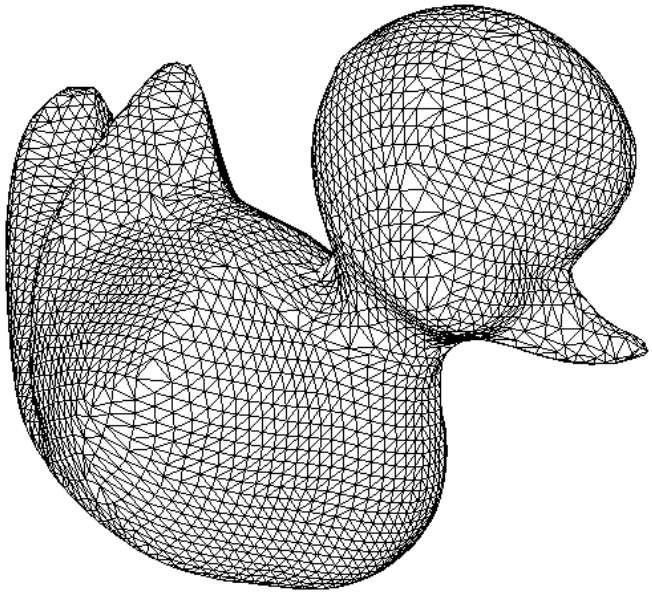
Fast Geometric Matching for Shape Registration



Timothée Jost

THÈSE SOUMISE À LA FACULTÉ DES SCIENCES
DE L'UNIVERSITÉ DE NEUCHÂTEL POUR L'OBTENTION
DU GRADE DE DOCTEUR ÈS SCIENCES

2002

Fast Geometric Matching for Shape Registration

Timothée Jost

THÈSE SOUMISE À LA FACULTÉ DES SCIENCES
DE L'UNIVERSITÉ DE NEUCHÂTEL POUR L'OBTENTION
DU GRADE DE DOCTEUR ÈS SCIENCES

2002

IMPRIMATUR POUR LA THESE

Fast geometric matching for shape registration

de M. Timothée Jost

UNIVERSITE DE NEUCHATEL

FACULTE DES SCIENCES

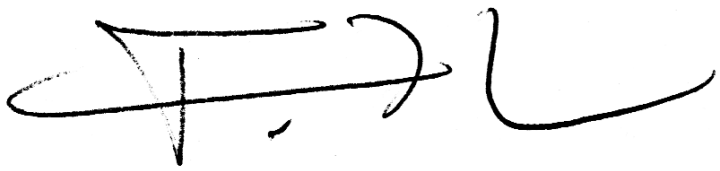
La Faculté des sciences de l'Université de Neuchâtel sur le rapport des membres du jury,

MM. H. Hügli (directeur de thèse),
P.J. Erard, R. Ingold (Fribourg) et
F. Marzani (Dijon, F)

autorise l'impression de la présente thèse.

Neuchâtel, le 21 octobre 2002

Le doyen:



F. Zwahlen

Summary

The research presented in this Ph.D. contributes to the development of the three-dimensional (3D) vision field. More precisely, it addresses the problem of the registration, or geometric matching, of 3D datasets, which consists in finding their correct relative alignment based on their intrinsic properties. Typical applications using registration as part of their working principle include the modelling of 3D objects, object recognition or quality inspection.

The iterative closest point (ICP) algorithm is considered in this work as our basic registration method. The ICP algorithm processes iteratively. At each iteration, it first creates closest point correspondences between two datasets and it then minimizes the average distance of the couplings by a rigid transformation.

The matching quality and the robustness of the ICP can be improved by considering additional features, such as colour or curvature and by adding weights to the couplings found in the first step of each iteration.

The main practical difficulty of the ICP algorithm is that it requires heavy computations and, thus, several speeding up methods have been proposed. A fairly complete review of the different methods is proposed in this work. The main conclusion of this review is that most of the existing solutions lead to a tradeoff between speeding up and quality of the matching.

Two new algorithms are proposed to accelerate the ICP. First of all, the neighbour search algorithm, which relies on neighbourhood relationships in the datasets to restrict the search of the closest point to a local subset. Then, a multi-resolution

scheme is also proposed and analysed. It proceeds from coarse to fine and successively improves a previous solution at the finer representation level. One can note that both solutions for the speeding up of the ICP have been developed in a perspective to avoid the tradeoff with matching quality that is imposed by most existing solutions.

Finally, a complete object digitising system is presented. It combines data from several unpositioned range images or 3D meshes taken from different viewpoints to create a complete virtual model. In addition to showing a practical use for the registration techniques described above, building the system also permitted the development of new algorithms and methods for colour digitising, mesh fusion and texture handling.

Résumé

La recherche présentée dans ce travail de thèse contribue au développement de la vision tridimensionnelle (3D). Plus précisément, elle aborde le problème de la mise en correspondance géométrique de données 3D, qui consiste à déterminer l'alignement relatif correct de ces ensembles de données en se basant uniquement sur leurs propriétés intrinsèques. Parmi les applications typiques utilisant la mise en correspondance, on peut citer la modélisation d'objet 3D, la reconnaissance d'objet ou encore l'inspection de qualité.

L'algorithme ICP (pour « Iterative Closest Point ») est la méthode de mise en correspondance de base considérée dans ce travail. L'ICP procède itérativement, comme son nom l'indique. A chaque itération, les points correspondants les plus proches entre deux ensembles de données sont tout d'abord définis. La distance moyenne des couples précédemment créés est ensuite minimisée par une transformation rigide.

La qualité de la mise en correspondance ainsi que sa robustesse peuvent être améliorées en faisant appel à des caractéristiques supplémentaires liées aux objets, comme la couleur ou la courbure, et en pondérant les couples créés dans la première partie de chaque itération.

La principale difficulté pratique de l'ICP est qu'il nécessite un nombre élevé de calculs. En conséquence, plusieurs méthodes ont été développées afin de l'accélérer. Une revue aussi complète que possible de ces méthodes est proposée dans ce travail. Sa conclusion majeure est que la plupart des solutions existantes nécessitent un compromis entre accélération et qualité de la mise en correspondance.

Deux nouveaux algorithmes pour l'accélération de l'ICP sont aussi proposés. Premièrement l'algorithme de recherche de voisinage (neighbour search algorithm) qui compte sur les relations de voisinage existant dans les données pour restreindre l'espace de recherche des points les plus proches à un sous-ensemble des données. Deuxièmement, un schéma multi-résolution est suggéré et analysé. Il procède de façon « grossière à fine » et améliore successivement la mise en correspondance avec un niveau de représentation des données de plus en plus fin. On notera que ces deux solutions ont été développées dans l'optique d'éviter tout compromis avec la qualité de la mise en correspondance.

Finalement, un système complet de digitalisation 3D d'objets est présenté. Il combine les données de plusieurs images de profondeur ou de maillages 3D pris sous différents points de vue afin de créer un modèle virtuel complet. En plus d'illustrer une utilisation pratique des techniques de mise en correspondance décrites précédemment, la mise au point de ce système a également permis le développement de nouveaux algorithmes pour l'acquisition de la couleur, la fusion de maillages ainsi que l'utilisation de textures.

Table of Contents

Chapter 1
Introduction

1.1 Motivations

The recent availability of cheap commercial range scanners, capable of easily measuring three-dimensional (3D) shapes, had a very beneficial effect on the development of 3D vision processing. Several types of range scanners are available on the market today, generally working on optical principles like stereo matching, active triangulation, time of flight or focus / defocus. Each single measurement is generally provided in a range image, where the value of each pixel represents the scanner-to-object distance. Range images are also often called 3D views since they represent a 3D image of an object or a scene seen from a certain point of view.

Geometric matching, or shape registration, belongs to the main techniques used in 3D vision. It basically consists in finding the correct alignment between two or more sets of data, based on their intrinsic properties, and plays an important role in today's computer vision.

Several important applications use registration as a part of their working principle:

• **Modelling of 3D objects**. Modern techniques for the modelling of 3D objects often rely on registration to match the set of 3D views that is needed to cover the whole surface of the object.

• **Object recognition**. In object recognition, a measured surface data can be registered with different existing models to recognize the object and its position.

• **Quality inspection**. As a last example, in quality inspection, a measured surface data of an existing object can be registered and compared with its model to detect eventual anomalies.

Generally, the following demands have to be observed in the previously cited applications: robustness of the matching, handling of free-form objects and speed of the process.

1.2 Scope of the research

One can distinguish two main types of geometric matching: the high-level one and the low-level one. High-level solutions basically rely on features extraction and matching. On the opposite, low-level solutions apply matching directly on data primitives, like points or triangles. The later type of matching has several advantages compared to high-level matching, especially when dealing with free-form objects, because segmentation and features extraction can be very unreliable in this case.

The iterative closest point (ICP) algorithm figures among the principal and widely used low-level registration methods. If the original algorithm doesn't fit all of the applications and their demands, as described in paragraph 1.1, several variations that do fit them were since then proposed.

Starting from an initial rough alignment of the data, the ICP processes iteratively. With each iteration, it first creates closest point correspondences between two sets of points (or more generally geometric data) and then minimizes the average distance of the previously found correspondences by a rigid transformation – i.e. a translation and a rotation.

The main practical difficulty of the ICP algorithm is that it requires heavy computations. When working with clouds of points or triangulated meshes, the complexity of the original algorithm is O(MN), where M and N represent the number of points of the clouds to be matched. Consequently, matching high-resolution shapes takes a lot of time, even on current computers, and there is a need for ways to reduce the ICP computation time.

The research presented in this report mainly addresses the acceleration of the ICP algorithm. Existing methods are presented

and compared and new methods are proposed and analysed. Besides the speeding up of the ICP, special care is given to preserving the quality of the matching. Finally, a complete object digitising system has been developed during the course of the research and is described at the end of this report. The main contributions of this work are presented into more details in the next section.

1.3 Main contributions

The main contributions of the Ph.D. work presented in this report are:

The neighbour search algorithm

A novel heuristic algorithm for fast closest point search applied to the ICP has been proposed, developed and tested. It is presented in Chapter 4. It consists of a heuristic that uses neighbourhood relationships to obtain a first approximation of the closest points and refines the results by a local search. Researches have been mainly conducted with range images, but the neighbour search can be applied to clouds of points or triangle meshes as well.

The multiresolution scheme ICP

A multiresolution scheme applied to the ICP algorithm has been proposed and analysed. The combination of this multiresolution approach and the neighbour search algorithm has been taken into account to obtain a very fast and robust ICP algorithm that can be found in Chapter 5.

A complete digitising system

During the course of the research, a complete digitising system has been built and is presented in Chapter 6. It consists of two structured light range finders (which differ mainly by the size of

their scan area and their resolution) and a workstation onto which an experimental digitising program has been developed. Most aspects of object digitising have been analysed and new algorithms for colour digitising, mesh fusion and texture handling have been proposed and implemented.

1.4 Organization of the report

The following chapters are organized as follows:

The ICP algorithm, its situation among the other registration methods and its main variants are presented in Chapter 2. Many solutions for the acceleration of the ICP algorithm have been proposed. A description and comparison of most of them are presented in Chapter 3.

The new closest point search method to speed up the ICP algorithm, the neighbour search, can be found in Chapter 4. The proposed multiresolution scheme ICP is presented in Chapter 5. A description of the main features of the built digitising system, as well as some results can be found in Chapter 6.

Finally, the general conclusions of this work are presented in Chapter 7.

Chapter 2
Surface Registration and 3D Points Matching

This chapter situates the ICP algorithm among other existing matching algorithms. Then, it presents and analyses the different steps of the algorithm, as well as their main variants.

2.1 An overview of matching methods

Registration basically consists in finding the correct alignment between two or more sets of data. The registration is a straightforward process if the relative position is known for each view. Proposed solutions in this way include using orthogonal views [Chi88], placing the object onto a turntable [Bha84] [Hou95] or using specific hardware positioning systems to get the scanner position [3Dsca]. The main problem of these methods is that parts of the object are hard or impossible to measure, typically its "bottom" or some concave surfaces.

If no a priori knowledge of the positioning is known, a common approach for shape registration is to use techniques based on the intrinsic properties of datasets, or geometric matching. One can distinguish two main types of geometric matching: high-level ones and low-level ones. High-level solutions basically rely on features extraction and matching - see [Ste92]

or [Chu96] for examples. On the opposite, low-level solutions apply matching directly on data, like points or triangles [Bes92] [Che92]. The later type of matching has several advantages compared to high-level matching, especially when dealing with free-form objects, because segmentation and features extraction can be very unreliable in this case [Kam89].

Among the existing low-level solutions, Potmesil [Pot83] proposed to use a heuristic search in the whole transformation parameter space and Munch [Mun93] tried to maximize the correlation between the datasets in order to match them. Unfortunately, these methods are not suitable for dense 3D data, because of their very high computational costs.

Another approach to solve low-level matching is to minimize the least square distances between pairs of corresponding points found in the datasets to be matched, or geometric point matching. The next section details the geometric point matching problem and the existing solutions.

2.2 Geometric point matching

As said above, the geometric point matching basically consists in minimizing the least square distances between pairs of corresponding points in the datasets to be matched. It can be defined as follows:

Given two datasets P and X and a set of points of P which have been paired with a set of corresponding points on X, denoted respectively by $\{\mathbf{p}_i\}$ and $\{\mathbf{y}_i\}$. Find the rigid transformation, defined by the rotation $\mathbf{R}$ and the translation $\mathbf{t}$, which minimizes the following mean-squares objective function

$$e(\mathbf{R}, \mathbf{t}) = \frac{1}{N}\sum_{i=1}^{N} \left\|(\mathbf{R}\mathbf{p}_i + \mathbf{t}) - \mathbf{y}_i\right\|^2 = \frac{1}{N}\sum_{i=1}^{N} \left\|(\mathbf{R}\mathbf{p}_i + \mathbf{t}) - c(\mathbf{p}_i)\right\|^2 \qquad (2.1)$$

where the function c associates every point $\{\mathbf{p_i}\}$ with a corresponding point of X

$$c : P \rightarrow X \left| \forall \mathbf{p}_i \in \{\mathbf{p}_i\} \in P, \mathbf{y}_i = c(\mathbf{p}_i) \in X \right. \qquad (2.2)$$

and N is the number of pairs.

Faugeras [Fau86] was among the first to make research in the 3D free-form shape matching and proposed a robust solution

to minimizing (2.1) in 1986 already. A presentation and discussion of existing solutions can be found in 2.6.

The main limitation of Faugeras's work is that it relied on the existence of large planar patches on the considered data to establish correspondences. This illustrates the difficulty to find a direct solution for the geometric point-matching problem, mainly because the correspondence function c is generally unknown.

2.2.1 Iterative geometric matching

In 1992, several authors [Bes92] [Che92] [Men92] [Cha92] came up with new algorithms to solve the geometric point-matching problem that exploit very similar techniques. The main idea of these algorithms is the following. Given that the two datasets to be matched are roughly aligned, a point of the first dataset is close to its corresponding point in the second dataset. Thus, the transformation that brings the two sets closer can be found by matching points of the first dataset with their closest counterpart in the second dataset. Then, one can refine the resulting transformation by applying this procedure iteratively.

Consequently, the mean-squares objective function of equation (2.1) must be solved at each iteration k and becomes

$$e(\mathbf{R}_k, \mathbf{t}_k) = \frac{1}{N} \sum_{i=1}^{N} \left\| (\mathbf{R}_k(\mathbf{R}\mathbf{p}_i + \mathbf{t}) + \mathbf{t}_k) - c(\mathbf{p}_i) \right\|^2 \qquad (2.3)$$

The global transformation $(\mathbf{R}, \mathbf{t})$ is incrementally updated as follows: $\mathbf{R} = \mathbf{R}\mathbf{R}_k$ and $\mathbf{t} = \mathbf{t} + \mathbf{t}_k$. Finally, the correspondence function c (2.2) is now defined by

$$c(\mathbf{p}_i) = \mathbf{x} \mid \min_{\mathbf{x} \in X} d((\mathbf{R}\mathbf{p}_i + \mathbf{t}), \mathbf{x}) \qquad (2.4)$$

The methods described above differ mainly in how they establish point correspondences. Besl [Bes92] uses the Euclidean distance between two point sets to compute closest points. Chen [Che92] calculates the closest point for a point from one surface by intersecting its surface normal vector (or simply normal) with the second surface. Menq [Men92] finds closest points between a point set and a set of parametric surface patches by solving non-linear equations. Finally, Champleboux [Cha92] converts a point set into an octree-spline and computes Euclidean distances.

2.2.2 Rough matching estimate

As said above, iterative geometric matching algorithms converge to a local minimum and need an initial rough matching to converge correctly. The problem of finding this initial alignment can be solved by many different methods. Some of them have already been exposed in 2.1, like high-level feature matching [Ste92] or using hardware positioning systems [3Dsca]. Methods proposed in the last years include interactive pose estimation, see 6.3.1, alignment of major axes of datasets [Dor96a] [Dor97], matching oriented points (spin-image) [Joh97a], exhaustive search of corresponding triangulation [Che98], robust fuzzy clustering [Tar99] or alignment of laser dot projection [Ber00].

One of the interesting practical questions that arise at this point is "how far from the correct matching can the initial rough matching be so that the algorithm still converges correctly?" Part of the answer to this question can be found in section 2.8, "Matching quality comparison, SIC ranges".

2.3 The iterative closest point (ICP) algorithm

Besl's iterative closest point or ICP [Bes92] algorithm has been chosen for this work. This choice is mainly motivated by the fact that it is a very widely and successfully used matching algorithm.

The ICP algorithm can register several types of geometric data like point sets, triangle sets, implicit surfaces or parametric surfaces. However, it establishes point-to-point correspondence using Euclidean distance, so the data may need decomposition into point sets.

2.3.1 ICP algorithm statement

The iterative closest point algorithm can be stated as follows:

- input: 2 point sets, $P = \{\mathbf{p}_i\}$ with N_p points (data) and $X = \{\mathbf{x}_i\}$ with N_x points (model)
- output: A transformation ($\mathbf{R}$, $\mathbf{t}$) that registers P and X

- initialisation: $k = 0$, $P_0 = P$, $\mathbf{R}_0=\mathbf{I}$ and $\mathbf{t}_0=(0,0,0)$
- iteration k:

1. Compute the closest points:

   Use the squared Euclidean distance

$$d(\mathbf{p}, \mathbf{x}) = \|\mathbf{p} - \mathbf{x}\|^2 \tag{2.5}$$

   to compute the set of N_p closest points, $Y_k = \{\mathbf{y}_{i,k}\}$, of $P_k = \{\mathbf{p}_{i,k}\}$ defined as

$$\mathbf{y}_{i,k} = c(\mathbf{p}_{i,k}) = \mathbf{x} | \min_{\mathbf{x}\in X} d(\mathbf{p}_{i,k}, \mathbf{x}) \tag{2.6}$$

2. Compute the registration:

   Define the mean squared error of the couplings $\{\mathbf{p}_{i,0}, \mathbf{y}_{i,k}\}$ as a function of $\mathbf{R}_k$ and $\mathbf{t}_k$.

$$e(\mathbf{R}_k, \mathbf{t}_k) = \frac{1}{N_p} \sum_{i=1}^{N_p} \left\| (\mathbf{R}_k \mathbf{p}_{i,0} + \mathbf{t}_k) - \mathbf{y}_{i,k} \right\|^2 \tag{2.7}$$

   and compute the rigid transformation $(\mathbf{R}_k, \mathbf{t}_k)$ such as

$$e_k = \min_{\mathbf{R}_k, \mathbf{t}_k} e(\mathbf{R}_k, \mathbf{t}_k) \tag{2.8}$$

3. Apply the registration:

   Apply the best rigid transformation to obtain the set $P_{k+1} = \{\mathbf{p}_{i,k+1}\}$ defined as

$$\mathbf{p}_{i,k+1} = \mathbf{R}_k \mathbf{p}_{i,0} + \mathbf{t}_k \tag{2.9}$$

4. Iteration termination

   Stop when the maximum number of iterations or a defined criterion is reached (see 2.7). Set $\mathbf{R} = \mathbf{R}_k$ and $\mathbf{t} = \mathbf{t}_k$

Figure 2.1 presents the ICP algorithm principle as well as a small example.

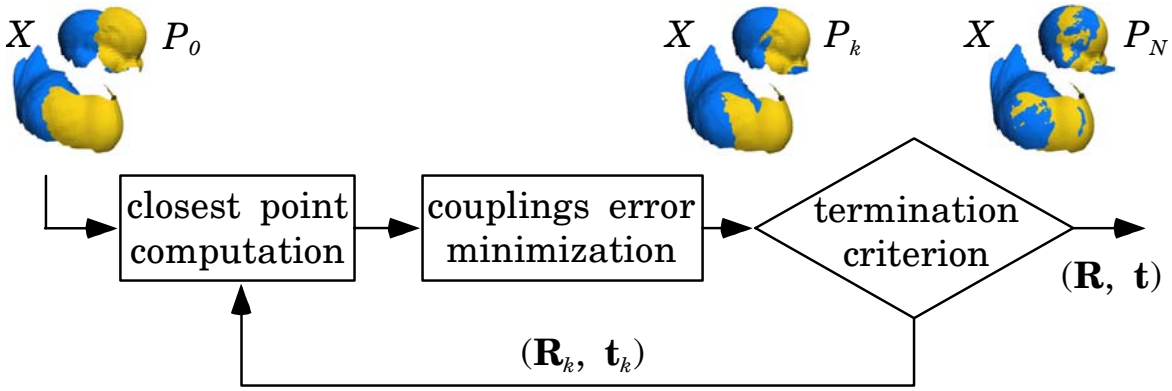


Figure 2.1: ICP algorithm principle

2.3.2 Convergence theorem

The following convergence theorem can be stated: The ICP algorithm presented above always converges monotonically to a local minimum. A proof of this theorem can be found below.

At any given iteration k, the mean squared distance, d_k, between P_k and Y_k is defined by

$$d_k = \frac{1}{\mathrm{N_p}} \sum_{i=1}^{\mathrm{N_p}} \left\| \mathbf{p}_{i,k} - \mathbf{y}_{i,k} \right\|^2 \tag{2.10}$$

After its minimization, the mean squared error of the couplings is given by

$$e_k = \frac{1}{\mathrm{N_p}} \sum_{i=1}^{\mathrm{N_p}} \left\| \left(\mathbf{R}_k \mathbf{p}_{i,0} + \mathbf{t}_k \right) - \mathbf{y}_{i,k} \right\|^2 = \frac{1}{\mathrm{N_p}} \sum_{i=1}^{\mathrm{N_p}} \left\| \mathbf{p}_{i,k+1} - \mathbf{y}_{i,k} \right\|^2 \tag{2.11}$$

By definition, it is always the case that $e_k \leq d_k$. If $e_k > d_k$ was true, it would mean that the identity transformation on the point set would yield a smaller mean squared error than the given ($\mathbf{R}_k$, $\mathbf{t}_k$), which isn't possible.

At iteration k+1, a new closest points search of P_{k+1} is done and a set Y_{k+1} obtained. It is straightforward that

$$\left\| \mathbf{p}_{i,k+1} - \mathbf{y}_{i,k+1} \right\|^2 \leq \left\| \mathbf{p}_{i,k+1} - \mathbf{y}_{i,k} \right\|^2 \text{ for each } i = 1, \mathrm{N_p} \tag{2.12}$$

simply because $\mathbf{y}_{i,k+1}$ is the closest point of $\mathbf{p}_{i,k+1}$ by definition and, consequently, that $d_{k+1} \leq e_k$. Finally, one can state that $0 \leq e_k$

since mean square errors cannot be negative. Therefore, the following inequality can be stated

$$0 \leq e_{k+1} \leq d_{k+1} \leq e_k \leq d_k \text{ for all } k \tag{2.13}$$

This proves that the error sequence is non-increasing and bounded below. Consequently, the ICP algorithm must converge monotonically to a minimum. Q.E.D.

2.3.3 Discussion

The basic ICP algorithm has two main disadvantages: poor robustness to outliers and speed. Specifically, it is very sensible to noisy data and hidden geometry, and has a high computation cost, especially its closest points search step. The next paragraphs of this chapter present the different steps of the algorithm, as well as the main variants proposed to improve its robustness. A description and comparison of the main existing solutions to speed up the ICP can be found in Chapter 3.

Variant names

Most variants of the ICP don't rely on all closest points anymore, which leads some authors to propose other appellations. For example, Zhang [Zha94] called his variation "Iterative Pseudo Point Matching" algorithm or Rusinkiewicz [Rus01] proposed "Iterative Corresponding Point" as a better expansion for the abbreviation of ICP. This being noted, and for clarity, we will keep the denomination ICP for all variations of the original algorithm, like it is generally done in literature.

Global registration

Basically, the ICP algorithm registers two point sets. If more than 2 datasets need to be registered together, one speaks of global registration. Object modelling from range views is a typical example of applications that require registering more than two datasets. A straightforward solution to this problem is to register views sequentially, adding and registering views one after

another. A problem of this method is that a propagation and accumulation of the registration errors can occur. Several smarter solutions for global registration have been proposed. We won't detail them here but some solutions based on ICP can be found in [Ber96] [Sto96a] [Ben97] [Pul99] [Ber00].

Non-rigid registration

In this work, we only consider rigid registration, i.e., the considered transformations are exclusively rotations and translations. However, in some cases, the datasets could be a generic deformable shape model to be registered with a specific shape data like, for example, an organ in a medical application. Then, we speak about non-rigid registration. A typical adaptation of the ICP algorithm for non-rigid registration can be found in [Fel97].

2.4 Closest point computation

The ICP algorithm assumes that a good approximation of corresponding points is their closest points. The better the calculated closest points reflect the real correspondences, the faster the ICP algorithm will converge. Furthermore, the Euclidean distance is sometimes not sufficient to establish good enough correspondences and to obtain a successful convergence of the ICP algorithm.

Additional features, such as colour, can often be available along with 3D data. Furthermore, extra geometric features like surface normal or curvature can be estimated easily in range images or triangle meshes. This section presents alternative distance computations that consider such additional features in order to better couple the different points and increase the robustness of the algorithm. We can note however that all the methods presented below are still based on the "classic" Euclidean distance since the ICP algorithm principle is based on it.

2.4.1 Point-to-surface matching

So far, we considered datasets P and X where surfaces are represented by sets of points. Basically, the denser the sampling of the data, the better the surfaces are represented and the smaller the matching error is. Given that the datasets are described by oriented points (point $\mathbf{p}$ and associated surface normal $\mathbf{n_p}$) or triangulated point sets, a point-to-surface distance can be used to improve the matching error introduced by the discreetization of the data.

- Schütz uses the point-to-triangle distance [Sch98b]. A method based on the k-D tree (see section 3.4.3) is used to accelerate the calculation of the distance from a point to a triangle mesh.
- Chen [Che92] defines the point-to-plane distance d_{Si} between the points $\mathbf{p}_i$ of the first surface and the plane S_i containing their coupled point $\mathbf{q}_i$ and oriented perpendicularly to the normal of their coupled point $\mathbf{n}_{\mathbf{q}i}$ (see Figure 2.2).

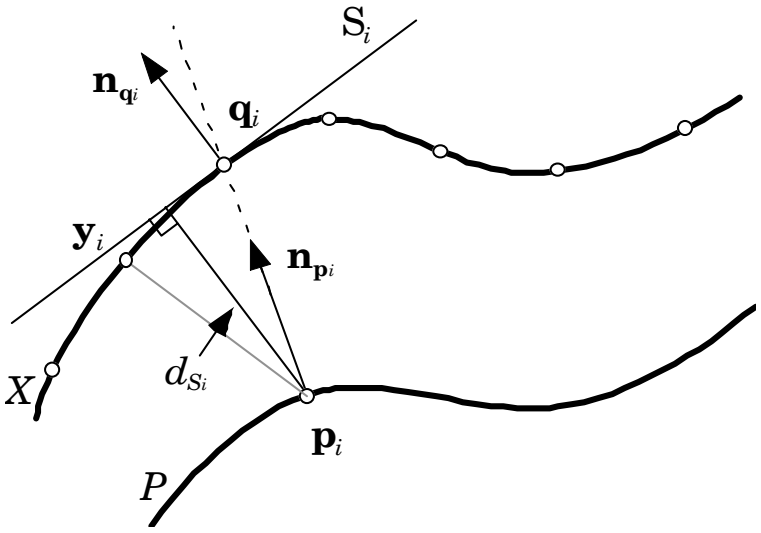


Figure 2.2: Point-to-plane vs. point-to-point metric

A difference between the point-to-point and the point-to-plane matching is that the minimization solutions are not the same (§2.6) and that there exists no closed-form solution for the latter. Consequently, a generic non-linear least-squares

technique (typically the Levenberg-Marquardt algorithm [Mar63]) is applied to find the best rigid transformation.

Practically, the point-to-plane minimization lowers the number of iterations needed for convergence versus a classis ICP using Euclidean distances. Some recent discussions on point-to-point versus point-to-plane matching can be found in [Pul99] or [God01]. Basically, point-to-plane matching converges faster than point-to-point matching but the alternative feature matching found in the rest of this chapter can't be used with it and it is less robust in some situations.

2.4.2 Colour matching

The shape-matching problem can be intrinsically ambiguous. It is the case when the surfaces, represented by point sets, lack geometric "features", like flat surfaces, or when the surfaces come from objects that possess one or more axis of symmetry. However, if intensity or colour data are available, they can be used to improve the matching (Figure 2.3). We can distinguish two different uses of colour or intensity to improve matching:

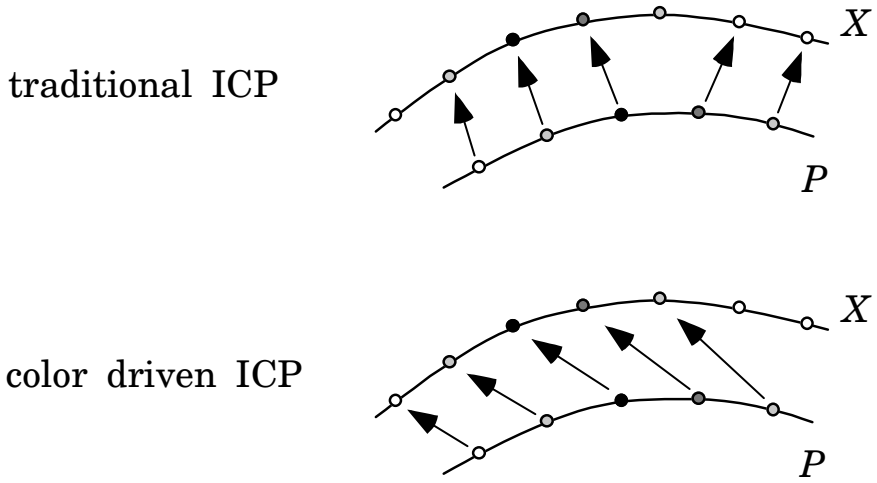


Figure 2.3: Influence of colour or intensity on matching

- Use the closest compatible point, i.e., first build a subset of the closest points that have a colour distance below a certain threshold, then find the closest point (using geometric distance) among these "compatible" candidates [God94].

- Use the colour ICP, where the colour information is added directly in the distance computation [Joh97b]. If the colour components are defined in a vector $\mathbf{c} = (r,g,b)$[1], the square distance (2.5) can be written

$$d(\hat{\mathbf{p}}, \hat{\mathbf{x}}) = \begin{bmatrix} (x_p - x_x)^2 + (y_p - y_x)^2 + (z_p - z_x)^2 + \\ \Gamma_1 (r_p - r_x)^2 + \Gamma_2 (g_p - g_x)^2 + \Gamma_3 (b_p - b_x)^2 \end{bmatrix} \quad (2.14)$$

with

$$\hat{\mathbf{p}} = (x_p, y_p, z_p, r_p, g_p, b_p) \text{ and } \hat{\mathbf{x}} = (x_x, y_x, z_x, r_x, g_x, b_x)$$

where $\Gamma = (\Gamma_1, \Gamma_2, \Gamma_3)$ are the scale factors that weight the importance of colours against shape. The main problem with this method is to define the scale factors. A smart solution is found in 2.5.1.

2.4.3 Orientation matching

Some local geometric features like normal vectors (Figure 2.4) or curvature can also be used to improve the coupling and, more generally, the registration. Solutions to using geometric features include the following:

- Chen [Che92] calculates the closest point $\mathbf{q}_i$ for a point $\mathbf{p}_i$ from one surface by intersecting its surface normal $\mathbf{n}_{\mathbf{p}i}$ with the second surface (see Figure 2.2), using an iterative algorithm that needs an approximation of the closest point. Chen used a simple orthographic projection along the z axis. A more subtle solution is presented in 3.4.2.
- Use the closest compatible point presented above but create a subset of "compatible" candidates based on local curvatures [God95] or on the normals [Pul99].
- Add the normals to the distance computation in the same manner presented for colour in (2.14), swapping $\mathbf{c} = (r,g,b)$ for the normal vectors $\mathbf{n} = (u,v,w)$ [Fel94].

[1] The red green blue space is used as an example but any other colour space could be used.

- Brett [Bre99] uses the normals as a weighting of the Euclidean distance, replacing the squared distance (2.5) by:

$$d(\mathbf{p}, \mathbf{x}) = \frac{2}{\mathbf{n}_p \cdot \mathbf{n}_x} \|\mathbf{p} - \mathbf{x}\|^2 \qquad (2.15)$$

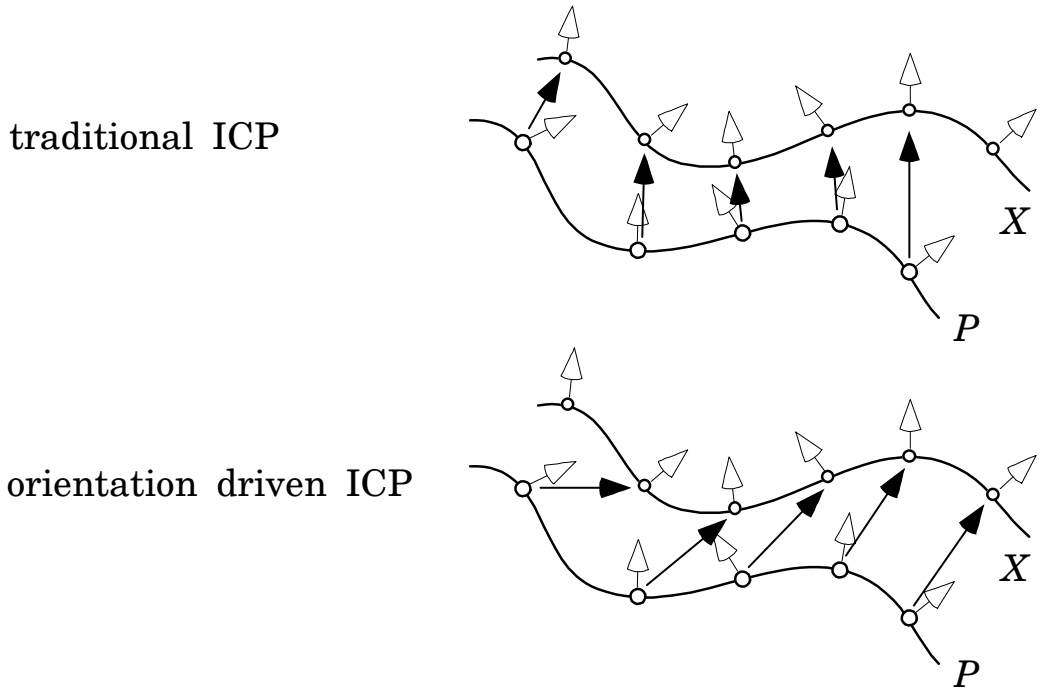


Figure 2.4: Influence of orientation on matching

2.4.4 Multi-feature matching

Some existing solutions combine several features with the geometric distance.

- Godin [God01] recently presented a solution based on the closest compatible point principle using multiple features. It also includes a pseudo-random sampling selection that chooses points with “interesting” alternative features. These features can be based on colour or geometry, but only quantities independent of the coordinate system, like curvatures.
- Schütz [Sch98c] proposed a single distance definition that combines surface geometry, normal and colour. Consequently, the feature vectors associated with the data can contain up to

9 components and the distance is the weighted sum of the squared distances of the different features:

$$d(\hat{\mathbf{p}}, \hat{\mathbf{x}}) = \frac{1}{\alpha_g}\left\|\mathbf{p} - \mathbf{x}\right\|^2 + \frac{1}{\alpha_n}\left\|\mathbf{n}_p - \mathbf{n}_x\right\|^2 + \frac{1}{\alpha_c}\left\|\mathbf{c}_p - \mathbf{c}_x\right\|^2$$

with

$$\hat{\mathbf{p}} = (\mathbf{p}, \mathbf{n}_p, \mathbf{c}_p) = (x_p, y_p, z_p, u_p, v_p, w_p, r_p, g_p, b_p)$$
$$\hat{\mathbf{x}} = (\mathbf{x}, \mathbf{n}_x, \mathbf{c}_x) = (x_x, y_x, z_x, u_x, v_x, w_x, r_x, g_x, b_x) \qquad (2.16)$$

where α_g, α_n and α_c are the scale factors that weight the different features. A proposed solution for the selection of the weights is found in 2.5.1. Schütz [Sch98c] showed that using this distance can greatly increase the quality of the matching (§2.8) of the ICP.

2.4.5 Influence on the convergence

The convergence theorem of section 2.3.2 isn't always valid anymore when using alternate distance for d_k computation. Consequently, the monotonous convergence of the ICP algorithm can't be proven anymore in some cases.

When features independent of the coordinate system, like curvatures or colours, are used in the distance d_k computation (2.10), the inequalities $e_k \leq d_k$ and $d_{k+1} \leq e_k$ remain true and the theorem is still valid. Since the features are not affected by the calculated rigid transformation (**R**, **t**) there is no need to include them in the matching error minimization indeed. This allows the use of the closed-form solution based on quaternions (section 2.6).

On the opposite, surface normals are affected by the rigid transformation. Therefore, and if the error minimization e_k remains only based on the geometric distance (2.11), the inequality $e_k \leq d_k$ and, consequently, the convergence theorem, aren't necessarily true anymore. However, results from the literature [Sch98c] and from our own experiments showed that practically, the behavior of the convergence is still nearly monotonous.

A solution to insure the monotonous convergence again consists in including the normals in the matching error minimization e_k. Unluckily, no closed-form exists anymore in this

case and a non-linear least-square method must be used to minimize e_k [Cha92] [Fel94], which is generally slower [Aru87].

2.5 Weighting the couplings

As said in 2.3, the basic ICP algorithm is sensible to noisy data and hidden geometry, e.g. outliers. Also, the algorithm was originally intended for object recognition and, thus, matches a P dataset that is a subset of X. This is not the case in object modelling or augmented reality [Sch97a], for example, where P and X generally only share a part of their respective geometry (see 6.3 and Figure 2.5). More generally, a problem occurs when the surface defined by point set P is not totally included in set X because some points of P don't have a correspondent in X and using a closest point approximation yields errors.

A solution to these problems consists in qualifying the closest point pairs by assigning weights to the couplings. In such case, the mean squared error of the couplings (2.7), becomes

$$e(\mathbf{R}_k, \mathbf{t}_k) = \frac{1}{W} \sum_{i=1}^{N_p} w_i \left\| \left(\mathbf{R}_k \mathbf{p}_{i,0} + \mathbf{t}_k \right) - \mathbf{y}_{i,k} \right\|^2 \qquad (2.17)$$

and a new step is added to the algorithm between step 1 and 2 to define the weights w_i. Fortunately, it doesn't change the way the minimization is performed (see 2.6.5).

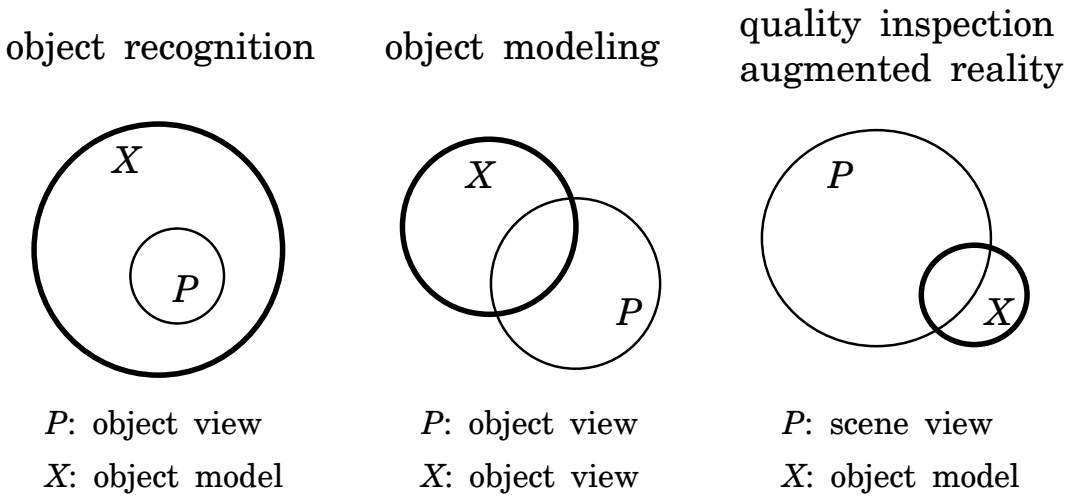


Figure 2.5: Different surface dispositions for the closest point search

We can recognize two distinct cases of weighting strategies. The first one consists in the elimination of bad couplings or binary weighting, where w_i can only take values of 0 or 1. The second case is the general case where weights can take any values.

2.5.1 Binary weighting

The main purpose of the elimination of couplings is to get rid of outliers, as explained above. Different strategies exist:

- Reject a coupling when its distance is bigger than a fixed threshold. τ typically depending on the resolution of the data.

$$w_i = \begin{cases} 1 & d(\mathbf{p}_i, \mathbf{y}_i) < \tau \\ 0 & \text{else} \end{cases} \tag{2.18}$$

  This method can easily be adapted to the multi-feature matching distance shown in equation (2.16) [Sch98b]. In this case, a threshold needs to be defined for each used feature. The normal threshold τ_n is derived from the corresponding maximal angle difference of two normal vectors. The colour threshold τ_c is defined by the maximal Euclidean distance between the corresponding colour vectors. Then, one can set the scale factors equal to the defined thresholds

$$\alpha_g = \tau_g,\ \alpha_n = \tau_n \text{ and } \alpha_c = \tau_c \tag{2.19}$$

  and the weighting function (2.18) can be expressed by

$$w_i = \begin{cases} 1 & d(\hat{\mathbf{p}}_i, \hat{\mathbf{y}}_i) < 3 \\ 0 & \text{else} \end{cases} \tag{2.20}$$

- Use an adaptive threshold depending on the mean μ and/or standard deviation σ of the distances. For example, Masuda [Mas96] used $\tau = 2.5\sigma$ and Zhang [Zha94] proposed $\tau = \mu + a\sigma$, where the integer a depends on μ.
- Discard the couplings that are not compatible with neighbouring couplings [Dor96b]. Couplings are considered incompatible if the distance d_i between a point and his closest point differs more that a threshold from that of the neighbouring couplings d_j.

- Eliminate couplings possessing points on mesh borders [Tur94].
- Eliminate the n% of couplings (typically 10%) that are farther apart [Pul99]. This method is a complement of the previous one and would be ineffective on its own.

Generally speaking, the methods that use a distance threshold give good results with the registration of partially overlapping datasets. Adaptive thresholds have a little advantage over fixed ones when the initial rough matching isn't very close to the final matching (see 6.3.2). The rejection of pairs possessing points on mesh borders is also used quite often as it avoids overlapping datasets to slip one over another [Tur94].

2.5.2 Other weighting methods

Generally, the idea behind assigning different weights to each pairings is to try to give more importance to the "better couplings". Proposed solutions to this problem include:

- Use the coupling distance d. Godin [God94] defined the weights as follows:

$$w_i = 1 - \frac{d(\mathbf{p}_i, \mathbf{y}_i)}{d_{\max}} \tag{2.21}$$

  where d_{max} is the biggest coupling distance. Another possibility is to use a function similar to the double threshold one (see section 3.2.3) with soft transitions from one class to another [Kre96].

- Use the normals of the points.

$$w_i = \mathbf{n}_{\mathbf{p}_i} \cdot \mathbf{n}_{\mathbf{y}_i} \tag{2.22}$$

- Use the colour of the points [God94].
- Use "uncertainty" of the measured points [Rus01]. A classic evaluation of the uncertainty of a measurement done with a range scanner consists in using the cosine of the angle Φ between the view direction $\mathbf{v}$ and the surface normal $\mathbf{n}$.

Basically, results from the literature show that general weighting methods don't have a big impact on the registration except in

some very specific cases. A beginning of explanation is that most of these weighting functions can be seen as "softer" - less effective - equivalents to solutions presented above in alternative distance computation (§2.4) and elimination of couplings (§2.5.1). Consequently, most ICP implementations rely on binary weightings only.

2.5.3 Influence on the convergence

Since the weights w_i can vary at each iteration, the inequality $d_{k+1} \leq e_k$ isn't always true. Consequently, the monotonic convergence of the ICP algorithm can't be proven anymore when using weights w_i in the error minimization e_k (2.17).

However, Zhang showed that the algorithm is well-behaved when using the adaptive threshold method showed in section 2.5.1. Furthermore, experimental results from the literature based on different weighing functions also show that the ICP algorithm converges successfully even if the weights change during the iterations [Zha94] [Sch98b] [Rus01].

2.6 Best transformation computation

As stated in equations (2.7) and (2.8), the problem of computing the best transformation consists in minimizing the mean squared error of the couplings $\{\mathbf{p}_{i,0}, \mathbf{y}_{i,k}\}$ as a function of $\mathbf{R}_k$ and $\mathbf{t}_k$. For the sake of simplicity, we will get rid of the iteration indexes k and we will minimize the sum of squared errors instead of their mean value (which produces exactly the same result).

The function we now consider for minimization is

$$e(\mathbf{R}, \mathbf{t}) = \sum_{i=1}^{N} \left\| (\mathbf{R}\mathbf{p}_i + \mathbf{t}) - \mathbf{y}_i \right\|^2 \qquad (2.23)$$

This problem could be solved by about any optimisation method, such as steepest descent for example, but it wouldn't be very efficient. Fortunately, several closed-form solutions exist to find the values of $\mathbf{R}$ and $\mathbf{t}$ that minimize e. They include using quaternions [Fau86] [Hor87], singular value decomposition (SVD) [Aru87], or dual number quaternions [Wal91].

Some evaluations and comparisons of them can be found in [Zha94] and [Egg97]. Basically, the conclusion of these works is that the different methods produce very similar results in term accuracy and stability. Speed isn't really a concern either since all these methods are relatively fast compared to the global running time of the ICP algorithm.

The quaternion method was chosen for this work and is summarized in the following sections.

2.6.1 Simplification of the problem with centroids

The main strategy to minimize e is to split equation (2.23) into simpler expressions and to try to decouple $\mathbf{R}$ and $\mathbf{t}$. Expanding (2.23) we obtain:

$$e(\mathbf{R}, \mathbf{t}) = \sum_{i=1}^{N} \left\| \mathbf{R}\mathbf{p}_i - \mathbf{y}_i \right\|^2 + 2\mathbf{t} \cdot \sum_{i=1}^{N} \left(\mathbf{R}\mathbf{p}_i - \mathbf{y}_i \right) + \sum_{i=1}^{N} \left\| \mathbf{t} \right\|^2 \qquad (2.24)$$

The first and third term of this expression contain only $\mathbf{R}$ or $\mathbf{t}$ and can only be positive or null. The second term on the other hand is the only one that still contains both $\mathbf{R}$ and $\mathbf{t}$. It means that this term needs to be eliminated to obtain an expression of e with decoupled $\mathbf{R}$ and $\mathbf{t}$. To do so, the datasets $\mathbf{p}_i$ and $\mathbf{y}_i$ are referred to their centroids:

$$\begin{aligned} \boldsymbol{\mu}_p &= \frac{1}{N} \sum_{i=1}^{N} \mathbf{p}_i, \quad \boldsymbol{\mu}_y = \frac{1}{N} \sum_{i=1}^{N} \mathbf{y}_i \\ \mathbf{p}'_i &= \mathbf{p}_i - \boldsymbol{\mu}_p, \quad \mathbf{y}'_i = \mathbf{y}_i - \boldsymbol{\mu}_y \end{aligned} \qquad (2.25)$$

Equation (2.23) can then be rewritten as

$$e(\mathbf{R}, \mathbf{t}) = \sum_{i=1}^{N} \left\| \left(\mathbf{R}\mathbf{p}_i - \mathbf{R}\boldsymbol{\mu}_p + \mathbf{t} \right) - \left(\mathbf{y}_i - \boldsymbol{\mu}_y \right) + \mathbf{R}\boldsymbol{\mu}_p - \boldsymbol{\mu}_y \right\|^2 \qquad (2.26)$$

and finally

$$e(\mathbf{R}, \mathbf{t}) = \sum_{i=1}^{N} \left\| \left(\mathbf{R}\mathbf{p}'_i + \mathbf{t}' \right) - \mathbf{y}'_i \right\|^2, \text{ with } \mathbf{t}' = \mathbf{R}\boldsymbol{\mu}_p - \boldsymbol{\mu}_y + \mathbf{t} \qquad (2.27)$$

We can expand this expression like we did in (2.24) except this time the mixed term is null because it basically sums up all the data points, which are referred to their centroids.

$$e(\mathbf{R}, \mathbf{t}) = \sum_{i=1}^{N} \left\| \mathbf{R}\mathbf{p}'_i - \mathbf{y}'_i \right\|^2 + \sum_{i=1}^{N} \left\| \mathbf{t}' \right\|^2 \tag{2.28}$$

The second term can now be set to zero, which leads to the best translation vector:

$$\mathbf{t} = \boldsymbol{\mu}_y - \mathbf{R}\boldsymbol{\mu}_p \tag{2.29}$$

Finally, only one term that isn't equal to zero remains. Consequently, the following equation has to be minimized to find the best rotation **R**.

$$e(\mathbf{R}) = \sum_{i=1}^{N} \left\| \mathbf{R}\mathbf{p}'_i - \mathbf{y}'_i \right\|^2 \tag{2.30}$$

To summarize, one can see that the problem is now to find the best rotation **R** that minimizes (2.30). After that, finding the best translation is trivial following (2.29).

2.6.2 Quaternions

This section presents some definitions about quaternions and their properties, needed for the computation of the best rotation.

Basically, a quaternion $\dot{\mathbf{q}}$ is a complex number containing four elements of the real space **R**, thus containing three imaginary components (i, j, k). It can be written as

$$\begin{aligned} &\dot{\mathbf{q}} = q_0 + q_x \mathrm{i} + q_y \mathrm{j} + q_z \mathrm{k} \quad \text{with} \\ &\mathrm{ii} = -1, \mathrm{ij} = \mathrm{k}, \mathrm{ji} = -\mathrm{k}, \mathrm{jk} = \mathrm{i}, \ldots \end{aligned} \tag{2.31}$$

A quaternion can also be represented as a vector with four components. The first one is a scalar while the remaining three correspond to an imaginary vector. It can be represented by

$$\dot{\mathbf{q}} = (q_0, q_x, q_y, q_z) = (q_0, \mathbf{q}) \tag{2.32}$$

A data vector is basically represented by a purely imaginary quaternion:

$$\dot{\mathbf{p}} = (0, \mathrm{p}_x, \mathrm{p}_y, \mathrm{p}_z) = (0, \mathbf{p}) \tag{2.33}$$

One can define the conjugate of a quaternion as

$$\dot{\mathbf{q}}^* = (q_0, -\mathbf{q}) \tag{2.34}$$

The product of two quaternions is defined by

$$\dot{\mathbf{q}}\dot{\mathbf{r}} = \left(q_0 r_0 - \mathbf{q} \cdot \mathbf{r}, q_0\mathbf{r} + r_0\mathbf{q} + \mathbf{q} \times \mathbf{r}\right) \neq \dot{\mathbf{r}}\dot{\mathbf{q}} \tag{2.35}$$

and one will note that it is not symmetric. Next, one can associate an orthogonal matrix $\mathbf{Q}$ to a quaternion so that the quaternion product can be expressed as a product of that matrix with the other quaternion. If

$$\mathbf{Q} = \begin{bmatrix} q_0 & -q_x & -q_y & -q_z \\ q_x & q_0 & -q_z & q_y \\ q_y & q_z & q_0 & -q_x \\ q_z & -q_y & q_x & q_0 \end{bmatrix} \text{ and } \overline{\mathbf{Q}} = \begin{bmatrix} q_0 & -q_x & -q_y & -q_z \\ q_x & q_0 & q_z & -q_y \\ q_y & -q_z & q_0 & q_x \\ q_z & q_y & -q_x & q_0 \end{bmatrix} \tag{2.36}$$

then, the quaternion product can be expressed by

$$\dot{\mathbf{q}}\dot{\mathbf{r}} = \mathbf{Q}\dot{\mathbf{r}} \neq \dot{\mathbf{r}}\dot{\mathbf{q}} = \overline{\mathbf{Q}}\dot{\mathbf{r}} \text{ and } \dot{\mathbf{r}}\dot{\mathbf{q}}^* = \overline{\mathbf{Q}}^T\dot{\mathbf{r}} \tag{2.37}$$

Rotations expressed by quaternions

Let's examine the following composite product

$$\dot{\mathbf{p}}' = \dot{\mathbf{q}}\dot{\mathbf{p}}\dot{\mathbf{q}}^* \tag{2.38}$$

where $\dot{\mathbf{p}}$ is a data vector in a quaternion form (2.33) and $\dot{\mathbf{q}}$ is a unit quaternion, where $\|\dot{\mathbf{q}}\| = 1$. It can be proved that this product represents a rotation of the data vector $\dot{\mathbf{p}}$. To do so, it is shown that the length of the data vector is not changed by this operation and that the dot and cross product are preserved [Hor87].

The properties of (2.37) are used to extract the rotation matrix defined by the quaternion $\dot{\mathbf{q}}$:

$$\dot{\mathbf{p}}' = \dot{\mathbf{q}}\dot{\mathbf{p}}\dot{\mathbf{q}}^* = (\mathbf{Q}\dot{\mathbf{p}})\dot{\mathbf{q}}^* = \overline{\mathbf{Q}}^T(\mathbf{Q}\dot{\mathbf{p}}) = (\overline{\mathbf{Q}}^T\mathbf{Q})\dot{\mathbf{p}} \tag{2.39}$$

The matrix $\overline{\mathbf{Q}}^T\mathbf{Q}$ has the form

$$\overline{\mathbf{Q}}^T\mathbf{Q} = \begin{bmatrix} 1 & \mathbf{0} \\ \mathbf{0} & \mathbf{R} \end{bmatrix} \tag{2.40}$$

where $\mathbf{R}$ is the 3x3 rotation matrix. It has the following value:

$$\mathbf{R} = \begin{bmatrix} q_0^2 + q_x^2 - q_y^2 - q_z^2 & 2(q_x q_y - q_0 q_z) & 2(q_x q_z + q_0 q_y) \\ 2(q_x q_y + q_0 q_z) & q_0^2 - q_x^2 + q_y^2 - q_z^2 & 2(q_y q_z - q_0 q_x) \\ 2(q_x q_z - q_0 q_y) & 2(q_y q_z + q_0 q_x) & q_0^2 - q_x^2 - q_y^2 + q_z^2 \end{bmatrix} \tag{2.41}$$

2.6.3 Finding the best rotation

Now we can look back at the minimization function (2.30) and use the quaternions as defined in (2.39).

$$e(\mathbf{R}) = \sum_{i=1}^{N} \left\| \mathbf{R}\mathbf{p}'_i - \mathbf{y}'_i \right\|^2 = \sum_{i=1}^{N} \left\| \dot{\mathbf{q}}\dot{\mathbf{p}}'_i\dot{\mathbf{q}}^* - \dot{\mathbf{y}}'_i \right\|^2 = e(\dot{\mathbf{q}}) \tag{2.42}$$

A development of $e(\dot{\mathbf{q}})$ (not shown here) leads to

$$e(\dot{\mathbf{q}}) = \sum_{i=1}^{N} \dot{\mathbf{q}}^T \left(\overline{\mathbf{P}}_i^T \overline{\mathbf{P}}'_i - 2\overline{\mathbf{P}}_i^T \mathbf{Y}'_i + \mathbf{Y}_i^T \mathbf{Y}'_i \right) \dot{\mathbf{q}} = \sum_{i=1}^{N} \dot{\mathbf{q}}^T \mathbf{A}_i \dot{\mathbf{q}} \tag{2.43}$$

Finally, the minimization function can be written

$$e(\dot{\mathbf{q}}) = \sum_{i=1}^{N} \dot{\mathbf{q}}^T \mathbf{A}_i \dot{\mathbf{q}} = \dot{\mathbf{q}}^T \mathbf{A}\, \dot{\mathbf{q}}, \text{ with } \mathbf{A} = \sum_{i=1}^{N} \mathbf{A}_i \tag{2.44}$$

Matrix $\mathbf{A}$ can be expressed again with data $\mathbf{p}'_i$ and $\mathbf{y}'_i$

$$\mathbf{A} = \sum_{i=1}^{N} \left\| \mathbf{p}_i \right\|^2 \mathbf{I} - 2\mathbf{B} + \sum_{i=1}^{N} \left\| \mathbf{y}_i \right\|^2 \mathbf{I}$$

$$\text{with } \mathbf{B} = \begin{bmatrix} S_{xx} + S_{yy} + S_{zz} & S_{yz} - S_{zy} & S_{zx} - S_{xz} & S_{xy} - S_{yx} \\ S_{yz} - S_{zy} & S_{xx} - S_{yy} - S_{zz} & S_{xy} + S_{yx} & S_{zx} + S_{xz} \\ S_{zx} - S_{xz} & S_{xy} + S_{yx} & -S_{xx} + S_{yy} - S_{zz} & S_{yz} + S_{zy} \\ S_{xy} - S_{yx} & S_{zx} + S_{xz} & S_{yz} + S_{zy} & -S_{xx} - S_{yy} + S_{zz} \end{bmatrix} \tag{2.45}$$

where S_{mn} are the sums of cross-correlations defined by

$$S_{mn} = \sum_{i=1}^{N} \mathrm{p}'_{i,m} \cdot \mathrm{y}'_{i,n} \text{ with } m, n \in (x, y, z) \tag{2.46}$$

Applying the technique of Lagrange multipliers to (2.43), we have to solve

$$\text{find } \min_{\dot{\mathbf{q}}} L = \left[\dot{\mathbf{q}}^T \mathbf{A} \dot{\mathbf{q}} + \lambda\left(1 - \|\dot{\mathbf{q}}\|^2\right)\right] \tag{2.47}$$

and do it by setting the partial derivatives of L to zero

$$\frac{\partial L}{\partial \dot{\mathbf{q}}} = \mathbf{A}\dot{\mathbf{q}} + \left(\dot{\mathbf{q}}^T \mathbf{A}\right)^T - 2\lambda\dot{\mathbf{q}} = \mathbf{0} \tag{2.48}$$

Since the matrix **A** is symmetric, (2.48) becomes

$$\mathbf{A}\dot{\mathbf{q}} = \lambda\dot{\mathbf{q}} \tag{2.49}$$

which shows that $\dot{\mathbf{q}}$ is an eigenvector of matrix **A**. Putting together (2.44) and (2.49) we can write

$$e_{\min}(\dot{\mathbf{q}}) = \dot{\mathbf{q}}^T \mathbf{A}\, \dot{\mathbf{q}} = \dot{\mathbf{q}}^T \lambda\dot{\mathbf{q}} = \lambda\dot{\mathbf{q}}^T\dot{\mathbf{q}} = \lambda\|\dot{\mathbf{q}}\|^2 = \lambda \tag{2.50}$$

and conclude that the quaternion that minimizes e is the eigenvector of matrix **A** corresponding to the smallest eigenvalue.

2.6.4 Summary of the quaternion method

The quaternion method to solve the minimization of the coupling error can be summarized as follows:

1. Refer data points to their centroid (2.25)
2. Build matrix **A** (2.45)
3. Find the smallest eigenvalue of **A** and its associated quaternion $\dot{\mathbf{q}}$
4. Get the best rotation matrix $\mathbf{R}(\dot{\mathbf{q}})$ (2.41)
5. Get the best translation vector **t** (2.29)

2.6.5 Influence of weightings

If weightings are used (see 2.5), the objective function (2.23) is

$$e(\mathbf{R}, \mathbf{t}) = \sum_{i=1}^{N} w_i \left\| (\mathbf{R}\mathbf{p}_i + \mathbf{t}) - \mathbf{y}_i \right\|^2 \tag{2.51}$$

The method remains the same but the weights make their apparition in two different steps. First, the centroids of both point sets (2.25) is now defined by

$$\boldsymbol{\mu}_p = \frac{1}{W} \sum_{i=1}^{N} w_i \mathbf{p}_i, \quad \boldsymbol{\mu}_y = \frac{1}{W} \sum_{i=1}^{N} w_i \mathbf{y}_i, \quad W = \sum_{i=1}^{N} w_i \tag{2.52}$$

Then, the matrix $\mathbf{A}$ (2.45) is

$$\mathbf{A} = \sum_{i=1}^{N} w_i \left\| \mathbf{p}_i \right\|^2 \mathbf{I} - 2\mathbf{B} + \sum_{i=1}^{N} w_i \left\| \mathbf{y}_i \right\|^2 \mathbf{I} \tag{2.53}$$

and the sums of cross-correlations S_{mn} are now defined by

$$S_{mn} = \sum_{i=1}^{N} w_i \left(\mathrm{p}'_{i,m} \cdot \mathrm{y}'_{i,n} \right) \text{ with } m, n \in (x, y, z) \tag{2.54}$$

2.7 Iteration termination

Using a fixed number of iteration isn't really a suitable option when using the ICP algorithm, mainly because of its high computational cost. Different methods have been proposed to automatically stop the iteration of the algorithm. A short commented summary follows.

- The absolute error criterion

  The iteration stops when the mean square error of the couplings falls below a threshold, $e_k \leq \tau$. The problem is that e_k is very sensitive to noise and τ is hard to set because it depends a lot on the data P and X and their configuration: basically, if τ is set too large, the iteration may stop too early, when the data are not correctly aligned. On the other hand, if τ is too low, it is very possible that the error e_k remains bigger even after complete convergence.

- The error change criterion

  The iteration stops when the change in mean square error of the couplings falls below a threshold, $e_{k-1} - e_k \leq \tau$ [Bes92]. This criterion is much less sensitive to noise and the threshold depends less on the data shape and configuration than in the absolute error case. It can be found in many implementations of the ICP.

- The pose change criterion

  The iteration stops when the change in the motion estimate falls below a threshold [Zha94]. The change in translation is defined as $\delta\mathbf{t}_k = \|\mathbf{t}_k - \mathbf{t}_{k-1}\|$. The change in rotation is computed using the rotation axis representation $\mathbf{r}$ defined as follows: $\theta = \|\mathbf{r}\|$ and $\mathbf{n} = \mathbf{r}/\|\mathbf{r}\|$. The rotation parameters can be extracted from the quaternion representation $\dot{\mathbf{q}}$ with

  $$\dot{\mathbf{q}} = \left(\cos\left(\tfrac{\theta}{2}\right), \mathbf{n}\sin\left(\tfrac{\theta}{2}\right)\right) \tag{2.55}$$

  The iteration stops when

  $$\delta\mathbf{t} = \|\mathbf{t}_k - \mathbf{t}_{k-1}\| \leq \tau_{\mathbf{t}} \text{ and } \delta\mathbf{r} = \|\mathbf{r}_k - \mathbf{r}_{k-1}\| \leq \tau_{\mathbf{r}} \tag{2.56}$$

- The complex error change criterion

  A new matching error ε is defined as the sum of the mean e and the deviation σ_e of the minimized squared error of the couplings (getting rid of the iteration index k for clarity):

  $$e = \frac{1}{N}\sum_{i=1}^{N} e_i \text{ and } \sigma_e = \sqrt{\frac{1}{N-1}\sum_{i=1}^{N}(e_i - e)^2} \tag{2.57}$$

  where $e_i = \|(\mathbf{R}\mathbf{p}_i + \mathbf{t}) - \mathbf{y}_i\|^2$ after the minimization of e. The error change criterion can then be applied to ε.

  $$\varepsilon_k = e_k + \sigma_{e_k}, \text{ stop when } |\varepsilon_{k-1} - \varepsilon_k| \leq \tau \tag{2.58}$$

  This error change criterion is mainly robust to bad couplings in object recognition [Sch98b].

2.8 Matching quality comparison, SIC ranges

Two measures can be considered to examine the quality of the matching of the ICP or one of its variants: the matching error and the domain of convergence.

Some applications need a very precise matching while others are less dependent on it. To observe the matching error, the algorithm must converge and the resulting alignment has to be compared to a defined "correct" matching (see below). Practically, the transformations are compared and the results are given as a rotation error e_φ and a translation error e_t.

As said before, the ICP algorithm needs a rough initial matching to converge correctly. To be efficient, some applications need to handle quite rough initial matching. It is the case for object recognition for example. The successful initial configuration (SIC) range provides a measure of how sensitive to the "roughness" of the initial matching an ICP variant and/or some data are.

2.8.1 SIC range

Examining the domain of convergence consists in finding the rough initial poses that lead to a successful matching. A successful initial configuration (SIC) is basically defined by a relative initial pose of the two datasets that leads to a successful matching. The initial configuration space possesses 6 dimensions (3 translation and 3 rotation parameters), so examining all of it isn't conceivable, due to both heavy computations and difficulty to handle the results.

We choose to use a dedicated setup, presented in [Hug97], to measure the range of successful convergence of two surfaces. The considered initial configurations are defined in a 3 dimensional space as follows: the SIC setup moves one surface (P) in several points on the circumsphere of the other one (X) (see Figure 2.6).

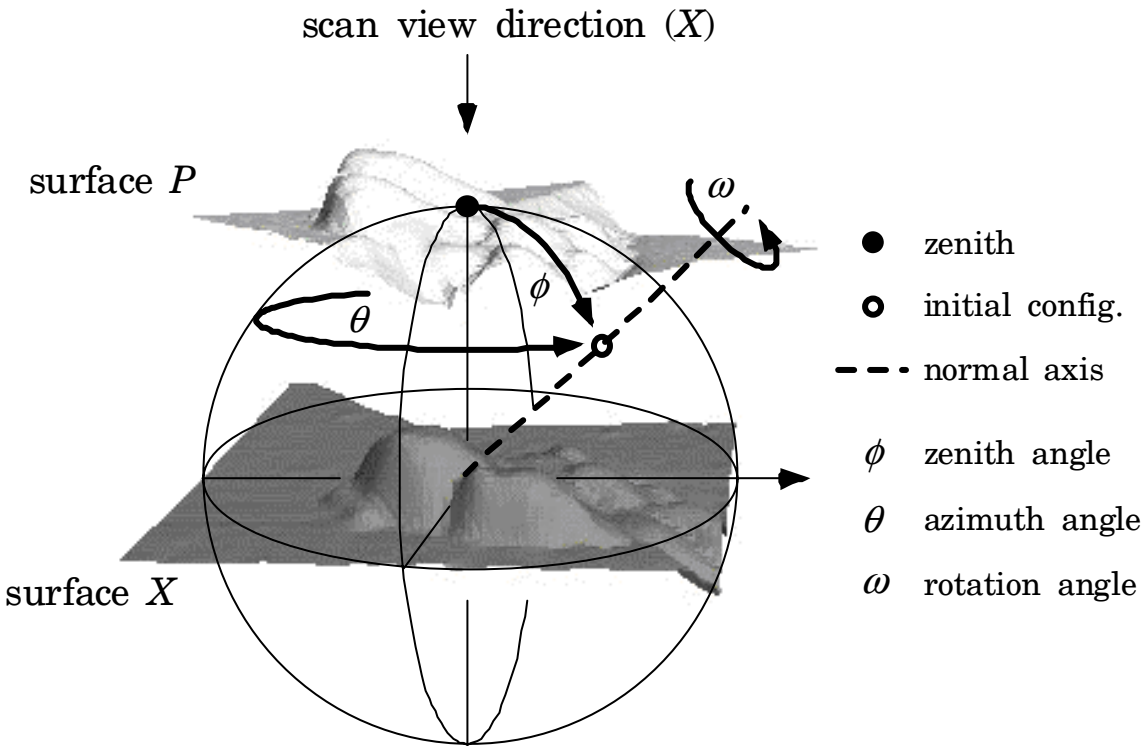


Figure 2.6: SIC range setup

Each of these configurations can be expressed by a zenith angle ϕ and an azimuth angle θ. Furthermore, at each current position on the circumsphere, the surface is rotated around its normal axis with an angle ω. The initial position, $\phi = 0$, is defined by the view axis of *P*. An initial configuration is thus defined by the 3 angles (ϕ, θ, ω).

2.8.2 SIC maps

Registrations are performed for each initial configuration and the results are plotted in a SIC-map. Each point on the circumsphere is represented by a circle using the following polar coordinates (radius, angle) = (ϕ, θ). ω is represented by sectors in the circles. Figure 2.7 shows the different angles on the SIC-map. An initial configuration is considered successful when the resulting registration position is closer to the correct matching than a certain threshold, in both rotation and translation. Those are plotted as black sectors on the SIC-map. Consequently, black areas in a SIC-map represent the range of successful initial configurations.

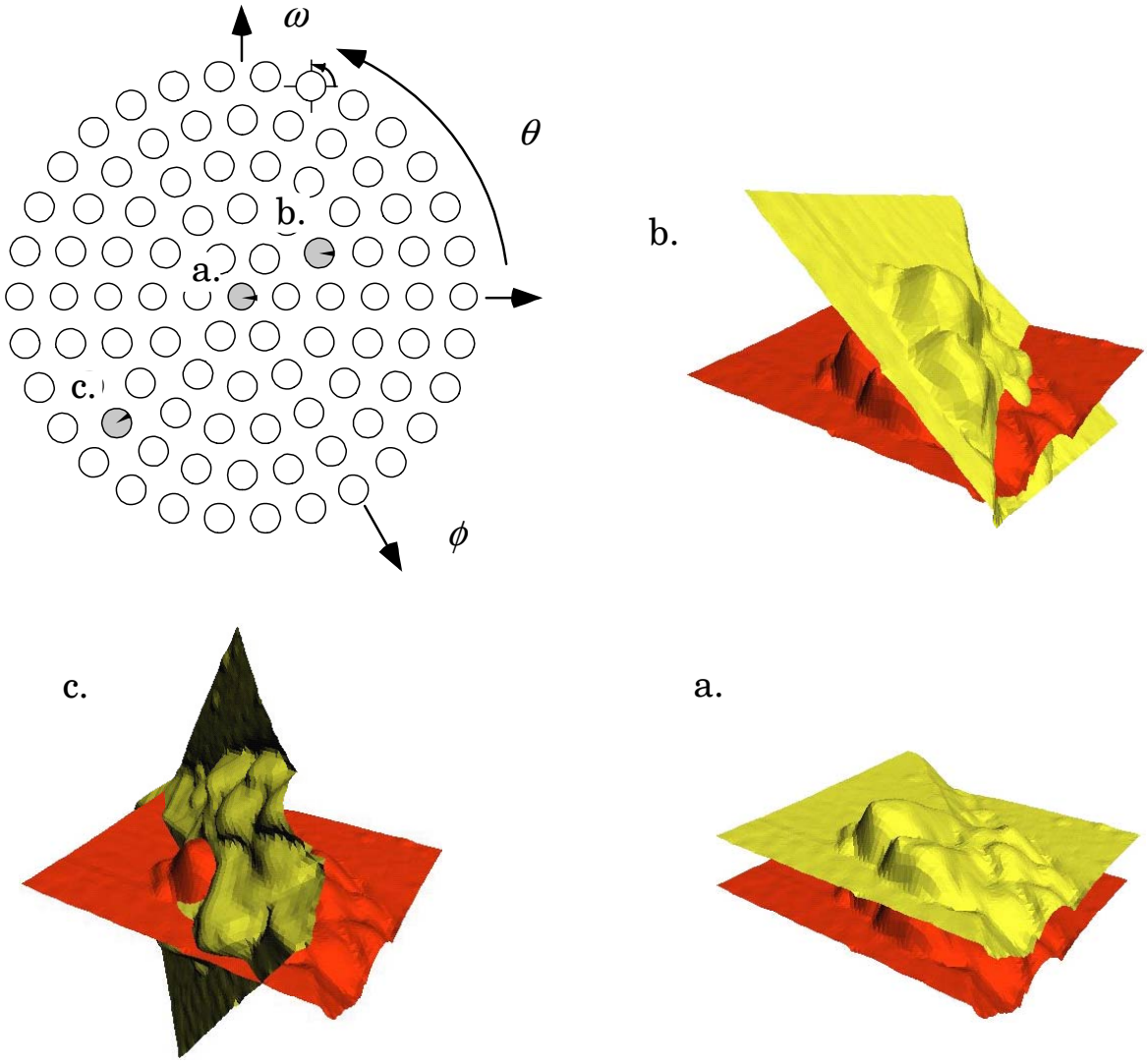


Figure 2.7: SIC map model and three example configurations

Chapter 3
Acceleration of the ICP Algorithm

Many solutions for the acceleration of the ICP algorithm have been proposed. A description and comparison of most of them are presented in this chapter.

3.1 Classification of acceleration methods

The time complexity of the different steps of the ICP algorithm, in its basic form, is as follows. The first step, the closest points computation, has a complexity of $O(N_p N_x)$. The next three steps, assigning weights, computing the registration and applying the registration, possess a complexity of $O(N_p)$. Consequently, the complexity of the ICP algorithm is $O(N_p N_x)$.

One of the main practical difficulties of the ICP algorithm is that it requires heavy computations. This is mainly due to closest points computation step, as its complexity is quadratic ($O(N_p N_x)$). Consequently, there is a need to accelerate the process, especially when working with large amounts of data.

Several authors have proposed solutions to speed up the algorithm. Langis [Lan01] recently proposed a parallel implementation of the ICP. He showed that a nearly linear performance improvement with the number of processors can be obtained with up to 16 processors. Beside this hardware-specialized implementation, one can separate the different

methods into three main classes: reduction of the number of iterations n, reduction of the number of data points N_p and N_x and acceleration of the closest points computation. A review of the different methods existing for each class and their results is given in the next paragraphs.

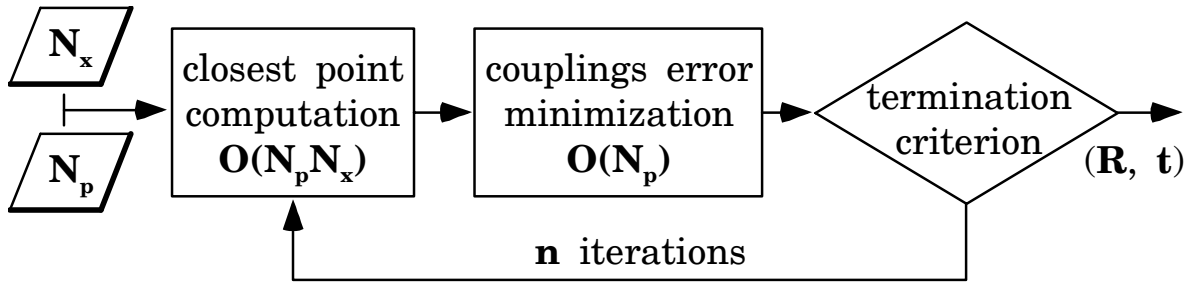


Figure 3.1: ICP principle

3.2 Reducing the number of iterations n

One of the possibilities to speed up the ICP algorithm consists in reducing the number of iterations needed for convergence. Of course, it means that an iteration stop criterion (§2.7) must be set instead of using a fixed number of iterations.

Solutions that can reduce the number of iterations include extrapolating the motion parameters and using extra features or special weighting functions. These methods are developed in the rest of section 3.2. One can also note here that Rusinkiewicz actually tested and evaluated the influence of many existing variations of the ICP on the number of iterations [Rus01]. Most of them didn't have much influence and, thus, are not cited in this section.

3.2.1 Extrapolation of matching parameters

A way to reduce the number of iterations is to extrapolate the matching parameters to estimate the minimum of the objective function. Besl proposed an *"accelerated ICP"* based on this principle [Bes92].

The registration parameters $\dot{\mathbf{q}}$ (the relation between $\dot{\mathbf{q}}$ and the rotation matrix $\mathbf{R}$ is found in 2.6) and $\mathbf{t}$ can be combined in a 7-dimensional global registration vector

$$\overline{\mathbf{q}} = (\dot{\mathbf{q}}, \mathbf{t}) = (q_o, q_x, q_y, q_z, t_x, t_y, t_z) \tag{3.1}$$

Let's consider the vector representing the difference between two consecutive iterations k-1 and k

$$\Delta\overline{\mathbf{q}}_k = \overline{\mathbf{q}}_k - \overline{\mathbf{q}}_{k-1} \tag{3.2}$$

which defines a direction in the registration state space. The angle between the last two directions in the 7D space is

$$\cos\theta_k = \frac{\Delta\overline{\mathbf{q}}_k \cdot \Delta\overline{\mathbf{q}}_{k-1}}{\|\Delta\overline{\mathbf{q}}_k\|\|\Delta\overline{\mathbf{q}}_{k-1}\|} \tag{3.3}$$

An extrapolation of the objective function e in the registration state space can be executed if the last three registration vectors $\overline{\mathbf{q}}_k$, $\overline{\mathbf{q}}_{k-1}$ and $\overline{\mathbf{q}}_{k-2}$ are well aligned, i.e., if

$$\theta_k < \delta\theta \text{ and } \theta_{k-1} < \delta\theta \tag{3.4}$$

with $\delta\theta$ being a sufficiently small angular threshold ($\delta\theta$ = 10° typically).

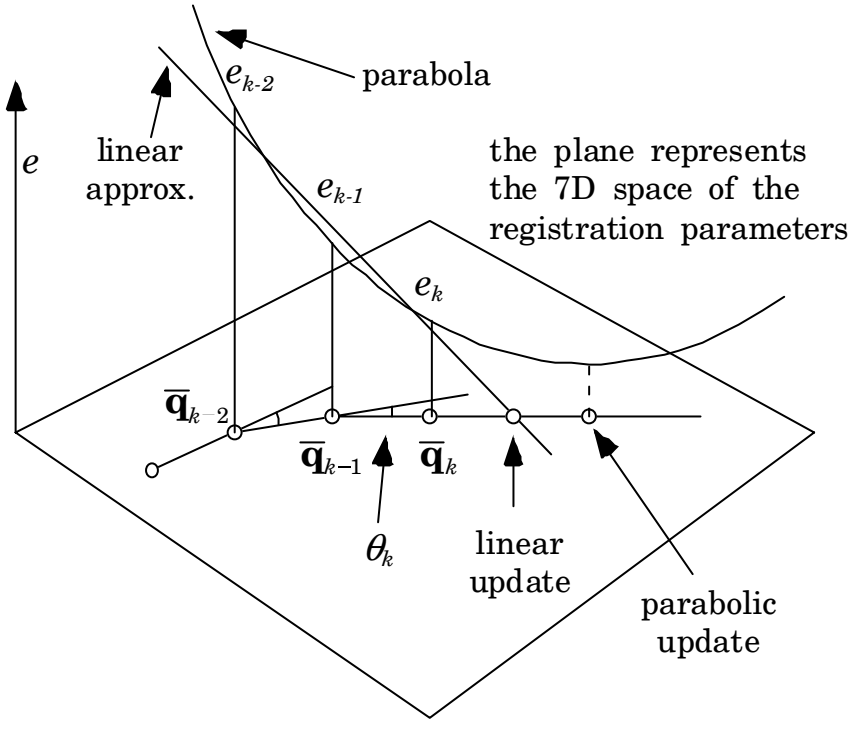


Figure 3.2: Extrapolation of the registration parameters

For clarity, let's write the associated mean square errors e_k, e_{k-1} and e_{k-2}, and the associated approximate arc length

$$v_k = 0, v_{k-1} = -\left\|\Delta\overline{\mathbf{q}}_k\right\| \text{ and } v_{k-2} = -\left\|\Delta\overline{\mathbf{q}}_{k-1}\right\| + v_{k-1} \tag{3.5}$$

Now, we can extrapolate the objective function e using a linear approximation e_1 and a parabolic interpolation e_2 of the last three points we just defined (see Figure 3.2):

$$e_1(v) = a_1 v + b_1 \text{ and } e_2(v) = a_2 v^2 + b_2 v + c_2 \tag{3.6}$$

Using these two functions gives us a possible linear update v_1, based on zero crossing of the line, and a possible parabolic update v_2, based on the minimum point of the parabola:

$$v_1 = -\frac{b_1}{a_1} \text{ and } v_2 = -\frac{b_2}{2a_2} \tag{3.7}$$

A maximum allowable value v_{max} is then set to avoid extreme values ($v_{max} = 25v_{k-1}$). The final update value v_0 is chosen among those three possible values being the smallest positive one.

$$v_0 = \min(v_1, v_2, v_{\max}) \text{ and } v_0 > 0 \tag{3.8}$$

Finally, the updated registration vector is defined by

$$\overline{\mathbf{q}}'_k = \overline{\mathbf{q}}_k + v_0 \frac{\Delta\overline{\mathbf{q}}_k}{\left\|\Delta\overline{\mathbf{q}}_k\right\|} \tag{3.9}$$

Simon [Sim95] proposed to decouple rotation and translation in the accelerated ICP to reduce the number of iterations further more.

One of the main problems of this method is that overshoot can happen, which would at best eliminate the beneficial effect of the method and which can be especially annoying if it causes the algorithm to converge to a wrong local minimum. Therefore, a test is necessary to ignore updated registration vectors that cause a mean square error worse than the original registration.

A solution to reduce the chances of overshoot is to multiply the amount of extrapolation v_0 by a dampening factor like 0.5 [Rus01]. Of course this also reduces the benefits of the extrapolation.

Practically, the accelerated ICP and its decoupled version reduce the number of iterations by factors of up to 3 and 4.5 respectively.

3.2.2 Additional features

Some of the methods to improve coupling with extra features, presented in 2.4, can have a significant action on the number of iterations.

Figure 3.3 shows an example of matching of two coloured surfaces of a toy rabbit (result from [Sch98b]). The error e is plotted at each iteration for the multi-feature distance (2.16). One can basically see that using more features reduces the number of iterations needed for convergence.

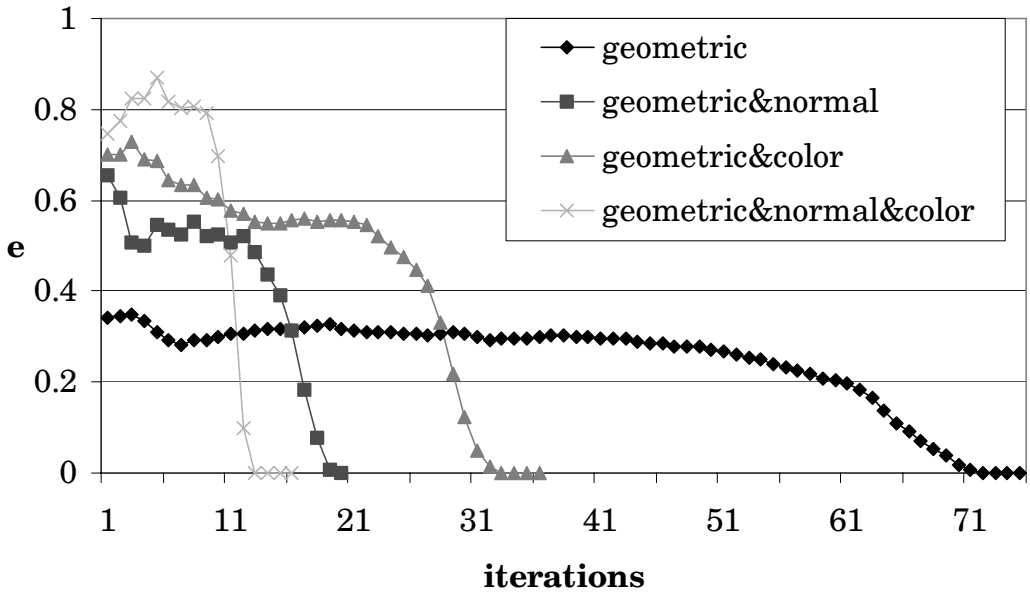


Figure 3.3: Convergence of ICP using multiple features [Sch98b]

Practically, the reduction of the number of iterations due to the use of extra features is of course very dependent on the data and features that are considered. Colours, for example, are only useful in certain cases. On the other hand, experiments showed that the normals have a very beneficial effect on the number of iterations in most cases.

Figure 3.4 shows the convergence of the error e versus time. This permits to see that the gain in number of iterations is counterbalanced by the longer searches due to the use of the multi-feature distance (see §3.4.1). All in all, the use of extra

features doesn't have much influence on the global matching time but mainly on the quality of the matching.

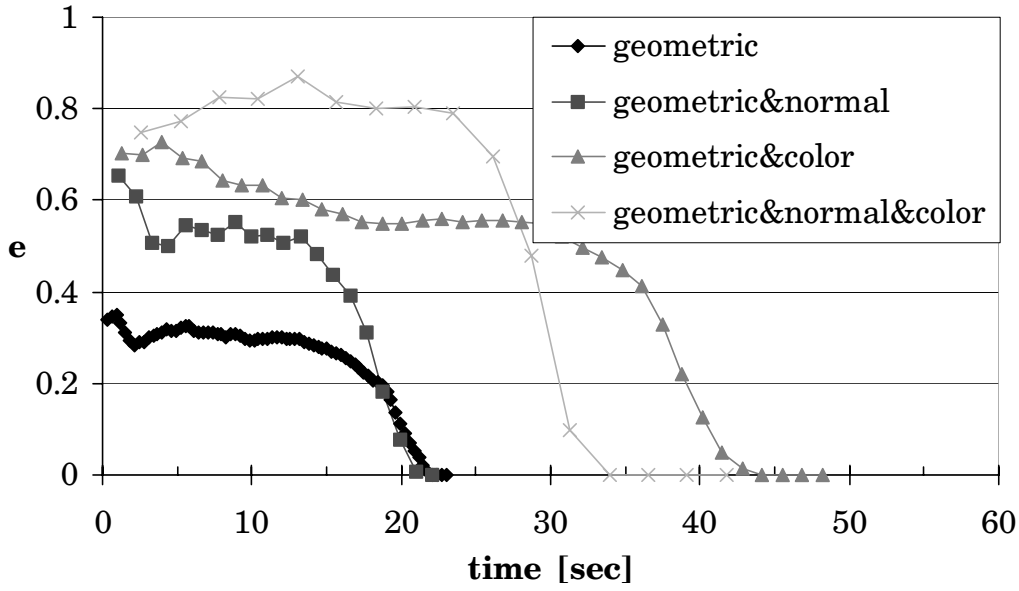


Figure 3.4: Convergence of ICP versus time using multiple features [Sch98b]

3.2.3 Weighting the couplings

Most weighting methods don't have much influence on the number of iterations. However, the basic binary weighting function to eliminate outliers, found in (2.18), can be slightly modified to accelerate convergence:

In this case, we use two distance thresholds and, thus, three different values for w_i (0, 1 and 4 in this case) as shown in Figure 3.5 [Sch98b]. Instead of a single threshold separating inliers and outliers, this function omits outliers as well but also attributes a lower weight to points that possess a nearby closest point. This results in a faster convergence since inliers that are still farther apart get more influence and outliers do not disturb the matching error minimization.

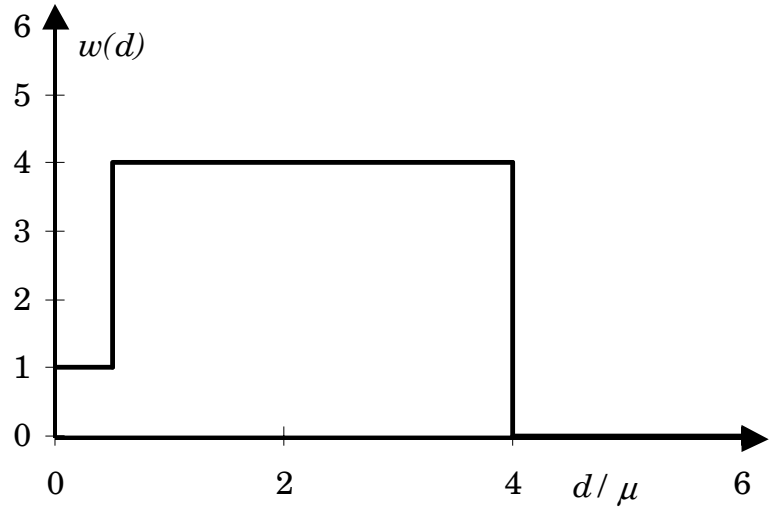


Figure 3.5: Double threshold weight function for closest point couplings [Sch98b]

Figure 3.6 shows the evolution of the matching error over several iterations for a typical ICP matching converging successfully. The double threshold weight function permits the reduction of the number of iterations needed to reach a minimum error by 50%.

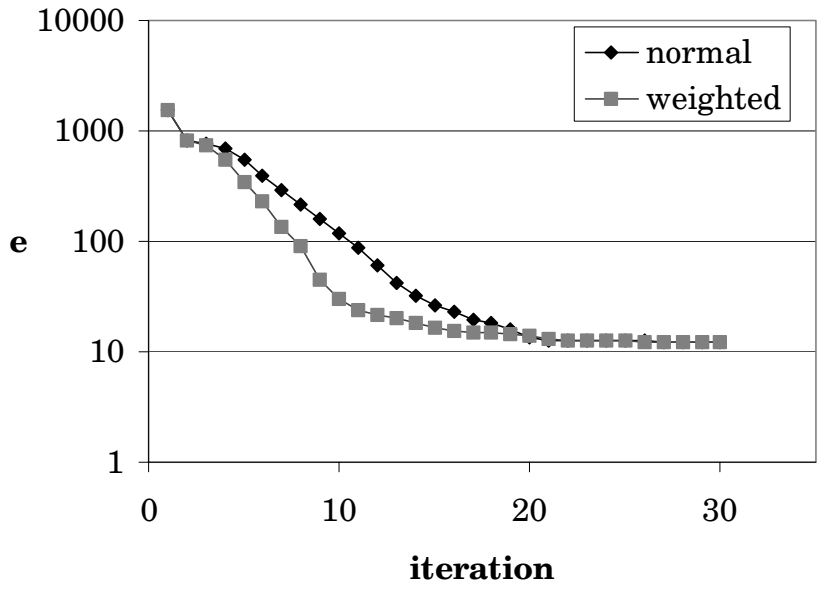


Figure 3.6: Matching acceleration with weighted couplings [Sch98b]

3.3 Reducing the number of data points N_i

Another good solution to speed up the ICP algorithm is to use subsets of the data P and, sometimes, X instead of the full datasets. Part of the solutions following this principle simply use a single subset of the data, called control points, through all iterations while others use a coarse to fine strategy. The closest compatible point principle exposed in 2.4 also reduces the number of points in X in its own way as discussed below.

3.3.1 Closest compatible point

The *closest compatible points* principle consists in first building a subset of the points of X that have a feature distance below a certain threshold, then finding the closest point (using geometric distance) among these "compatible" candidates.

As long as the considered features are independent of the coordinate system, like curvature or colour [God94] [God01], the subsets of compatible candidates only need to be computed once. On the other hand, using normals is computationally expensive since the subsets need to be recomputed at each iteration.

We can note that it is impossible to estimate the possible gain in speed because the size of the subsets depends a lot on the feature used and on the data. As long as the extra features aren't very discriminating, there is a good chance that the subsets still contains a high proportion of the points of X. So generally, this method requires a big memory use and doesn't have such a high potential for speeding up. An additional problem is that is that no further speeding up method can be easily applied to the closest points search (§3.4) since there are many different subsets of X.

3.3.2 Control points

Several authors proposed to only select a subset of points, called *control points* [Che92], instead of using full data. Generally, control points are selected as a subset of dataset P. On the other hand, the dataset X is kept at full resolution to still permit a precise matching between both sets.

The basic benefit of using control points regarding speed can be estimated easily. If N_c control points are selected in a dataset P that contains N_p points, the computation of each ICP iteration is basically N_p / N_c time faster since all steps of the algorithms are performed using only the N_c control points.

Different strategies have been proposed to select the control points and are presented below.

- Perform a uniform subsampling of points in the range image.
- Use a random subset. Masuda [Mas96] suggested using new random subsets at each iteration.
- Select points with help from additional features. For example, keep points with high image gradients in the intensity image [Wei97] or select points having a distribution of the normals as large as possible [Rus01]. Finally, Godin proposed generalization of the use of additional features (called attributes in this work) to orient a random sampling [God01].
- Use a mesh decimation algorithm that keeps significant feature (high curvature) on triangle meshes [Bre99].
- Keep points sitting in smooth areas [Che92]. This argument is especially valid when using point-to-plane distances because normals and line-plane intersections are more reliable in that case.

The main problem with this solution is that the resulting registration can lack precision because the whole matching is done using only a part of the data (see 5.4.1). Solutions that try to select points “intelligently”, like using mesh decimation, may have a positive effect on the result precision but they also have a fairly high computational cost.

3.3.3 Coarse to fine strategy

A *coarse to fine* strategy can easily be applied to the ICP algorithm. The main advantage of a coarse to fine strategy over choosing control points is that the final precision of the matching is expected to be the same as when using the full resolution all the way.

Proposed solutions execute the first iterations using a subset of X, like 1/4 of the points uniformly sampled [Tur94] or 1/5 randomly chosen points [Zha94], and finish with fine matching using all the points. Zhang found an experimental gain in speed of about 2 to 4 using this method.

In this case, the acceleration gain is greatly dependent on the number of iterations performed at the different resolutions. So far, few results have been published concerning coarse to fine strategies for the ICP. A generalized *multiresolution* scheme applied to the ICP and combined with fast closest points search is formulated and analysed in Chapter 5 of the present work.

3.4 Speeding up the closest points computation

As seen before, the closest points search step has a basic complexity of $O(N_p N_x)$ and most computation time of the ICP algorithm is spent in it. To illustrate the problem, Table 3.1 shows the percentage of computation time spent for closest points search for different number of points, as provided by a practical experiment.

Number of points N	1000	4000	16000	64000
Closest points search comp. %	93.0%	97.2%	98.2%	98.8%

Table 3.1: Percentage of computation time spent for closest points search

The acceleration of the closest points search can be done using either search structures, as shown in the next section, or using projection methods, which are very specific to the ICP, as shown in section 3.4.2. The section 3.4.3 presents the k-D tree search into more details, as it is the reference fast closest points search method chosen in this work.

3.4.1 Search structures

There exist different structures that permit to find the closest point in a dataset without browsing the full set and, thus, that

speed up the process. The main methods using a search structure and proposed to speed up the closest point computations of the ICP algorithm are summarized below.

- The *multidimensional binary search tree, or k-D tree* uses a divide-and-conquer strategy to speed up the closest points computation [Bes92] [Zha94]. This method is the classical fast closest point search method used to speed up the ICP and we consider it as our reference fast ICP solution in this work. Consequently, it is discussed and presented into more detail in section 3.4.3.

- The *Elias method* [Gre00] divides the space into congruent sub-regions called bins. Each bin is assigned a list of the points of X that it contains. The closest point of the considered point $\mathbf{p}$ is first searched in the bin it belongs to. If the distance to the found closest point is bigger than the distance to any other bin or if the bin is empty, then the search is performed in these bins until the distance to the closest point is smaller than the distance to any unexplored bins.

  The exact complexity of the method can't be defined but if all the bins contain at least a point, the maximum number of adjacent bins that may need to be examined is 3^k-2. This number basically increases if some of the bins are empty. Thus, this method is not suited for multi-feature data since the search time rises up quickly in this case.

- The *distance field method* [Tub02] also divides the space into congruent sub-regions called voxels (from volume cells). The closest point correspondences between the considered point set X and the voxel centers are stored, which allows direct access to the closest points.

  One of the biggest drawbacks of this method is that there is a big tradeoff between memory requirements and resolution, and consequently quality of the matching. Furthermore, this method is not suited for multi-feature (high dimensional) data since the memory requirement of the distance field explodes in this case. Consequently, it has been rarely considered in the context of the ICP until recently and for specific applications (see [Tub02]).

- The *Spherical Triangle Constraint Nearest Neighbour (STCNN)* method has been developed specifically for the speedup of the ICP [Gre01]. The idea is to keep a list of the points situated in the ε-neighbourhood – all points closer than ε – of each point of *X*, sorted by distance. At runtime, the correspondence of the previous iteration is used as a first approximation of the closest point. A first constraint (the spherical one) permits to tell if the real closest point falls in within the ε-neighbourhood of the estimate. If it is the case, the triangle inequality is used to efficiently prune the points of the ε-neighbourhood. Else a classical search method (like the ones presented above) is used.

  There isn't much feedback since the method is new, but some conclusions can be taken from Greenspan paper. The good side is that, globally, the method is a bit faster than the k-D tree and Elias methods. Furthermore, and in the best case, less than 2 distance computations are needed on average. On the other hand, the method seems to have several downsides. One of them is that it requires an important storage size and the pre-processing is complicated (finding all the points within a distance threshold and sorting them, for each points of *X*). Another downside is common to most of the structure search methods and is explained below.

Most of these search structure methods are effective if the data cover the whole parameter space. Consequently, they suffer from the fact that the datasets considered in the ICP are surfaces, and that a lot of the space is "free" of data. First of all, most methods are ineffective for points of *P* that aren't close to the set *X*. This is the case for the first iterations when datasets are only roughly aligned and also when they only partially overlap. The problem is detailed for the k-D tree in 3.4.3. Since a lot of the voxels or bins are empty, it also leads to a very inefficient memory use in the distance field method.

3.4.2 Projection methods

The goal of projection methods is to speed up the closest points search by projecting points into one or more planes, reducing the problem to a 2D search. Unlike the methods presented above,

these solutions are typically adapted to the ICP algorithm and use the fact that data are usually representing surfaces to their advantage. It should be noted here that due to their nature, they generally define approximations of the closest points instead of real ones. Of course the approximations can well be the real closest points in some cases.

Two main solutions based on projection methods have been proposed in the literature.

- The *reverse calibration* [Bla95] consists in projecting the points of one dataset into the range image of the second one, in the direction of the range camera. Practically, one must invert the scanner calibration parameters so that one gets the coordinates (u, v) in range image X of a point (x, y, z) of P, expressed in dataset X reference frame.

  Weik proposed to refine the search in the range image by using the gradient of the corresponding intensity image [Wei97]. Neugebauer used this method to obtain the first approximation of the closest point needed for point-to-plane matching (§2.4.1) [Neu97]. Of course, this method relies on working with range images and, more importantly, on knowing the calibration parameters of the 3D scanner, which can be restrictive.

- Benjemaa proposed to use one [Ben95] or several [Ben96] reference projecting planes called *Z-buffers*. In case a single Z-buffer is used, both point sets are projected on the same plane and local searches -in an n x n window- are performed to refine the correspondences. The multi-Z-buffers solution uses the normal information to assign a point to a plane (Z-buffer). The normal space is basically divided in 216 parts, each of those defining a plane perpendicular to its mean normal (the associated Z-buffer). Then, points are projected in their respective Z-buffers and the local searches are performed in each Z-buffer.

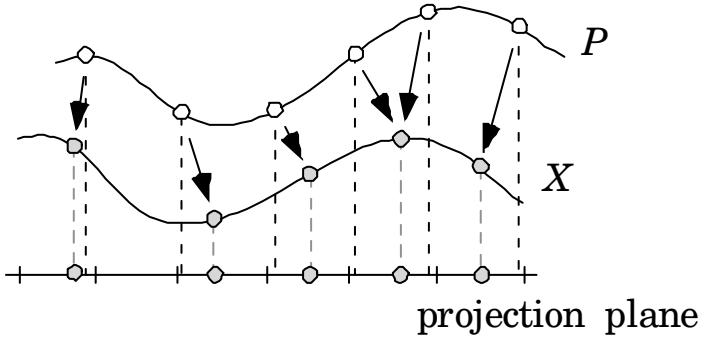


Figure 3.7: Projection method principle

One of the problem of this method is that the points of *P* need to be projected in their respective Z-buffer at each iteration of the ICP, which isn't very optimal computationally speaking. Another drawback, common to all projection methods, is that a very good initial matching is required for correct convergence. Practically, the size of the *n* x *n* window is chosen to be of the same order as the error of the initial matching.

Most of these methods possess a complexity of $O(N_p)$, which is much better than the original $O(N_pN_x)$ one and often permits a very good speeding up of the ICP algorithm. On the other hand projection methods rely on a very good initial approximation of the matching and can't be adapted very well to using extra matching features, i.e., the resulting ICP is less robust [Rus01]. Consequently, they imply an important tradeoff between speedup and quality of matching (as defined in 2.8).

3.4.3 k-D tree search

Bentley introduced the general idea of the k-D tree back in 1975 [Ben75]. The main idea of the method is to generalize bisection in one dimension to the n-dimensional case [Pre86]. The main advantage of the k-D tree is that it permits to use multi-feature data (defined in equation (2.16)) and to compute the real closest point, unlike projection methods. Therefore, it doesn't imply a tradeoff between the speedup of the ICP and the quality of the matching.

Building the k-D tree

An example of the construction of a k-D tree is shown in Figure 3.8 for a 2D case (*xy*-plane, k=2). A starting point of the set is first chosen, as well as one of the feature space dimensions, t, named key. Both items are assigned to the first node of the tree called root node. In our example, the root node contains point p_5 and key x. Then, the points are subdivided by a line of equation $x = x(p_5)$. Every point that possesses a smaller x coordinate than p_5 will be found on the left side of the root node (p_1, p_2, p_3 and p_4), the other points (p_6, p_7 and p_8) being on the right side. One of the points of the left side and a key are then chosen and assigned to the left son of the root. In our case, it is point p_2 and key y. Once again, the remaining points are subdivided, this time by the line $y = y(p_2)$.

The construction continues like this until all points have been assigned to a node of the tree. If a subdivision implies that no points are left on a side of a node, then an empty node, called a leaf, is placed. In our example, a square and a letter represent the leaves of the tree to see the relation between them and the empty rectangles they define.

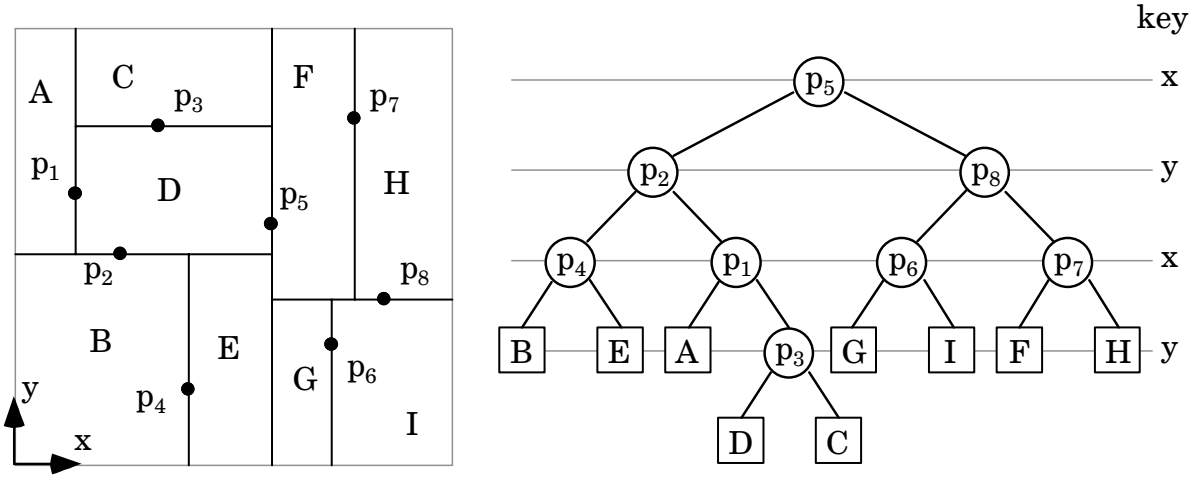


Figure 3.8: Example of a 2-D tree construction

The generalization to k-dimensions is straightforward. The cutting lines in 2D are replaced by hyperplanes in k-dimensions, keeping the same type of equation ($t = t(p_i)$). The complexity for building the tree is O(NlogN) and it uses a storage θ(N).

Closest point search in the k-D tree

Different algorithms to search closest point using a k-D tree have been proposed by Friedman [Fri77] and Zhang [Zha94]. Both are recursive and their main idea is to perform the search in a k-dimensional sphere centred on the test point **p** for which the closest point is searched. Furthermore, the radius of the sphere is reduced during the search, which permits the search to be more efficient. We used Zhang's search algorithm in this work since it is simpler and easier to understand than Friedman's one.

The search algorithm works as follows:

First, the sphere radius D is set to a maximum value D_{max} and the current node ν is the root of the k-D tree T. Then, if the sphere overlaps the hyperplane defined by the key $t(\nu)$ of the current node, the search algorithm is called recursively for both left and right sons. Otherwise, only the son lying on the side of the test point **p** needs to be searched. Furthermore, if the distance between **p** and the current node point $\mathbf{x}(\nu)$ is smaller than D, D is reduced to this distance and the closest point **y** is set to $\mathbf{x}(\nu)$. The recursive search stops if a node has no children. Finally, and unless the closest point was further than D_{max}, **y** corresponds to the correct closest point and D is equal to the closest point distance.

Formally, the search for the closest point to **p** is made by calling Search(root(T), **p**) of the following procedure:

- **input**: a test point **p**, a k-D tree T of a point set X, a closest point **y** and a distance $D = D_{max}$
- **output**: the closest point **y** and the corresponding distance D
- **procedure** Search(ν,**p**)

```
if(ν == leaf) return;
if(¦p[t(ν)] - x(ν)[t(ν)]¦ < D) then
 if(d(p, x(ν)) < D) then
   y = x(ν), D = d(p, x(ν));
if(p[t(ν)] – D < x(ν)[t(ν)]) then
   Search(leftson(ν),p);
```

```
if(p[t(v)] + D > x(v)[t(v)]) then
    Search(rightson(v),p);
```

Optimization of the k-D trees

To be of maximum efficiency, the tree needs to be as balanced as possible, i.e., the branches have to have similar lengths. The selection of the point and key assigned to each node has an influence on the shape of the k-D tree. Points can be selected randomly, but this could result in an unbalanced tree, especially in the ICP case, because data generally represent only a part of the space (surfaces). Bentley proposed to use the median value for the selected key [Ben75], which effectively permits to create a balanced tree of with branches of average length $\log N_x$.

Usually, the keys are chosen in a cyclic order, a key being assigned to each level in the tree. It is the case in the example of Figure 3.8. Friedman proposed to use, at every non-terminal node, the key with the largest spread in values [Fri77]. The advantage of doing so is that the partitioning is more regular and the probability of the sphere overlapping the hyperplanes is least (averaged over all the query locations).

Another optimisation proposed by Friedman is to keep small subsets of data in the terminal nodes, called buckets (instead of leaves). Then, when a bucket is reached in the search, distances are computed with each points in the bucket. If one examines only the number of distance computations, then the tree is optimal for buckets containing a single point. On the other hand, it generally involves more backtracking with the recursive search. A value of around 10 to 20 points per bucket was shown to be optimal regarding the time of execution.

Performance and limitations related to the ICP

The best case complexity of the closest point search in a k-D tree is $O(\log N_x)$, where N_x is the number of point in X [Ben75] [Fri77]. The number of distance computations due to backtracking is expected to be 2^k and, thus, constant for a given dimension k. On the other hand, the worst-case complexity is $O(N_x^{1-1/k})$ [Pre86]

[Zha94], which is particularly inefficient. Fortunately, results from the literature and from our experiments (see sections 4.6 and 5.4) show that, practically, the performances of the k-D tree search are closer to the expected behavior.

Let's examine the particular case of the ICP. As said before, the particularity of the data considered in the ICP is that they are surfaces, which only represent a small part of the 3D space. When a test point **p** is far from the data X, the search is much less effective since all coupling distances are of similar length and much bigger than the average distance between the points of X. The discrimination power of single keys falls short of separate points with similar distances. This effect can be seen in Figure 3.9. It shows the time required for the closest point search at each iteration of the ICP for a typical registration.

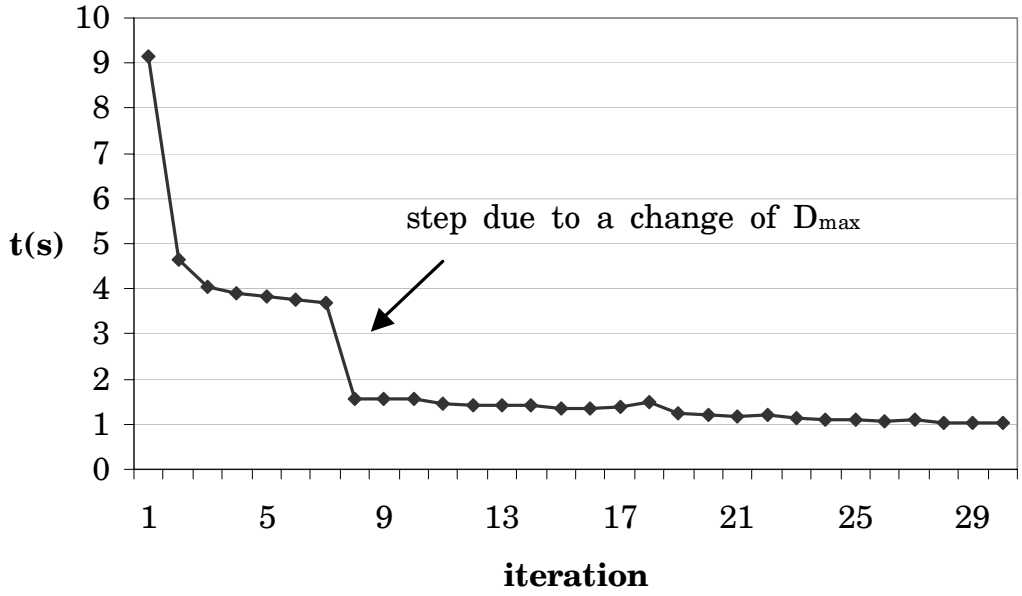


Figure 3.9: Average search time using a k-D tree vs. iteration for a typical registration with a very rough initial matching

Another specificity of the closest point search in the ICP is the value of D_{max}. In the general case, D_{max} is set to infinity to be sure that the closest point is found. This is not very efficient since the larger D_{max} is, the more branches must be inspected. As seen in 2.5.1, a distance threshold τ can be used to get rid of outliers in the ICP. All couplings with a distance bigger than τ

are not considered in the registration computation. This can be used to our advantage because we can then set $D_{max} = \tau$. The effect of setting a different D_{max} can be seen in Figure 3.9. The value of τ decreases between iterations 7 and 8 and the resulting reduction of the search time can be clearly seen.

Other smart methods to set D_{max} exist, like using the distance of the previous coupling, be it spatially [Sch98b] or temporally [Sim95].

3.5 Summary of acceleration methods

Table 3.2 presents a summary of the ICP acceleration methods.

General class	**Name/author**	**Method**	**Comments**
Hardware acceleration	Parallel ICP algorithm [Lan01]	Parallel implementation	• linear gain with nb. CPUs (up to 16 CPUs)
Reduction of n	Accelerated ICP [Bes92]	Extrapolation of the registration parameters	• gain factor of up to 3 - risk of overshoot
Reduction of n	Decoupled Accelerated ICP [Sim95]	Extrapolation of rotation and translation parameters	• gain factor of up to 4 - risk of overshoot
Reduction of n	Double threshold weightings [Sch98b]	Give more weight to farther inliers	• gain factor of around 2
Reduction of N_i	Random control points [Mas96]	Random selection of a subset of N_c points	• gain of N_p / N_c - lack of precision
Reduction of N_i	Control points [Che92] [Bre99] [God01]	"Intelligent" selection of a subset of N_c points	• gain of N_p / N_c - risk of a lack of precision - big computational cost
Reduction of N_i	Coarse to fine matching [Zha94] [Tur94]	Execute the first iterations at a lower resolution	• gain factor of around 2-4

Acceleration of the closest point search	k-D tree [Bes92] [Zha94]	Use a multidimensional binary search tree	• $O(N_p \log N_x)$ - not very effective on rough initial matching
Acceleration of the closest point search	Elias method or distance fields [Gre01] [Tub02]	Divide the space into congruent sub-regions	- big memory use - not adapted for extra-features
Acceleration of the closest point search	STCNN method [Gre01]	Use spherical and triangle constraints with previous corresp. caching	• $O(N_p)$ best case - ineffective on rough initial matching
Acceleration of the closest point search	Reverse calibration [Bla95] [Wei97]	Project points in the range image plane using calibration parameters	• complexity $O(N_p)$ - need a very good initial matching - scanners calibration parameters must be known
Acceleration of the closest point search	Z-buffer(s) projection [Ben95] [Ben96]	Project points in Z-buffers and perform local searches	• complexity $O(N_p)$ - need a very good initial matching

Table 3.2: A summary of ICP acceleration methods

3.6 Discussion and conclusion

This chapter presented and compared most of the solutions for the acceleration of the ICP algorithm. Three main types of acceleration methods have been defined: reduction of the number of iterations, reduction of the number of data points and acceleration of the closest points search.

One can note here that all three types of acceleration methods are quite independent and consequently can be combined together. For example, Simon [SIM95] mixed accelerated ICP with k-D tree and Zhang [ZHA94] used both coarse to fine strategy and k-D tree. Recently, Rusinkiewicz [Rus01] mixed a projection method with a selection of control points to obtain a very fast ICP.

Reduction of the number of iterations n only has a minor impact on the total running time of the ICP compared to the other two classes of solutions. Reduction of the number of points can be very efficient, computationally speaking, especially when

using only a subset of the data (control points). However the quality of the matching can suffer since details can disappear. Finally, the acceleration of the closest points search permits to reduce the complexity of the ICP - to $O(N_p \log N_x)$ or even $O(N_p)$ - and generally has the biggest impact on the speedup, especially for large datasets. Projection methods are the most efficient in term of speed but their main downside is that a very good starting approximation is generally needed for a successful matching.

As we can see, a lot of the existing solutions generally imply a tradeoff between speed and quality of matching. Keeping this in mind, this work concentrates on trying to maintain the quality of matching while speeding up the ICP.

In this context and examining the existing solutions, a coarse to fine approach and a k-D tree closest point search seems to be the most appropriate methods. In the next chapter, a solution for fast closest point search is presented and compared to the k-D tree approach in terms of speed and quality of matching. Then, a general multiresolution coarse to fine scheme is proposed and analysed in Chapter 5.

Chapter 4
The Neighbour Search Algorithm

In this chapter, a novel algorithm for fast closest point search to speed up the ICP is presented. It consists of a heuristic that uses neighbourhood relationships to obtain a first approximation of the closest points and refines the results by a local search. Part of the presented work has been published in [Jos02a].

4.1 The neighbourhood relationship hypothesis

The proposed algorithm assumes the existence of a neighbourhood relationship between the two sets of points P and X. Given that there exist neighbourhoods V and V' defined in respectively datasets P and X, the relationship hypothesis is that two neighbours in a dataset possess closest points that are neighbours in the other dataset. Formally, the principle of this neighbourhood relationship is exposed in Figure 4.1: given a point $\mathbf{p}_k$ in dataset P and its corresponding closest point $\mathbf{x}_k$ in dataset X, the closest point $\mathbf{x}_i$ of $\mathbf{p}_i$, if $\mathbf{p}_k$ belongs to neighbourhood V of $\mathbf{p}_i$, V($\mathbf{p}_i$), is found in the neighbourhood V' of $\mathbf{x}_k$, V'($\mathbf{x}_k$).

The proposed idea towards a faster search is to use good approximations of the closest points instead of exact closest points. The neighbourhood relationship is used to get a first approximation of the closest point and, then, a local search can be performed to refine the result instead of an exhaustive one: if

$\mathbf{p}_i$ possesses a neighbour $\mathbf{p}_k$ in dataset P, with a know closest point $\mathbf{x}_k$ in dataset X, finding the closest point of $\mathbf{p}_i$ can be reduced to searching the closest point in the neighbourhood V' of $\mathbf{x}_k$, V'($\mathbf{x}_k$).

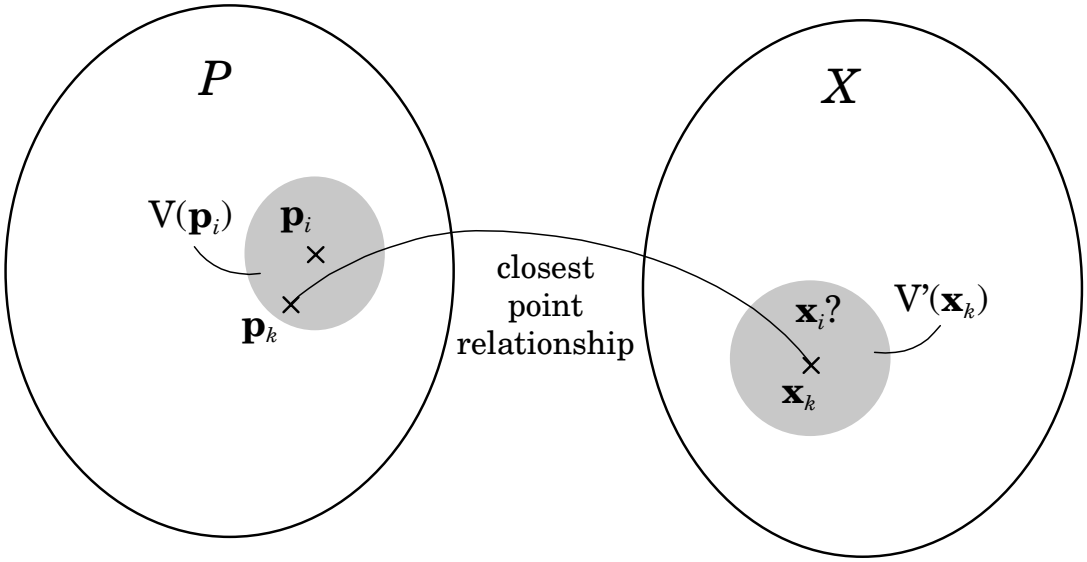


Figure 4.1: The neighbourhood relationship assumption

4.2 Basic algorithm

The following pseudo-code formulates the closest point neighbour search algorithm:

- **input**: datasets P and X, with associated neighbourhood, V(P) and V'(X)
- **output**: for each $\mathbf{p_i}$ of P, an approximation $\mathbf{x_i}$ of its closest point in X
- **procedure** neighbour_closest_point_search (P, X)

```
for (each p_i of P) do
 if (∃ x_k closest point of p_k ∈ V(p_i)) then
   x_i = search_closest_point (p_i, V'(x_k) );
 else
   x_i = search_closest_point (p_i, X);
```

It appears that for each point $\mathbf{p}_i$, the closest point search is performed either in the full set X or only in the neighbourhood $V'(\mathbf{x}_k)$, depending wether or not at least one neighbour of $\mathbf{p}_i$ has already a known closest point. Formally, this closest point search algorithm has therefore a theoretical best-case complexity of $O(N_p)$. This is better than the one using k-D tree, $O(N_p \log N_x)$ and lets us expect an overall gain in speed over using a tree search.

The worst-case complexity corresponds to the conventional full search discussed so far. A good suggestion here is to use a tree search, instead of an exhaustive search, when searching the full set X - other potential ideas would be to use a temporal cache or a heuristic method like 3 steps algorithm, in range images -. Using this method, it is interesting to note that the neighbour search algorithm cost worst case is basically equal to the tree search $O(N_p \log N_x)$, as long as the cost of the local search is smaller than a tree search of course.

Of course, the order in which points $\mathbf{p}_i$ of P are scanned is also important. Using a random method is a bad idea, as it would create a high number of global searches and pushes complexity toward the worst case. Consequently, the basic idea is to scan points using a diffusion method, such as the next point to be scanned is chosen in the neighbourhood of the points that already have a known neighbour.

Concerning convergence, ICP can't be proven to converge anymore when using approximations of the closest points.

One can note here that the proposed algorithm can be combined with the other methods of acceleration like reduction of the number of iteration or reduction of the number of data points.

4.3 Data structure considerations

Structured datasets are needed to have a defined neighbourhood. In many cases, it already exists a priori or at least can be built. Generally speaking, applications using 3D registration rely on either range images, clouds of points or triangle meshes as their input data. A neighbourhood exists in a range image, where direct neighbours in the image are also neighbours on the surface

(with a few exceptions explained below). It also exists in the case of triangle meshes, where neighbourhood is defined by belonging to the same triangle. In the case of a cloud of points, an existing triangulation algorithm (like the ball-pivoting algorithm [Ber99] for example) can be used to create a triangle mesh. Such a neighbourhood is needed in both the P mesh, to obtain closest point approximation, and the X mesh, to make the local search.

4.3.1 Resolution variation

The efficiency of the presented algorithm is related to the existence of the closest point $\mathbf{x}_i$ of $\mathbf{p}_i$ in the neighbourhood $V'(\mathbf{x}_k)$. Practically, the resolution of the data has a direct influence on this and is affected by different factors like the resolution of the scanner, the direction of the scan and the existence of hidden surfaces.

If the datasets that are being matched don't have approximately the same resolution, or at least resolutions in the same range, the neighbourhood relationship hypothesis is still valid but the size of the local search relatively to the resolution isn't anymore. In this case, the size of the local search needs to be adapted. Fortunately, for most applications the resolution of the datasets is roughly the same, as long as the used scanning tools remain in the same configuration for the different measurements.

The problem that remains is that, even with the same scanning resolution, some local changes in resolution can appear. This is caused by the fact that the viewing angle on an overlapping surface part is often different between scans, as shown in Figure 4.2. The size of the local search directly influences the exactness of the matching: the bigger the size, the higher the chance that the real closest point will be found. A discussion about this subject oriented toward range images can be found in section 4.4.1.

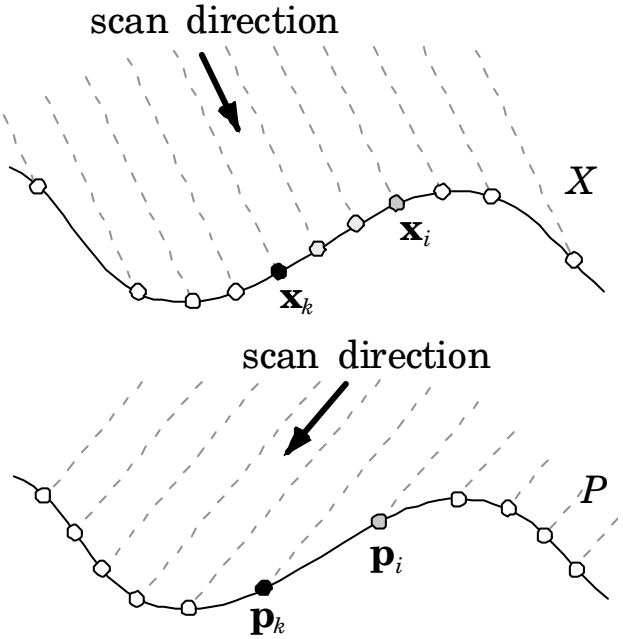


Figure 4.2: Influence of local resolution variation on the neighbour search

The situation is worse when hidden surfaces are found in the data. Figure 4.3 presents such a case. It presents a surface with a "hole" in it. The concavity created by the hole is mostly hidden in one of the view, set as *P* in this example. Consequently, points 2 and 3 are neighbours in the range image (they correspond to two successive measurements) but they are not on the surface. On the other hand, most of the surface is visible in view *X*. The example uses a local search in a neighbourhood of size n=5 and starts from point 1. Unfortunately, using the neighbour search results in some very bad couplings in this case. Basically, this problem exists only when using range images since points 2 and 3 wouldn't be considered neighbours otherwise, as explained in the next section. A fairly easy solution to this problem is proposed in 4.4.2.

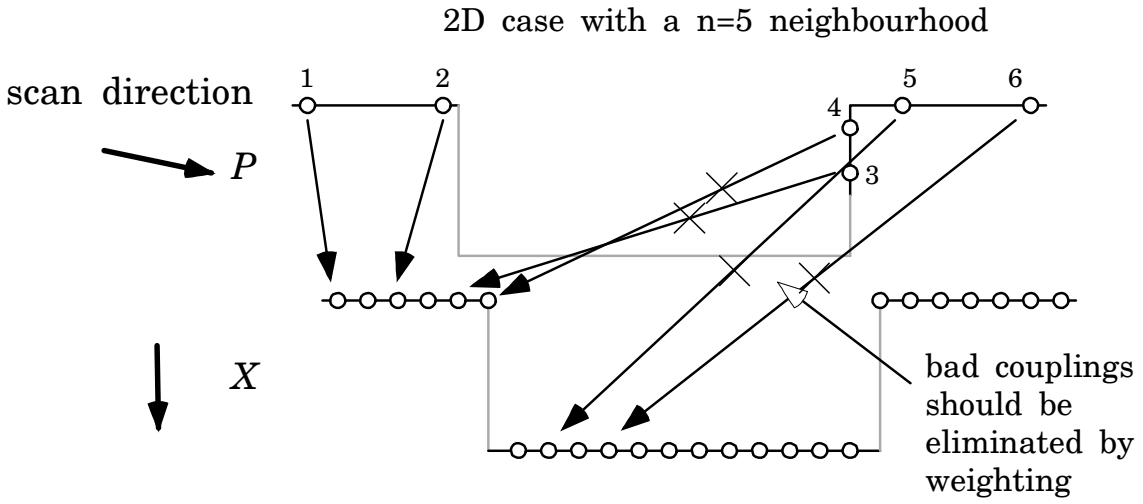


Figure 4.3: Bad couplings created by hidden surfaces

4.3.2 Unconnected datasets

A problem that has to be taken into consideration is that some dataset can contain different subsets that aren't connected together in term of neighbourhood. This is the case for triangle meshes in which multiple surface patches could be found. For example, a triangulation of the dataset P of Figure 4.3 would certainly create 2 unconnected surface patches separated between points 2 and 3, instead of a single surface "covering" the hole (see section 6.2). Background, shadows and specular reflections can be responsible for missing data points in range images obtained from optical range finders. The presence of several missing data points can also create unconnected subsets of data in range images. These unconnected datasets can have negative effects in both P and X datasets, as explained below.

On dataset P, there can be situations where several points $\mathbf{p}_i$ have not a single valid neighbour $\mathbf{p}_k$. In that case, the number of global closest point searches increases, which can degrade the performance of the algorithm in term of speed. However, as long as a smart method is used to scan the points of set P, the number of global search is basically equal to the number of unconnected subsets. This means that in most datasets, the number of global search is insignificant versus the number of points.

On dataset X, this can indeed lead to cases where closest point local searches get "stuck" in a subset due to neighbourhood restrictions. It creates situations with more bad approximations of closest points and bigger errors. An interesting remark can be done before trying to actually solve this problem: assuming weights are considered in the closest point couplings in ICP, to get rid of outliers (2.5.1), most points belonging to an erroneous coupling will be seen as outliers and consequently won't have an impact on the registration (as for the bad couplings of Figure 4.3). This means that the quality of the registration will be affected a bit by the fact that less closest point pairs are taken into account to compute the transformations but won't be too affected by bad approximations.

That said, one can easily conclude the dataset containing the less subsets is preferably considered as X, even if the cost of the search may be a bit higher. A good way to handle this could be to alternate P and X for both datasets. In the case of a triangle mesh, the only way to reduce the number of bad approximations is to create additional neighbourhood relationships for points sitting on boundaries of the subsets. These could be for example the closest point in a different subset. Simpler solutions exist for range images and are discussed in the next section.

4.4 Algorithm applied to range images

When considering range images, each point possesses either 4 or 8 direct neighbours (except points on borders), depending on the considered topology (V-4 or V-8 neighbourhood). A very basic algorithm is considered here. Neighbourhood V is the 3x3 window surrounding the point $\mathbf{p}_i$ in P (V-8 neighbourhood) and neighbourhood V' is a nxn window in X. We choose to scan the points of range image P row by row, starting from upper left. That way, the possible direct neighbours of $\mathbf{p}_i$ with a known closest point $\mathbf{p}_k$ can be found on the previous point in the same row and in the previous row (see image P on Figure 4.4). Those 4 possible candidates are just checked sequentially and the first one that possesses a known closest point is chosen as $\mathbf{p}_k$.

Normally, any of the candidate neighbours possesses a known closest point, except for the first scanned point and in case of missing data points.

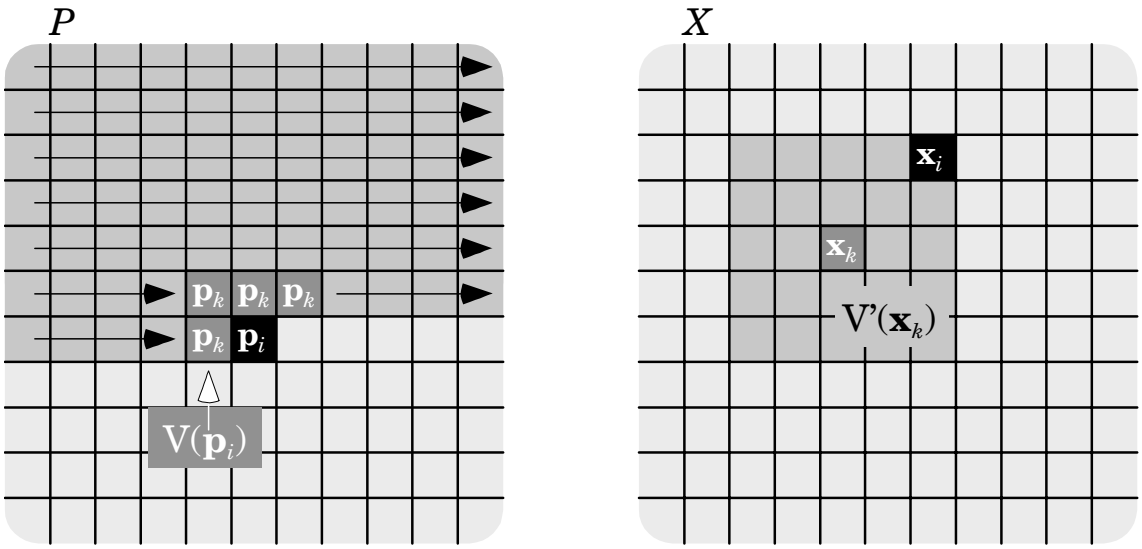


Figure 4.4: Neighbour search using range images

Once $\mathbf{p}_k$ and its corresponding closest point $\mathbf{x}_k$ are known, the local closest point search of $\mathbf{p}_i$ is done in a square neighbourhood zone of size nxn, centred on the approximation $\mathbf{x}_k$ (see image X in Figure 4.4). If no point $\mathbf{p}_k$ can be found, a k-D tree search is performed in X, as suggested previously.

The pseudo-code of the closest point neighbour search algorithm for range images, combined with a tree search, is looking as follows:

- **input**: 2 range images, P and X, and a tree search structure for image X, Xtree
- **output**: for each point $\mathbf{p_i}$ of P, an approximation $\mathbf{x_i}$ of its closest point on X
- **procedure** neighbour_closest_point_search (P, X, Xtree)

```
for (each pi of P sequentially) do
 for (each neighbour candidate pk)
  if (pk ≠ NULL ) then
    xi = zone-search_closest_point (pi, X, xk);
    end for each pk;
 xi = tree-search_closest_point (pi, X, Xtree);
```

4.4.1 Selection of the local search size *n*x*n*

As seen above, the size of the local search directly influences the exactness of the matching. Of course, the bigger the zone, the better the approximation, but the longer the search as well, so a compromise has to be found. A simple reflection to estimate an efficient size for the local search is made in this section.

If the datasets P and X have been taken using the same scanner resolution 1 / *r*, the relation between the resolution of the scanner and the resolution of the data 1 / *d* depends of the angle of view and is given by

$$d_p = \frac{r}{\cos \alpha_p} \text{ and } d_x = \frac{r}{\cos \alpha_x} \tag{4.1}$$

as shown in Figure 4.5.

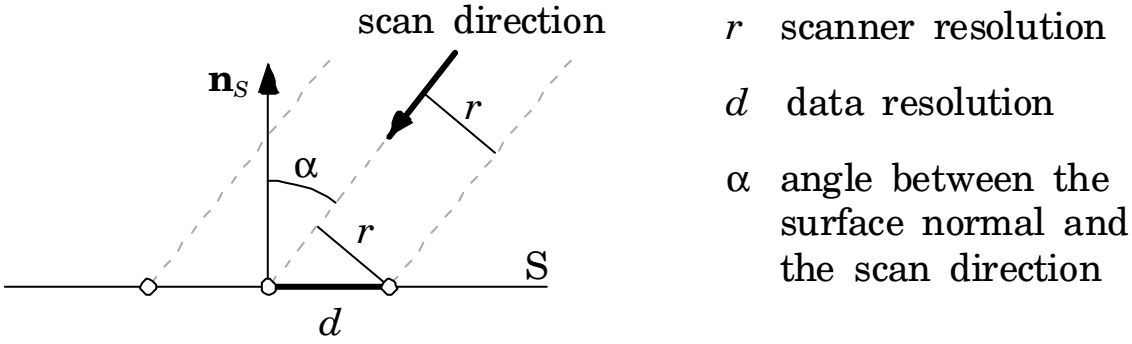


Figure 4.5: Comparison of the resolution of the data vs. angle of view.

The distance between a point $\mathbf{p}_i$ and its chosen neighbour $\mathbf{p}_k$ is approximately d_p. Similarly, the distance between the point $\mathbf{x}_k$ and its direct neighbours $\mathbf{x}_j$ in the range image is roughly d_x. If we consider that there is no problem related to hidden surface or missing data (§4.3), we can also estimate that the distance between their respective closest points $\mathbf{x}_i$ and $\mathbf{x}_k$ is equal to d_p. Consequently, the size of the *n*x*n* search window in the range image *X* must at least be equal to

$$n \geq 2 \frac{d_p}{d_x} + 1 = 2 \frac{\cos \alpha_x}{\cos \alpha_p} + 1 \tag{4.2}$$

to be sure that $\mathbf{x}_i$ is found in it.

The worst situation appears when a plane is scanned and dataset X has maximum resolution (a small d_x), which happens when $\alpha_x = 0$. Table 4.1 shows the relation between the required size of the local search and the maximum angle of view of dataset P.

n	5	7	9	11
α_p **max**	60°	70°	75°	78°
number of distance computation (nxn)	25	49	81	121

Table 4.1: Relation between the required size of the local search and the maximum angle of view of dataset P

These results mean, for example, that with a 9x9 search zone, all couplings should be correct as long as the surface of P doesn't present angles of more than 75° with the scanner. Practically, we expect a 7x7 or 9x9 zone to be enough since most of the data will present a smaller view angle. If the angle of view is bigger, the correspondence is erroneous and there is a risk that errors will spread and add up. Fortunately, there are simple checks to avoid this case and they are presented in the next chapter.

4.4.2 Problems and performance

It was shown in 4.3.1 and 4.4.1 that problems could arise when the distance between $\mathbf{p}_i$ and a candidate neighbour $\mathbf{p}_k$ in the range image P is too big. A simple way to avoid this problem is to compare the range values of the point and its neighbour. Basically, if the difference between the ranges is bigger than a threshold, the candidate shouldn't be valid. This check only adds a maximum of 4 subtractions / comparisons per points of P, which isn't very costly. Furthermore, one could imagine stocking a list of the valid candidate neighbours to avoid repeating the checks at each iteration.

Another check can be used to avoid the propagation of bad matching. It verifies the coupling weight of the candidate

neighbour $\mathbf{p}_k$, which is chosen only if it has a valid coupling, unless none of the candidates have one.

4.5 Experimental setup

The neighbour search ICP algorithm was tested on several different data. In this section, we introduce a few typical datasets we used to obtain the presented results. A general description of the different experiments follows.

4.5.1 Datasets used

Set 1

This first set of experiments is done using two copies of the same dataset. The advantage of matching a dataset with itself is that the exact matching is known a priori. It would also have been the case with data P being a subpart of X.

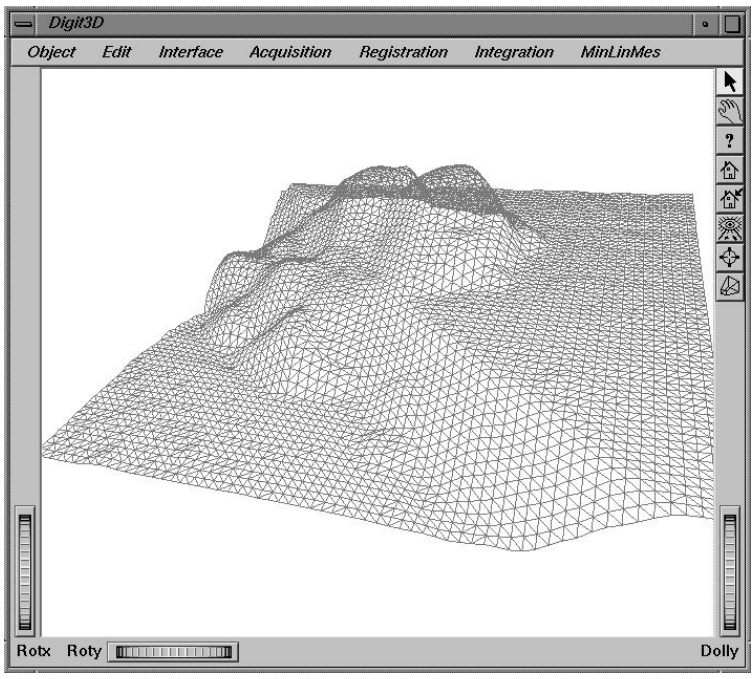


Figure 4.6: Triangulated representation of the AFM range image in the test program

Data represent a very small impurity measured by an atomic force microscope (AFM). The range image contains 256x256 points and a down-sampled triangulated mesh containing approximately 1000 points is shown in Figure 4.6. This AFM depth map possesses no missing data point, insuring that only the first closest point computation doesn't possess at least one candidate neighbour.

Set 2

The second set of measurements uses two different, partially overlapping datasets. Registering partially overlapping datasets typically represents the problem of views registration for virtual modelling.

Presently, the data represent the surface of an irregular substrate measured by an AFM. The second dataset has been obtained during a second acquisition, after a shift of the sample in the *xy* plan. The overlap of both sets is approximately 50% of their total surface. Both images are 256x256 points big and Figure 4.7 shows a render of the two coarsely aligned sets.

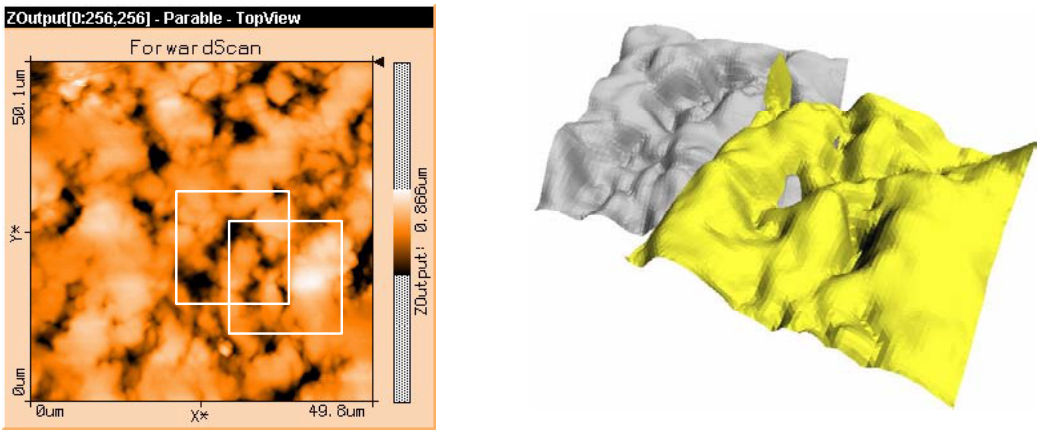


Figure 4.7: Range image and surface rendering of two partially overlapping AFM datasets

Set 3

The third set of measurements uses two partially overlapping datasets as well. The data are range images of a duck toy taken

with our structured light range finder and have been used to build a full model. The overlap of both views is approximately 50% of their total surface. Figure 4.8 shows both range images and the two coarsely aligned sets, rendered in the working environment.

The main difference between this set and the previous one is that it contains missing points (with a null value, shown in black), hidden surfaces and noise. Each range image contains 300x300 points (90000) but only 20000 to 30000 actually contain non-null values.

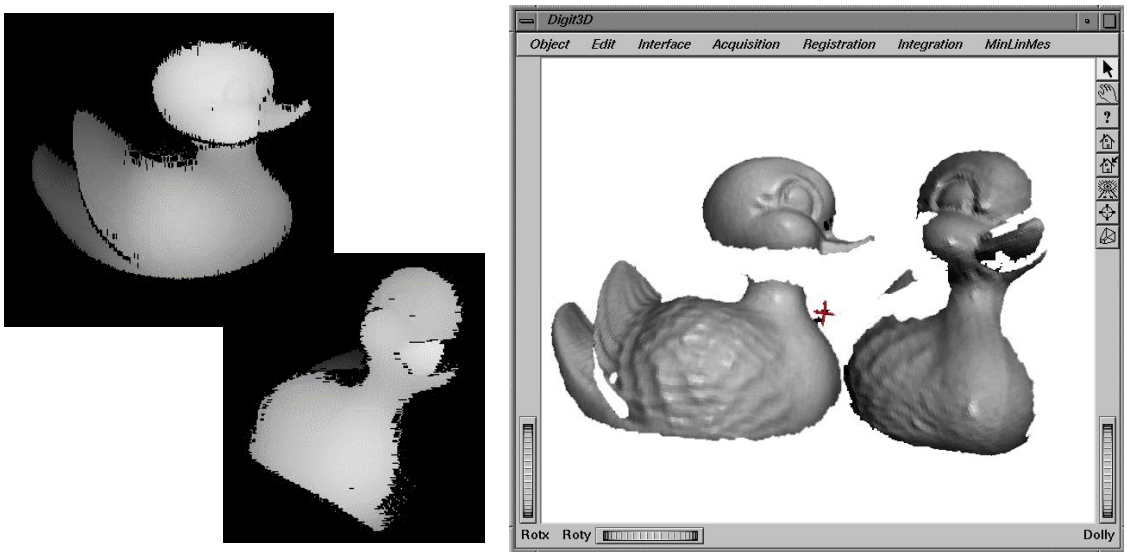


Figure 4.8: Partially overlapping range images of a duck toy and their corresponding surface rendering

4.5.2 Experiments description

Our experimental program has been coded and ran on an Indigo^2 175 MHz SGI workstation. The implemented ICP algorithm uses a multi-feature distance (2.16) and binary weighting (2.20). Practically, we did not investigate the use of colour in our tests so the multi-feature distance is restricted to using points and normals. Concerning the binary weighting, we used two fixed thresholds depending on the resolution of the data. The first iterations are made with a larger threshold, to ensure the convergence of datasets far from each other, then a smaller

threshold is used to finish the matching. Finally, the complex error change criterion is used to define when to change the threshold and to automatically stop the iterations.

As said before, for comparisons, we chose to use a k-D tree as our reference fast closest point algorithm for the ICP. The main reason is that it still permits a reasonable speeding of the matching while maintaining the quality of the matching. The implemented k-D tree method is based on the original optimised k-D tree proposed by Bentley (balanced tree, with fixed key at each level and empty buckets). The closest point search is made with Zhang's algorithm (§3.4.3).

To measure the domain of convergence, we used the SIC setup presented in section 2.8. A slight modification of the size of the sphere onto which the different initial configurations are chosen has been made: the diameter of the circumsphere of dataset X is divided by a factor four. This has been made since we use surfaces from range images instead of complete object models and the data would really be too far from each other. We also limit the value of the zenith angle to the range $\phi = [0–100]$.

For practical reasons, some down sampled data have been used to compute the SIC ranges. It was done that way because the time requested for the whole computation at full resolution was too long. On the other hand, a sub set of 30 initial configurations has been chosen to test the matching of high or full resolution data.

The absolute error measurements are done by comparing the actual position of the dataset P with the position of the correct matching. Practically, the transformations are compared and the results are given as a rotation and a translation error. The rotation error e_φ is given in degrees and the translation error e_t is given as a percentage of the circumsphere radius of X.

Finally, we use two representations to measure the acceleration of the different methods. The first one is the actual execution time, be it for the closest point search or the global ICP matching. The second representation is the gain G over using the basic exhaustive search method.

$$G = \frac{t_{exh}}{t_{fast}} \tag{4.3}$$

with t_{exh} and t_{fast} are the measured run times when using respectively the exhaustive search and a fast search method.

4.6 Experimental results

This section summarizes the principal results obtained while testing the neighbour search algorithm. For simplification, we will use the term "*n*x*n* n-search" for neighbour search using a local *n*x*n* window in this section. We will also call d_g the classical geometric distance and d_n the multi-feature distance using normals.

4.6.1 Matching quality

As said before, the matching error and the domain of convergence are the two essential measurements needed to quantify the quality of convergence. Our first observation concerns the matching errors, since they are needed to define the successful matches.

Matching error

A matching is considered successful when both the rotation and translation errors fall below some fixed thresholds. Figure 4.9 shows the rotation error e_φ evolution for three typical registrations: an unsuccessful matching using the neighbour search and two successful ones using the neighbour search and a k-D tree. We can see that the choice of the thresholds is not critical since the error of the unsuccessful registrations is generally much bigger than the threshold (set to 2° in this example).

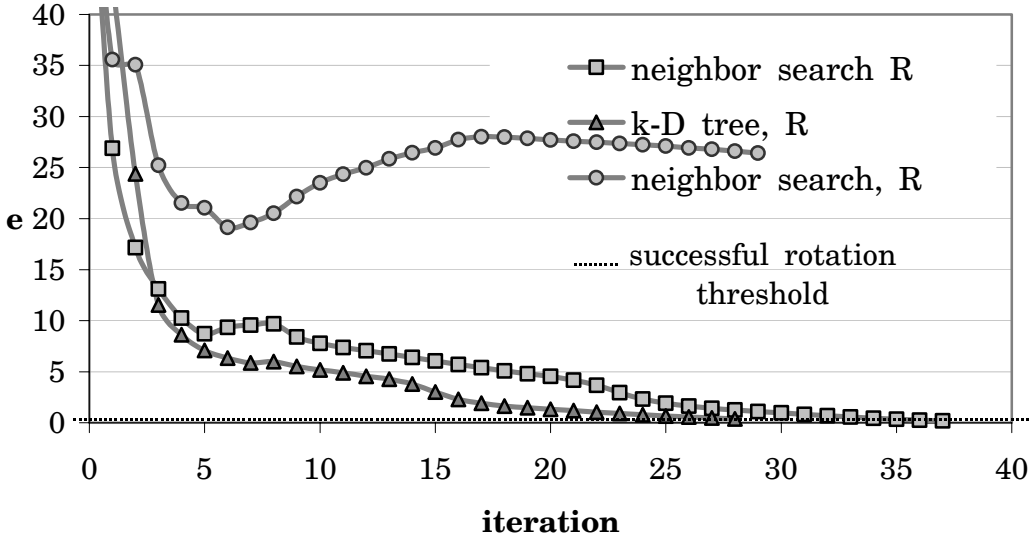


Figure 4.9: Evolution of the rotation error for 3 typical registrations

Practically, we want to compare the registration errors when using a neighbour search and when using a k-D tree search. Table 4.2 presents the average rotation and translation errors for successful convergences with the different datasets. The principal observation we can make is that, in case of successful convergence, all methods possess errors in the same range. It shows that using the neighbour search doesn't have a bad influence on the final registration.

	set 1 (d_g)	**set 1 (d_n)**	**set 2 (d_n)**	**set 3 (d_n)**
k-D tree	3.9° / 0.54%	0.0° / 0.0%	1.7° / 0.97%	0.5° / 0.20%
5x5 n-search	3.9° / 0.58%	0.0° / 0.0%	2.5° / 1.20%	0.6° / 0.22%
9x9 n-search	3.9° / 0.58%	0.0° / 0.0%	1.9° / 0.99%	0.5° / 0.20%

Table 4.2: Average registration errors (e_φ / e_t) with the different datasets.

Domain of convergence

The SIC-maps of Figure 4.10 present the domains of successful initial configuration of set 1, using distance d_g. Figure 4.11 does the same for set 2, using distance d_n. Globally, it appears that the neighbour search performs as well as a k-D tree in terms of SIC-range. However, there are some nuances presented below.

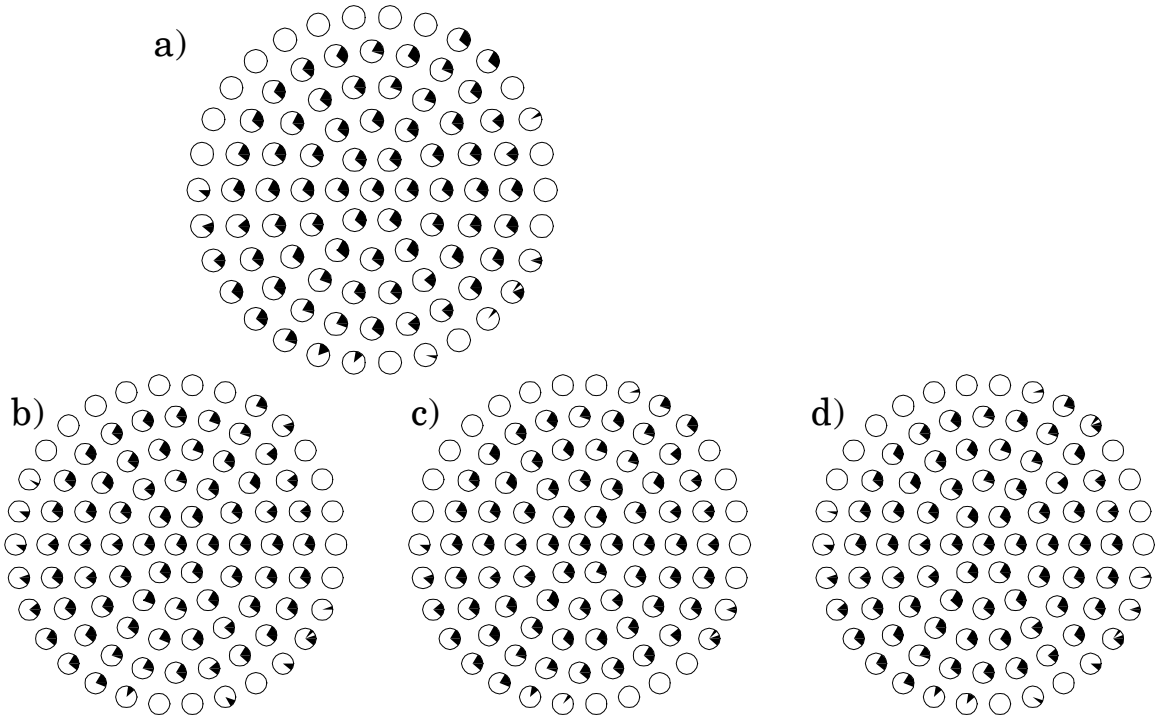


Figure 4.10: SIC-maps of set 1, using distance $\mathbf{d_g}$:a) k-D tree b) 5x5 n-search c) 7x7 n-search d) 9x9 n-search

The influence of the sizes of local search *n*x*n* is examined here. When using the Euclidean distance d_g, a 5x5 n-search was sufficient to maintain the same SIC-range. On the other hand, when using d_n, the range of successful initial configuration is practically reduced to nothing with a 5x5 n-search. A 7x7 n-search improves the SIC-range but the latter is still reduced. Finally, a 9x9 n-search is shown to work fine.

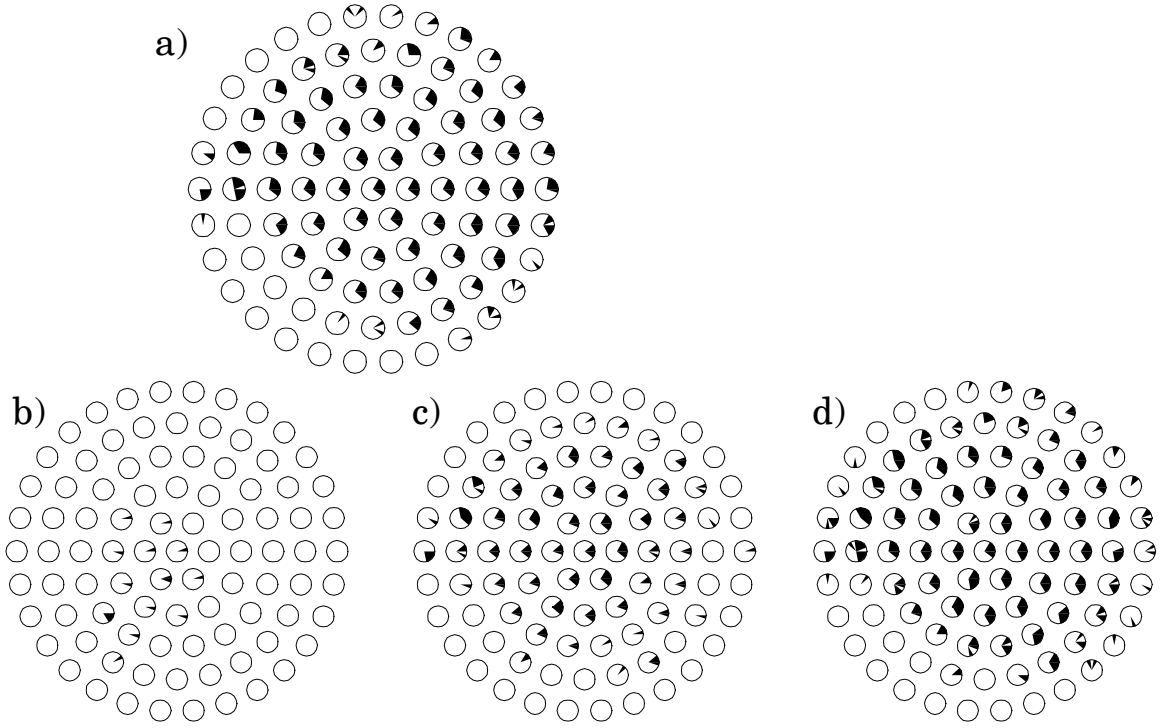


Figure 4.11: SIC-maps of set 2, using distance $\mathbf{d_n}$:a) k-D tree b) 5x5 n-search c) 7x7 n-search d) 9x9 n-search

The fact that the neighbour search requires a larger window for correct convergence when using d_n can be explained by the fact that the neighbourhood relationship assumption is mainly based on the Euclidean distance d_g. Using the multi-feature distance d_n basically reduces the validity of the neighbourhood relationship assumption, especially when surfaces are poorly aligned.

Practically, when trying to register datasets 2 and 3 using the sole point distance d_g implies that the ICP algorithm practically never converges to the right positioning. In fact, a very good initial matching is required in this case, which confirms that the use of the multi-feature distance d_n can have a significant impact on the quality of the registration. Consequently, we focused on using distance d_n with these sets.

Figure 4.12 shows the influence of the resolution on the SIC-range. The SIC-maps of set 3 are compared at different resolutions.

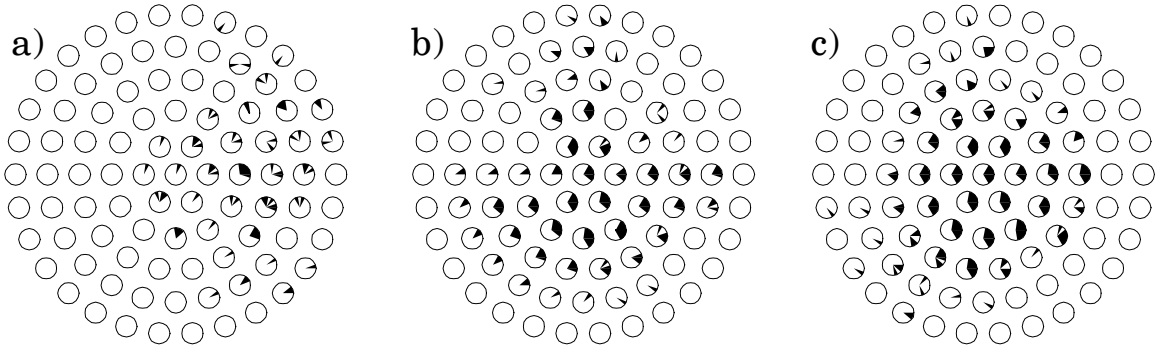


Figure 4.12: SIC-maps of set 3, using distance d_n and a 9x9 n-search: a) 6000 points b) 1500 points c) 350 points

One can see that, unfortunately, the SIC-range tends to decrease when the number of points increases. Basically, the SIC-range also diminishes a bit when using a k-D tree, but much less dramatically. A tentative explanation of this problem is that the absolute size of the local search diminishes if the resolution of the data increases and the local search window remains the same. A solution to this problem is discussed below, in section 4.7.

4.6.2 Search times

Figure 4.13 and Figure 4.14 present the closest point computation time per point of P as a function of the number of point of X. The neighbour search and k-D tree search methods are compared at different resolutions. Two values of k-D tree are given at each resolution, a minimal time and a maximal time. This is to reflect the difference in the computation time depending on the distance between both datasets, as explained in 3.4.3. It must be considered as a best-case / worst-case type of measure. The average k-D tree search value depends a lot on the data and the initial matching but we can estimate it to approximately 2 times the k-D tree min. value.

In the case distance d_g is used (Figure 4.13), the gain in speed of the neighbour search algorithm over minimal tree search lies approximately between 2 and 3. When using distance d_n (Figure 4.14), the gain can go up to 5. This is bigger than the first case, even if a 9x9 n-search is used vs. a 5x5 search. It basically shows the problem of the k-D tree when using a larger

dimension space, in this case d_n (k=6 instead of k=3 with an estimated 2^k distance computations).

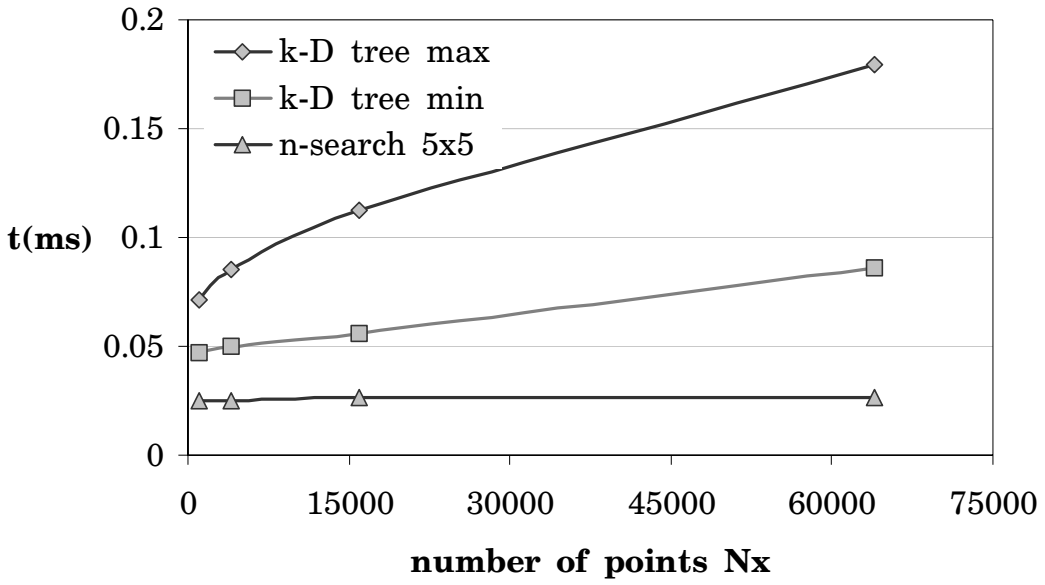


Figure 4.13: Comparison of closest point search time for set 1, using d_g

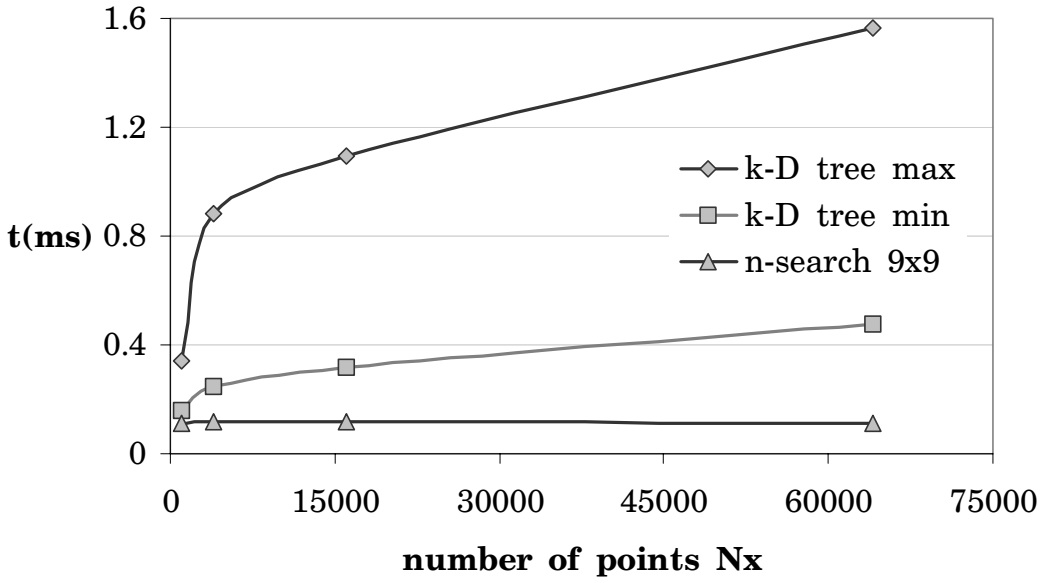


Figure 4.14: Comparison of closest point search time for set 2, using d_n

Of course, this is using the best-case for k-D tree, which means it is basically the worst-case gain of the neighbour search. Considering the maximal k-D tree values, the gain can go up to 15 times!

If we examine the general behaviour of the curves, we can see that the expected linear complexity $O(N_p)$ of the ICP is reached since the duration of the neighbour search remains constant. The $O(\log N_x)$ behaviour of the k-D tree can also be observed.

4.7 Results summary and discussion

The different results show that the proposed neighbour search algorithm performs significantly better than a k-D tree search with gains up to 5. Furthermore, the theoretical complexity of $O(N_p)$ was practically reached, which means that the gain over a k-D tree would increase with larger datasets.

The principal problem of the neighbour search is that the absolute size of the local search diminishes if the resolution of the data increases and the local search window remains the same. It was shown that the range of successful initial configuration actually decreases in this case. Increasing the size of the local search window would certainly permit to avoid this problem but it isn't efficient, computationally speaking.

The main idea to maintain the SIC-range and still keep a 9x9 n-search is to go toward a coarse to fine multiresolution scheme. Indeed, it was shown that the SIC-range is maintained with low-resolution data. Consequently, the SIC-range should be maintained on high-resolution data as well if the neighbour search is applied on lower resolution data for the first iterations.

A multiresolution adaptation of this algorithm is also very easy to implement in the case of range image. Using different resolution simply corresponds to skipping points in the range images when doing the neighbour search, and this for both P and X sets. One could for example use 1/16 points for the first iterations, continue with a few iterations using 1/4 points and finish with the whole datasets. Description and analysis of a multiresolution scheme can be found in Chapter 5.

4.7.1 Future considerations

One can note that the square neighbourhood zone search is basic and simple for testing purposes, but a smarter local search method could be considered later on. A steepest descent or a local 3 steps algorithm, for example, could both diminish the number of local searches and augment the exactness of the result. A local 3 steps algorithm could also help to reduce the subset connection problems described in 4.3.2 by creating a larger neighbourhood search zone with the same number of scanned points.

4.8 Chapter conclusion

A new closest point algorithm for fast ICP registration has been proposed and analysed in this chapter. It permits to reduce ICP complexity to $O(N_p)$. The method uses the assumption that two neighbours on a surface possess closest points that are neighbours on the other surface to easily obtain a first approximation of the closest point and then proceeds with a local search for refining the result.

Results from a series of registration experiments show that the proposed neighbour search algorithm performs significantly better than a tree search. The theoretical complexity of $O(N_p)$ was practically reached. Also, the method improves the computation speed of the ICP algorithm, without altering the matching error and the domain of convergence. The results show that, in nearly all cases, the ICP registration that uses the neighbour search still converges toward the same position as when using an exact closest points search. This confirms the good potential of the proposed method.

The neighbourhood relationship isn't always valid, but when working with data from range image, which is often he case in 3D vision, it is shown to be working in most cases.

The main problem of the neighbour search is that the absolute size of the local search diminishes if the resolution of the data increases and the local search window remains the same. It was shown that the range of successful initial configuration actually decreases in this case. A multiresolution

approach, presented in next chapter, is expected to “cure” this problem.

Finally, we can note that the algorithm was mainly tested with range images but that it extends to cloud of points where the neighbourhood relation exists.

Chapter 5
The Multiresolution Scheme ICP

A multiresolution scheme applied to the ICP algorithm is proposed and analysed in this chapter. Then, the combination of this multiresolution approach and the neighbour search algorithm is taken into account to obtain a very fast and robust ICP algorithm. Part of the content of this chapter has been published in [Jos02b].

5.1 The basic multiresolution scheme

The principle of the multiresolution ICP is to make the first few iterations using down sampled data and to further increase the resolution of the data in the following iterations, creating a coarse to fine matching (Figure 5.1). The main expected advantage of the multiresolution is the reduction of the computational cost, given that the duration of each iteration made at lower resolutions is reduced. The precision of the final matching is expected to be the same as when using all the points for the whole registration. In addition, the total number of iterations should be reduced this way, mainly because a lower resolution matching generally implies more important rotations and translations, meaning a faster convergence.

The multiresolution coarse to fine strategy is not a new concept and has been widely used in image processing and other

domains for years. A few publications (as seen in 3.3.3) also briefly presented some coarse to fine solutions applied to ICP. On the other hand, and to the author knowledge, no real analysis of such solutions have been presented. This basically motivated the existence of the present chapter, which presents and analyses such a multiresolution coarse to fine solution, applied to the ICP.

5.1.1 Chosen multiresolution pattern

The multiresolution pattern chosen here is to divide the number of points by a factor N for each resolution step i, i = 1 being the full resolution data step. If $N_{p,i}$ and $N_{x,i}$, are the number of points of P and X at the i^{th} step, we can write:

$$\left(N_{p,i}, N_{x,i}\right) = \left(\frac{N_{p,i-1}}{N}, \frac{N_{x,i-1}}{N}\right) = \left(\frac{N_p}{N^{i-1}}, \frac{N_x}{N^{i-1}}\right) \tag{5.1}$$

The lowest possible resolution is defined by keeping the number of points of the reduced datasets above a minimum value N_{min}. It is typically chosen between 50 and 100 as a lower number may result in too few pairings for correct computation of best transformation.

This can be formalized by

$$\left(\frac{N_p}{N^{k-1}}, \frac{N_x}{N^{k-1}}\right) > \left(N_{min}, N_{min}\right) > \left(\frac{N_p}{N^{k}}, \frac{N_x}{N^{k}}\right) \tag{5.2}$$

where k is the number of resolution steps.

For example, if we have datasets P and X which contains $N_p = N_x = 5000$ points each, a multiresolution factor N=4 and N_{min}=50, the number of resolution steps k is equal to 4.

$$\left(\frac{5000}{4^3}, \frac{5000}{4^3}\right) > (50,50) > \left(\frac{5000}{4^4}, \frac{5000}{4^4}\right) \tag{5.3}$$

The number of points N_i at each steps i is

$$\left(N_{p,4}, N_{x,4}\right) = \left(\frac{N_{p,3}}{4}, \frac{N_{x,3}}{4}\right) = \left(\frac{N_{p,2}}{16}, \frac{N_{x,2}}{16}\right) = \left(\frac{N_{p,1}}{64}, \frac{N_{x,1}}{64}\right) \tag{5.4}$$

where $N_{p,1}= N_p$ and $N_{x,1}= N_x$.

The number of iterations at each resolution step isn't set and can vary from case to case. Instead, the algorithm goes to the next resolution step automatically when a defined stop criterion is reached at the current one. In our case the stop criterion is the complex error change (see 2.7). Figure 5.1 summarized the multiresolution ICP principle.

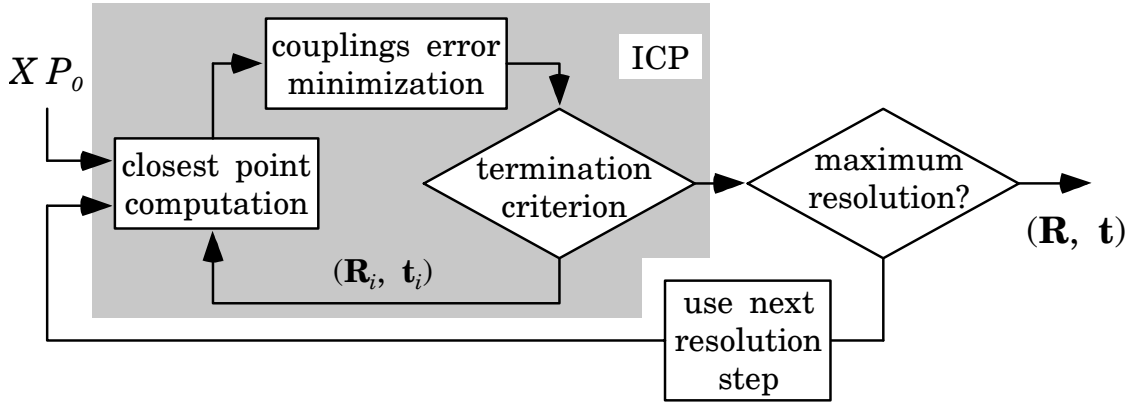


Figure 5.1: The multiresolution ICP principle

5.1.2 Cost reduction factor

The possible gain in speed can be estimated by a simple development. As said before, the basic ICP algorithm has a complexity of $O(N_p N_x)$ and the costs for an iteration are as follows:

- at resolution step i:

$$C_i \approx c N_{p,i} N_{x,i} \tag{5.5}$$

- at step i+1, $N_{p,i}$ and $N_{x,i}$ reduced by a factor N:

$$C_{i+1} \approx c \frac{N_{p,i}}{N} \frac{N_{x,i}}{N} = c \frac{N_{p,i} N_{x,i}}{N^2} = \frac{C_i}{N^2} \tag{5.6}$$

This means that if both datasets are reduced by a factor N, the cost reduction factor for one iteration is:

$$\frac{C_i}{C_{i+1}} = N^2 \tag{5.7}$$

5.1.3 Estimation of the speedup gain

Now, say m is the number of iterations needed for a registration in monoresolution and n_i is the number of iterations performed at resolution step i for the same registration in multiresolution, the cost of the complete registration is:

- monoresolution:

$$m\ C_1 \tag{5.8}$$

- multiresolution:

$$n_1C_1 + ... + n_kC_k = n_1C_1 + \frac{n_2C_1}{N^2} + ... + \frac{n_kC_1}{N^{2(k-1)}} \tag{5.9}$$

and the gain of multiresolution is:

$$G^N = \frac{m}{n_1 + \frac{n_2}{N^2} + \frac{n_3}{N^4} + ... + \frac{n_k}{N^{2(k-1)}}} \tag{5.10}$$

To analyse the numerical value of the gain, we make the hypothesis that the total number of iterations remains the same in both cases. We will also distinguish 3 cases here on how iterations are distributed for each step:

1. a constant number of iterations at each step
2. an increasing number of iterations with the higher resolutions
3. a decreasing number of iterations with the higher resolutions

Basically, they can be seen as medium, worst and best theoretical cases. The associated linear functions of n_i are:

$$n_i^{(1)} = m\frac{1}{k}\ , \quad n_i^{(2)} = m\frac{k+1-i}{(1+2+...+k)}$$
$$n_i^{(3)} = m\frac{i}{(1+2+...+k)} \tag{5.11}$$

where k is the number of resolution steps.

This gives us the following result for estimated gains:

$$G_1{}^N(k) = \frac{k}{\left(\sum_{i=1}^{k} \frac{1}{N^{2(i-1)}}\right)}, \qquad G_2{}^N(k) = \frac{\sum_{j=1}^{k} j}{\left(\sum_{i=1}^{k} \frac{k+1-i}{N^{2(i-1)}}\right)}$$
$$G_3{}^N(k) = \frac{\sum_{j=1}^{k} j}{\left(\sum_{i=1}^{k} \frac{i}{N^{2(i-1)}}\right)} \tag{5.12}$$

Figure 5.2 shows a graphical presentation of these functions for N=4. One can see that the higher the number of steps, the higher the expected gain is. The theoretical best case (3) has a quadratic behaviour, while gain functions 1 and 2 are nearly linear.

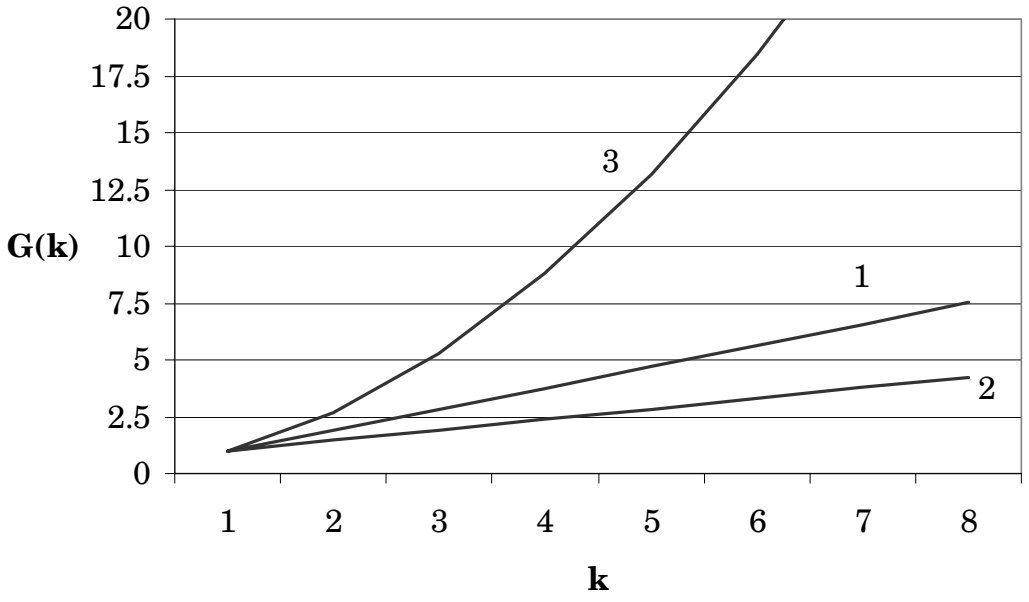


Figure 5.2: Expected multiresolution gains G(k) for the basic ICP algorithm, for N= 4

5.2 Coupling multiresolution with fast closest point search

Using the sole multiresolution scheme to speed up the ICP algorithm would not be really effective when compared to solutions that reduce the complexity of the closest points search. But both types of acceleration methods are quite independent and consequently can be combined together. Thus, we consider multiresolution combined with the fast ICP algorithms tree search and neighbour search. Such a combination should permit to create a very fast ICP algorithm.

Moreover, multiresolution matching is expected to have an added beneficial impact on both tree search and neighbour search methods. In the case of a k-D tree search, the first iterations require a longer search time because of the coarse alignment of the data [Zha94] (see Figure 3.9). With multiresolution, these iterations are typically done with a very low resolution, which greatly reduces the search time. In the case of the neighbour search closest point algorithm, higher resolutions and coarser matching decrease the exactness of the closest points pairing which typically reduces the range of successful initial configurations (SIC), as seen in the previous chapter. Using a lower resolution for the initial iterations, when matching is coarse, should permit to avoid this problem.

5.2.1 Cost reduction factor

As seen before, the complexity of the ICP algorithm changes when using tree search or neighbour search. It is $O(N_p \log N_x)$ with a tree search and $O(N_p)$ with the neighbour search. Consequently, the cost reduction factors are not the same in these cases.

If both datasets are reduced by a factor N, the cost of iteration in the tree search case is:

$$C_{i+1} \approx c\,\frac{N_p}{N}\log\left(\frac{N_x}{N}\right) = c\,\frac{N_p}{N}\,\frac{\log N_x}{\log N_x}\log\left(\frac{N_x}{N}\right) = C_i\,\frac{\log\left(\frac{N_x}{N}\right)}{N\log N_x} \qquad (5.13)$$

and, consequently, the cost reduction factor for one iteration is:

$$\frac{C_i}{C_{i+1}} = \frac{N\log N_x}{\log\left(\frac{N_x}{N}\right)} = N\frac{\log N_x}{\log N_x - \log N} \tag{5.14}$$

Table 5.1 shows a summary and comparison of the different cost reduction factors, when datasets *X* and *P* are reduced by a factor N.

	Reduce N_p	**Reduce N_x**	**Reduce N_p and N_x**
normal ICP	N	N	N^2
k-D tree ICP	N	$\frac{\log N_x}{\log N_x - \log N}$	$N\frac{\log N_x}{\log N_x - \log N}$
neighbour search ICP	N	1	N

Table 5.1: Cost reduction factors for data reduced with a factor N

If we examine the k-D tree case, the value of the expression "$\log(N_x)/(\log(N_x)-\log(N))$" is typically situated between 1 and 2, as shown in Figure 5.3. This means that it is basically worthless to use a lower resolution on dataset *X* when using a tree search. Firstly because the cost reduction value is much lower than when using a similar resolution on dataset *P* and secondly because it would imply to build a new tree at each resolution.

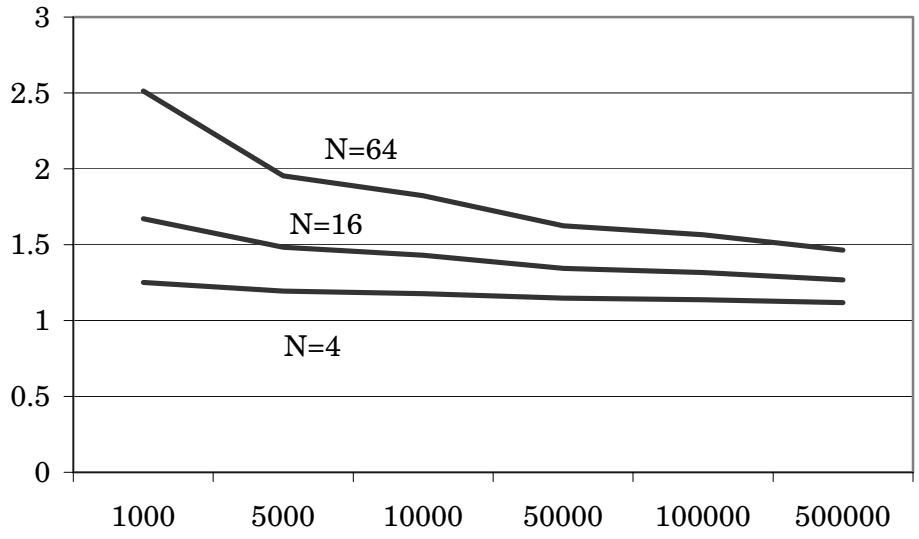


Figure 5.3: $\log(N_x)/(\log(N_x)-\log(N))$ plot for typical values of N

In the case of the neighbour search, the reduction of dataset X doesn't have an influence on the cost at all. Note however that in practice, dataset X needs to be reduced as well in order that the relative size of the local search area stays the same.

Taking the above into account, we can consider that the cost reduction factor for one iteration is N if both datasets are reduced by a factor N, in both k-D tree and neighbour search cases.

5.2.2 Estimation of the speedup gain

Referring to equation (5.10) and if full resolution is kept on dataset X when using a k-D tree, the relative gain of the multiresolution registration for both tree search and neighbour search methods is:

$$G^N = \frac{\mathrm{m}}{\mathrm{n}_1 + \frac{\mathrm{n}_2}{\mathrm{N}} + \frac{\mathrm{n}_3}{\mathrm{N}^2} + \ldots + \frac{\mathrm{n}_k}{\mathrm{N}^{k-1}}} \tag{5.15}$$

Making the same assumptions as in the section 5.1.3, we now find the following multiresolution gains for the medium, worst and best hypothetical cases:

$$G_1{}^N(k) = \frac{k}{\left(\sum_{i=1}^{k} \frac{1}{N^{i-1}}\right)}, \qquad G_2{}^N(k) = \frac{\sum_{j=1}^{k} j}{\left(\sum_{i=1}^{k} \frac{k+1-i}{N^{i-1}}\right)}$$
$$G_3{}^N(k) = \frac{\sum_{j=1}^{k} j}{\left(\sum_{i=1}^{k} \frac{i}{N^{i-1}}\right)} \tag{5.16}$$

The expected gains have the same functional behaviours than for the basic ICP case, exposed in §5.1.3, but the values are slightly lower in this case, as seen in Figure 5.4. On the other hand, one can note that the global acceleration, regarding a non-accelerated ICP, is much higher thanks to the reduction of complexity of the closest points search.

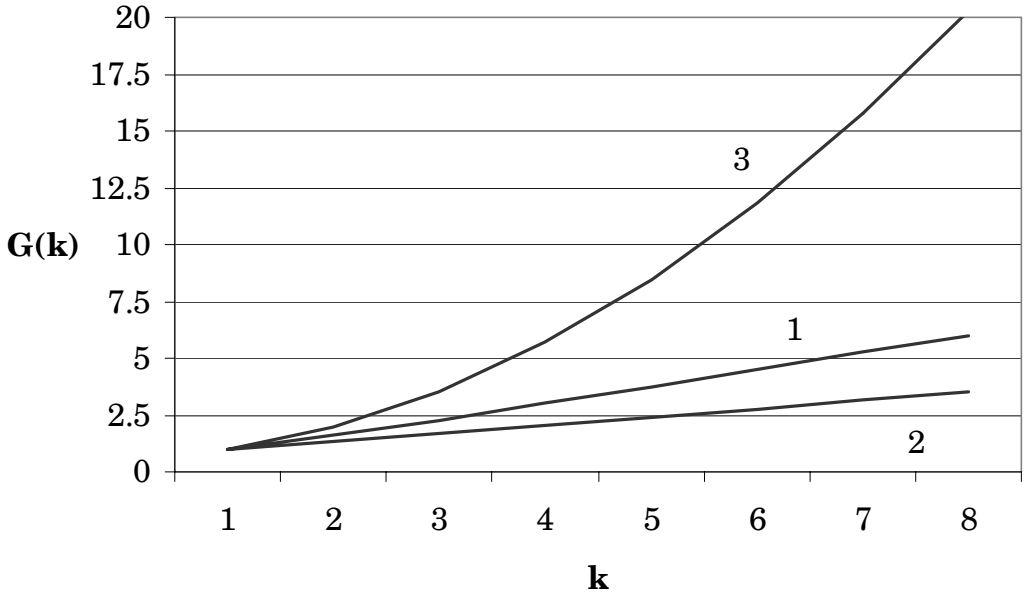


Figure 5.4: Expected multiresolution gains G(k) for fast ICP algorithms, for N= 4

The maximum number of resolution steps k is directly related to the number of points of the data, N_p and N_x and to the reduction factor N. A first consequence of this is that the bigger the datasets, the higher the expected gain is.

When the reduction factor N increases, the value of the gain increases as well for a similar value of k (not shown). However, the number of steps k decreases in this case, which means a smaller gain (following Figure 5.4). Consequently, an interesting representation is to actually express the gain G as a function of the reduction factor N. It is the case for different values of N_p or N_x in Figure 5.5, using the medium case gain G_1.

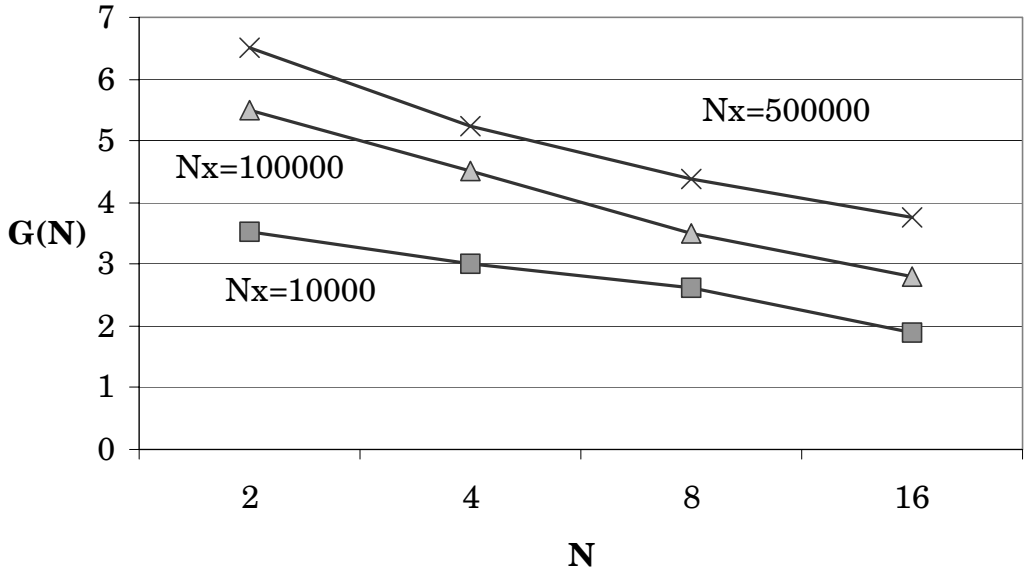


Figure 5.5: Expected multiresolution gains G(N) for fast ICP algorithms, for different values of N_p or N_x

The results of Figure 5.5 show that the smaller the reduction factor N, the bigger the expected gain. It also permits to see the gain increase with the number of points in the datasets.

5.3 Experimental setup

Following the previous theoretical results, we would be tempted to choose a value of N as small as possible. On the other hand, a small value of N also means a lot of resolution steps. Practically, we chose a value of N=4 for the multiresolution scheme, which represents a middle value for expected gain and number of resolution steps. Also, N=4 was basically easier to implement for range images than a multiresolution with N=2 and the expected gains are just a bit smaller.

Unless said otherwise, we used the multi-feature distance d_n and a 9x9 n-search.

Beside these specific considerations, the same datasets and different algorithms presented in section 4.5 are used again in this chapter.

5.4 Experimental results

The principal results obtained while testing the different multiresolution algorithms are summarized in this section.

5.4.1 Matching quality

Once again we want to verify that the matching error and the domain of convergence aren't badly influenced by the usage of the multiresolution scheme and the neighbour search, since we want to avoid any quality versus speedup compromise.

Matching error

Table 5.2 shows a comparison of the average registration errors of dataset 3 when using and not using multiresolution, for both k-D tree search and neighbour search. The results show that, in case of successful convergence, all methods possess errors in the same range, multiresolution giving even slightly better results.

	monoresolution	multiresolution
k-D tree	0.5° / 0.20%	0.01° / 0.01%
9x9 n-search	0.5° / 0.20%	0.03° / 0.02%

Table 5.2: Average registration errors (e_φ / e_t) for successful convergences, between monoresolution and multiresolution

Figure 5.6 and Figure 5.7 show the evolution of the rotation and translation errors during the iterations. An arrow indicates when the resolution step changes. One can clearly see that the errors are generally nearly constant before a step change and that the following step permits to further improve the quality of the matching.

The first step of the multiresolution k-D tree search is equivalent to using a few control points, as defined in section 3.3.2. This result basically confirms the advantage of a multiresolution coarse to fine scheme over using control points, in terms of matching quality.

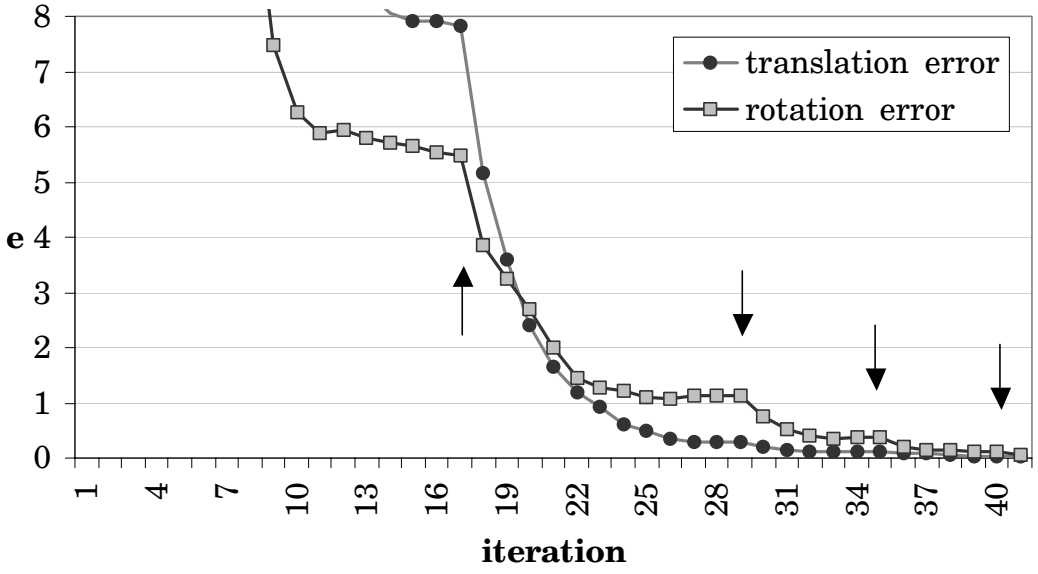


Figure 5.6: Evolution of the rotation error e_φ and translation errors e_t for the multiresolution n-search case.

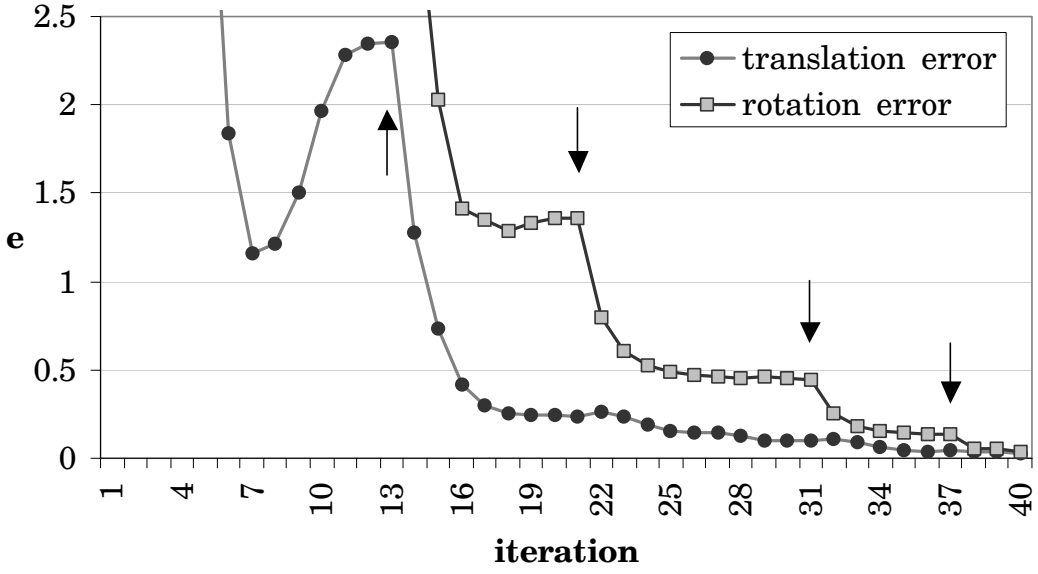


Figure 5.7: Evolution of the rotation error e_φ and translation errors e_t for the multiresolution k-D tree case.

Domain of convergence

The SIC-maps of the 4 cases presented above can be found in Figure 5.8. The results show that multiresolution does not affect the domain of successful convergence when coupled with a k-D tree search and has the expected beneficial effects on it when coupled with the neighbour search. More precisely, one can first see that when using a k-D tree, the SIC range remains the same when using multiresolution (a, b), given some small difference, mainly for very twisted initial configurations ($\phi > 80°$). On the other hand, the resolution was a bit too big for the neighbour search to work correctly and the SIC range is much reduced in this case (c). However, the SIC range obtained combining multiresolution and neighbour search (d) is again very similar to the one obtained while using a k-D tree.

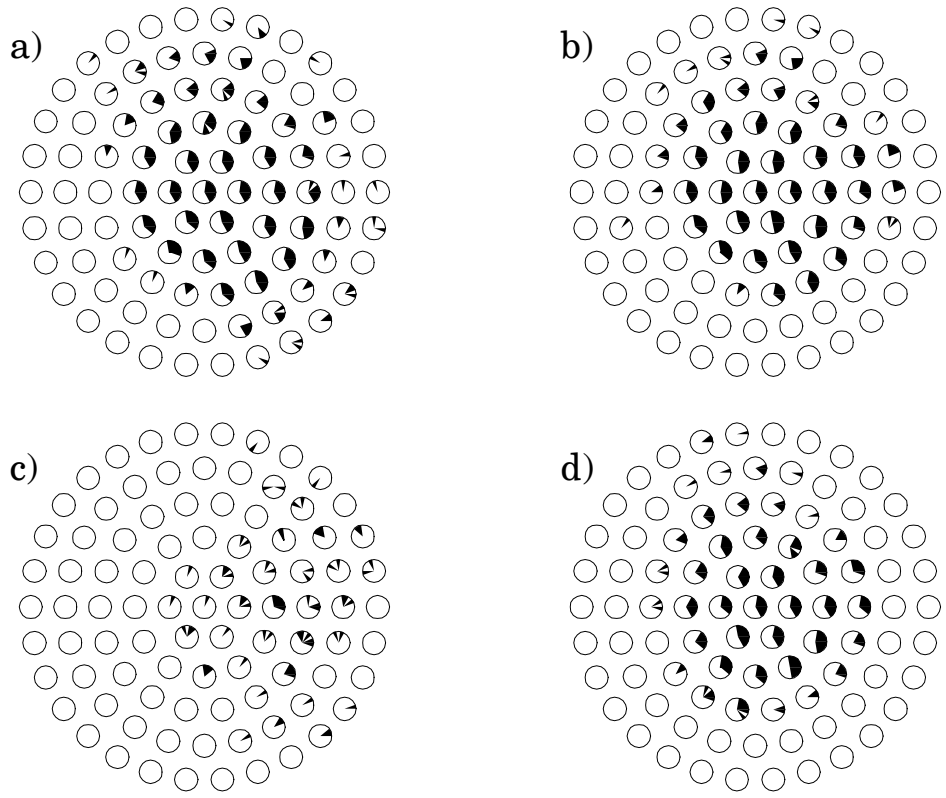


Figure 5.8: SIC-maps of set 3: a) k-D tree search, b) multiresolution k-D tree search, c) n-search, d) multiresolution n-search

The results are very similar when using full resolution data (4.5.2). In this last case, the effect of multiresolution is even more visible since using full resolution also tends to decrease the

SIC-range when using a k-D tree. Basically, multiresolution permits to keep a large SIC-range in most cases.

All in all, these results confirm that the multiresolution scheme does not affect the matching quality for both the matching error and the domain of convergence. Furthermore, it has beneficial effects on the SIC range, especially when combined with the neighbour search algorithm.

5.4.2 Search times

Table 5.3 presents a comparison of the average total computation time for the successful registrations, using the different acceleration methods on the full resolution dataset 3. We will note that the results related to the neighbour search have been greyed out to reflect that they come from a single measurement.

	total time (s)	**number of iterations**	**relative gain over k-D tree**	**absolute gain G**
k-D tree	504.1	42	1.0	60
n-search	259.0	90	1.9	117
MR k-D tree	59.3	36	8.5	513
MR n-search	18.8	34	26.8	1621

Table 5.3: Comparison of the total computation time and gains of the registration (at full resolution) using the different acceleration methods (MR = multiresolution)

The multiresolution scheme permits to reduce the total registration time by an average factor between 8.5 and 14 times for the different cases. These results are at least equal to or better than the theoretical best case value found with equation (5.16), $G_3^4(5) = 8.5$ (k=5 in this case). Practically, one can effectively see a decreasing number of iterations in the higher resolutions steps, as seen in Figure 5.9, although close to but not as important as the best theoretical case. The high gain results can be further explained by the fact that other factors were not taken into account in the theoretical estimations, like the reduction of the total number of iterations, especially when using

the neighbour search, or the beneficial effects of the very low resolution in the first iterations when using a k-D tree search.

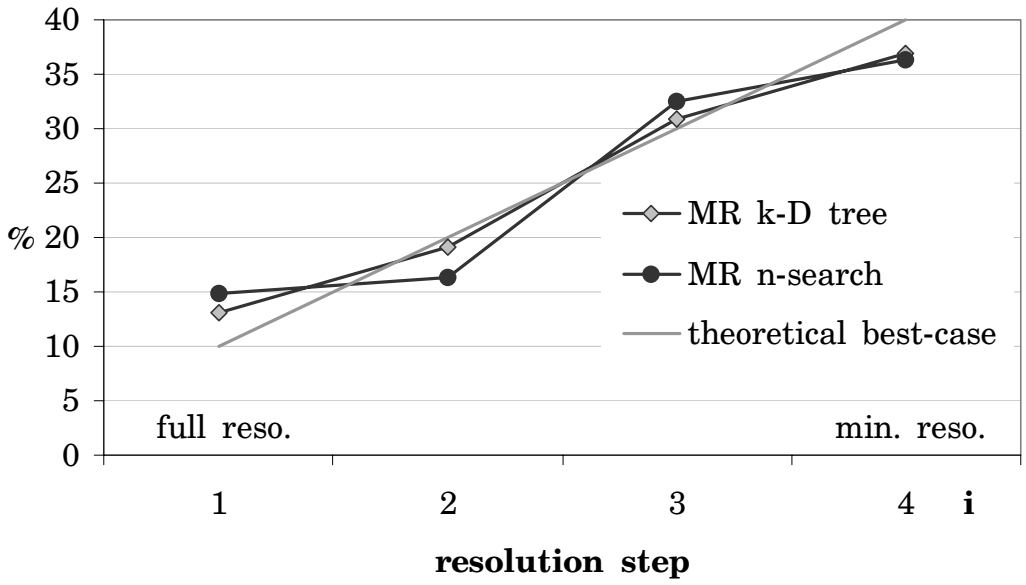


Figure 5.9: Average percentage of iterations vs. resolution step i of the multiresolution ICP

Finally, the multiresolution neighbour search ICP can be up to 3 times faster than the multiresolution tree search ICP and nearly 27 times faster than a k-D tree ICP, which shows the advantage of neighbour search. This gain is also expected to be even higher for bigger datasets, due to the smaller complexity of the neighbour search algorithm.

One can also note that these relative gains refer to a fast ICP algorithm. The approximate gain in speed over a non-accelerated ICP algorithm is over 1600.

5.5 Chapter conclusion

This chapter proposed to further accelerate fast shape registration by application of a multiresolution scheme to the ICP algorithm. The chosen multiresolution scheme proceeds from coarse to fine and successively improves a previous solution at the finer representation level.

In a first part, using simple hypotheses, estimates of the speedup gain of the multiresolution approach with respect to a standard monoresolution approach were established for the standard ICP and also for two fast ICP methods based on tree search and neighbour search. The results show a nearly linear behaviour of the gain with the number of resolution steps for the worst and medium cases and a quadratic behaviour for the best case. Typical expected gains are in the order of 4 to 10.

In a second part, the various algorithms are experimentally compared in a 3D shape-matching test. In these experiments, special attention is given to allow only good quality matches and not to go in any quality versus speedup compromise. Under these circumstances, the results show large speedup gains that can reach the best theoretical expectations.

Specifically, in both cases of fast ICP matching using a tree search or a neighbour search, multiresolution improves the registration speed by factors up to 8. These results are as good or even slightly better than the expected theoretical best case. This can be explained by the reduction of the total number of iterations and by the beneficial effects of the very low resolution in the first iterations when using a tree search.

Combining multiresolution with the neighbour search method, the registration can be up to around 25 times faster than when using a k-D tree search, which really represents a very high-performance ICP algorithm in term of speed. The approximate gain in speed over a non-accelerated ICP algorithm is over 1600.

Finally, the pure speedup potential goes together with improvements observed with respect to the convergence speed and the matching quality. Practically and as expected, multiresolution permits to fully eliminate the decrease of matching quality that can appear when using the neighbour search. The advantage of a multiresolution coarse to fine scheme over using control points in terms of matching quality was also demonstrated. These results clearly show that the multiresolution scheme exploits the fundamental nature of shape registration to substantially contribute to improve its computation.

Chapter 6
3D Object Modelling from Range Images

Application like virtual museums, reverse engineering or industrial inspection generally need virtual models that are very close to their real counterpart. Range scanners now provide a simple and fast way to capture 3D data. However, most of them suffer from the same problem: as data is measured from a single point of view, only a part of the object surface can be scanned at a time. Consequently, acquisitions from several viewpoints must be performed, in order to create a model covering the whole surface of the considered object. These views must then be combined to create the final model.

During the course of the research, a complete digitising system has been built. It uses range images as input and works with triangulated meshes. A description of its main features and proposed methods, as well as some results can be found in this chapter. The presented work has been published in [Sch97b] [Jos98] [Sch98a] [Sch98d] [Hug99a] [Hug99b] [Jos99] [Hug00] [Jos01] and [Hug02].

6.1 System architecture

A general diagram of the modelling system is presented in Figure 6.1. Of course, the input of the system is the object to be

scanned. Three main blocks are then defined: view digitising, view registration and view fusion. The view-digitising block first captures range images with the help of a range scanner and creates virtual triangulated views from them. Then, the view registration block aligns the different views. Finally, these virtual views are combined in the view fusion block, which outputs the expected virtual model.

The different possible registration schemes are showed in Figure 6.1. One can distinguish between the global registration, where all views are registered at the same time, the sequential registration, where the different views are sequentially registered, and the sequential registration with partial modelling, where each new view is registered and fused with the partial model being built.

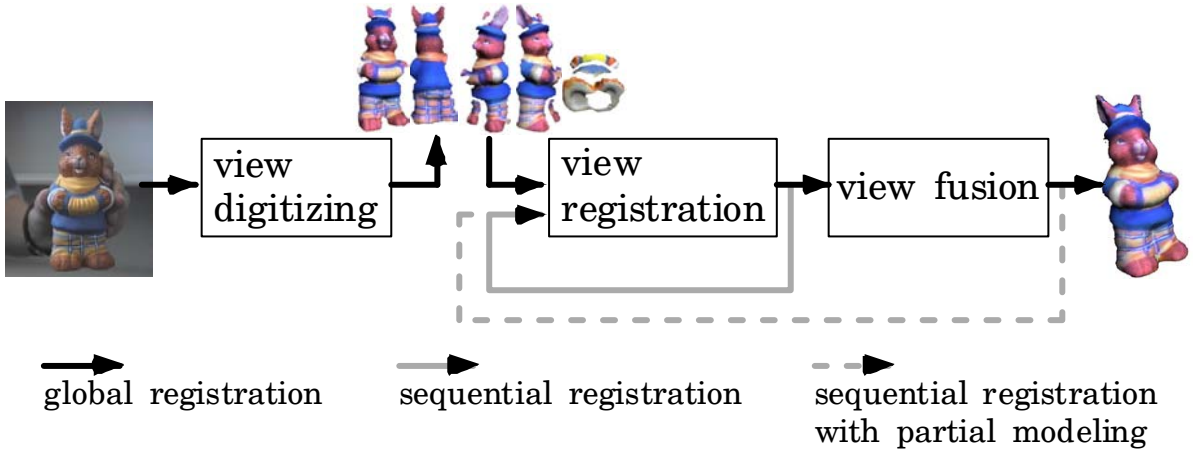


Figure 6.1: Modelling system

These different blocks are explained into more details in the next sections.

6.2 View digitising

The main goal of view digitising is to acquire the 3D data for each view and to present them in a virtual view format that preserves the topology of the scanned surfaces. Three main parts compose the view-digitising block: data acquisition, data processing and view triangulation.

Data acquisition

Data acquisition consists in measuring the 3D geometry and range scanners now provide a simple and fast way to capture 3D data. Several types of range scanners are available on the market today, generally working on optical principles like stereo matching, active triangulation or focus / defocus [Sch96].

Each single measurement provides a range image, where each pixel represents the camera-to-object distance. In certain cases, colour can also be measured. Such a measure is essential to creating realistic looking virtual models. Typically, colour will be used for application where the appearance is important, like virtual museums or web shopping.

The 3D scanners we use in our digitising system work using a strip pattern coding system [Abw3d], which is part of the active triangulation systems. Such scanners also permit to easily obtain colour measurements (see §6.5).

Data processing

During data processing, geometric data are filtered to remove noise and missing points. This step is also important if colour is used. Light reflection effects must be eliminated to retrieve the intrinsic colour of the object and to avoid artifacts during integration. A more complete explanation of the acquisition and processing of colour can be found in §6.5.

View triangulation

Finally, in view triangulation, the measured surface points are triangulated to create an adequate mesh representation of the surface. The basic method is straightforward because measurements in range images are ordered in a regular grid where topology is maintained [Rut94]. However, additional checks are necessary to avoid the linking of neighbouring range points separated by a discontinuity step, which, basically, has the unwanted effect to fill occluded parts with triangles.

These checks typically eliminate triangles that are bigger than a threshold [Tur94] or possess a normal nearly perpendicular to the viewing axis [Rut94].

Finally, a colour can be assigned to each vertex to create a coloured mesh or the whole colour image can be used as a texture map. Handling of textures is detailed in §6.6.

6.3 View registration

As one of the main purposes of the built scanning system is to test the presented geometric matching algorithms (mainly fast ICP variants), the scanning process is kept as simple as possible and allows an operator to place the real world object in any pose on the acquisition field.

We chose not to rely on a fully automatic registration but rather let the user do the rough matching. Thus, one can separate registration in two distinct actions: interactive pose estimation and automatic matching with the ICP algorithm.

6.3.1 Interactive pose estimation

Human perception easily identifies corresponding surface parts for any object type and shape. Therefore, the user can easily enter a hint for the computer, which will then calculate the precise alignment using the ICP algorithm.

The system provides an interactive interface that permits an operator to enter a pose estimate for the objects to be aligned. The different views are rendered in 3D and can be manipulated in all six degree of freedoms using a space mouse [Space] as input device (Figure 6.2). The manipulation of the space mouse can be a little tricky at first but the learning curve is fast and it then permits a very fast rough matching. An alternate solution for interactive pose estimation, when no space mouse is available, is to manually designate some point pairs between datasets and then perform a best transformation computation. This is the case in the commercial software IMAlign (now included in Polyworks [Innov]) for example. This solution is slower and generally more tedious than using a space mouse but has the advantage of creating a good rough matching.

Figure 6.2: Space mouse

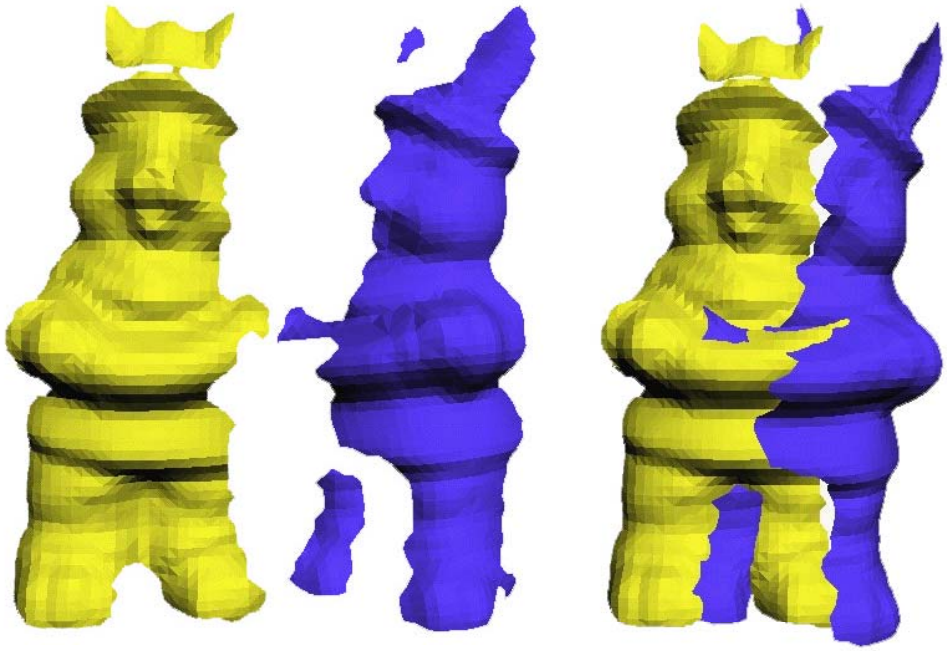


Figure 6.3: Two views of a clay rabbit, before and after the interactive pose estimation

6.3.2 Automatic matching with ICP

The implemented ICP algorithm has the following characteristics:

- it uses binary weightings (§2.5.1)
- it uses multiple feature point matching (2.16)
- it uses fixed relative distance thresholds (2.20)
- it uses either multiresolution and k-D tree or neighbour search
- no global registration method is used

We think this combination gives a robust matching algorithm with a very good tradeoff between quality of the matching, range of successful convergence and speed of execution. The range of successful convergence doesn't have to be neglected because of the rough matching method we use. The SIC range needs to be sufficiently big so that the interactive rough alignment can be done fast and easily. It would have been of less importance with a manual point pairing solution.

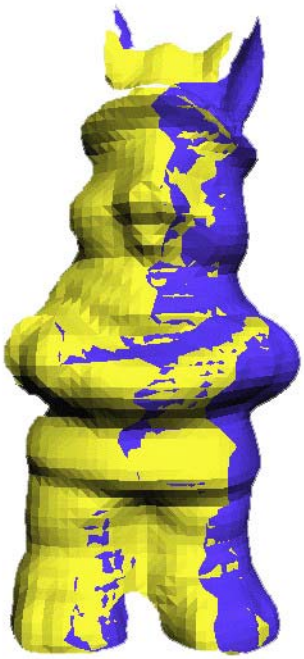


Figure 6.4: Fully registered views

6.4 View fusion

Once the surfaces are matched, they must be fused together in order to eliminate redundant data and to create a unique mesh. Several methods have been proposed to integrate 3D views. They mainly differ in how they treat redundant data. They can be separated into two groups: partial erosion of surfaces and complete retriangulation of the surface points. Several authors [Pit96] [Tur94] erode the overlapping surfaces until the overlap disappears. The two surface meshes are then recombined at their frontiers in order to have one unique mesh for the union of the two surfaces. Other authors [Hil96] [Rut94] [Sou95] discard the mesh information from the triangulated views, if calculated at all, and retriangulate the overlapping zone or even the complete point set [Lor87] [Ber99].

The method we use here is part of the partial erosion solutions. It relies on a modelling that preserves the range images topology all along the reconstruction and introduces a new fusion algorithm that is coupled with registration. Topology preservation is an advantage because it permits to avoid later triangulation and it also permits to easily handle textured views. The new fusion algorithm, unlike other works where the fusion is a totally separated task involving 2D projection or similar complex algorithms, couples fusion with registration and takes full advantage of the available mesh correspondence to treat overlapping areas [Sch98a] [Hug02].

6.4.1 Mesh fusion algorithm

The mesh fusion algorithm is characterized by the following steps:

1. overlap detection: The valid couplings from the previous automatic matching are used to easily identify the parts of surface P which overlap surface X where P and X are defined as in the previous chapters.
2. overlap erosion: The overlapping part of surface P is eroded.
3. frontier detection: A gap separates the surface X and the eroded surface P. The frontier on P is

calculated during the overlap erosion where a closest point search detects the start of the frontier on X.

4. gap filling: The gap enclosed by the two frontiers is filled with triangles. The filling algorithm works in 3D space and does not need any projections into tangential planes, which increases its reliability.

Some details of the algorithm are discussed below and illustrated by examples in Figure 6.5 and Figure 6.6 below.

6.4.2 Overlap detection

The closest point routine of the automatic matching module marks the vertices on surface P that overlap surface X. To ensure that only close points are marked, the constant c of the coupling distance threshold (2.18) is set to a small value during the last iterations of the automatic matching. This results in a set C_V of coupled vertices with:

$$C_V = \{\mathbf{p}_k \in P | w_k = 1\} \tag{6.1}$$

6.4.3 Overlap erosion

The erosion process eliminates all the vertices members of C_V on surface P. A small gap of about the size of the distance threshold appears where the surface X faces the eroded surface P_C.

$$P_C = P - C_V, \; P_C = \{\mathbf{p}_k \in P | \mathbf{p}_k \notin C_V\} \tag{6.2}$$

6.4.4 Frontier detection

Triangles with only one marked vertex are used to extract the frontier on surface P facing surface X. For every triangle with one marked vertex, the edge that is not connected to the marked vertex is put into a frontier edge list F. The list F is build as follows:

During the automatic matching the list

$$T_F = \left\{ \mathbf{t}_l = \{\mathbf{v}_{l,0}, \mathbf{v}_{l,1}, \mathbf{v}_{l,2}\} \middle| \mathbf{v}_{l,i} \in P \text{ and } \sum_{i=0}^{2} w_{l,i} = 1 \right\} \quad (6.3)$$

that contains the triangles with only one coupled vertex is established. For every triangle in T_F, the vertices that are not coupled are inserted in

$$F = \left\{ \mathbf{v}_{l,i} \in \mathbf{t}_l \text{ with } \mathbf{t}_l \in T_F \text{ and } w_{l,i} = 0 \right\} \quad (6.4)$$

F is a two dimensional list which contains several potential frontiers on P composed of a sequence of ordered edges. The order of the edges is clockwise and defined by the normal vector of the triangle which an edge is part of. F is traced continuously in order to merge touching frontiers. Figure 6.5 shows the erosion and frontier extraction process for a typical triangle mesh.

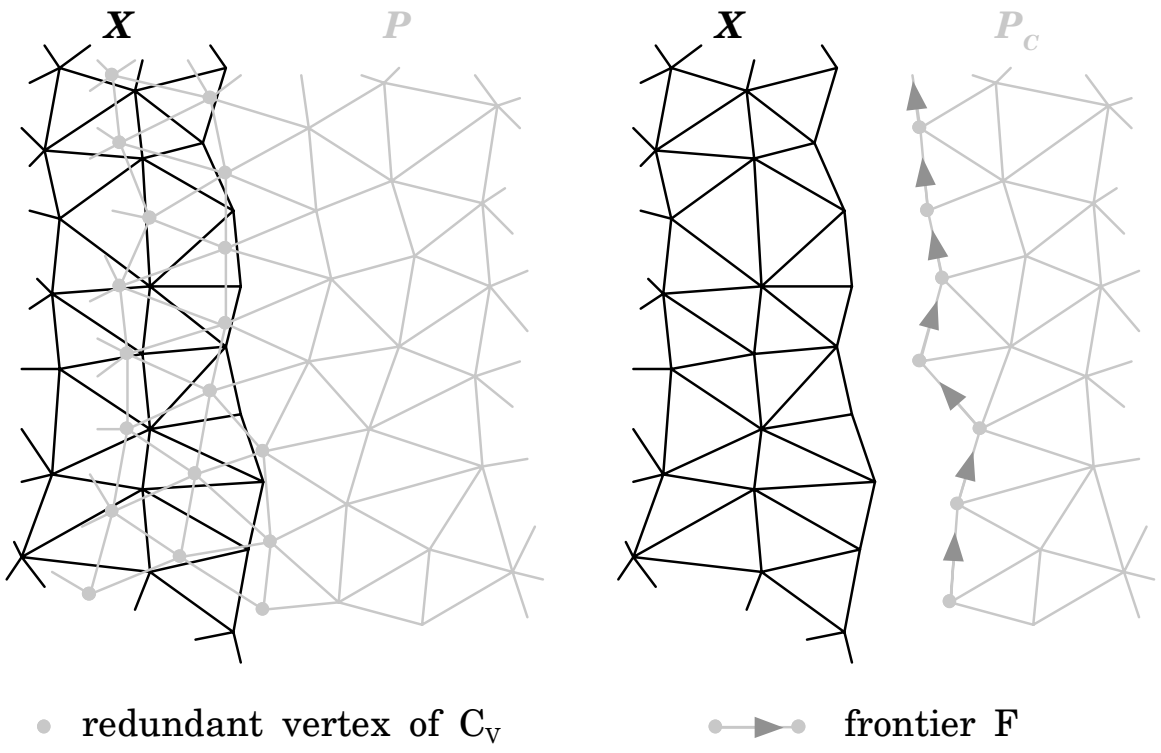


Figure 6.5: Erosion of the redundant zone and frontier detection

6.4.5 Gap filling

The gap between the two surfaces X and P_C is filled with triangles in order to join the two meshes. The different frontiers on P_C delimiting these gaps are processed sequentially. The filling process is initialised for a frontier on P_C with the search of the first vertex $\mathbf{x_N}$ on the frontier of X. To do so, the first two vertices $\mathbf{f_0}$ and $\mathbf{f_1}$ of the frontier list F are selected and the nearest point $\mathbf{x_N}$ to $\mathbf{f_0}$ and $\mathbf{f_1}$ on the frontier of X is calculated. Then, the first bridge triangle joining the two frontiers is constructed with the vertices $\mathbf{x_N}$, $\mathbf{f_0}$ and $\mathbf{f_1}$ as shown in Figure 6.6. The frontier list F is updated by setting its first vertex $\mathbf{f_0}$ equal to $\mathbf{x_N}$.

The following algorithm fills the gap iteratively starting with the above initialisation. Two candidate vertices are selected to build the next bridge triangle. One is $\mathbf{f_2}$, the third vertex in the frontier list F and the other one is $\mathbf{x_C}$, the next vertex on the frontier of X. These two candidates form together with the vertices $\mathbf{f_0}$ and $\mathbf{f_1}$ of F the next potential bride triangles as shown in Figure 6.6. The candidate that encloses the maximal angle is selected in order to obtain a regular triangulation. The frontier list F is updated with the new vertices as follows: $\mathbf{f_0}$ is set equal to $\mathbf{x_C}$ if $\mathbf{x_C}$ is chosen or $\mathbf{f_1}$ is removed from F if the candidate $\mathbf{f_2}$ is selected. The candidate selection starts again with the modified frontier list and the above procedure is applied until F contains only two vertices.

The following deteriorations of the gap-filling algorithm have to be avoided. First, if the frontiers of the two surfaces diverge which results in a large gap or second if the bridge triangle normal is negative indicating a filling in the wrong direction. In these cases new candidates are calculated, the filling process is initialised with the next edge from the edge list or the filling is stopped.

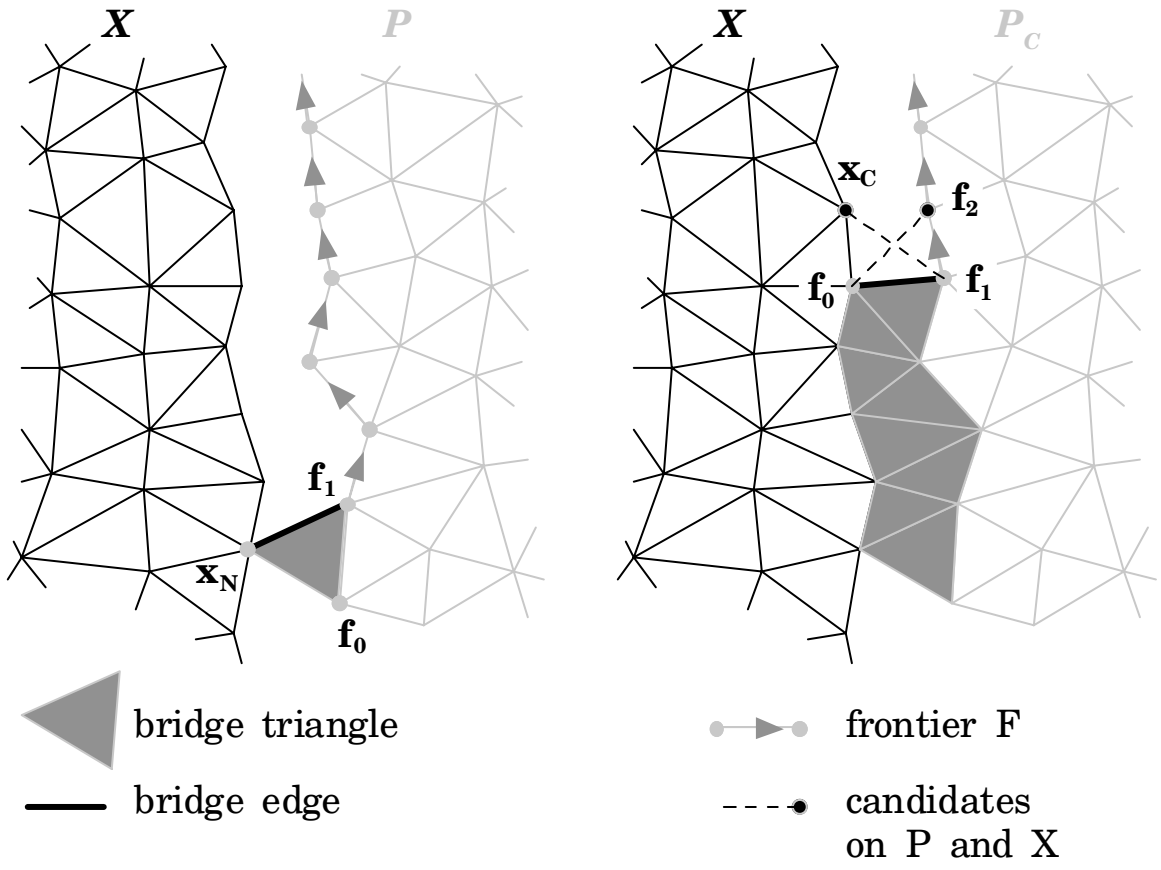


Figure 6.6: Gap filling initialisation and iteration procedure

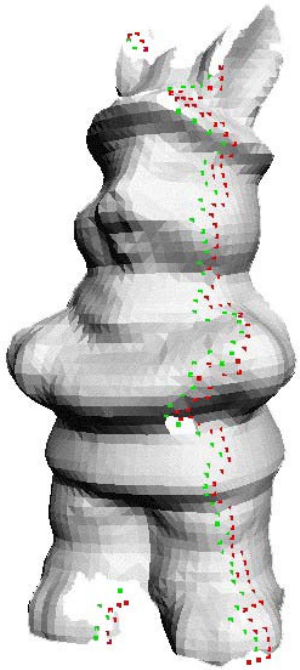


Figure 6.7: Fusion result

6.5 Colour acquisition

Beside geometry, colour data is essential to create a realistic model of an object: 3D models won't look realistic if only their geometrical aspect is taken into account; the appearance of an object is also important. Realistic appearance is obtained by assigning either a colour or a sub-image to each model mesh, respectively leading to a coloured or a texture-mapped object model.

Range finders based on structured lighting permit an easy measurement of both colour and geometric information since they mainly uses a CCD camera and a projector to retrieve range data. In addition, the colour picture of the object that is provided by the camera is in perfect registration with the range image. Unfortunately, the colour components delivered by these scanners express the reflected colour intensity of the object and not its intrinsic colour as required for the object model. A consequence of this is therefore the existence on the reconstructed model of strong colour discontinuities, which appear on the boundaries of two object views because their acquisition was done under different illumination conditions.

Two main reflectance types can be observed on a physical object: diffuse and specular. To account for it, we successively considered the Lambertian and then the Phong model. Three different approaches have been considered to obtain the desired intrinsic colour.

The first one converts the reflected colour intensity into the intrinsic colour by computation. It operates with the projector of the range finder as only light source, and uses therefore the colour image as provided by the video camera of the 3D scanner. In order to compensate for varying light illumination, it uses a reflectance model and operates then on known illumination, camera and surface orientations to compute the compensation.

The second approach aims at using a nearly constant, diffuse and omnidirectional illumination over the visible parts of the object. Creating such a uniform diffuse illumination around the object can be done with several white light sources that must be distributed as homogeneously as possible.

Finally, the third method proceeds according to the first approach but tries additionally to get rid of specularities by using several known illumination sources.

Figure 6.8 presents a comparison of the methods applied to a round spray can. The diffuse reflections compensation can be seen using all three methods. Basically, the first two methods can't do anything against specular reflections, which is annoying with very shiny objects. On the other hand, the multiple light compensation method shows to be useful here, the specular reflections being nearly invisible.

A detailed description of the different algorithms as well as more results can be found in [Jos99] and [Hug00].

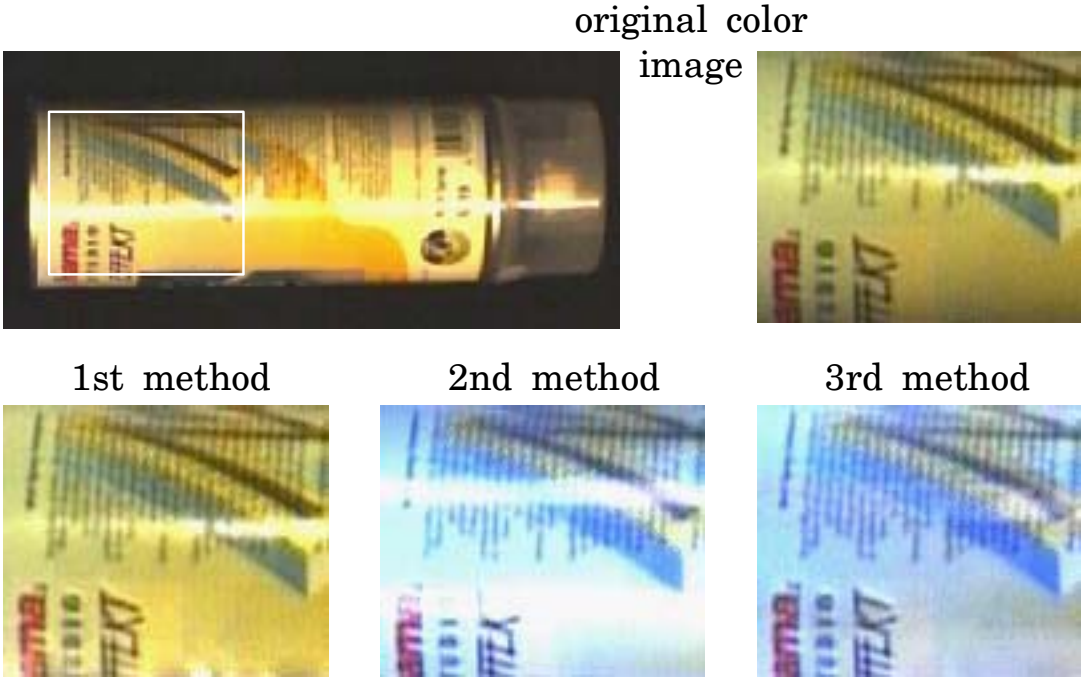


Figure 6.8: Comparison of the methods applied to a spray can

6.6 Handling textures

An extension of our reconstruction method that applies to textured views has been presented in [Jos98]. Advantages compared to other approaches are the capability to handle textured views and the lower complexity of object reconstruction.

The basic principle of the fusion of two views is to remove the overlapping part of one of the views, then to fill the resulting gap with new triangles and map them with the texture of the removed part. The resolution of the texture is checked to try to

keep the best texture resolution in each triangle. Finally, textures from the different views are blended at their boundaries to insure a smooth colour transition.

The work about the handling of textures in object digitising is found in [Jos98] and an example of textured object obtained with the presented method is shown in Figure 6.11.

6.7 Results, some models and applications

Some examples of objects scanned with the presented system are shown in this chapter. At the same time, they illustrate some applications of such a digitising system.

6.7.1 Reverse engineering

Reverse engineering consists in recreating a CAD model of an existing prototype object so as to make a copy of the prototype or to put it into production.

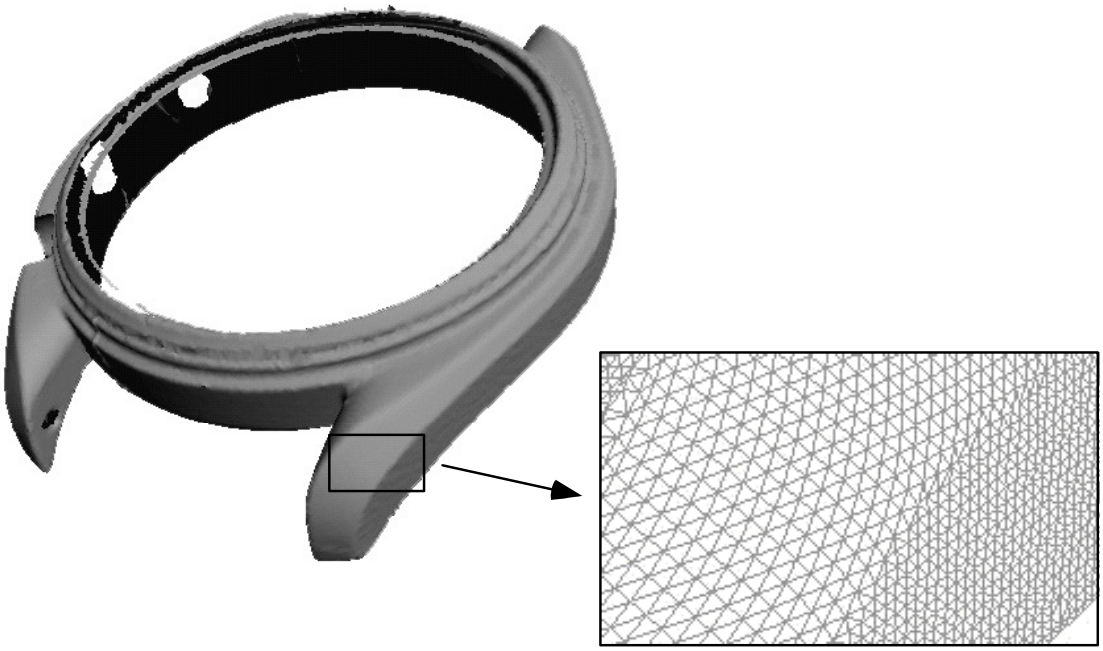


Figure 6.9: High resolution model of a watch frame

The presented example of reverse engineering modelling is a reconstruction of a watch frame. The goal of the work was to precisely measure the modified edges of the frame and then to update the existing original CAD model for mass production. In such case, high resolution is required to have a good precision. The resolution of this model is about 0.15mm. It contains 60000 vertices and 115000 faces and required about 20 different views.

6.7.2 Modelling from AFM images

Some tests have been made to register and model micrometric or even nanometric objects, specifically small quartz particles measured with an AFM microscope. This is definitely a domain where calibration devices or markers are very hard to use and where shape registration is required. Figure 6.10 shows a triangulated surface and the extracted triangulated view of a measured quartz particle. The resolution of the mesh is about 10 nm and it contains about 10000 points. A more complete publication of the research with AFM measurements and modelling can be found in [Jos01].

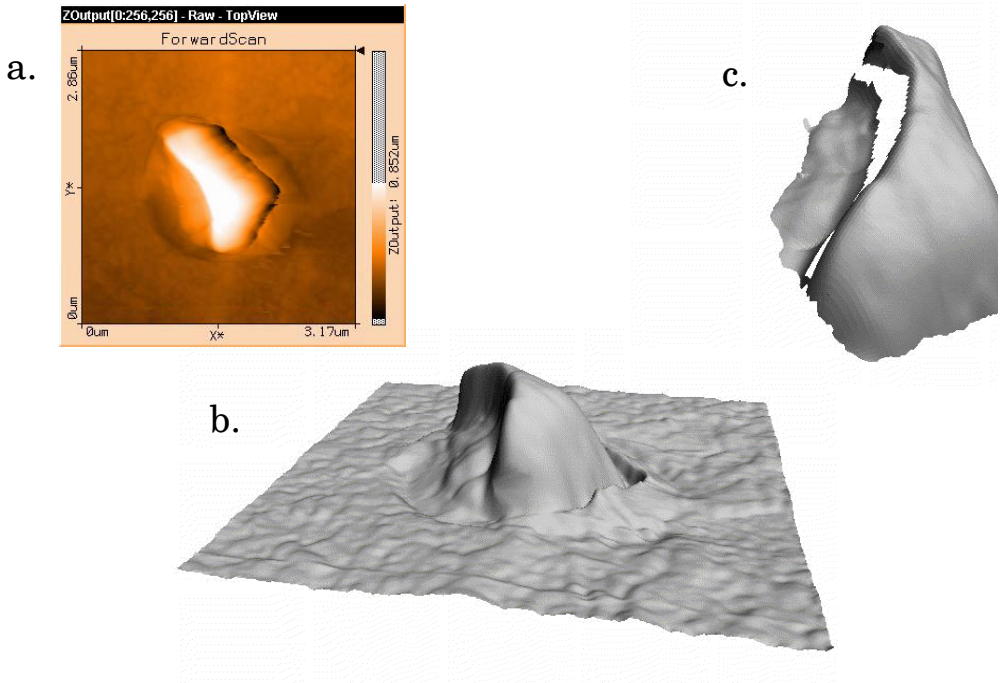


Figure 6.10: Range image (a), triangulated surface (b) and extracted triangulated view (c) of a quartz particle

6.7.3 Various objects

The first example, shown in Figure 6.11 is a texture-mapped model of a clay rabbit. This model was built using a decimation of the vertices and texture mapping. It permits to reduce the amount of data and to create a realistic looking model. The resolution of this model is about 2mm. It was reconstructed using 8 views and contains about 4000 vertices and 7800 faces. This result is a typical example of a multimedia type model, used for online shopping for example, which requires a good appearance and low data size but where high geometric precision is not necessary.

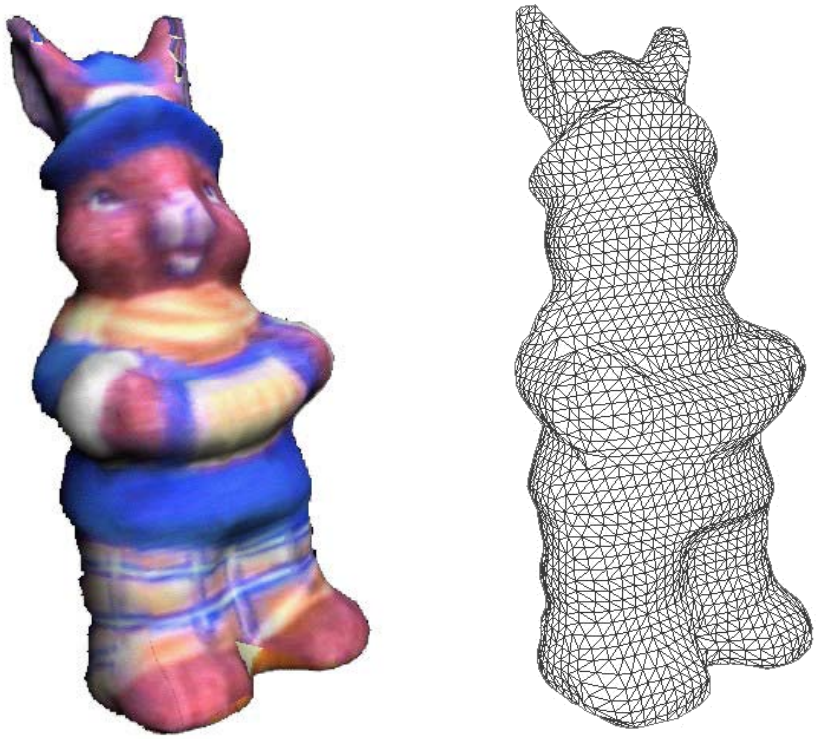


Figure 6.11: Textured model of a clay rabbit

The last example is some coloured models of two bones, presented in Figure 6.12. The resolution of the models is about 2mm. This example illustrates the use of virtual models for museums or in archaeology. Such models permit to study fragile objects under every angle without taking the risk of damaging the original. It also opens the possibility to easily examine objects to many people worldwide.

Figure 6.12: Coloured models of bones

6.8 Chapter conclusion

A complete digitising system that uses a new fusion algorithm has been presented in this chapter. It combines several range images to create a complete virtual model of an object. It considers triangulated views from unpositioned range images. The registration is separated into two steps, a first interactive rough positioning followed by an automatic matching using a variant of the ICP algorithm. The view fusion algorithm is coupled with registration, taking advantage of the closest point correspondences established during the automatic registration process. It keeps most of the existing view triangulation, removing redundant surfaces and linking remaining meshes together.

Other aspects of the object digitising, like colour digitising or texture handling have been analysed and new algorithms have been proposed and implemented as well. Several objects have been scanned and results proved the presented system to be effective, be it for precise, high-resolution models or for realistic looking, textured models.

Chapter 7
Conclusions

The research presented in this Ph.D. report mainly addresses the problem of the acceleration of the iterative closest point (ICP) algorithm.

The ICP algorithm is used for the registration of geometric data, which consists in finding the correct alignment of two or more datasets, based on their intrinsic properties. The working principle of the ICP consists in iteratively creating closest point correspondences between two sets of points and minimizing the average distance between both sets by a rigid transformation.

The main practical difficulty of the ICP algorithm is that it requires heavy computations and, thus, several speeding up methods have been proposed. A fairly complete review of the different methods has been proposed in this work. The main conclusion of this review is that most of the existing solutions lead to a tradeoff between speeding up and the quality of the matching.

A new closest points search algorithm for fast ICP has been presented. It consists of a heuristic that uses neighbourhood relationships to obtain a first approximation of the closest points and refines the results by a local search. Results showed that the proposed neighbour search algorithm performs significantly better than a k-D tree search, which is the standard fast closest point search. The theoretical complexity of $O(N_p)$ was practically reached.

The major drawback of the neighbour search is that the range of successful initial configuration tends to decrease when the resolution of the data increases. The main explanation of this problem is that the absolute size of the local search diminishes if the resolution of the data increases and the local search window remains the same.

A multiresolution scheme applied to the ICP has been proposed and analysed. This multiresolution scheme proceeds from coarse to fine and successively improves a previous solution at the finer representation level. Under these circumstances, the results show large speedup gains that can reach the best theoretical expectations. Combining the multiresolution scheme with the neighbour search presented above can lead to gains of more than 1600 over a non-accelerated ICP. Furthermore, the pure speedup potential of this combination goes together with improvements observed with respect to the convergence speed and the matching quality. Practically, multiresolution permits to fully eliminate the decrease of matching quality that can appear when using the neighbour search. Consequently, this combination permits to create a very fast ICP algorithm while avoiding a tradeoff between speedup and quality of the matching.

Finally, a complete digitising system has been built during the course of this research and is presented at the end of this report. It combines data from several unpositioned range images to create a complete virtual model of an object and uses a new fusion algorithm working on triangulated views. The registration is separated into two steps, a first interactive rough positioning followed by an automatic matching using a variant of the ICP algorithm.

Other aspects of the object digitising, like colour digitising or texture handling have been analysed and implemented. Several objects have been scanned and the results proved the presented system to be effective, be it for precise, high-resolution models or for realistic looking, textured models.

7.1 Possible extension and future work

The square zone local search is basic and simple but a smarter local search method could be considered later on. A steepest descent or a local 3 steps algorithm (coarse to fine), for example,

could both diminish the number of local searches and augment the exactness of the result.

The algorithm was mainly tested with range images but it extends to cloud of points where the neighbourhood relation exists. One of the next steps could be to actually implement the neighbour search for clouds of points and triangle meshes to assess its usability in this case.

When datasets are getting close together, the resulting rigid transformation is getting smaller at each iteration. If it is sufficiently small, we could consider using the correspondences of the previous iterations as a first approximation when a point possesses no valid neighbour or even for every points.

In this work, we only considered the registration of two datasets. Adapting a global registration, to register several datasets at once, with the multiresolution scheme and neighbour search could be interesting to decrease the propagation of errors that happens when matching datasets sequentially.

Finally, one of the challenge of object digitising concerns edges. The objects used in industry often have sharp edges. The scanning and reconstruction of such surfaces lead to new problems that are not fully solved yet.

Acknowledgements

Several people have contributed in different ways to the successful realization of this work.

First of all, I am deeply grateful to Prof. Heinz Hügli, head of the Pattern Recognition Laboratory at IMT Uni-Neuchâtel, who gave me the opportunity to work in his team and who kindly accepted to supervise the present dissertation. Working in Prof. Hügli's laboratory permitted me to be in an environment where freedom, encouragement and guidance all mixed up and made it possible to increase my knowledge of Vision and Pattern Recognition far beyond my main subject.

I would like to express my gratitude to Prof. Pierre-Jean Erard, director of the IIIA Uni-Neuchâtel, Prof. Rolf Ingold, head of the DIVA Research Group at the University of Fribourg and Dr. Franck Marzani, member of the Le2i at the University of Bourgogne, France, for their time and efforts in evaluating this dissertation as members of the jury. Their positive feedback during our discussions and during the examination is also largely appreciated.

My particular thanks go to Dr. Christian Schütz, first for introducing me to the 3D vision world, then for our 18 months collaboration during which profitable discussions and many knowledge exchanges took place.

I want to thank all of my colleagues and ex-colleagues at the IMT for the excellent ambiance that exists in our laboratory, making it a very pleasant place to work at. François Tièche, Christian Schütz, Olivier Hüsser and Nabil Ouerhani all provided support and were always open to discuss about technical or not-so-technical topics. I wish to thank our neighbours from the

Electronics & Signal Processing laboratory as well for all the extra-academic activities we got involved in.

I also express my gratitude to the people of the administrative team of the IMT for their efficiency and their amiability. Special thanks to Heinz Burri and Laurent Jeanrenaud, our system managers, who helped us master the mysteries of the network of workstations.

As a research and teaching assistant, I had the opportunity to supervise several students and trainees involved in different projects. Let them all find in these lines my appreciation for these interactions that gave place to many interesting technical discussions.

Part of this research has been funded by the Swiss National Science Foundation under project number 2100-43530. Its support is here thankfully acknowledged.

Finally, but most important, I want to thank my family and Marcel with all my heart for their encouragements and Myriam for supporting and encouraging me during the writing of this dissertation and for all her love.

Timothée Jost
September 2002

References

[3Dsca] http://www.3dscanners.com

[Abw3d] http://www.abw-3d.de/home_e.html

[Aru87] K.S. Arun, T.S. Huang and S.D. Blostein, "Least-Squares Fitting of Two 3-D Point Sets," *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, vol. 9, no. 5, pp. 698-700, 1987.

[Ben75] J.L. Bentley, "Multidimensional Binary Search Tree Used for Associative Searching," *Communications of the ACM*, vol. 8, no. 9, pp. 509-517, 1975.

[Ben95] R. Benjemaa and F. Schmitt, "Registering range views of complex objects," *Proceedings of the 4th European Conferences on Rapid Prototyping*, Paris, 12 pages, 1995.

[Ben96] R. Benjemaa and F. Schmitt, "Recalage rapide de surfaces 3D après projection dans des multi-zbuffers," *Proceedings of the 5th European Conferences on Rapid Prototyping*, Paris, 11 pages, 1996.

[Ben97] R. Benjemaa and F. Schmitt, "Fast Global Registration of 3D Sampled Surfaces Using a Multi-Z-Buffer Technique," *Proceedings of the International Conference on Recent Advances in 3D Imaging and Modelling (3DIM97)*, Ottawa, pp. 113-119, 1997.

[Ber96] R. Bergevin, M. Soucy, H. Gagnon and D. Laurendeau, "Towards a General Multi-View Registration Technique," *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, vol. 18, no. 5, pp. 540-547, 1996.

[Ber99] F. Bernardini, J. Mittleman, H. Rushmeier, C. Silva, G. Taubin, "The Ball-Pivoting algorithm for surface reconstruction," *IEEE Transactions on Vision and Computer Graphics*, vol. 5, pp. 349-359, 1999.

[Ber00] F. Bernardini, H. Rushmeier, "Strategies for registering range images from unknown camera positions," *Proceedings of SPIE EI00*, San Jose, vol. 3958, pp. 200-206, 2000.

[Bes92] P.J. Besl and N.D. McKay, "A Method for Registration of 3-D Shapes," *Proc. of IEEE Trans. on Pattern Analysis and Machine Intelligence*, vol. 14, no. 2, pp. 239-256, 1992.

[Bha84] B. Bhanu, "Representation and shape matching of 3-D objects," *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, vol. 6, no. 3, pp. 340-351, 1984.

[Bla95] G. Blais and M. Levine, "Registering multiview range data to create 3D computer objects," *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, vol. 17, no. 8, pp. 820-824, 1995.

[Bre99] A.D. Brett, A. Hills, C.J. Taylor, "A Method of 3D Surface Correspondence and Interpolation for Merging Shape", *Image and Vision Computing*, vol. 17, no. 8, pp. 635-642, 1999.

[Cha92] G. Champleboux, S. Lavallée, R. Szeliski, L. Brunie, "From accurate imaging sensor calibration to accurate model-based 3-D object localization," *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, pp. 83-89, 1992.

[Che92] Y. Chen and G. Medioni, "Object modelling by registration of multiple range images," *Int. Journal of Image and Vision Computing*, vol. 10, no. 3, pp. 145-155, 1992.

[Che98] C.S. Chen, Y.P. Hung, J.B. Cheng, "A Fast Automatic Method for Registration of Partially Overlapping Range Images," *Proceedings of International Conference on Computer Vision*, ICCV98, pp. 242-248, 1998.

[Chi88] C.H. Chien, Y.B. Sim, J.K. Aggarwal, "Generation of volume / surface octree from range data," *Proceedings of Conference on Computer Vision and Pattern Recognition*, Ann Arbor, pp. 254-260, 1988.

[Chu96] C.S. Chua, R. Jarvis, "3D Free-Form Surface Registration And Object Recognition," *International Journal of Computer Vision*, vol. 17, no.1, pp. 77-99, 1996.

[Dor96a] C. Dorai, "COSMOS: A Framework for Representation and Recognition of 3D Free-Form Objects," *PhD thesis, Dept. of Computer Science*, Michigan State Univ., 1996.

[Dor96b] C. Dorai, J. Weng, A.K. Jain, C. Mercer, "From images to models: Automatic 3D model construction from multiple views," *Proceedings of the International Conference on Pattern Recognition (ICPR96)*, pp. 770-774, 1996.

[Dor97] C. Dorai, J. Weng, A.K. Jain, "Optimal Registration of Object Views Using Range Data," *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, vol. 19, no. 10, pp. 1131-1138, 1997.

[Egg97] D.W. Eggert, A. Lorusso, R.B. Fisher, "Estimating 3-D rigid body transformations: A comparison of four major algorithms," *MVA*, vol. 9, no. 5/6, pp. 272-290, 1997.

[Fau86] O.R. Faugeras and M. Hebert, "The Representation, Recognition, and Locating of 3-D Objects," *The International Journal of Robotics Research*, vol. 5, no. 3, pp. 27-52, 1986.

[Fel94] J. Feldmar, N. Ayache, F. Betting, "3D-2D projective registration of free-form curves and surfaces," *research report, INRIA*, no. 2434, Sophia Antipolis, 1994.

[Fel97] J. Feldmar, J. Declerck, G. Malandain, N. Ayache, "Extension of the ICP algorithm to nonrigid intensity-based registration of 3D volumes," *Computer Vision and Image Understanding*, vol. 66, no. 2, pp. 193-206, 1997.

[Fri77] J.H. Friedman, J.L. Bentley, R.A. Finkel, "An Algorithm for Finding Best Matches in Logarithmic Expected Time," *ACM Transactions on Mathematical Software*, vol. 3, no. 3, pp. 209-226, 1977.

[God94] G. Godin, M. Rioux, and R. Baribeau, "3-D registration using range and intensity information," *Proceedings of SPIE Videometrics III*, Boston, vol. 2350, pp. 279-290, 1994.

[God95] G. Godin and P. Boulanger, "Range image registration through viewpoint invariant computation of curvature," *From Pixels to Sequences, Proc. conf. ISPRS*, Zurich, vol. 30, pp. 170-175, 1995.

[God01] G. Godin, D. Laurendeau, R. Bergevin, "A method for the registration of attributed range images," *Proceedings of Int. Conf. on Recent Advances in 3-D Digital Imaging and Modelling (3DIM01)*, Quebec, pp. 179-186, 2001.

[Gre00] M. Greenspan, G. Godin, J. Talbot, "Acceleration of binning nearest neighbor methods," *Proceedings of Vision Interface 2000*, Montreal, pp. 337-344, 2000.

[Gre01] M. Greenspan, G. Godin, "A nearest neighbor method for efficient ICP," *Proceedings of Int. Conf. on Recent Advances in 3-D Digital Imaging and Modelling (3DIM01)*, Quebec, pp. 161-168, 2001.

[Hil96] A. Hilton, A.J. Stoddart, J. Illingworth and T. Windeatt, "Marching triangles: range image fusion for complex object modelling," *IEEE International Conference on Image Processing*, Lausanne, vol. 2, pp. 381-384, 1996.

[Hor87] B.K.P. Horn, "Closed-form solution of absolute orientation using unit quaternions," *Journal of the Optical Society of America*, vol. 4, no. 4, pp. 629-642, 1987.

[Hou95] J.-C. Houg, "Reconstruction de formes tridimensionnelles," *report IMT/HU*, University of Neuchâtel, no. 384, 1995.

[Hug97] H. Hügli, C. Schütz, "Geometric Matching of 3D Objects: Assessing the Range of Successful Initial Configurations," *International Conference on Recent Advances in 3-D Digital Imaging and Modelling (3DIM97)*, Ottawa, pp. 101-106, 1997.

[Hug99a] H. Hügli, T. Jost, "Object modelling by geometric matching for a prospective portable 3D scanner," *Proceedings of Neuchatel COST 254 Workshop*, Neuchatel, pp. 67-70, 1999.

[Hug99b] H. Hügli, T. Jost, "3D models from unpositioned range images," *SPIE's International Technical Group Newsletter on Electronic Imaging*, vol. 10, no. 1, pp. 12 & 3, 1999.

[Hug00] H. Hügli, T. Jost, "Digitalisation et modélisation d'objets 3D colorés," *Actes du Congrès Numérisation 3D – Scanning 2000*, Paris, 8 pages, 2000.

[Hug02] H. Hügli, T. Jost, "A match and merge method for 3D modelling from range images," *Proceedings of the Int. Conf. On Signal Processing (ICSP02)*, Beijing, pp. 1783-1786, 2002.

[Innov] http://www.innovmetric.com

[Joh97a] A. Johnson and M. Hebert, "Surface registration by matching oriented points," *International Conference on Recent Advances in 3-D Digital Imaging and Modelling (3DIM97)*, Ottawa, pp. 121-128, 1997.

[Joh97b] A. Johnson and S.B. Kang, "Registration and Integration of Textured 3-D Data," *International Conference on Recent Advances in 3-D Digital Imaging and Modelling (3DIM97)*, Ottawa, pp. 234-241, 1997.

[Jos98] T .Jost, C. Schütz, H. Hügli, "Modelling 3D Textured Objects by Fusion of Multiple Views," *Proceedings of Signal Processing IX, Eusipco 98*, Rhodes, vol. 2, pp. 1073-1076, 1998.

[Jos99] T. Jost, C. Schütz, H. Hügli, "Colour digitising and modelling of free-form 3D objects," *Proceedings of SPIE EI99*, San Jose, vol. 3640, pp. 38-48, 1999.

[Jos01] T. Jost, H. Hügli, "Three-Dimensional Modelling from AFM Measurements," *Proceedings of SPIE Lase01*, San Jose, vol. 4275-08, pp. 61-70, 2001.

[Jos02a] T. Jost, H. Hügli, "Fast ICP algorithms for shape registration," *Proceedings of DAGM02*, Zürich, pp. 91-99, 2002.

[Jos02b] T. Jost, H. Hügli, "A multi-resolution scheme ICP algorithm for fast shape registration," *Proceedings of the 1st Int. Symp. on 3D Data Processing, Visualisation and Transmission (3DPVT02)*, Padova, pp. 540-543, 2002.

[Kam89] B. Kamgar-Parsi, L. Jones, A. Rosenfeld, "Registration of multiple overlapping range images: scenes without distinctive features," *Proceedings of Conference on Computer Vision and Pattern Recognition*, San Diego, pp. 282-290, 1989.

[Kre96] B. Krebs, P. Sieverding and B. Korn, "Correct 3D Matching via a Fuzzy ICP Algorithm for Arbitrary Shaped Objects," *DAGM96, Proc. of Mustererkennung*, Heidelberg, pp. 521-528, 1996.

[Lan01] C. Langis, M. Greenspan, G. Godin, "The parallel Iterative closest point algorithm," *Proceedings of Int. Conf. on Recent Advances in 3-D Digital Imaging and Modelling (3DIM01)*, Quebec, pp. 195-204, 2001.

[Lor87] W. Lorensen, H. Cline, "Marching Cubes: a high resolution 3D surface construction algorithm," *Computer Graphics (SIGGRAPH'87)*, pp. 163-169. 1987.

[Mar63] D.W. Marquardt, "An algorithm for least-squares estimation of nonlinear parameters," *Journal of Soc. Indust. Appl. Math.*, vol. 11, no. 2, pp. 431-441, 1963.

[Mas96] T. Masuda, K. Sakaue, N. Yokoya, "Registration and integration of multiple range images for 3-D model construction," *Proceedings CVPR*, Vienna, pp. 879-883, 1996.

[Men92] C.H. Menq, H.T. Yau and G.-Y. Lai, "Automated Precision Measurement of Surface Profile in CAD-Directed Inspection," *IEEE Trans. on Robotics and Automation*, vol. 8, no. 2, pp. 268-278, 1992.

[Mun93] B. Munch, P. Ruegsegger, "3D repositioning and differential images of volumetric CT measurements," *IEEE Transactions on Medical Imaging*, vol. 12, no. 3, pp. 509-514, 1993.

[Neu97] P. Neugebauer, "Geometrical cloning of 3D objects via simultaneous registration of multiple range images," *Proceedings SMA*, Aizu-Wakamatsu, pp. 130-139, 1997.

[Pit96] R. Pito, "Mesh Integration Based on Co-Measurements," *IEEE International Conference on Image Processing*, Lausanne, vol. 2, pp. 397-400, 1996.

[Pot83] M. Potmesil, "Generating Models of Solid Objects by Matching 3D Surface Segments," *Proc. Int. Joint Conf. on Artificial Intelligence*, Karlsruhe, pp. 1089-1093, 1983.

[Pre86] F.P. Preparata, M.I. Shamos, "Computational geometry: an introduction," *Springer-Verlag*, 1986.

[Pul99] K. Pulli, "Multiview registration for large data sets," *Proceedings of the International Conference on Recent Advances in 3D Imaging and Modelling (3DIM99)*, Ottawa, pp. 160-168, 1999.

[Rus01] S. Rusinkiewicz, M. Levoy, "Efficient Variants of the ICP Algorithm," *Proceedings of Int. Conf. on Recent Advances in 3-D Digital Imaging and Modelling (3DIM01)*, Quebec, pp. 145-152, 2001.

[Rut94] M. Rutishauser, M. Stricker, M. Trobina, "Merging Range Images of Arbitrarily Shaped Objects," *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, Seattle, pp. 573-580, 1994.

[Sch96] C. Schütz, "Range finder systems - a tentative overview," *report IMT/HU*, University of Neuchatel, no. 410, 1996.

[Sch97a] C. Schütz, H. Hügli, "Augmented reality using range images," *Proceedings of SPIE Stereoscopic Displays and Virtual Reality Systems*, San Jose, vol. 3012, pp. 472-478, 1997.

[Sch97b] C. Schütz, T. Jost, H. Hügli, "Free-Form 3D Object Reconstruction from Range Images," *Proceedings of VSMM'97*, Geneva, pp. 69-70, 1997.

[Sch98a] C. Schütz, T. Jost, H. Hügli, "Semi-Automatic 3D Object Digitising System Using Range Images," *Proceedings of ACCV98, Lecture Notes in Computer Science*, Springer, vol. 1351, pp. 490-497, 1998.

[Sch98b] C. Schütz, "Geometric point matching of free-form 3D objects," *PHD thesis*, University of Neuchatel, 1998.

[Sch98c] C. Schütz, T. Jost, H. Hügli, "Multi-Feature Matching Algorithm for Free-Form 3D Surface Registration," *International Conference on Pattern Recognition (ICPR98)*, Brisbane, pp. 982-984, 1998.

[Sch98d] C. Schütz, T. Jost, H. Hügli, "Numérisation d'objets 3D," *MSM Marchés Systèmes Management*, vol. 12, pp. 20-21, 1998.

[Sim95] D.A. Simon, M. Hebert and T. Kanade, "Techniques for Fast and Accurate Intra-Surgical Registration," *Journal of Image Guided Surgery*, vol. 1, no. 1, pp. 17-29, 1995.

[Sou95] M. Soucy, D. Laurendeau, "A general surface approach to the integration of a set of range views," *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, vol. 17, no. 4, pp. 344-358, 1995.

[Space] http://www.spacemouse.com

[Ste92] F. Stein, G.G. Medioni, "Structural Hashing: Efficient Three Dimensional Object Recognition," *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, vol. 14, no. 2, pp. 125-145, 1992.

[Sto96a] A.J. Stoddart, A. Hilton, "Registration of multiple point sets," *Int. Conf. on Pattern Recognition (ICPR)*, Vienna, pp. 40-44, 1996.

[Tar99] J.P. Tarel, N. Boujema, "A Coarse to Fine 3D Registration Method Based on Robust Fuzzy Clustering," *Computer Vision and Image Understanding*, vol. 73, no. 1, pp. 14-28, 1999.

[Tub02] D. Tubic, P. Hébert, D. laurendeau, "A volumetric approach for interactive 3D modeling," *Proceedings of the 1st Int. Symp. on 3D Data Processing, Visualisation and Transmission (3DPVT02)*, Padova, pp. 150-158, 2002.

[Tur94] G. Turk and M. Levoy, "Zippered polygon meshes from range images," *Proceedings ACM Siggraph*, Orlando, pp. 311-318, 1994.

[Wal91] M.W. Walker, L. Shao, R. Volz, "Estimating 3-D location parameters using dual number quaternions," *CVGIP: Image Understanding*, vol. 54, no. 3, pp. 358-367, 1991.

[Wei97] S. Weik, "Registration of 3-D partial surface models using luminance and depth information," *International Conference on Recent Advances in 3-D Digital Imaging and Modelling (3DIM97)*, Ottawa, pp. 93-100, 1997.

[Zha94] Z. Zhang, "Iterative point matching for registration of free-form curves and surfaces," *Int. Journal of Computer Vision*, vol. 13, no. 2, pp. 119-152, 1994.