Available online at www.sciencedirect.com
SCIENCE DIRECT
Image and Vision Computing 23 (2005) 299–309
image AND vision COMPUTING
www.elsevier.com/locate/imavis

Robust Euclidean alignment of 3D point sets: the trimmed iterative closest point algorithm

Dmitry Chetverikov[a,*,1], Dmitry Stepanov[a], Pavel Krsek[b,2]

[a]*Computer and Automation Research Institute, Eötvös Loránd University, Kende u. 13-17, H-1111 Budapest, Hungary*
[b]*Center for Applied Cybernetics, FEE, Czech Technical University, 166 27 Prague 6, Technicka 2, Czech Republic*

Received 9 June 2002; received in revised form 11 March 2004; accepted 18 May 2004

Abstract

The problem of geometric alignment of two roughly pre-registered, partially overlapping, rigid, noisy 3D point sets is considered. A new natural and simple, robustified extension of the popular Iterative Closest Point (ICP) algorithm [IEEE Trans. Pattern Anal. Machine Intell. 14 (1992) 239] is presented, called Trimmed ICP (TrICP). The new algorithm is based on the consistent use of the Least Trimmed Squares approach in all phases of the operation. Convergence is proved and an efficient implementation is discussed. TrICP is fast, applicable to overlaps under 50%, robust to erroneous and incomplete measurements, and has easy-to-set parameters. ICP is a special case of TrICP when the overlap parameter is 100%. Results of a performance evaluation study on the SQUID database of 1100 shapes are presented. The tests compare TrICP and the Iterative Closest Reciprocal Point algorithm [Fifth International Conference on Computer Vision, 1995].

Keywords: Registration; Point sets; Iterative closest point; Least trimmed squares; Robustness

1. Introduction

In this study, we address the problem of Euclidean alignment of two roughly pre-registered, partially overlapping, 3D point sets in presence of noise and outliers. This problem has been considered in 3D model acquisition (reverse engineering, scene reconstruction) and motion analysis, including model-based tracking. (See Ref. [3] for an overview of recent applications.) Given two 3D point sets, $\mathscr{P}$ and $\mathscr{M}$, the task is to find the Euclidean motion that brings $\mathscr{P}$ into the best possible alignment with $\mathscr{M}$.

The Iterative Closest Point (ICP) algorithm proposed by Besl and McKay [1] is a standard solution to the alignment problem. ICP has three basic steps: (1) pair each point of $\mathscr{P}$ to the closest point in $\mathscr{M}$; (2) compute the motion that minimises mean square error (MSE) between the paired points; (3) apply the optimal motion to $\mathscr{P}$ and update MSE. The three steps are iterated; the iterations have been proved to converge in terms of MSE.

Independently, Chen and Medioni [4] published a similar iterative scheme using a different pairing procedure based on surface normal vector. This formulation is applicable to points on relatively smooth surfaces. When surfaces are deteriorated by substantial noise, the normals become unreliable. In this paper, we consider the formulation by Besl and McKay which is applicable to noisy volumetric and surface measurements.

The idea of ICP proved very fruitful as it was followed by numerous applications, improvements and modifications. A comprehensive survey oriented towards range images can be found in the PhD thesis by Pulli [5]. Rusinkiewicz and Levoy [6] give a fresh update of the variants of ICP algorithm. They classify the variants according to the way the algorithms: (1) select subsets of $\mathscr{P}$ and $\mathscr{M}$; (2) match (pair) points; (3) weight the pairs; (4) reject some pairs;(5) assign error metric; (6) minimise the error metric.

Selection usually refers to random sampling of points when using a Monte-Carlo technique, such as the Least Median of Squares (LMedS) [3,7]. Pairs can be weighed or rejected based on the distribution of distances [8] or some geometric constraints [2]. Different cost functions and minimisation procedures are applied. For example,

* Corresponding author. Tel.: +36-1-2096510; fax: +36-1-1667503.
E-mail address: csetverikov@sztaki.hu (D. Chetverikov).
[1] Supported by the EU grant ICA1-CT-2000-70002: MIRACLE—Centre of Excellence and by the Hungarian Scientific Research Fund (OTKA) under grants M28078 and T038355.
[2] Supported by the Czech Ministry of Education under project LN00B096.

0262-8856/$ - see front matter © 2004 Elsevier B.V. All rights reserved.
doi:10.1016/j.imavis.2004.05.007

a recent paper by Fitzgibbon [9] presents an attempt of minimisation of the MSE cost function using the Levenberg-Marquardt algorithm.

These modifications of ICP seek to improve robustness, speed and precision. The most critical issue is probably that of robustness, as the original algorithm assumes outlier-free data and $\mathscr{P}$ being a subset of $\mathscr{M}$, in the sense that each point of $\mathscr{P}$ has a valid correspondence in $\mathscr{M}$. Numerous attempts have been made to robustify ICP by rejecting wrong pairs. In particular, robust statistics have been applied, such as LMedS or the Least Trimmed Squares (LTS) [5,10].

Pajdla and Van Gool [2,11] proposed the Iterative Closest Reciprocal Point (ICRP) algorithm that exploits the ϵ-reciprocal correspondence: given a point $\mathbf{p} \in \mathscr{P}$ and the closest point $\mathbf{m} \in \mathscr{M}$, $\mathbf{m}$ is back-projected onto $\mathscr{P}$ by finding the closest point $\mathbf{p}' \in \mathscr{P}$. If $\|\mathbf{p} - \mathbf{p}'\| > \epsilon$, the pair $(\mathbf{p}, \mathbf{m})$ is rejected.

Often, different heuristics are combined, making the resulting ICP-variant efficient in cases when the underlying assumptions are met. Such heterogeneous combinations are difficult to analyse; in particular, convergence properties remain unclear.

Computational efficiency is another important issue, since some applications require fast real-time operation for medium-size datasets, such as range images [5]. Various data structures, like k-D tree [12] or spatial bins [8], are used to facilitate search of the closest point. To speed up the convergence, normal vectors are considered, which is mainly helpful in the beginning of the iteration process [5].

In this paper, we concentrate on the issue of robustness and address the following question: *can one make ICP robust while preserving its structure and convergence*? A new robustified extension of ICP is presented, called Trimmed ICP (TrICP). The new algorithm is based on the consistent use of the LTS approach in all phases of the operation. LTS [13,14] means sorting the squared errors (residuals) in increasing order and minimising the sum of a certain number of smaller values. LMedS [3,7] minimises the median, that is, the value in the middle of the sorted sequence; LTS 50% minimises the sum up to the median. (See Section 2 for precise definitions and a brief discussion of the two methods.)

Previously, LTS has only been used in the context of randomised, Monte-Carlo type initial estimation of alignment parameters [5], following the guidelines of the standard approach [7] to robust regression and outlier detection. In this approach, model parameters are repeatedly estimated as random samples are drawn whose size is sufficient for the estimation. After the initial estimation, outliers are detected and rejected, and the final least squares solution is obtained for inliers only.

LTS is preferred to LMedS because it has better convergence rate and a smoother objective function [14]. However, as robust statistics in the context of a randomised approach, LTS and LMedS have the same breakdown point of 50%. This means that the overlap between the two point sets has to exceed 50%.

Our basic observation is that LTS fits the original scheme of ICP without any significant modification. At each step of iteration, the optimal motion can be computed for trimmed squares in exactly the same way as it is done in ICP for all squares. (The median of squares does not facilitate this computation, rendering the LMedS variant [3] inapplicable to large point sets.) At the same time, trimming the squares makes the algorithm robust in the original deterministic framework, without randomisation. The resulting algorithm, TrICP, is applicable to overlaps under 50%. As no additional heuristics are used, the convergence of the algorithm is easy to prove.

The paper is organised as follows. In Section 2, we give definitions of the LMedS and the LTS and briefly compare the two methods in relation to surface alignment and robust regression where these methods have originally been introduced. In Section 3, we formulate the problem and present the new algorithm. Relevant details of its implementation are given in Section 4; the convergence is proved in Section 5. Results of tests are shown in Section 6. The tests include a systematic comparative performance evaluation of the proposed method and the ICRP algorithm [2]. Finally, in Section 7 we give a brief critical analysis of the algorithms and outline possible ways to improve their performance.

2. The least median of squares and the least trimmed squares

The two methods, the LMedS and the LTS, were introduced by Rousseeuw [13]to robustify the standard Least (Mean) Squares technique, that is, to make the linear regression insensitive to *outliers*. Regression outliers are defined as observations (measurements) that do not comply with the linear pattern formed by the majority of the data [14]. (The discussion below is based on this paper. See also Ref. [15] for an early but still useful survey of robust regression in computer vision.)

To better understand the differences between the LMedS and the LTS, let us summarise the two methods in the context of the linear regression model

$$y_i = \beta_0 + \beta_1 x_{i1} + \cdots + \beta_m x_{im} + e_i, \tag{1}$$

for $i = 1, \ldots, n$. Here y_i are the response variables, x_{i1} to x_{im} the explanatory variables, β_j for $j = 0, \ldots, m$ the coefficients (the model parameters).

In the classical regression theory the errors e_i are assumed to have a Gaussian distribution with zero mean. The standard Least (Mean) Squares method computes the parameters $\hat{\beta}_j$ such that the sum of squares of the residuals $r_i(\hat{\beta}_0, \ldots, \hat{\beta}_m)$ is minimal:

$$\underset{(\hat{\beta}_0,\ldots,\hat{\beta}_m)}{\text{minimise}} \sum_{i=1}^{n} r_i^2. \tag{2}$$

(For simplicity of notation, we omit the parameters in $r_i(\hat{\beta}_0, \ldots, \hat{\beta}_m)$.)

A simple but useful measure of robustness is the *breakdown point*, also called the breakdown value. Assume that the input data to the model (1) is a mixture of inliers and outliers. The inliers obey the model, the outliers do not. The fraction of the outliers in the mixture is ϵ, the fraction of the inliers is $\theta = 1 - \epsilon$. The outliers can be generated by any distribution, or can even be deterministic. The deviation of an outlier from the model can be arbitrarily large.

Now, consider a statistical estimator of the model parameters β_j. Apply the estimator to the inliers and obtain an unbiased estimate of the parameters. Then contaminate the input data by adding the outliers. The smallest fraction ϵ^* that can cause the estimator to run arbitrarily far from the original unbiased estimate is called the breakdown point of the estimator. To be able to estimate the original parameters, we need $\epsilon < \epsilon^*$, or, equivalently, $\theta > \theta^*$.

It is obvious that the Least Squares method (2) is not robust, since a single outlier can destroy the estimator: $\epsilon^* = \frac{1}{n} \to 0$ as $n \to \infty$. The LMedS replaces the mean by the median of the residuals

$$\underset{(\hat{\beta}_0,\ldots,\hat{\beta}_m)}{\text{minimise}} \ \underset{i=1}{\overset{n}{\text{med}}} r_i^2. \tag{3}$$

The breakdown point of the LMedS is 50%. In other words, up to a half of the input data may be contaminated: until the inliers form a majority, the median is selected from the inliers.

A similar effect can be achieved by the LTS method defined by

$$\underset{(\hat{\beta}_0,\ldots,\hat{\beta}_m)}{\text{minimise}} \ \sum_{i=1}^{h} (r_i^2)_{i:n}, \tag{4}$$

where $(r_i^2)_{1:n} \le (r_i^2)_{2:n} \le \cdots \le (r_i^2)_{n:n}$ are the ordered squared residuals. $1 \le h \le n$ is the parameter that can be tuned to discard the outliers depending on the contamination of the data. When $h = n$, all residuals are taken into account; the method is equivalent to the non-robust Least Squares (2). Setting $h \approx n/2$ we have $\epsilon^* \approx 0.5$ (50%); for larger h we obtain $\epsilon^* \approx (n - h)/n$.

It should be emphasised that the goal of the LMedS and LTS *is not* just to 'throw away' a portion of data. The majority fit is used to detect and discard the outliers while selecting as many inliers as possible for another, more accurate fit. It may happen that the actual level of data contamination is less than originally expected (0.5 for LMedS, $(n - h)/n$ for LTS). Both methods will then try to use all useful data.

Let us now discuss the two robust statistics in the context of the alignment problem. Following the trends in robust regression, the LMedS has been used in computer vision and image analysis more frequently than the LTS. (The early survey [15] on robust regression in computer vision does not mention the LTS at all.) In particular, the LMedS has already been used to design a robust variant of the ICP algorithm [3]. What are the advantages of the LTS that make it much more suitable for this purpose than the LMedS? To answer this question, we have to consider the statistical properties of the two estimators, the nature of the alignment problem and the structure of the ICP algorithm.

In robust regression, a traditional reason to prefer the LMedS to the LTS was the assumed higher computational efficiency of the former. Indeed, one can find the median without sorting all of the numbers. The trimmed squares were believed to need more sorting. Later, fast implementations of the LTS [14] were designed; the sorting time is not a critical issue anymore. In particular, sorting the residuals is not an issue in the ICP-like algorithms, where the computation of the residuals takes much more time: finding the closest point is the most time-consuming operation.

With the development of the theory and practice of robust regression, the attention switched to the statistical properties of the methods, and this is where the LTS is superior. Currently, the author of the compared methods prefers the LTS [14] because: (1) its objective function is smoother, making the LTS less sensitive to local effects; (2) its statistical efficiency, in the sense of precision, is better. LTS has a higher convergence rate, which makes it more suitable for an iterative estimator.

Intuitively, these advantages of the LTS over the LMedS can be explained by the simple fact that the median outputs a single observation (although implicitly supported by a majority), while the trimmed squares explicitly integrate a large number of observations. Recently, the advantages of the LTS have been recognised within the computer vision community [16]; more examples of LTS applications in computer vision are given in Ref. [14].

Turning to the surface alignment problem, we see that this problem differs from the regression problem in several aspects. A critical theoretical aspect is the nature of the outliers: here, the outliers are (1) the correctly measured points that have no pair in the other set and (2) the incorrectly measured points. The former are deterministic and lie on the non-overlapping parts of surfaces to be matched; the latter are indeterministic and can appear in any part of the data. Normally, there are much more deterministic than indeterministic outliers in the data. (In the regression problem, the situation is usually just the opposite.)

The median is efficient in discarding strongly deviating outliers. However, the deterministic outliers are not necessarily strong outliers: the spectrum of residuals is often continuous. The efficiency of the median in such situation is questionable. The trimmed squares seem to better fit the nature of the surface alignment problem.

The final argument in favour of the LTS is computational rather than statistical. As it will be shown below, the LTS perfectly fits the framework of the ICP algorithm, while the LMedS does not. The LMedS-based robustified ICP [3] is slow and inefficient; its convergence properties are hard to analyse. The LTS-based variant proposed in this paper is

much faster, more efficient, and it inherits the provable convergence of the original algorithm.

3. The new algorithm

Following the notation of Ref. [3], consider two sets of 3D points to align: the *data* set $\mathscr{P} = \{\mathbf{p}_i\}_1^{N_p}$ and the *model* set $\mathscr{M} = \{\mathbf{m}_i\}_1^{N_m}$. Usually, the number of points in the two sets are different: $N_p \neq N_m$. A large portion of the data points may have no correspondence in the model set. Assume a minimum guaranteed rate of the data points that can be paired is known; we will call this rate the *minimum overlap* and denote it by ξ. Then, the number of the data points that can be paired is $N_{\text{po}} = \xi N_p$.

If the value of ξ is unknown, one can run TrICP several times and select a result that combines a good MSE with the highest possible overlap. A procedure for automatic setting of ξ is given in Section 4.

Like most iterative algorithms, including ICP, our algorithm assumes that $\mathscr{P}$ and $\mathscr{M}$ have been *roughly pre-registered*, either manually or automatically. This can be done, for example, by aligning a few characteristic points or, in a controlled measurement setup, by calculating the sensor motion between the two views. It should be emphasised, however, that the initial alignment can be fairly rough: TrICP has been successfully applied to initial relative rotations of up to 20°.

Also, it is assumed that the overlapping part of the two sets is characteristic enough to allow for unambiguous matching. In particular, this part should not be symmetric and 'featureless'. This assumption is typical for most point set registration algorithms. A possible way to cope with this problem is discussed in Section 4.

Under these assumptions, the problem is to find the Euclidean transformation that brings an N_{po}-point subset of $\mathscr{P}$ into the best possible alignment with $\mathscr{M}$. For an Euclidean motion with rotation matrix $\mathbf{R}$ and translation vector $\mathbf{t}$, denote the transformed points of the data set by

$$\mathbf{p}_i(\mathbf{R},\mathbf{t}) = \mathbf{R}\mathbf{p}_i + \mathbf{t}, \qquad \mathscr{P}(\mathbf{R},\mathbf{t}) = \{\mathbf{p}_i(\mathbf{R},\mathbf{t})\}_1^{N_p}. \tag{5}$$

Define the *individual distance* from a data point $\mathbf{p}_i(\mathbf{R},\mathbf{t})$ to the model set $\mathscr{M}$ as the distance to the closest point of $\mathscr{M}$:

$$\mathbf{m}_{\text{cl}}(i,\mathbf{R},\mathbf{t}) = \arg\min_{\mathbf{m}\in\mathscr{M}} \|\mathbf{m} - \mathbf{p}_i(\mathbf{R},\mathbf{t})\|, \tag{6}$$

$$d_i(\mathbf{R},\mathbf{t}) = \|\mathbf{m}_{\text{cl}}(i,\mathbf{R},\mathbf{t}) - \mathbf{p}_i(\mathbf{R},\mathbf{t})\|. \tag{7}$$

We wish to find the motion $(\mathbf{R},\mathbf{t})$ that minimises the sum of the *least* N_{po} squared individual distances $d_i^2(\mathbf{R},\mathbf{t})$.

The conventional ICP algorithm assumes that all data points can be paired: $\xi = 1$ and $N_{\text{po}} = N_p$. TrICP provides a smooth transition to ICP as $\xi \to 1$.

The structure of TrICP is similar to that of ICP. The basic idea of TrICP is to consistently use the LTS in all major aspects of operation: to cope with outliers and incomplete data; to estimate the optimal transformation at each iteration step; and to form the global cost function which is minimised. The main steps of the algorithm are described below. These steps are iterated until any of the stopping conditions described below is satisfied. The iterations are started with the previous sum of trimmed squares $S'_{\text{TS}} = \text{huge\_number}$.

Algorithm 1. Trimmed iterative closest point

(1) For each point of $\mathscr{P}$, find the closest point in $\mathscr{M}$ and compute the individual distances d_i^2 (Eq. (7)).
(2) Sort d_i^2 in the increasing order, select the N_{po} least values and calculate their sum S_{TS}.
(3) If any of the stopping conditions is satisfied, exit; otherwise, set $S'_{\text{TS}} = S_{\text{TS}}$ and continue.
(4) Compute for the N_{po} selected pairs the optimal motion $(\mathbf{R},\mathbf{t})$ that minimises S_{TS}.
(5) Transform $\mathscr{P}$ according to $(\mathbf{R},\mathbf{t})$ (Eq. (5)) and go to 1.

We use the standard stopping conditions [3] related to the number of iterations N_{iter} and MSE for the N_{po} selected pairs:

(1) the maximum allowed N_{iter} has been reached, or
(2) the *trimmed* MSE

$$e = \frac{S_{\text{TS}}}{N_{\text{po}}}, \tag{8}$$

is sufficiently small, or
(3) change of trimmed MSE $|e' - e|$ is sufficiently small, where e' is the previous value.

4. Implementation details

4.1. The basic algorithm

Like any variant of ICP, a fast implementation of TrICP needs an efficient data structure supporting the closest point search. In step 1, we use a simple boxing structure [17] that partitions the space into uniform boxes, cubes. Given a point in space, only the box containing this point and the adjacent boxes are to be considered during the search. The box size is updated as the two sets get closer. (The updating is discussed later in relation to step 5.)

The heap sort [18] is used to efficiently sort the squared individual distances in step 2. Denote by $\{d^2_{i:N_p}\}_1^{N_p}$ the sorted distances:

$$d^2_{1:N_p} \le d^2_{2:N_p} \le \cdots \le d^2_{N_{\text{po}}:N_p} \le \cdots \le d^2_{N_p:N_p}.$$

The sum of the trimmed squares is calculated as

$$S_{\text{TS}} = \sum_{i=1}^{N_{\text{po}}} d^2_{i:N_p}. \tag{9}$$

(Compare to Eq. (4).) Initially, S'_{TS} is set to a huge value to avoid occasional stopping at the first iteration because of small relative change.

The optimal motion $(\mathbf{R}, \mathbf{t})$ in step 4 is computed by the unit quaternion method due to Horn [19]. The same method was used in the original version of ICP [1]. There are different analytical ways to calculate the 3D rigid motion that minimises the sum of the squared distances between the corresponding points. In Ref. [19], four such techniques were compared and the unit quaternion method was found to be robust with respect to noise, stable in presence of degenerate data ('flat' point sets) and relatively fast.

In step 5, $\mathscr{P}$ is transformed according to the optimal motion. The transformation cannot increase the sum of the N_{po} least squared distances $\{d^2_{i:N_p}\}_1^{N_{\text{po}}}$. However, some of the individual distances, including the largest one, may occasionally increase. TrICP keeps track of the largest distance $d_{\max}$ and updates it after the transformation in step 5.

$d_{\max}$ falls sharply during the first few iterations. It is used to adaptively set the box size in step 1: at kth iteration, we set $D_{\text{box}}(k) = d_{\max}(k-1)$. This speeds up the search while ensuring that the N_{po} least distances will be obtained. Indeed, all the N_{po} pairs of the previous step will be found. If any other pair has to enter the list of the N_{po} least distances, then for this pair $d_i < d_{\max}(k-1)$. Therefore, the pair will be found with the box size $d_{\max}(k-1)$.

The box size and the search space normally decrease as the two sets get closer. The initial size $D_{\text{box}} \doteq D_{\text{box}}(0)$ is a parameter of TrICP specifying the largest possible distance between the corresponding points. This parameter depends on dimensions and initial orientations of the two sets. Updating the box size is especially efficient at the beginning of the iterations.

4.2. Finding optimal overlap and setting other parameters

When the value of the overlap parameter ξ is unknown, we set it automatically by minimising the objective function

$$\psi(\xi) = \frac{e(\xi)}{\xi^{1+\lambda}}, \tag{10}$$

where $\lambda \geq 0$ is a preset parameter. (In the tests described in Section 6, we used $\lambda = 2$.) $\psi(\xi)$ minimises the trimmed MSE $e(\xi)$ while trying to use as many points as possible. Increasing λ, one can attempt to avoid undesirable alignments of symmetric and/or 'featureless' parts of the two sets.

Typical idealised shapes of the objective functions $e(\xi)$ and $\psi(\xi)$ are shown in Fig. 1. The functions are smooth; they start to increase drastically as ξ exceeds the actual overlap ξ_a. The processing time also tends to increase because of the points that cannot be paired. Usually, $\psi(\xi)$ has a distinct single minimum while $e(\xi)$ has none, or its minimum is less distinct.

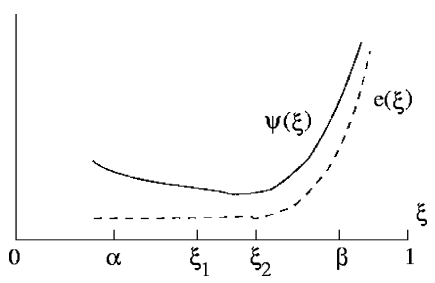

Fig. 1. Typical shapes of the objective functions $e(\xi)$ and $\psi(\xi)$.

The minimisation procedure is given a search interval $[\alpha, \beta]$. It assumes that in this interval $\psi(\xi)$ has a single minimum and locates the minimum by iterative bracketing with the Golden Section Search algorithm [18]. The algorithm has been modified so as to avoid unnecessary computations of $\psi(\xi)$ for large values of ξ, especially when $\xi > \xi_a$. The reason is that in our case, calculating the objective function is much more time-consuming than calculating the Golden Section and comparing the function values. Also, computing $\psi(\xi)$ is usually faster for smaller ξ. The Golden Section Search algorithm is described in Ref. [18]. Our modification is as follows.

At the first iteration, the Golden Section Search splits the interval $[\alpha, \beta]$ into three pieces $\alpha < \xi_1 < \xi_2 < \beta$

$$\xi_1 = \alpha + w(\beta - \alpha),$$

$$\xi_2 = \beta - w(\beta - \alpha),$$

where the fraction $w = (3 - \sqrt{5})/2 \approx 0.38197$.

To speed up the computation, we first obtain $\psi(\alpha)$ and $\psi(\xi_1)$. If $\psi(\alpha) < \psi(\xi_1)$, the search interval is reduced to $[\alpha, \xi_1]$. Otherwise, $\psi(\xi_2)$ is also obtained. If $\psi(\xi_1) < \psi(\xi_2)$, values $\xi > \xi_2$ are not considered. These constraints can be applied at any iteration.

By default, the minimum of $\psi(\xi)$ is searched in the interval $[0.4, 1.0]$, which is a typical range of overlaps. Specifying the interval more strictly improves the accuracy and the computational efficiency of the method. In any case, about 10 iterations are usually sufficient to locate the minimum with an acceptable precision of 0.01, that is, 1%.

Other adjustable parameters of TrICP are the parameters of the stopping conditions and the initial box size D_{box}. Given the convergence of the algorithm, setting the stopping conditions is an easy task. The initial box size must be set so as to find all corresponding points at the initial rotation. In other words, it must exceed the maximum distance between two corresponding points. On the other hand, to speed up the operation it is advisable to set D_{box} as close to the maximum distance as possible. In a sense, one can trade safety for speed.

5. Convergence

Many previous attempts to robustify ICP used some additional geometric or statistical heuristics, which were not mathematically coherent with the original idea. TrICP incorporates the robust LTS statistics in a way compatible with the philosophy and data structure of ICP. An important advantage of this natural extension is that convergence of

TrICP can be proved similar to ICP. Besl and McKay [1] prove that the ICP algorithm always converges monotonically to a local minimum with respect to the MSE cost function. They do this by showing that the sequence of MSE values is nonincreasing and bounded below. With TrICP, the situation is exactly the same. The following theorem is valid.

Theorem. *The Trimmed Iterative Closest Point algorithm always converges monotonically to a local minimum with respect to the trimmed mean-square distance objective function* (8).

Proof. An iteration of TrICP includes three basic operations in the following order: (a) the optimal motion is computed, (b) the closest points are found and (c) the N_{po} least distances are selected. (In the first iteration, steps 1 and 2 just initialise trimmed MSE.)
Consider the algorithm after the kth iteration. Let $e(k)$ be trimmed MSE before the optimal motion. The optimal motion does not increase S_{TS} [1]: if it did it would be inferior to the identity transformation, loosing its optimality. Therefore, $0 \leq e_a(k+1) \leq e(k)$.
Operations (b) and (c) can also modify the trimmed squares. Consider the list $\mathscr{L}$ of the N_{po} pairs forming $e_a(k+1)$. View this list as the starting point for updating the trimmed squares. Updating the closest points for those data points that are in $\mathscr{L}$ does not increase the MSE of $\mathscr{L}$, since no individual distance increases: $0 \leq e_b(k+1) \leq e_a(k+1) \leq e(k)$.
Now, consider those points of $\mathscr{P}$ which are not in $\mathscr{L}$ When selecting the N_{po} least distances, any of these points can only enter $\mathscr{L}$ if its pair substitutes in $\mathscr{L}$ a pair with a larger d_i^2. Consequently, the sum of the N_{po} least squares cannot increase:

$$0 \leq e(k+1) \doteq e_c(k+1) \leq e_b(k+1) \leq e_a(k+1) \leq e(k).$$

The sequence of trimmed MSE values is nonincreasing and bounded below by zero. This proves convergence of TrICP to a local minimum. □

Convergence to global minimum depends on the starting point. To avoid local minima, ICP is usually run several times at different conditions. Varying ξ we also run TrICP at different conditions and select the best result.

Note. In some cases, it may be preferable to treat the model set $\mathscr{M}$ as a surface and use the distance between a point in the data set and that surface. (For example, by triangulating $\mathscr{M}$ and finding the distance to the closest triangle.) The convergence will be valid in this case as well, as it is valid for *any* sampling of the surface. Roughly speaking, a continuous surface can be viewed as a point set resulting from infinitely dense sampling.

6. Tests

In this section, we show examples of operation of TrICP on real measured 3D data. Then results of a performance evaluation study on the SQUID database [20] of 1100 shapes are presented.

6.1. Examples of operation

Fig. 2 compares ICP and TrICP in aligning two partially overlapping and differently rotated measurements of Frog. Each of the two sets has about 3000 points. Some numerical results are shown in Table 1, including number of iterations and the execution time on a 1.6 GHz PC. TrICP alignment is better and faster. (Note: as ICP is a special case of TrICP, the same program was run with different values of ξ.)

In this case, the parameter ξ was set manually. The automatic setting procedure based on the minimisation of the objective function (10) yields the optimal value $\xi = 67\%$ and an MSE very close to that of Table 1, but needs more processing time, 15 s.

All further TrICP results presented below were obtained with the automatic setting of the overlap parameter. Fig. 3 illustrates the process of alignment of four laser scanner measurements of an industrial part, made from different positions. The first dataset is the basic measurement made from the top. This point set overlaps with the other three, which were made from different sides.

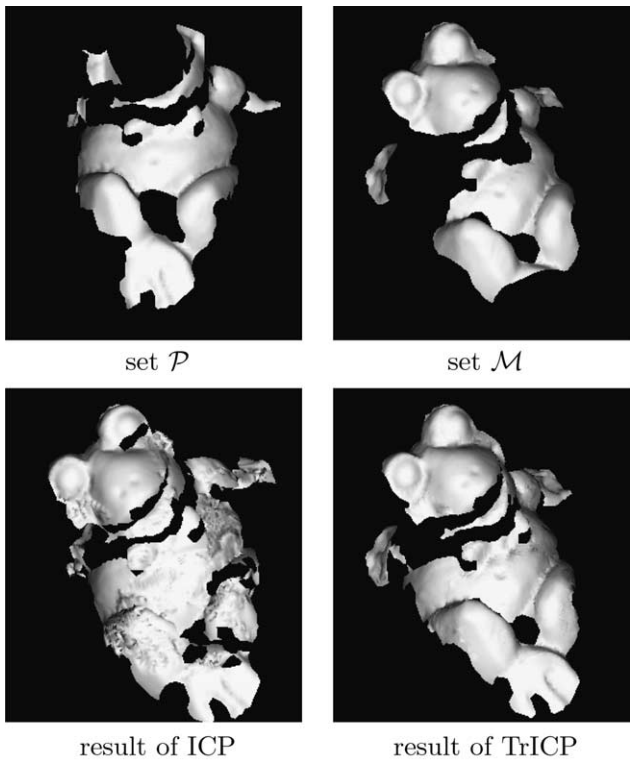

Fig. 2. Aligning two measurements of Frog.

Table 1
Numerical results for Frog data

Method	N_{iter}	MSE	Exec. time (s)
ICP (100%)	45	5.83	7.4
TrICP (70%)	88	0.10	2.5

As TrICP aligns the three point sets with the basic one, the model becomes more and more complete.

Another example of industrial measurement is shown in Fig. 4. Note that the surface is smooth and contains few characteristic details suitable for matching. However, due to a relatively good initial alignment done manually, the algorithm works properly. This example is interesting also because the overlap between the last two sets is only 20–25%.

Fig. 5 displays an TrICP alignment of two measurements of a chimpanzee skull, one from the top, the other from the bottom. The two data sets are shown in different colour. Note that the two sets 'interweave', which is a sign of a good alignment.

The sizes of the data sets considered in this section and the execution times for these data are given in Table 8.

6.2. Performance evaluation

The systematic tests compare TrICP and the ICRP algorithm by Pajdla and Van Gool [2]. As our goal was to design a robust variant of ICP that preserves its basic structure, we did not consider sophisticated and principal modifications, such as methods relying on surface normals. The LMedS variant [3] could have been another natural candidate for comparison. However, due to its computational load this method is only applicable to a small number—up to a few hundreds—of points. TrICP has been successfully applied to sets containing hundreds of thousands of points.

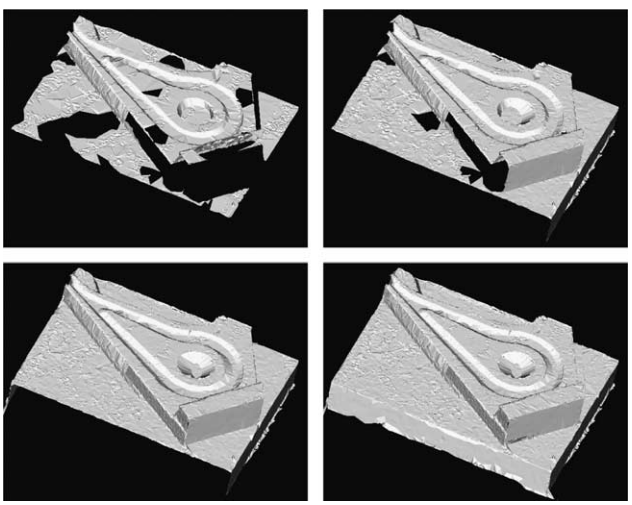

Fig. 3. Aligning four measurements of Skoda part.

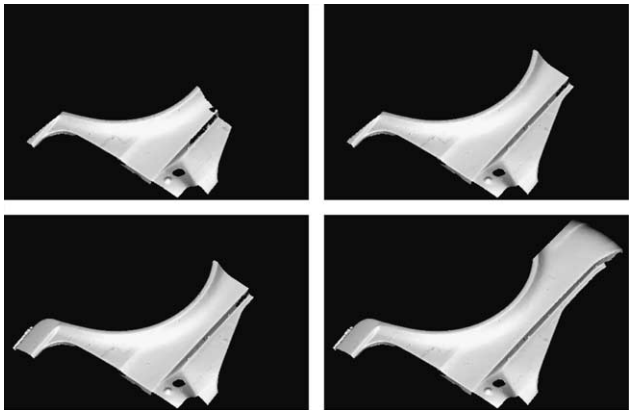

Fig. 4. Aligning four measurements of Fiat part.

ICRP is similar to TrICP in the sense that it also preserves the structure of ICP and makes it robust by incorporating a simple and efficient idea of the reciprocal correspondence. Backward matching [21] is a standard way of removing erroneous correspondences in dense stereo matching.

ICRP is applicable to large point sets. The algorithm has two major adjustable parameters, the radius of correspondence search R and the reciprocal correspondence parameter ϵ. (Recall Section 1). Setting of the parameters is discussed later in this section. Monotonic convergence is not guaranteed; MSE for points in the reciprocal correspondence may oscillate. Despite this, ICRP has been successfully used by several European laboratories in various applications. A common practice is to run the algorithm at least twice, each time improving the previous alignment.

The results given below were obtained for the SQUID fish contour database available at the web site [20] of the University of Surrey, UK. The database contains 2D shapes (contours) of 1100 different fishes. It has been accepted by the MPEG-7 group as a standard benchmark for shape retrieval. A fish contour has 200–700 raster points. A typical dimension of a shape is a few hundred pixels. The database

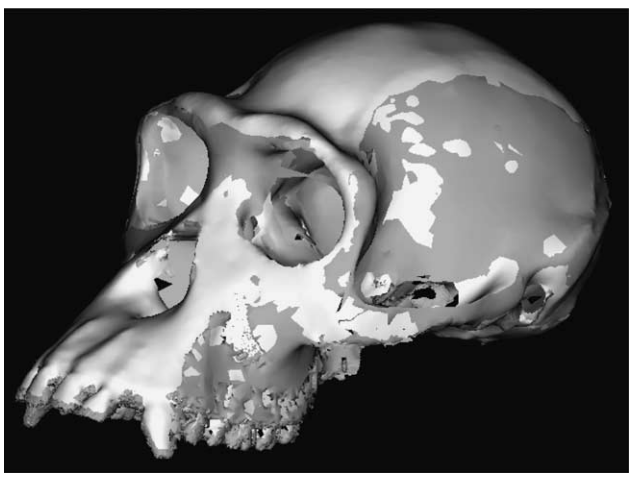

Fig. 5. Aligning two measurements of Skull.

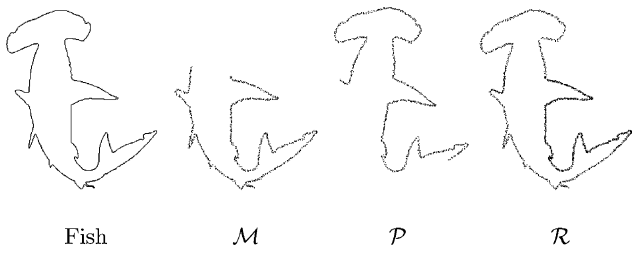

Fig. 6. Aligning deteriorated SQUID shapes. 'Fish' is the original noise-free shape.

reflects the natural diversity of fish shapes ranging from very simple to quite sophisticated.

The reasons for selecting a 2D database are twofold. Firstly, to our best knowledge, no large-scale database of 3D shapes is available. Secondly, the alignment results are much easier to interpret and assess in 2D than in 3D. In addition, running the algorithms many thousand times for many points in 3D would require too much time even on a modern computer.

To form $\mathcal{P}$, the original shape was rotated by a known angle. $\mathcal{M}$ preserved the original orientation. Then, different non-overlapping parts of $\mathcal{M}$ and $\mathcal{P}$ were randomly deleted so as to provide a desired overlap. Finally, noise was added to both shapes: ± 1 or 0 was randomly added to co-ordinates of each point. For each fish at each angle, the experiment was repeated 10 times with varying random perturbation, that is, deletion and noise. Fig. 6 exemplifies the experimental protocol of the test. $\mathcal{R}$ denotes the resulting alignment.

The fully automatic version of TrICP was applied. This means that the known overlap ξ_a was *not* passed to the algorithm: ξ was set automatically using the procedure described in Section 4. In most cases, the obtained ξ and the actual ξ_a were very close, which does not necessarily mean that ξ_a is always optimal for alignment. The automatic procedure was used for $\xi_a = 100\%$ as well. When the best ξ was found to be exactly 100%, ICP was implicitly run as a special case of TrICP; otherwise, TrICP was used.

ICRP was applied as follows. To obtain the best possible result, for each alignment the algorithm was run two or three times. Each time, the input was the output of the previous run, and the parameters of the ϵ-reciprocal correspondence were modified accordingly. For rotations below 20°, ICRP was run twice with the following parameters: $R = 20$, $\epsilon = 5$, then $R = 5$, $\epsilon = 2$. For rotation 20°, the algorithm was executed three times: prior to the above two runs, it was run with $R = 100$, $\epsilon = 25$.

Tables 2 and 3 present mean absolute differences between the ground-truth rotation and the rotations obtained by TrICP and ICRP for the 1100 *noise-free* shapes at various rotations (degrees) and overlaps (percents). Results for the *noisy* data are shown in Tables 4 and 5.

Before drawing any conclusion, we would like to emphasise that results of any performance evaluation and comparison depend on input data and conditions of experiments, including values of parameters and criteria

Table 2
TrICP errors for noise-free SQUID data (in degrees)

	100%	90%	80%	70%	60%
1°	0.0002	0.0021	0.0059	0.0142	0.0589
5°	0.0026	0.0118	0.0137	0.0487	0.2173
10°	0.0036	0.0187	0.0354	0.1428	0.4903
15°	0.0034	0.0312	0.0859	0.3333	1.1006
20°	0.0047	0.0454	0.1564	0.4917	1.5761

Table 3
ICRP errors for noise-free SQUID data (in degrees)

	100%	90%	80%	70%	60%
1°	0.0012	0.0036	0.0153	0.0426	0.0999
5°	0.0042	0.0114	0.0281	0.0980	0.1961
10°	0.0057	0.0193	0.1234	0.4196	1.6447
15°	0.0125	0.0218	0.1624	0.8895	2.1989
20°	0.0715	0.1507	0.4626	1.2854	2.7535

Table 4
TrICP errors for noisy SQUID data (in degrees)

	100%	90%	80%	70%	60%
1°	0.0512	0.0829	0.0701	0.0984	0.1879
5°	0.0509	0.0858	0.0797	0.1216	0.3411
10°	0.0517	0.0917	0.0984	0.1915	0.5800
15°	0.0509	0.1091	0.1646	0.3380	1.1430
20°	0.0502	0.0953	0.2025	0.6942	1.7949

Table 5
ICRP errors for noisy SQUID data (in degrees)

	100%	90%	80%	70%	60%
1°	0.0531	0.0612	0.0762	0.1226	0.2259
5°	0.0541	0.0652	0.1110	0.1763	0.3079
10°	0.0542	0.0655	0.1826	0.5988	1.7008
15°	0.0609	0.1108	0.3625	1.0878	2.5363
20°	0.1076	0.1614	0.4871	1.5114	3.0254

Table 6
$\mu_{ang} \pm \sigma_{ang}$ for rotation 10° (in degrees)

	100%	90%	80%	70%	60%
TrICP	9.98 ± 0.07	9.95 ± 0.13	9.95 ± 0.38	9.86 ± 0.75	9.49 ± 1.38
ICRP	9.98 ± 0.08	9.99 ± 0.11	9.92 ± 0.77	9.90 ± 1.87	9.89 ± 3.61

Table 7
Number of errors exceeding 5° for rotation 10°

	100%	90%	80%	70%	60%
TrICP	0	4	4	22	30
ICRP	0	0	9	84	190

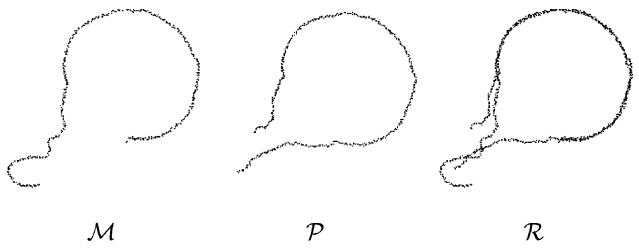

Fig. 7. Example of TrICP misalignment.

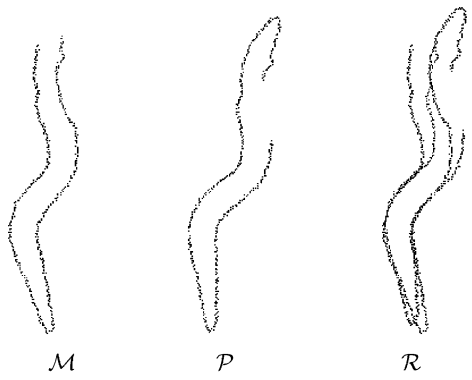

Fig. 8. Another example of TrICP misalignments.

of comparison. We did our best to provide a fair competition by studying the algorithms and the data, and setting the parameters so as to yield the optimal results. Despite this, one can never exclude a possibility that at different circumstances the algorithms will perform and compare differently.

For the SQUID dataset, TrICP is usually more accurate than ICRP. At small rotations, the difference is not significant. However, TrICP is more robust to rotation and to incomplete, noisy data. Table 6 provides an additional insight into the sources of the errors. The table shows mean rotations μ_{ang} and standard deviations σ_{ang} obtained on the noisy data for rotation 10° at varying overlaps. When the difference between μ_{ang} and 10° is small, the distribution of the 1100 measurements is centred close to the ground truth, meaning small systematic errors. (Note that $\mu_{\text{ang}} < 10°$ because the algorithms approach the true orientation from below.) Random errors are exhibited by σ_{ang}.

From Table 6, one can make a conclusion that the *systematic errors* of the two algorithms are very similar. At small overlaps, the distribution of ICRP is centred slightly better. However, ICRP's *random errors* are significantly larger. Table 7 provides an explanation: it demonstrates that at small overlaps ICRP yields a relatively large number of wrong alignments. (Recall that for each overlap the total number of alignments is 1100.) Apparently, ICRP gets stuck in a local minimum more frequently that TrICP. Also, convergence of ICRP is not guaranteed.

7. Discussion and conclusions

Like any ICP variant, TrICP occasionally obtains a wrong registration. Examples of TrICP misalignments are shown in Figs. 7 and 8. In these cases, ICRP gives correct results. Fig. 7 illuminates a major limitation of the method: most of the overlapping part is highly symmetric, and there is no regular way to force the algorithm to use the tiny informative part of the shape. In Fig. 8, TrICP converges to a local minimum.

Examples of ICRP misalignments are shown in Fig. 9. The corresponding TrICP solutions are correct. ICRP minimises the total distance between closest reciprocal points as defined in Section 1. One can imagine this process as connecting the points by springs. The springs force the alignment transformation to minimise energy (distance). When the reciprocal condition is not satisfied, connection between the corresponding points is not established, or is replaced by a wrong connection. This may result in an erroneous registration.

The objective function of ICRP does not encourage the algorithm to use as many points as possible. The first (left) example in Fig. 9 is the case when a small number of points are in good registration, while the rest are discarded as having no reciprocal correspondence. The other (right) example shows a situation similar to Fig. 7: small features have no impact on the outcome since the correspondences between the features are not established.

Typical execution times of TrICP and ICRP for different 2D and 3D data are shown in Table 8. The figures are given

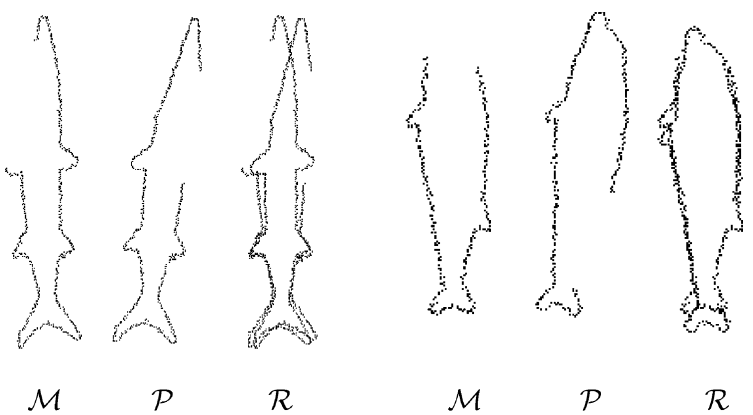

Fig. 9. Examples of ICRP misalignments.

Table 8
Execution times for different data (in seconds)

Data	Points in $\mathcal{M}/\mathcal{P}$	TrICP time	ICRP time
2D SQUID	248/248	0.05	0.06
2D SQUID	312/312	0.09	0.10
2D SQUID	526/526	0.13	0.12
3D Frog	2822/3177	2.53	2.46
3D Skoda	6534/6009	8.33	7.20
3D Skull	54,252/111,954	38.29	182.02

for a single run of an algorithm. In practice, ICRP may be run 2–3 times, while automatic setting of the overlap parameter in TrICP may require about 10 runs of the algorithm. If so, the figures given in the table should be modified accordingly. Otherwise, processing speeds of the two algorithms are comparable.

It should be emphasised that execution time depends on implementation of algorithm and optimisation of compiler. A critical operation is finding the closest point. ICRP uses a different procedure, which seems to be less efficient when distances between the corresponding points are relatively large. The Skull results in Table 8 are given for a good initial registration. When a poor initial alignment is provided, the execution time of ICRP increases to 16 min, while TrICP needs 2.7 min.

We conclude the paper by briefly characterising the compared algorithms and outlining possible ways to improve their performance. Our systematic tests confirm the practical experience of ICRP users showing that this algorithm is a reliable and reasonably fast tool for registration of partially overlapping 3D data sets *when a good initial estimation is available*. ICRP can be improved by adding a method for computing an initial estimation as proposed in Ref. [11]. Automatic adaptive setting of the parameters R, ϵ should also be considered. How to force the algorithm to use as many points as possible is a related open question to be addressed.

The advantages of the proposed TrICP algorithm are its *robustness* to *noise and to poor initial estimation*. TrICP has been successfully applied to large data sets of a few hundred thousand points, to overlaps much smaller than 50% and to initial rotations of up to 20°. A procedure for automatic setting of the critical overlap parameter is available that encourages TrICP to use most of the points that can be paired. The guaranteed convergence of the iterative algorithm makes its behaviour more stable and predictable.

We recommend using TrICP when a poor initial estimation is only available, or when high precision is needed. Prior knowledge of overlap is not required. However, the algorithm operates better and faster when the range of the overlap values considered is narrow. To more efficiently avoid local minima, it is planned to add an option that would allow TrICP to perturb the initial orientation of the data set.

We are also working on the automatic rough pre-alignment of the two point sets for subsequent application of the proposed algorithm. Genetic algorithms (GAs) have been used for pre-alignment [22] as well as final alignment [23]. GAs are slower and less precise than the iterative algorithms, but they do not require a good starting point for search of global optimum. An elegant way to automatically initialise the TrICP would be to include in genetic search the six Euclidean parameters of alignment and the overlap parameter ξ, using the objective function (10). Our preliminary experiments show that this is possible.

References

[1] P. Besl, N. McKay, A method for registration of 3-D shapes, IEEE Transactions on Pattern Analysis and Machine Intelligence 14 (1992) 239–256.
[2] T. Pajdla, L.V. Gool, Matching of 3-D curves using semi-differential invariants, in: Fifth International Conference on Computer Vision, 1995, pp. 390–395.
[3] E. Trucco, A. Fusiello, V. Roberto, Robust motion and correspondence of noisy 3-D point sets with missing data, Pattern Recognition Letters 20 (1999) 889–898.
[4] Y. Chen, G. Medioni, Object modelling by registration of multiple range images, Image and Vision Computing 10 (1992) 145–155.
[5] K. Pulli, Surface reconstruction and display from range and color data, PhD thesis, University of Washington, Seattle, 1997.
[6] S. Rusinkiewicz, M. Levoy, Efficient variants of the icp algorithm, in: Third International Conference on 3D Digital Imaging and Modeling, 2001.
[7] P. Rousseeuw, A. Leroy, Robust Regression and Outlier Detection, Wiley Series in Probability and Mathematical Statistics, 1987.
[8] Z. Zhang, Iterative point matching for registration of free-form curves and surfaces, International Journal of Computer Vision 13 (1994) 119–152.
[9] A. Fitzgibbon, Robust registration of 2D and 3D point sets, in: British Machine Vision Conference, 2001.
[10] P. Rousseeuw, B. van Zomeren, Unmasking multivariate outliers and leverage points, Journal of the American Statistical Association 85 (1990) 633–651.
[11] P. Krsek, T. Pajdla, V. Hlaváič, R. Martin, Range image registration driven by a hierarchy of surface differential features, in: M. Gengler, M. Prinz, E. Schuster (Eds.), 22nd Workshop of the Austrian Association for Pattern Recognition, Österreichische Computer Gesellschaft, 1998, pp. 175–183.
[12] J. Friedman, J. Bentley, R. Finkel, An algorithm for finding best matches in logarithmic expected time, ACM Transactions on Mathematical Software 3 (1977) 209–226.
[13] P. Rousseeuw, Least median of squares regression, Journal of the American Statistical Association 79 (1984) 871–880.
[14] P. Rousseeuw, S. Van Aelst, Positive-breakdown robust methods in computer vision, Computing Science and Statistics 31 (1999) 451–460.
[15] P. Meer, D. Mintz, A. Rosenfedl, D. Kim, Robust regression methods in computer vision: a review, International Journal of Computer Vision 6 (1991) 59–70.
[16] M. Ye, R. Haralick, Optical flow from a least-trimmed squares based adaptive approach, in: Proceedings of International Conference on Pattern Recognition, vol. 3, IEEE Computer Society, 2000, pp. 1064–1067.
[17] D. Chetverikov, Fast neighborhood search in planar point set, Pattern Recognition Letters 12 (1991) 409–412.

[18] W. Press, S. Teukolsky, W. Vetterling, B. Flannery, Numerical Recipes in C, Cambridge University Press, Cambridge, 1992.
[19] D. Eggert, A. Lorusso, R. Fisher, Estimating 3-D rigid body transformations: a comparison of four major algorithms, International Journal of Machine Vision and Applications 9 (1997) 272–290.
[20] University of Surrey, Guildford, UK, The SQUID database: Shape Queries Using Image Databases, www.ee.surrey.ac.uk/Research/VSSP/imagedb/demo.html.
[21] E. Trucco, A. Verri, Introductory Techniques for 3-D Computer Vision, Prentice Hall, Englewood Cliffs, NJ, 1998.
[22] K. Brunnström, A. Stoddart, Genetic algorithms for free-form surface matching, in: Proceedings of International Conference on Pattern Recognition, vol. 4, IEEE Computer Society, 1996, pp. 689–693.
[23] C. Chow, H. Tsui, T. Lee, Surface registration using a dynamic genetic algorithm, Pattern Recognition 37 (2004) 105–117.