Fast Image Registration

Ugo Jardonnet

Technical Report n^o0808, JUNE 2008
revision 1868

Abstract: Image registration is a process widely used in image processing. Considering two measurements A and B of the same object (say, a x-ray and a magnetic resonance image (MRI)), this technique estimates a transformation of A so that the object in A becomes aligned with the object in B. Basically this technique is able to superimpose the image A over the image B, allowing the client to see mixed information. This presentation will discuss the implementation of a fast image registration algorithm in Milena, the Cxx generic image processing library from the Olena platform, developed at the LRDE. Specific techniques used to improve this process will be introduced.

Résumé : Le recalage d'images est une technique classique en traitement d'images. Soit A et B deux images représentant le même objet (par exemple une radiographie et une image à résonance magnétique (IRM)), on calcul une transformation de A telle que le recalage de l'objet dans A soit aligné sur l'objet dans B. Typiquement, cette technique peut permettre la lecture simultanée de deux mesures A et B. Cet exposé discutera des procédés de recalage rapide utilisés dans Milena, la bibliothèque C++ générique de traitement d'images de la plate-forme Olena, développée au LRDE. Certaines amélioration seront présentées.

Keywords
Image processing, Image registration, Optimization

Laboratoire de Recherche et Développement de l'Epita
14-16, rue Voltaire – F-94276 Le Kremlin-Bicêtre cedex – France
Tél. +33 1 53 14 59 47 – Fax. +33 1 53 14 59 22
jardonnet@lrde.epita.fr – http://publis.lrde.epita.fr/200806-Seminar-jardonnet

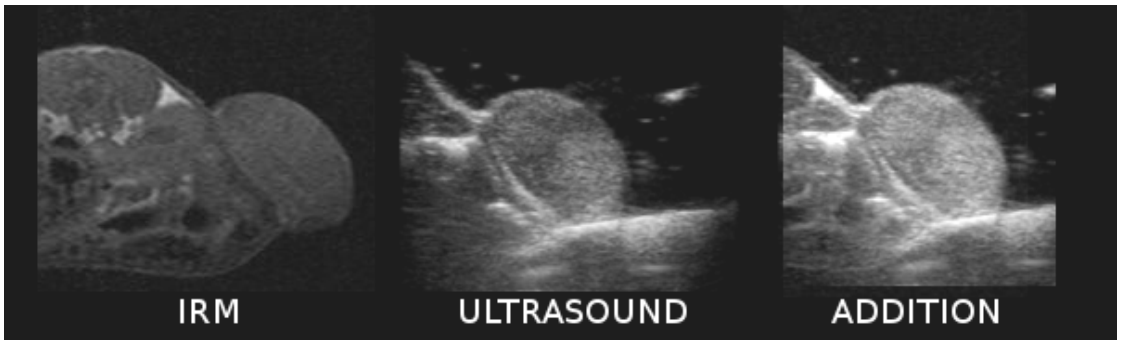
Copying this document

Contents

Chapter 1

Introduction

Image registration is to align objects from multimodal images. The following image shows the registration of a tumor coming from two different images, a magnetic resonance image (MRI) and an ultrasound. The resulting image on the right is the addition of these images. Information are clearly readable because registration has been correctly performed.



This paper describes a computationally efficient registration of 2-D and 3-D shapes. Based on the iterative closest point (ICP) algorithm, the method requires only the ability to find the closest point on a geometric entity to a given point. The ICP algorithm always converges monotonically to the nearest local minimum of a mean-square distance metric. Given a "model" shape and a sensed "data" shape that represents a major portion of the model shape, one can use our algorithm to register those shapes. Important applications of this method are shape registration, shape alignment or estimation of congruence between shapes.

This work is a part of the Olena project. Olena is a generic image processing library in C++ developped at the LRDE. This work on fast image registration has been performed in a context of cooperation with the Gustave Roussy Institute (Villejuif).

The first part of this report quickly recalls mathematical notions used further in the paper. In order to enable readers to reproduce our work, chapter two introduces in details an improved version of the iterative closest point algorithm from Bresl and McKay. The third chapter presents the new techniques we used to improved the efficiency of shape registration based on ICP.

Acknowledgments

I would like to thank Dr. Thierry Geraud and Roland Levillain for all the guidance throughout the project.

Thanks to Etienne Folio for his implementation of the Distance Map and Geraud Beguin for the test data.

Chapter 2

Mathematical Preliminaries

Expected value Expected value (or mathematical expectation, or mean) of a discrete random variable is the sum of the probability of each possible outcome of the experiment multiplied by the outcome value. Let X, be a random variable,

$$E(X) = \sum_i p_i x_i \tag{2.1}$$

Standard Deviation The standard deviation is a measure of statistical dispersion of values. Standard deviation of a random variable X is defined as:

$$\sigma = \sqrt{E((X - E(X))^2)} = \sqrt{E(X^2) - (E(X))^2} \tag{2.2}$$

Mean Square Error In statistical modelling, the MSE is defined as the difference between the actual observations and the response predicted by the model and is used to determine whether the model does not fit the data.

The MSE of an estimator $\hat{\theta}$ with respect to the estimated parameter θ is defined as

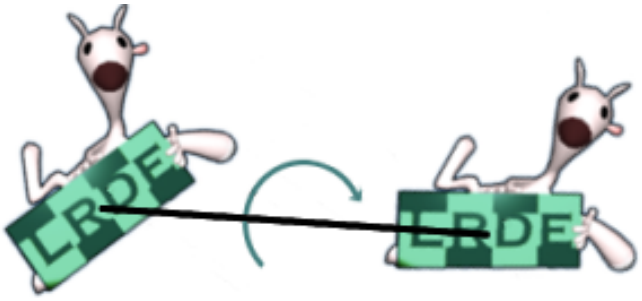
$$\widehat{MSE}(\hat{\theta}) = \frac{1}{n}\sum_{j=1}^{n}(\theta_j - \theta)^2 \tag{2.3}$$

Root Mean Square The Root Mean Square (RMS) is root square of the mean of the square values.

$$x_{\text{rms}} = \sqrt{\langle x^2 \rangle} \tag{2.4}$$

where $\langle \ldots \rangle$ denotes the arithmetic mean.

Rigid Transform An alignment or a **rigid** transform is the application of a translation and a rotation.



If P is a set of point, $\vec{q}(P)$ means that we apply the rigid transform $\vec{q}$ to every point in P.

Closest Point Given to shape C and X.
The closest point of a point p_c in C is the point p_x in X that minimizes the distance $d(p_c, p_x)$.

Projection The projection of C over X is the set of every closest points.

Chapter 3

Fast Iterative Closest Point (ICP)

3.1 ICP Algorithm

Given too point set P and X, this algorithm produces an optimal rigid transform $\vec{q}$ to a local minimum X_k such as the mean square error between X_k and $\vec{q}(P)$ is minimized.

Our implementation of the iterative closest point algorithm uses the quaternion-based algorithm that is only valid in two and three dimensions. However the singular value decomposition (SVD) algorithm can be used in order to generalize the method to n dimensions (Arun et al. [1], Gu et al. [11]).

The center of mass $\vec{\mu_p}$ of the measured point set P and the center of mass $\vec{\mu_x}$ for the X point set are given by

$$\vec{\mu_p} = \frac{1}{N_p}\sum_{i=1}^{N_p}\vec{p_i} \quad and \quad \vec{\mu_x} = \frac{1}{N_x}\sum_{i=1}^{N_p}\vec{x_i} \tag{3.1}$$

The cross-covariance matrix Σ_{px} of the set P and X is given by

$$\Sigma_{px} = \frac{1}{N^p}\sum_{i=1}^{N^P}[\,(\vec{p_i}-\vec{\mu_p})(\vec{x_i}-\vec{\mu_x})^t\,] = \frac{1}{N^p}\sum_{i=1}^{N^P}[\,\vec{p_i}\vec{x_i}^{\,t}] - \vec{\mu_p}\vec{\mu_x}^{\,t}. \tag{3.2}$$

The cyclic components of the anti-symmetric matrix $A_{ij} = (\Sigma_{px} - \Sigma_{px}^T)_{ij}$ are used to form the column vector $\Delta = [A_{23}A_{31}A_{12}]^T$. This vector is then used to form symmetric 4×4 matrix $Q(\Sigma_{px})$

$$Q(\Sigma_{px}) = \begin{bmatrix} tr(\Sigma_{px}) & \Delta^T \\ \Delta & \Sigma_{px} + \Sigma_{px}^T - tr(\Sigma_{px})I_3 \end{bmatrix} \tag{3.3}$$

Where I_3 is the 3×3 identity matrix. The unit eigenvector $\vec{q_R}$ corresponding to the maximum eigenvalue of the matrix $Q(\Sigma_{px})$ is selected as the optimal rotation. The *Jacobi* or the *power iteration* can be applied to the matrix $Q(\Sigma_{px})$ so as to obtain this eigenvector (Flannery et al. [6], Golub and Loan. [9]).

The optimal translation vector is given by

$$\vec{q_T} = \vec{\mu_x} - \mathbf{R}(\vec{a_R})\vec{\mu_p}. \tag{3.4}$$

The 7-space vector $\vec{q_k}$, representing the rigid transform in dimension 3, is constructed as a concatenation of $\vec{q_R}$ and $\vec{q_T}$. That is

$$\vec{q_k} = [\vec{q_R}|\vec{q_T}]^t \tag{3.5}$$

Initially, $\vec{q_k}$ is the identity vector and $P_k = P$.
At the beginning of every iteration, A cloud X_k, which is a projection of P_k over X, is built. Then, the current vector $\vec{q_k}$ is computed based on P and X_k. Finally, $\vec{q_k}$ is applied to P. Convergence is reach when the mean square error falls below a predefined value t.

1. Compute the projection: $X_k = CP(P_k, X)$
2. Compute the registration: $\vec{q_k} = Q(P_0, X_k)$
3. Apply the registration: $P_{k+1} = \vec{q_k}(P_0)$
4. Do it until convergence: $MSE(\vec{q_k}(P), X) < t$

3.2 Accelerated ICP

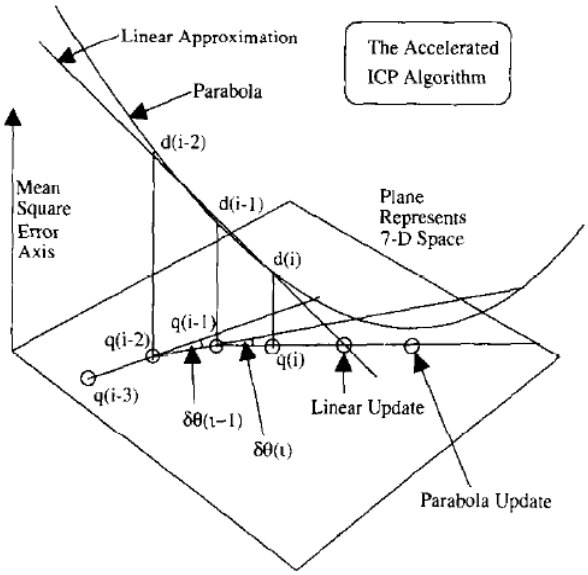
This optimization was originally proposed by Besl and McKay [2]. The idea of their paper was to prognosticate on the evolution of $\vec{qk}$, the rigid transform.

As the iterative closest point algorithm proceeds, a sequence of registration vectors is generated: $\vec{q_0}, \quad \vec{q_1}, \vec{q_2}, \vec{q_3}, \vec{q_4}, \ldots$, which traces out a path in the registration state space from the identity transformation toward a locally optimal shape match. Consider the difference vector sequence defined by

$$\Delta\vec{q_k} = \vec{q_k} - \vec{q_{k-1}} \tag{3.6}$$

which defines a direction in the registration state space.

Table 3.1: Variation of qk in the registration state space.



Let the angle in 7 space between the two last directions be denoted

$$\theta_k = \cos^{-1} \frac{\Delta\vec{q_k}^t \Delta\vec{q}_{k-1}}{\|\vec{q_k}\|\|\vec{q}_{k-1}\|} \tag{3.7}$$

and let $\delta\theta$ be a sufficiently small angular tolerance (e.g., 10°). If

$$\theta_k < \delta\theta \quad and \quad \theta_{k-1} < \delta\theta \tag{3.8}$$

then there is good direction alignment for last three registration state vectors: $\vec{q}_k, \vec{q}_{k-1}$, and $\vec{q}_{k-2}$. Let d_k, d_{k-1}, and d_{k-2} be the associated mean square errors, and let $v_k, v_{k-1}, and v_{k-2}$ be associated approximate arc length argument values:

$$v_k = 0, \tag{3.9}$$
$$v_{k-1} = -\|\Delta\vec{q_k}\|, \tag{3.10}$$
$$v_{k-2} = -\|\Delta\vec{q}_{k-1}\| + v_{k-1} \tag{3.11}$$

Next, a linear approximation and a parabolic interpolant to the last three data points are computed:

$$d_1(v) = a_1 v + b_1, \tag{3.12}$$
$$d_2(v) = a_2 v^2 + b_2 v + c_2 \tag{3.13}$$

which gives us a possible linear update, based on the zero crossing of the line, and a possible parabola update, based on the *extremum* point of the parabola:

$$v_1 = -b_1/a_1 > 0, v_2 = b_2/2a_2 \tag{3.14}$$

Besl and McKay adopt a maximum allowable value v_{max}. According to the following condition $\vec{q_k}$ is updated.

1. If $0 < v_2 < v_1 < v_{max}$ or $0 < v_2 < v_{max} < v_1$ use the parabola-based updated registration vector: $\vec{q_k}' = \vec{q_k} + v_2 \Delta\vec{q_k}/\|\Delta\vec{q_k}\|$ instead of the usual vector $\vec{q_k}$ when performing the update on the point set, i.e., $C_{k+l} = \vec{q_k}'(C_0)$.

2. If $0 < v_1 < v_2 < v_{max}$ or $0 < v_1 < v_{max} < v_2$ or $v_2 < 0$ and $0 < v_1 < v_{max}$, use the line-based updated registration vector $\vec{q_k}' = \vec{q_k} + v_1 \Delta\vec{q_k}/\|\Delta\vec{q_k}\|$ instead of the usual vector $\vec{q_k}$.

3. If both $v_1 > v_{max}$ and $v_2 > v_{max}$, use the maximum allowable update $\vec{q_k}' = \vec{q_k} + v_{max} \Delta\vec{q_k}/\|\Delta\vec{q_k}\|$ instead of the usual vector $\vec{q_k}$.

Based on Bresl and McKay recommendation, we used an arbitrary v_{max} value set to $25\|\Delta\vec{q_k}\|$.

This technique significantly improved the number of iterations needed to converge to the local minimum. Though sometimes without consequences on the number of iterations, this methods allowed us to earn an average of 20 iterations in some specific cases. Generally, this method improves results in terms of time and number of iterations (Besl and McKay [2]).

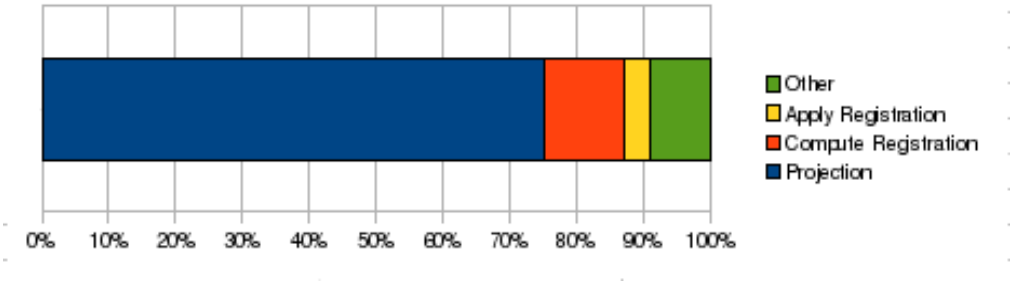
Chapter 4

Our proposals

This chapter introduce the work performed at the LRDE during the year 2007.

It is easy to see that the costliest part of the iterative closest point algorithm is the projection step (3.1). Indeed, finding closest points has a complexity in n^2. The following chart based on *gprof* analysis, clearly shows the importance of this step.

Table 4.1: ICP: Percentage of running time



We propose two original methods for the computing of closest points. Those methods, both improving running time, will be compared. Then an efficient implementation of the multi-scale registration is described. Finally, we present a new method introducing correction *a posteriori* of ICP results.

4.1 Distance Map Computation

Distance map Considering a binary image I on a domain $D(I)$, distance map is an image of points, that is an array holding points, that contains in every point p of $D(I)$ the closest point of p.

We saw that during the registration of a cloud of points P over an another cloud X, only the point set P was moving. Closest points are then taken in X for every point in P. We propose here to pre-compute closest point for all the domain of X so as to do not have to compute it

during the registration.

A distance map can be computed in $O(d)$, where d is the number of point in $D(I)$. We will not describe the implementation of such a map since E. Folio produced a consequent report on the subject [7].

Lazy map While still a map, this version is called lazy because it computes the distance only at call time, only the first though, since results are stored for every point. In this way we can say the version uses concepts of the dynamic programming paradigm. However, This calculus is performed in $O(n^2)$, which is quite costly.

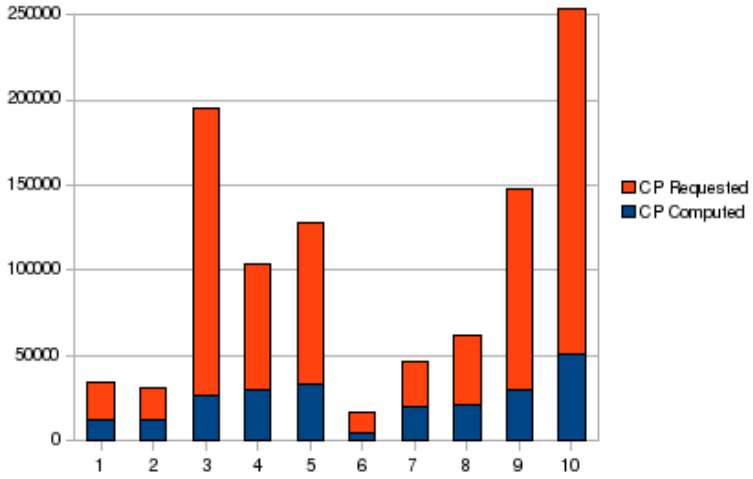
Listing 4.1: Lazy Map

```
class lazy_map
{
  ...
  const point
  operator () (const point& p) const
  {
    if (is_known(p))
      return value(p);
    value(p)    = closest_point(p);
    is_known(p) = true;
    return value(p);
  }
}
```

Using our test set (10 different registrations) we found out that more than 50% of closest point requests were computed at least twice. We saw that the transformation evolved first quickly then very slowly. Moreover we would be able to show that the translation only evolved even more quickly. Actually, most of the registration occurs in a very local area. Most of the closest point research is performed in the same small area. It results that closest point of the same points have a high probability to be computed twice.

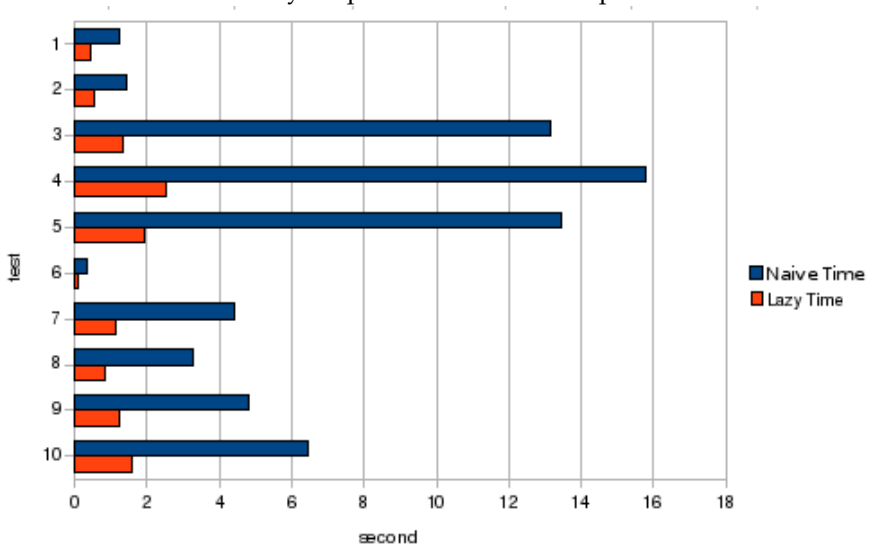
Typically we obtained an improvement of 39% using the lazy map compared to the original algorithm (not lazy 3.1). The following chart shows 10 different registrations and the part of closest point (CP) computed compared to the number of closest point requested (actually computed by the *not lazy* version).
We observe that the lazy map strongly decreases the number of closest points calculated (mean percentage of $CP\ Computed/CP\ requested$: 42.48, median: 39.04).

Table 4.2: Lazy Map: Closest Point (CP) Calculations



Projection being the costliest part of the algorithm, the execution time is seriously decreased (mean gain: 4.7, median: 3.92)[Athlon 2Ghz, RAM 1G]:

Table 4.3: Lazy Map: Execution Time Improvement



Discussion Let's naively compare both complexities. We observe that the lazy version is more effective than the distance map version if the number of points describing the object is lower than square root of the number of point in the domain.

Let n be the number of point describing the object and d the number of point in the domain. Complexities of the distance map and lazy map algorithms are respectively $O(d)$ and $O(n^2)$. We have $n^2 < d$ if and only if $n < \sqrt{d}$. That is, lazy map is more efficient than distance map if $n < \sqrt{d}$.

This assertion seems quite weak. First because we didn't compare complexities themselves but complexity classes. Then because the use of cache or such optimizations can deeply modify this kind of consideration. Finally we didn't mention the part of distances that are computed twice. This part definitely need further investigation.

We decided to avoid using distance map for the moment since we observed better time using the lazy map versions (between 5 and 25% of improvement). **Further parts of this report assume the use of the lazy map version.**

4.2 Multi-scale registration

Multi-scale techniques are often wise methods in image-processing (JL Starck and Bijaoui [12]). Such techniques aim at observing an image at different resolutions in order to obtain certain information in a less expensive way. Some properties are, indeed, very robust to resizing. For instance, morphological properties are pretty well preserved.

During image registration, real issues deal with morphological properties. We are continuously comparing 2 shapes. In fact, specialized press particularly recommends multi-scale image registration (Granger and Pennee [10], Rusinkiewicz and Levoy [16]). This techniques successively register the same image at low resolution first, higher resolutions next. Each consecutive registration is performed using informations acquired during the preceding registration.

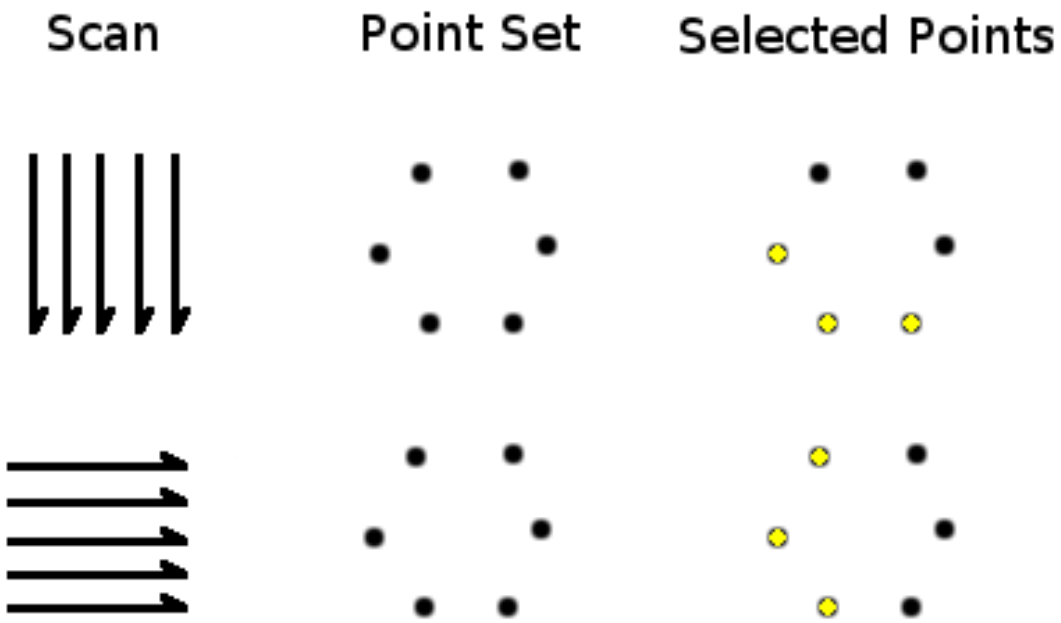
Image registration we used in this report uses clouds of points in order to represent objects. In practice, we use an array of point.

We would like to extract a representative sub-set of the general shape of the object from this vector. For the moment, let's say we take one point out of two in this vector. The fact that the sub-set is or not representative of the shape depend on the way points have been extracted.

Generally, such a vector is build upon a black and white image. This image is scanned and every white points is added to the vector. *De facto* points ordering strongly depends on the way the scan has been done; the vector building method is very central.

Let's consider our naive sub-set again (one point out of two). The following figure 4.4 shows that the shape of the subset is very sensitive to the scanning method. We notice that, generally speaking, this naive sub-set is not representative of the object's shape.

Table 4.4: Representativeness of a sub-set.



It has been shown that a sub-set extracted using a random selection generally conserves morphological properties of the object (McNeill and Vijayakumar [15]). Masuda et al. [14] uses Random sampling (with a different sample of points at each iteration). Turk and Levoy [18] also use uniform sub-sampling of the available points.

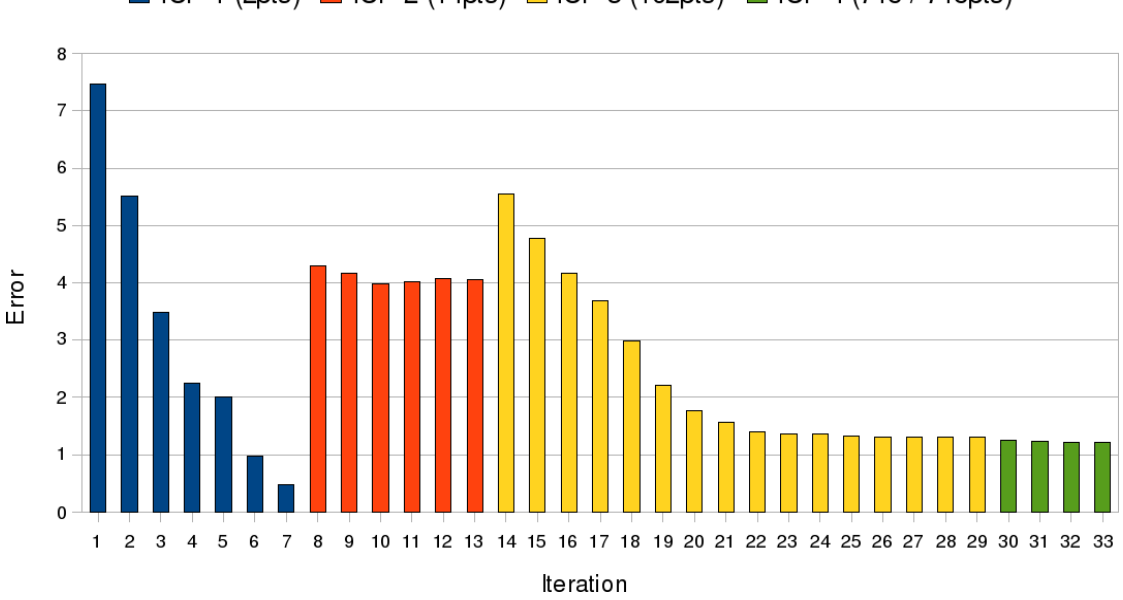
We suggest the following algorithm for multi-scale registration:

- e is the number of scales used
- q is a quotient chosen by users

Let N be the number of point in C.
We perform e registrations of a subset of N/q^i points from C, i varying from $e-1$ to 0.

The following charts represents the evolution of a multi-scale registration ($q = 7$,$e = 4$).

Table 4.5: Multi-Scale(7,4): Evolution of the Mean Square Error during 33 iterations

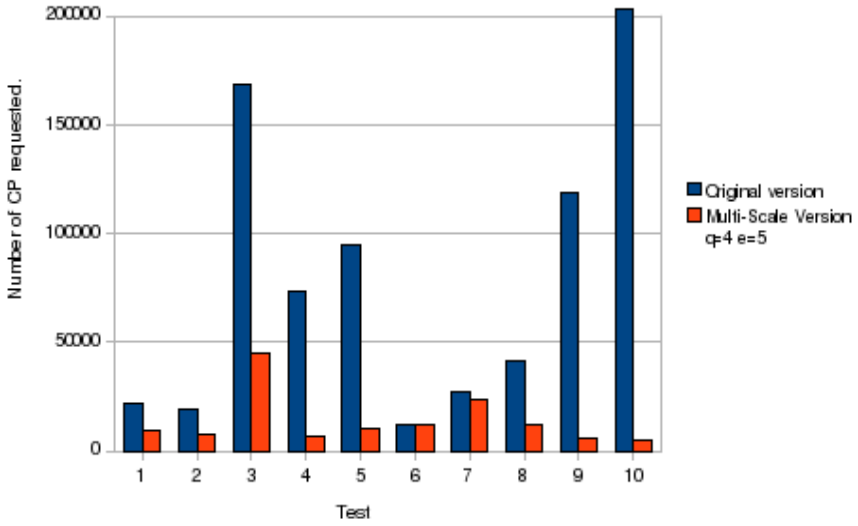


We can see that the last registration, which uses every point, has an extremely fast convergence. Indeed it benefits from information gained during the preceding registrations. As most of the information is acquired with only few points (registration 1, 2 and 3), only few closest points have been computed. More precisely, $\vec{q_k}$ is upgraded by ICP at every iteration but exists outside of the function. Thereby every registration is able to apply the last $\vec{q_k}$ before to start. Then it upgrade itself the $\vec{q_k}$.

This method offers different advantages. First the array is randomized only one time (which differs from Masuda et al. [14] version). Secondly, successive registrations re-use every points of the preceding registration so that every successive registration is guaranteed to be more accurate than its predecessor.

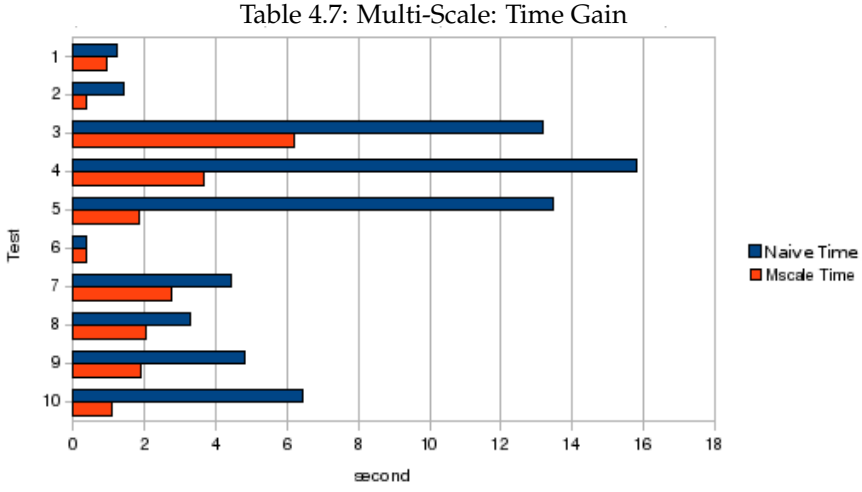
A multi-scale registration requests impressively less calculation of closest points than the non multi-scale version. The following table 4.6 compares the number of points processed using our test set for $q = 4$, $e = 5$. We observe that the multi-scale version processes an average of 65% of closest points less than the non multi-scale version.

Table 4.6: Multi-Scale: Improvement in terms of number of point processed



Projection being the costliest part of the algorithm, the execution time is again seriously decreased (mean gain: 3.12, median: 2.33) [Athlon 2Ghz, RAM 1G]:

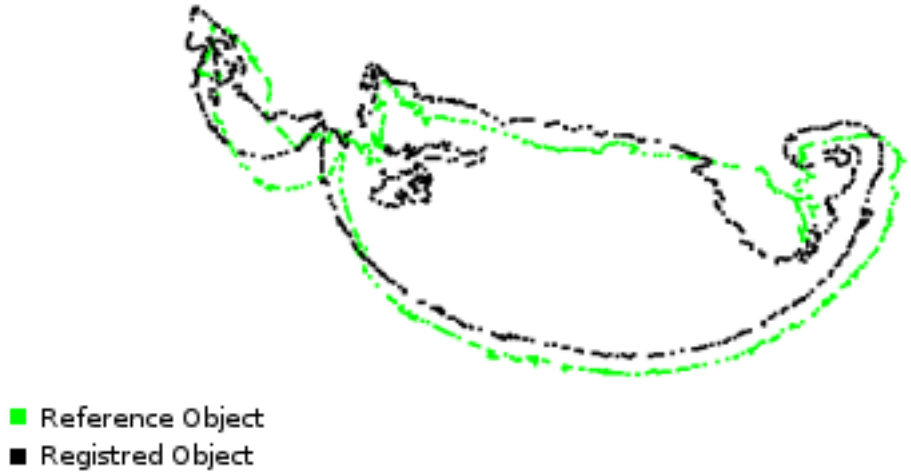
Table 4.7: Multi-Scale: Time Gain



4.3 Final Transform

Applying ICP may give bad results. Indeed, due to noise or huge deformations in the object, considering every points can be inadequate for a real shape alignment. The following figure shows an application of ICP which is not satisfactory.

Table 4.8: Instance of ICP registration.

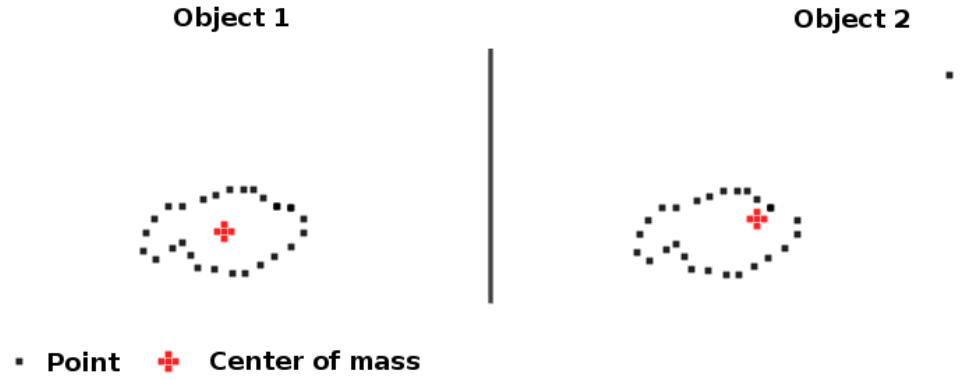


Indeed, the registered object presents a hole in its right side. This deformation compared to the reference object, modifies the value of the center of mass and of the greatest eigenvector of the cross-covariance matrix. Hence, the algorithm produce a results more or less far from the one a human would have done.

Visually, we would like to see the object's borders matching the reference object's borders almost everywhere. However ICP only minimizes the global mean square error 2. We would like a transformation minimizing error for "most of" the points.

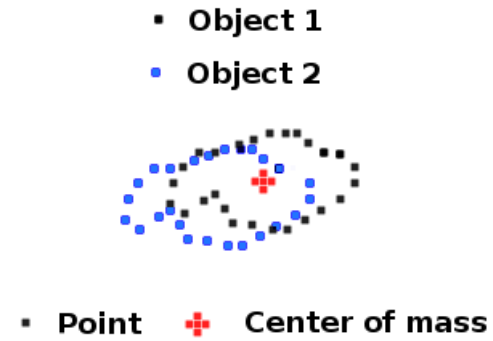
Consider two identical shapes, an object and a reference object. Let move one point of the object extremely far from its original location.

Table 4.9: One point only can influence a registration.



Existence of the right-most point in the object 2 prevents ICP to correctly align shapes. That is because the center of mass of the object is modified by the noised point. This leads to a wrong alignment:

Table 4.10: Registration disturbed by a noised point.

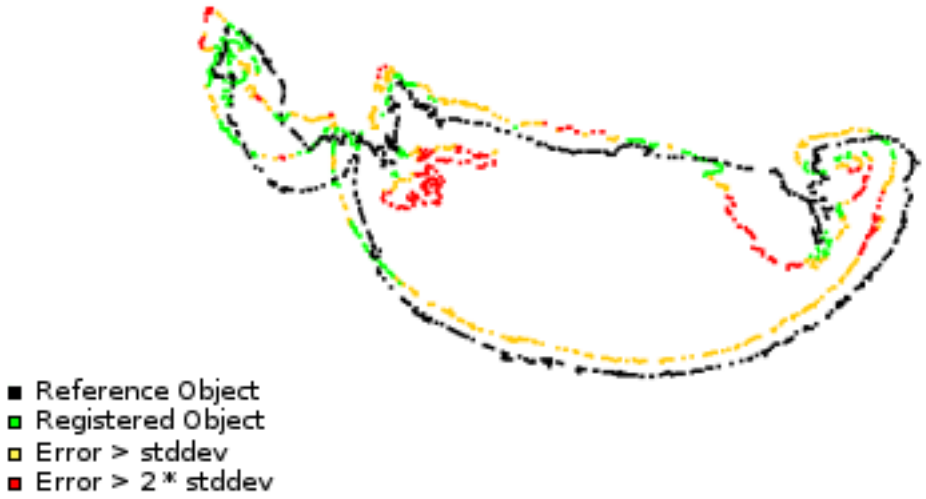


It is unfortunately not possible to say *a priori* if a point will or will not be significant for a given registration. We actually need to have shapes aligned so as to conclude.

The technique we are exposing here aims at correcting any registration, only once this one has been performed (Masuda et al. [14]). Thereby we take advantage of the fact the local mean square error minimum has been discovered and so of the final ICP projection, that is the set of closest point in *Object 1* for every points in *Object 2*. Hence, we only assume that the projection is correct.

Let's color every points according to their distances from their projection points. The standard deviation (stddev) is computed. Distances longer than 1 time the standard deviation are marked orange ■, distances longer than 2 times the standard deviation are marked red ■. Points from P and X are respectively in green ■ and black.

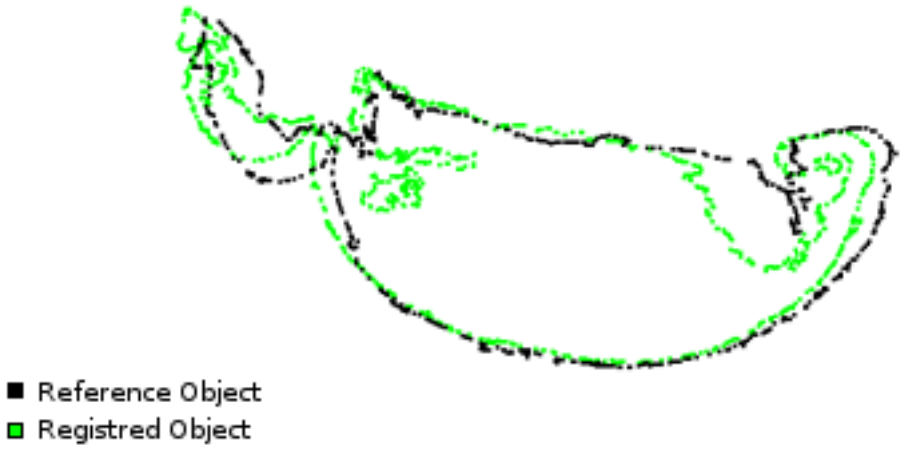
Table 4.11: Distance between points and their final projections.



We observe that filtering points with the standard deviation is relevant here. A big part of the hole on the left matches the *sup* $2 \times stddev$ range. The steam on the right, which does not exist in the reference object, matches too.

We propose to simply remove points further than 1 time the standard deviation and compute again a final rigid transform. Mean square error of the subset selected is mathematically decreased. Moreover, this way we achieve our goal to minimize the error for "most of" the points.

Table 4.12: Final transformation avoiding furthermost points.



Let's summarize this final registration process:

1. : Apply ICP registration.
2. : Compute every distances between each point and their projection on X.
3. : Compute the standard deviation of those distances.
4. : Construct a vector of points closer than the standard deviation from their projection.
5. : Compute the rigid transform between this vector and its projection on X.
6. : Apply the final transform to the registered object.

Alignement has been improved on every test of our test set.

Multi-scale point escaping We also suggest to take advantage of the multi-scale registration in order to pre-select furthermost points. Assuming that the good local minima is focused since the first scale of registration, we can mark furthermost points so that those points won't be evaluated during the following registration. Moreover, inasmuch as *bad* points are statically due to macro deformation, we propose a technique to anticipate on points to be escaped.

- Register at scale 0.
- Built A_0, the set of point farther than n times the standard deviation.
- Dilate A.
- Register at scale 1 while not using points from A_0.
- Built A_1 and dilate it.
- ...

This techniques unfortunately give bad results, even removing the dilation step. Future work should improve this technique.

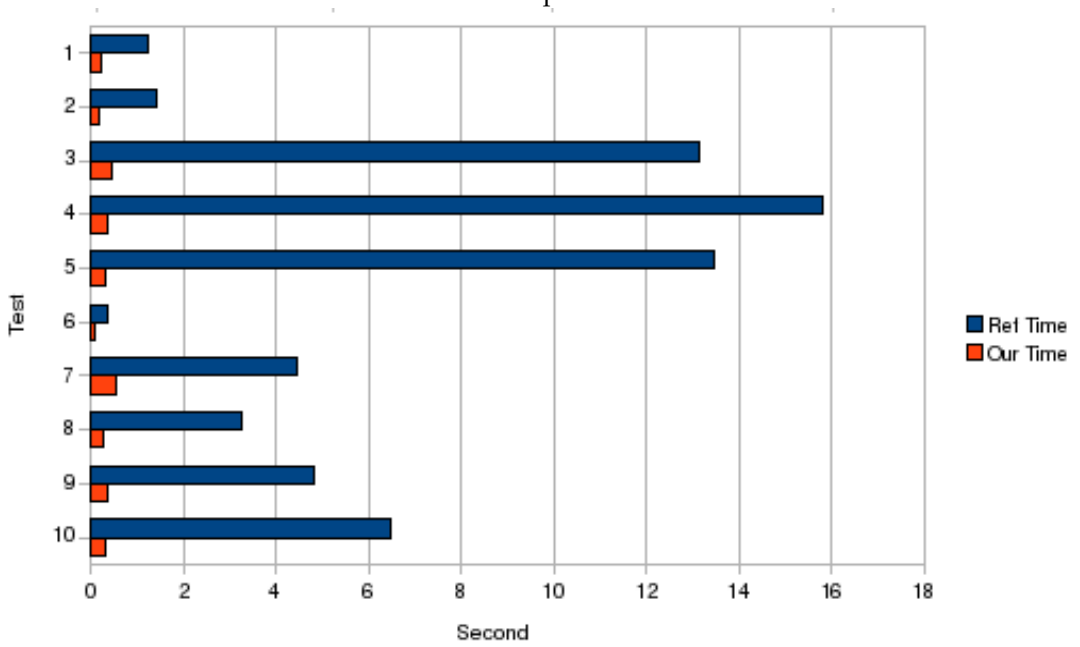
Chapter 5

Results

This chapter introduces results obtained mixing all of the three enhancements described before.

Combined together, multi-scaling and lazy map give interesting results in terms of time of execution. We observed an average gain of 18.8 $\times$ (ICP reference time divided by our modified version of ICP), median 12.94 [Athlon 2Ghz, RAM 1G]. Registration is performed with the same precision. Moreover if we apply the algorithm described in the section "Final transform" 4.12, the partial ("most of" the points) MSE decreases and the alignment (common meaning) seems better in any cases of our test set.

Table 5.1: Our Proposal: Time Gain



Chapter 6

Related Work

Many variants of ICP have been proposed, affecting all phases of the algorithm from the segmentation and matching of points to the minimization strategy.

Rusinkiewicz and Levoy [16] classify these variants as affecting one of six stages of the algorithm:

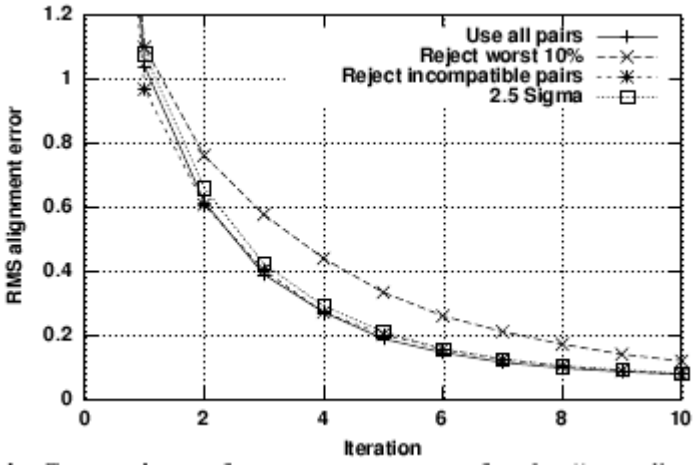
1. **Selection** of some set of points in one or both meshes.
2. **Matching** these points to samples in the other mesh.
3. **Rejecting** certain pairs based on looking at each pair individually or considering the entire set of pairs.
4. **Minimizing** the error metric.

Here follow some of the variants that distinguish from our work.

Selection of Points

- Always using all available points, Besl and McKay [2].
- Selection of points with high intensity gradient, in variants that use per-sample color or intensity to aid in alignment, Weik [19].
- Each of the preceding schemes may select points on only one object, or select source points from both objects, Godin et al. [8].
- Bucket points according to the position of the normals in angular space, then sample as uniformly as possible across the buckets [16].

Table 6.1: Average Convergence Rate According to Points Selection Methods



Besl and McKay [2] version is actually the one we used in the original ICP registration. About the Weik version, we actually doing almost the same thing since we pre-process shapes in order to extract boundary based on the difference of gradient.

Matching of points

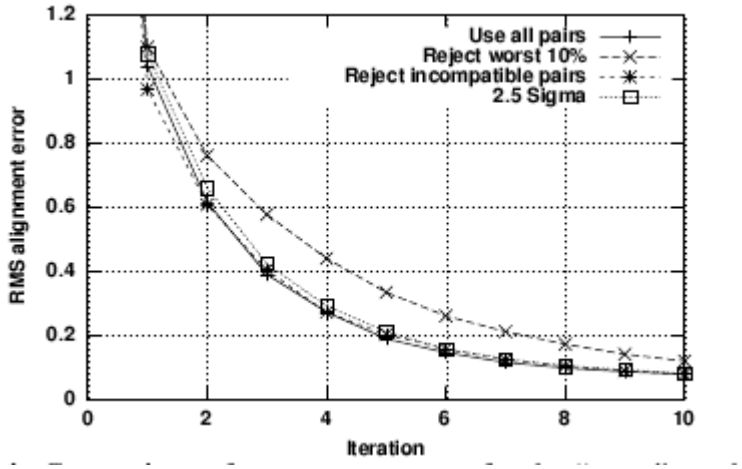
- Find the closest point in the other mesh (Besl and McKay [2]). This computation may be accelerated using a k-d tree and/or closest-point caching, Simon [17].
- Find the intersection of the ray originating at the source point in the direction of the source point's normal with the destination surface Chen and Medioni [4].

The use of k-d tree is a well known optimization for the projection step. This technique uses segmentation of the surface space (X) in order to reduce the cost of a closest point request. We plan to compare this method to our lazy map technique.

Rejecting Pairs

- Rejection of the worst n% of pairs based on some metric, usually point-to-point distance. As suggested by K [13], we reject 10% of pairs.
- Rejection of pairs that are not consistent with neighboring pairs, assuming surfaces move rigidly, Dorai et al. [5]. This scheme classifies two correspondences (p_1, q_1) and (p_2, q_2) as inconsistent iff $|Dist(p_1, p_2) - Dist(q_1, q_2)|$ is greater than some threshold.

Table 6.2: Average Convergence Rate According to Pair Rejection Methods



We did not used the Pulli version since the percentage of point rejected is purely subjective. We want something stronger and more objective. We could probably implement the Dorai optimization [16].

Error metric

- Performing the iterative minimization using various randomly-selected subsets of points, then selecting the optimal result using a robust (least median of squares metric Masuda et al. [14].
- Stochastic search for the best transform, using simulated an nealing, Blais and Levine [3].

We are also working on some tools that are sometimes considered as part of ICP registration. We would rather call it pre-processing.

- Image sub-sampling: Simplify initial image.
- Binarization (or Thresholding): Reduce quantity of useless information.
- Image auto-orientation: pre-align shapes under certain assumptions.

Chapter 7

Conclusion

The iterative closest point algorithm (ICP) is an efficient technique for shape registration, shape alignment or estimation of congruence between shapes.

In the first part of this report we described the method. We saw that ICP converges monotonically to the nearest local minimum of a mean-square distance metric. We also presented an optimization from Besl and McKay [2]. Those algorithms are now written in the Milena library and give efficient results for shape registration.

Then the report get onto three main optimization techniques, respectively treating problems from *Selection of Points* to *Rejection of Pairs*.
First we introduced the concept of Distance map. We saw that the use of a Lazy map drastically improves efficiency of the iterative closest point algorithm.
Then we addressed the problem of point selection with multi-scale registration. We also saw that the execution time could be really improved with such a technique.
Finally we spoke about pairs rejection while introducing an efficient way to decrease the mean square error at the end of the process.
Using those techniques, we obtained interesting results like a average improvement of 18.9 times concerning execution time.

Future works will deals with k-d trees so as to optimize our distance computation techniques. We also may consider the use of distance map for very local area in order to reduce the dimension of the domain and so to reduce the cost the distance map calculation. Fundamentally different methods for shape registration like heuristic methods also deserve to be discussed.

Chapter 8

Bibliography

[1] Arun, K. S., Huang, T., and Blostein, S. D. (1987). Least squere fitting of two 3-d point sets. *IEEE Trans. Pact. Anal. Machine Intell. vol. PAMI-9*.

[2] Besl, P. J. and McKay, N. D. (1992). A method for registration of 3-d shapes. *IEEE transactions on paltern analysis and machine intelligence, Vol. 14, NO. 2*.

[3] Blais, G. and Levine, M. (1995). Registering multiview range data to create 3d computer objects. *Trans. PAMI, Vol. 17, No. 8*.

[4] Chen, Y. and Medioni, G. (1991). Object modeling by registration of multiple range images. *Proc. IEEE Conf. on Robotics and Automation*.

[5] Dorai, C., Weng, J., and Jain, A. (1998). Registration and integration of multiple object views for 3d model constrution. *Trans. PAMI, Vol. 20, No. 1*.

[6] Flannery, B. P., Teukolsky, S. A., and Vetterling, W. T. (1992). *Numerical Recipes in FORTRAN: The Art of Scientific Computing, 2nd ed.* Cambridge England: Cambridge University Press.

[7] Folio, E. (2008). *Distance Transform*. LRDE Student Report.

[8] Godin, G., Rioux, M., and Baribeau, R. (1994). Three-dimensional registration using range and intensity information. *Proc. SPIE Videometrics III, Vol. 2350*.

[9] Golub, G. H. and Loan., C. F. V. (1989). *Matrix Computations, 2nd Edition. Johns Hopkins*. University Press.

[10] Granger, S. and Pennee, X. (2002). Multi-scale EM-ICP: A fast and robust approach for surface registration. Internal research report, INRIA.

[11] Gu, M., Demmel, J. W., and Dhillon, I. (1994). Efficient computation of the singular value decomposition with applications to least squares problems. Technical Report CS-94-257, institut, Knoxville, TN, USA.

[12] JL Starck, F. M. and Bijaoui, A. (1998). Image processing and data analysis: The multiscale approach. *Cambridge University Press*.

[13] K, P. (1999). Multiview registration for large data sets. *Proc. 3DIM*.

[14] Masuda, T., Sakaue, K., and Yokoya, N. (1996). Registration and integration of multiple range images for 3-d model construction. *Proc. CVPR*.

[15] McNeill, G. and Vijayakumar, S. (2001). 2d shape classification and retrieval. *Institute of Perception, Action and Behavior*.

[16] Rusinkiewicz, S. and Levoy, M. (1998). Efficient variants of the icp algorithm. *Stanford University*.

[17] Simon, M. (1996). Fast and accurate shape-based registration. *Carnegie Mellon University*.

[18] Turk, G. and Levoy, M. (1994). Zippered polygon meshes from range images. *Proc. SIGGRAPH*.

[19] Weik, S. (1997). Registration of 3-d partial surface models using luminance and depth information. *Proc. 3DIM*.