rusu, R.B., Cousins, S.: 3D is here: point cloud library. In: IEEE International Conference on Robotics and Automation, pp. 1 – 4 (2011)

[23] Sabata, B., Aggarwal, J.K.: Surface correspondence and motion computation from a pair of range images. Comput. Vis. Image Underst. **63**(2), 232–250 (1996)

[24] Salvi, J., Matabosch, C., Fofi, D., Forest, J.: A review of recent range image registration methods with accuracy evaluation. Image Vis. Comput. **25**(5), 578–596 (2007)

[25] Skotheim, Ø., Schumann-Olsen, H., et. al.: ZividLabs. Zivid 3d camera

[26] Sveier, A., Kleppe, A.L., Tingelstad, L., Egeland, O.: Object detection in point clouds using conformal geometric algebra. Adv. Appl. Clifford Algebras **27**(3), 1–16 (2017)

[27] Tombari, F., Salti, S., Di Stefano, L.: Unique signatures of histograms for local surface description. In: Lecture Notes in Computer Science (Including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics), 6313 LNCS(PART 3), pp. 356–369 (2010)

[28] Yang, J., Li, H., Jia, Y.: Go-ICP: Solving 3D registration efficiently and globally optimally. In: Proceedings of the IEEE International Conference on Computer Vision, pp. 1457–1464 (2013)

Adam Leon Kleppe and Olav Egeland
Department of Mechanical and Industrial Engineering
Norwegian University of Science and Technology (NTNU)
7491 Trondheim
Norway
e-mail: adam.l.kleppe@ntnu.no

Olav Egeland
e-mail: olav.egeland@ntnu.no

Received: February 13, 2018.
Accepted: April 19, 2018.