Efficient Variants of the ICP Algorithm

Szymon Rusinkiewicz
Marc Levoy

Stanford University

Abstract

The ICP (Iterative Closest Point) algorithm is widely used for geometric alignment of three-dimensional models when an initial estimate of the relative pose is known. Many variants of ICP have been proposed, affecting all phases of the algorithm from the selection and matching of points to the minimization strategy. We enumerate and classify many of these variants, and evaluate their effect on the speed with which the correct alignment is reached. In order to improve convergence for nearly-flat meshes with small features, such as inscribed surfaces, we introduce a new variant based on uniform sampling of the space of normals. We conclude by proposing a combination of ICP variants optimized for high speed. We demonstrate an implementation that is able to align two range images in a few tens of milliseconds, assuming a good initial guess. This capability has potential application to real-time 3D model acquisition and model-based tracking.

1 Introduction – Taxonomy of ICP Variants

The ICP (originally Iterative Closest Point, though Iterative Corresponding Point is perhaps a better expansion for the abbreviation) algorithm has become the dominant method for aligning three-dimensional models based purely on the geometry, and sometimes color, of the meshes. The algorithm is widely used for registering the outputs of 3D scanners, which typically only scan an object from one direction at a time. ICP starts with two meshes and an initial guess for their relative rigid-body transform, and iteratively refines the transform by repeatedly generating pairs of corresponding points on the meshes and minimizing an error metric. Generating the initial alignment may be done by a variety of methods, such as tracking scanner position, identification and indexing of surface features [Faugeras 86, Stein 92], "spin-image" surface signatures [Johnson 97a], computing principal axes of scans [Dorai 97], exhaustive search for corresponding points [Chen 98, Chen 99], or user input. In this paper, we assume that a rough initial alignment is always available. In addition, we focus only on aligning a single pair of meshes, and do not address the *global registration* problem [Bergevin 96, Stoddart 96, Pulli 97, Pulli 99].

Since the introduction of ICP by Chen and Medioni [Chen 91] and Besl and McKay [Besl 92], many variants have been introduced on the basic ICP concept. We may 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. **Weighting** the corresponding pairs appropriately.
4. **Rejecting** certain pairs based on looking at each pair individually or considering the entire set of pairs.
5. Assigning an **error metric** based on the point pairs.
6. **Minimizing** the error metric.

In this paper, we will look at variants in each of these six categories, and examine their effects on the performance of ICP. Although our main focus is on the speed of convergence, we also consider the accuracy of the final answer and the ability of ICP to reach the correct solution given "difficult" geometry. Our comparisons suggest a combination of ICP variants that is able to align a pair of meshes in a few tens of milliseconds, significantly faster than most commonly-used ICP systems. The availability of such a real-time ICP algorithm may enable significant new applications in model-based tracking and 3D scanning.

In this paper, we first present the methodology used for comparing ICP variants, and introduce a number of test scenes used throughout the paper. Next, we summarize several ICP variants in each of the above six categories, and compare their convergence performance. As part of the comparison, we introduce the concept of normal-space-directed sampling, and show that it improves convergence in scenes involving sparse, small-scale surface features. Finally, we examine a combination of variants optimized for high speed.

2 Comparison Methodology

Our goal is to compare the convergence characteristics of several ICP variants. In order to limit the scope of the problem, and avoid a combinatorial explosion in the number of possibilities, we adopt the methodology of choosing a *baseline* combination of variants, and examining performance as individual ICP stages are varied. The algorithm we will select as our baseline is essentially that of [Pulli 99], incorporating the following features:

- Random sampling of points on both meshes.
- Matching each selected point to the closest sample in the other mesh that has a normal within 45 degrees of the source normal.
- Uniform (constant) weighting of point pairs.
- Rejection of pairs containing edge vertices, as well as a percentage of pairs with the largest point-to-point distances.
- Point-to-plane error metric.
- The classic "select-match-minimize" iteration, rather than some other search for the alignment transform.

We pick this algorithm because it has received extensive use in a production environment [Levoy 00], and has been found to be robust for scanned data containing many kinds of surface features.

In addition, to ensure fair comparisons among variants, we make the following assumptions:

- The number of source points selected is always 2,000. Since the meshes we will consider have 100,000 samples, this corresponds to a sampling rate of 1% per mesh if source points are selected from both meshes, or 2% if points are selected from only one mesh.
- All meshes we use are simple perspective range images, as opposed to general irregular meshes, since this enables comparisons between "closest point" and "projected point" variants (see Section 3.2).
- Surface normals are computed simply based on the four nearest neighbors in the range grid.

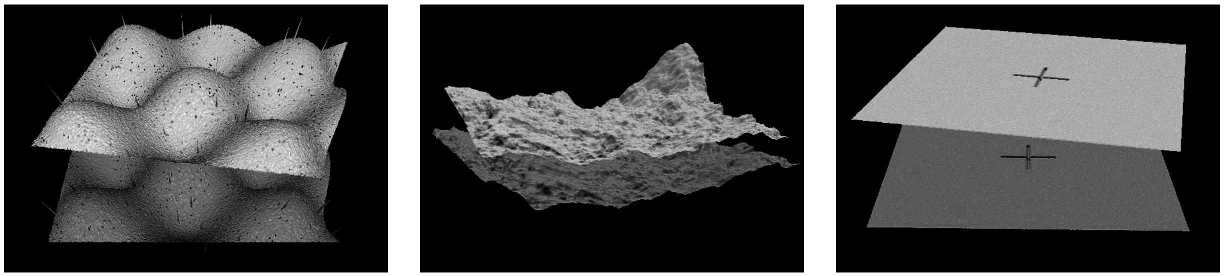

(a) Wave (b) Fractal landscape (c) Incised plane

Figure 1: Test scenes used throughout this paper.

- Only geometry is used for alignment, not color or intensity.

With the exception of the last one, we expect that changing any of these implementation choices would affect the quantitative, but not the qualitative, performance of our tests. Although we will not compare variants that use color or intensity, it is clearly advantageous to use such data when available, since it can provide necessary constraints in areas where there are few geometric features.

2.1 Test Scenes

We use three synthetically-generated scenes to evaluate variants. The "wave" scene (Figure 1a) is an easy case for most ICP variants, since it contains relatively smooth coarse-scale geometry. The two meshes have independently-added Gaussian noise, outliers, and dropouts. The "fractal landscape" test scene (Figure 1b) has features at all levels of detail. The "incised plane" scene (Figure 1c) consists of two planes with Gaussian noise and grooves in the shape of an "X." This is a difficult scene for ICP, and most variants do not converge to the correct alignment, even given the small relative rotation in this starting position. Note that the three test scenes consist of low-frequency, all-frequency, and high-frequency features, respectively. Though these scenes certainly do not cover all possible classes of scanned objects, they are representative of surfaces encountered in many classes of scanning applications. For example, the Digital Michelangelo Project [Levoy 00] involved scanning surfaces containing low-frequency features (e.g., smooth statues), fractal-like features (e.g., unfinished statues with visible chisel marks), and incisions (e.g., fragments of the Forma Urbis Romæ).

The motivation for using synthetic data for our comparisons is so that we know the correct transform exactly, and can evaluate the performance of ICP algorithms relative to this correct alignment. The metric we will use throughout this paper is root-mean-square point-to-point distance for the *actual* corresponding points in the two meshes. Using such a "ground truth" error metric allows for more objective comparisons of the performance of ICP variants than using the error metrics computed by the algorithms themselves.

We only present the results of one run for each tested variant. Although a single run clearly can not be taken as representing the performance of an algorithm in all situations, we have tried to show typical results that capture the significant differences in performance on various kinds of scenes. Any cases in which the presented results are not typical are noted in the text.

All reported running times are for a C++ implementation running on a 550 MHz Pentium III Xeon processor.

3 Comparisons of ICP Variants

We now examine ICP variants for each of the stages listed in Section 1. For each stage, we summarize the variants in the literature and compare their performance on our test scenes.

3.1 Selection of Points

We begin by examining the effect of the selection of point pairs on the convergence of ICP. The following strategies have been proposed:

- Always using all available points [Besl 92].
- Uniform subsampling of the available points [Turk 94].
- Random sampling (with a different sample of points at each iteration) [Masuda 96].
- Selection of points with high intensity gradient, in variants that use per-sample color or intensity to aid in alignment [Weik 97].
- Each of the preceding schemes may select points on only one mesh, or select source points from both meshes [Godin 94].

In addition to these, we introduce a new sampling strategy: choosing points such that the distribution of normals among selected points is as large as possible. The motivation for this strategy is the observation that for certain kinds of scenes (such as our "incised plane" data set) small features of the model are vital to determining the correct alignment. A strategy such as random sampling will often select only a few samples in these features, which leads to an inability to determine certain components of the correct rigid-body transformation. Thus, one way to improve the chances that enough constraints are present to determine all the components of the transformation is to bucket the points according to the position of the normals in angular space, then sample as uniformly as possible across the buckets. Normal-space sampling is therefore a very simple example of using surface features for alignment; it has lower computational cost, but lower robustness, than traditional feature-based methods [Faugeras 86, Stein 92, Johnson 97a].

Let us compare the performance of uniform subsampling, random sampling, and normal-space sampling on the "wave" scene (Figure 2). As we can see, the convergence performance is similar. This indicates that for a scene with a good distribution of normals the exact sampling strategy is not critical. The results for the "incised plane" scene look different, however (Figure 3). Only the normal-space sampling is able to converge for this data set.

The reason is that samples not in the grooves are only helpful in determining three of the six components of the rigid-body transformation (one translation and two rotations). The other three components (two translations and one rotation, within the plane)

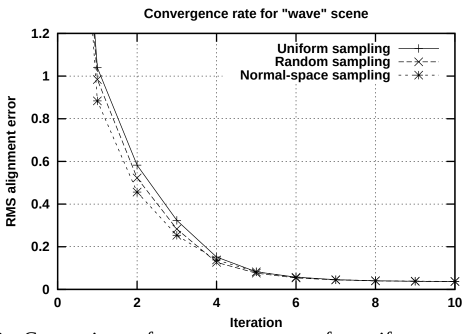


Figure 2: Comparison of convergence rates for uniform, random, and normal-space sampling for the "wave" meshes.

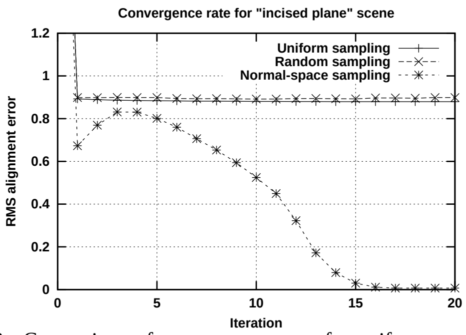


Figure 3: Comparison of convergence rates for uniform, random, and normal-space sampling for the "incised plane" meshes. Note that, on the lower curve, the ground truth error increases briefly in the early iterations. This illustrates the difference between the ground truth error and the algorithm's estimate of its own error.

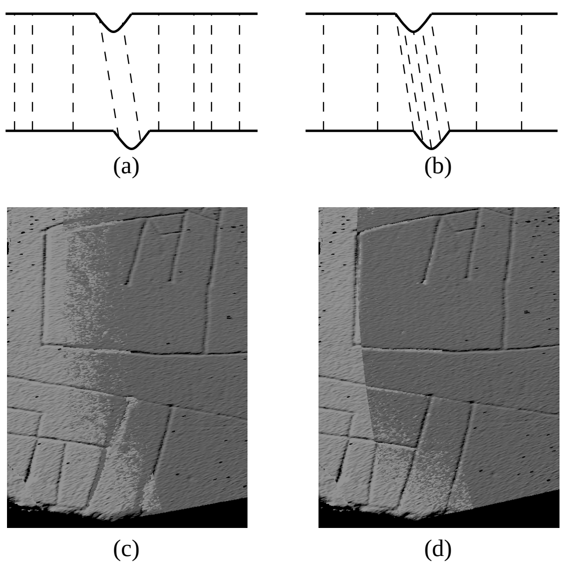


Figure 4: Corresponding point pairs selected by the (a) "random sampling" and (b) "normal-space sampling" strategies for an incised mesh. Using random sampling, the sparse features may be overwhelmed by presence of noise or distortion, causing the ICP algorithm to not converge to a correct alignment (c). The normal-space sampling strategy ensures that enough samples are placed in the feature to bring the surfaces into alignment (d). "Closest compatible point" matching (see Section 3.2) was used for this example. The meshes in (c) and (d) are scans of fragment 165d of the Forma Urbis Romæ.

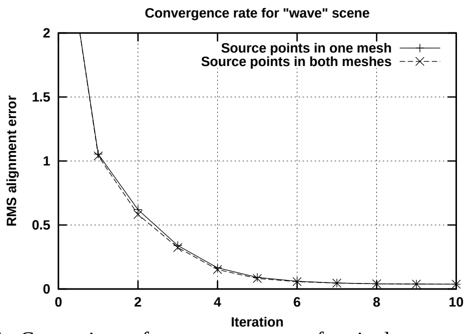


Figure 5: Comparison of convergence rates for single-source-mesh and both-source-mesh sampling strategies for the "wave" meshes.

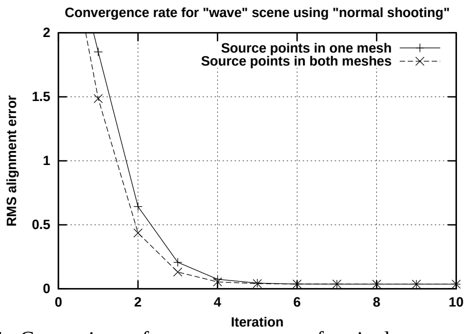


Figure 6: Comparison of convergence rates for single-source-mesh and both-source-mesh sampling strategies for the "wave" meshes, using normal shooting as the matching algorithm.

are determined entirely by samples within the incisions. The random and uniform sampling strategies only place a few samples in the grooves (Figure 4a). This, together with the fact that noise and distortion on the rest of the plane overwhelms the effect of those pairs that *are* sampled from the grooves, accounts for the inability of uniform and random sampling to converge to the correct alignment. Conversely, normal-space sampling selects a larger number of samples in the grooves (Figure 4b).

Sampling Direction: We now look at the relative advantages of choosing source points from both meshes, versus choosing points from only one mesh. For the "wave" test scene and the baseline algorithm, the difference is minimal (Figure 5). However, this is partly due to the fact that we used the closest compatible point matching algorithm (see Section 3.2), which is symmetric with respect to the two meshes. If we use a more "asymmetric" matching algorithm, such as projection or normal shooting (see Section 3.2), we see that sampling from both meshes appears to give slightly better results (Figure 6), especially during the early stages of the iteration when the two meshes are still far apart. In addition, we expect that sampling from both meshes would also improve results when the overlap of the meshes is small, or when the meshes contain many holes.

3.2 Matching Points

The next stage of ICP that we will examine is correspondence finding. Algorithms have been proposed that, for each sample point selected:

- Find the closest point in the other mesh [Besl 92]. This computation may be accelerated using a *k-d* tree and/or closest-point caching [Simon 96].

- 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 91]. We will refer to this as "normal shooting."

- Project the source point onto the destination mesh, from the point of view of the destination mesh's range camera [Blais 95, Neugebauer 97]. This has also been called "reverse calibration."

- Project the source point onto the destination mesh, then perform a search in the destination range image. The search might use a metric based on point-to-point distance [Benjemaa 97], point-to-ray distance [Dorai 98], or compatibility of intensity [Weik 97] or color [Pulli 97].

- Any of the above methods, restricted to only matching points compatible with the source point according to a given metric. Compatibility metrics based on color [Godin 94] and angle between normals [Pulli 99] have been explored.

Since we are not analyzing variants that use color, the particular variants we will compare are: closest point, closest compatible point (normals within 45 degrees), normal shooting, normal shooting to a compatible point (normals within 45 degrees), projection, and projection followed by search. The first four of these algorithms are accelerated using a *k-d* tree. For the last algorithm, the search is actually implemented as a steepest-descent neighbor-to-neighbor walk in the destination mesh that attempts to find the closest point. We chose this variation because it works nearly as well as projection followed by exhaustive search in some window, but has lower running time.

We first look at performance for the "fractal" scene (Figure 7). For this scene, normal shooting appears to produce the best results, followed by the projection algorithms. The closest-point algorithms, in contrast, perform relatively poorly. We hypothesize that the reason for this is that the closest-point algorithms are more sensitive to noise and tend to generate larger numbers of incorrect pairings than the other algorithms (Figure 8).

The situation in the "incised plane" scene, however, is different (Figure 9). Here, the closest-point algorithms were the only ones that converged to the correct solution. Thus, we conclude that although the closest-point algorithms might not have the fastest convergence rate for "easy" scenes, they are the most robust for "difficult" geometry.

Though so far we have been looking at error as a function of the number of iterations, it is also instructive to look at error as a function of running time. Because the matching stage of ICP is usually the one that takes the longest, applications that require ICP to run quickly (and that do not need to deal with the geometrically "difficult" cases) must choose the matching algorithm with the fastest performance. Let us therefore compare error as a function of time for these algorithms for the "fractal" scene (Figure 10). We see that although the projection algorithm does not offer the best convergence per iteration, each iteration is faster than an iteration of closest point finding or normal shooting because it is performed in constant time, rather than involving a closest-point search (which, even when accelerated by a *k-d* tree, takes $O(\log n)$ time). As a result, the projection-based algorithm has a significantly faster rate of convergence vs. time. Note that this graph does not include the time to compute the *k-d* trees used by all but the projection algorithms. Including the precomputation time (approximately 0.64 seconds for these meshes) would produce even more favorable results for the projection algorithm.

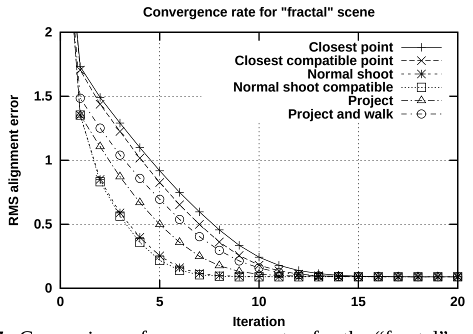


Figure 7: Comparison of convergence rates for the "fractal" meshes, for a variety of matching algorithms.

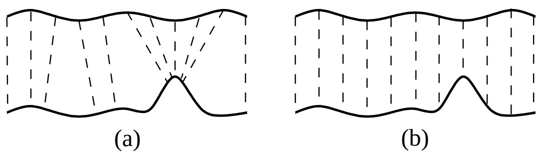


Figure 8: (a) In the presence of noise and outliers, the closest-point matching algorithm potentially generates large numbers of incorrect pairings when the meshes are still relatively far from each other, slowing the rate of convergence. (b) The "projection" matching strategy is less sensitive to the presence of noise.

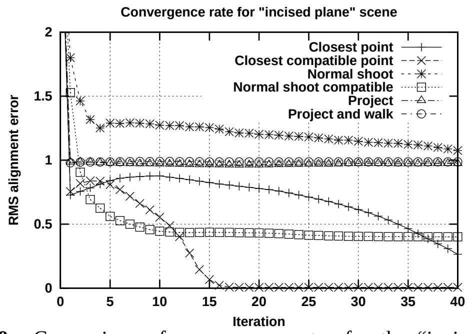


Figure 9: Comparison of convergence rates for the "incised plane" meshes, for a variety of matching algorithms. Normal-space-directed sampling was used for these measurements.

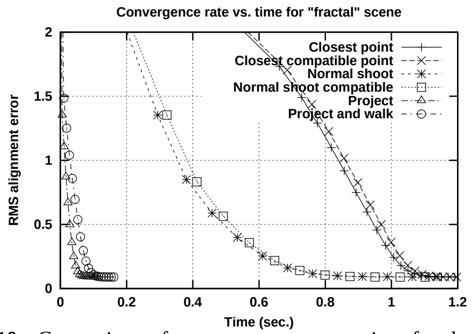


Figure 10: Comparison of convergence rate vs. time for the "fractal" meshes, for a variety of matching algorithms (cf. Figure 7). Note that these times do not include precomputation (in particular, computing the *k-d* trees used by the first four algorithms takes 0.64 seconds).

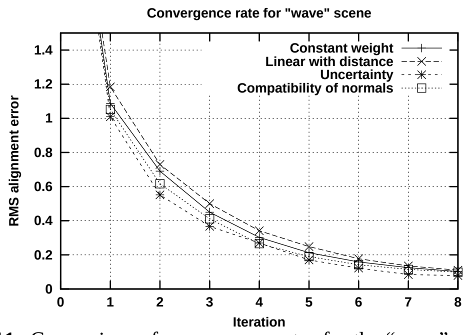


Figure 11: Comparison of convergence rates for the "wave" meshes, for several choices of weighting functions. In order to increase the differences among the variants we have doubled the amount of noise and outliers in the mesh.

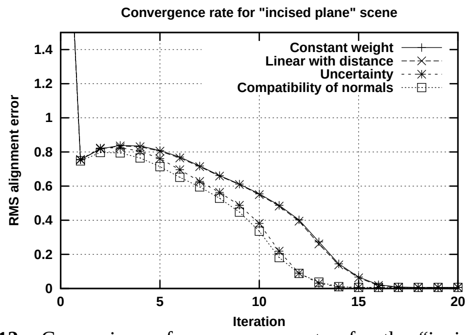


Figure 12: Comparison of convergence rates for the "incised plane" meshes, for several choices of weighting functions. Normal-space-directed sampling was used for these measurements.

3.3 Weighting of Pairs

We now examine the effect of assigning different weights to the corresponding point pairs found by the previous two steps. We consider four different algorithms for assigning these weights:

- Constant weight
- Assigning lower weights to pairs with greater point-to-point distances. This is similar in intent to dropping pairs with point-to-point distance greater than a threshold (see Section 3.4), but avoids the discontinuity of the latter approach. Following [Godin 94], we use

$$Weight = 1 - \frac{Dist(p_1, p_2)}{Dist_{max}}$$

- Weighting based on compatibility of normals:

$$Weight = n_1 \cdot n_2$$

  Weighting on compatibility of colors has also been used [Godin 94], though we do not consider it here.
- Weighting based on the expected effect of scanner noise on the uncertainty in the error metric. For the point-to-plane error metric (see Section 3.5), this depends on both uncertainty in the position of range points and uncertainty in surface normals. As shown in the Appendix, the result for a typical laser range scanner is that the uncertainty is lower, hence higher weight should be assigned, for surfaces tilted away from the range camera.

We first look at a version of the "wave" scene (Figure 11). Extra noise has been added in order to amplify the differences among the variants. We see that even with the addition of extra noise, all of the weighting strategies have similar performance, with the "uncertainty" and "compatibility of normals" options having marginally better performance than the others. For the "incised plane" scene (Figure 12), the results are similar, though there is a larger difference in performance. However, we must be cautious when interpreting this result, since the uncertainty-based weighting assigns higher weights to points on the model that have normals pointing away from the range scanner. For this scene, therefore, the uncertainty weighting assigns higher weight to points within the incisions, which improves the convergence rate. We conclude that, in general, the effect of weighting on convergence rate will be small and highly data-dependent, and that the choice of a weighting function should be based on other factors, such as the accuracy of the final result; we expect to explore this in a future paper.

3.4 Rejecting Pairs

Closely related to assigning weights to corresponding pairs is rejecting certain pairs entirely. The purpose of this is usually to eliminate outliers, which may have a large effect when performing least-squares minimization. The following rejection strategies have been proposed:

- Rejection of corresponding points more than a given (user-specified) distance apart.
- Rejection of the worst n% of pairs based on some metric, usually point-to-point distance. As suggested by [Pulli 99], we reject 10% of pairs.
- Rejection of pairs whose point-to-point distance is larger than some multiple of the standard deviation of distances. Following [Masuda 96], we reject pairs with distances more than 2.5 times the standard deviation.
- Rejection of pairs that are not consistent with neighboring pairs, assuming surfaces move rigidly [Dorai 98]. This scheme classifies two correspondences (p_1, q_1) and (p_2, q_2) as inconsistent iff

$$\left| Dist(p_1, p_2) - Dist(q_1, q_2) \right|$$

  is greater than some threshold. Following [Dorai 98], we use

$$0.1 \cdot \max(Dist(p_1, p_2), Dist(q_1, q_2))$$

  as the threshold. The algorithm then rejects those correspondences that are inconsistent with most others. Note that the algorithm as originally presented has running time $O(n^2)$ at each iteration of ICP. In order to reduce running time, we have chosen to only compare each correspondence to 10 others, and reject it if it is incompatible with more than 5.
- Rejection of pairs containing points on mesh boundaries [Turk 94].

The latter strategy, of excluding pairs that include points on mesh boundaries, is especially useful for avoiding erroneous pairings (that cause a systematic bias in the estimated transform) in cases when the overlap between scans is not complete (Figure 13). Since its cost is usually low and in most applications its use has few drawbacks, we always recommend using this strategy, and in fact we use it in all the comparisons in this paper.

Figure 14 compares the performance of no rejection, worst-10% rejection, pair-compatibility rejection, and $2.5\,\sigma$ rejection on the "wave" scene with extra noise and outliers. We see that rejection of outliers does not help with initial convergence. In fact, the algorithm that rejected pairs most aggressively (worst-10% rejection) tended to converge more slowly when the meshes were

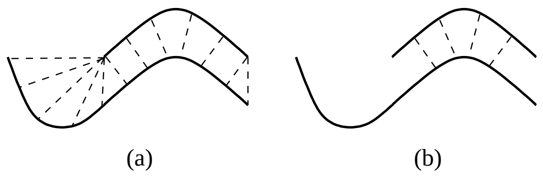


Figure 13: (a) When two meshes to be aligned do not overlap completely (as is the case for most real-world data), allowing correspondences involving points on mesh boundaries can introduce a systematic bias into the alignment. (b) Disallowing such pairs eliminates many of these incorrect correspondences.

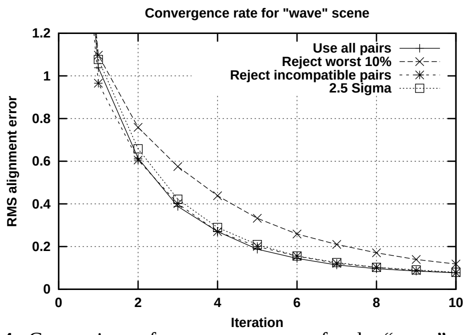


Figure 14: Comparison of convergence rates for the "wave" meshes, for several pair rejection strategies. As in Figure 11, we have added extra noise and outliers to increase the differences among the variants.

relatively far from aligned. Thus, we conclude that outlier rejection, though it may have effects on the accuracy and stability with which the correct alignment is determined, in general does not improve the speed of convergence.

3.5 Error Metric and Minimization

The final pieces of the ICP algorithm that we will look at are the error metric and the algorithm for minimizing the error metric. The following error metrics have been used:

- Sum of squared distances between corresponding points. For an error metric of this form, there exist closed-form solutions for determining the rigid-body transformation that minimizes the error. Solution methods based on singular value decomposition [Arun 87], quaternions [Horn 87], orthonormal matrices [Horn 88], and dual quaternions [Walker 91] have been proposed; Eggert et. al. have evaluated the numerical accuracy and stability of each of these [Eggert 97], concluding that the differences among them are small.
- The above "point-to-point" metric, taking into account both the distance between points and the difference in colors [Johnson 97b].
- Sum of squared distances from each source point to the plane containing the destination point and oriented perpendicular to the destination normal [Chen 91]. In this "point-to-plane" case, no closed-form solutions are available. The least-squares equations may be solved using a generic non-linear method (e.g. Levenberg-Marquardt), or by simply linearizing the problem (i.e., assuming incremental rotations are small, so $\sin\theta \sim \theta$ and $\cos\theta \sim 1$).

There are several ways to formulate the search for the alignment:

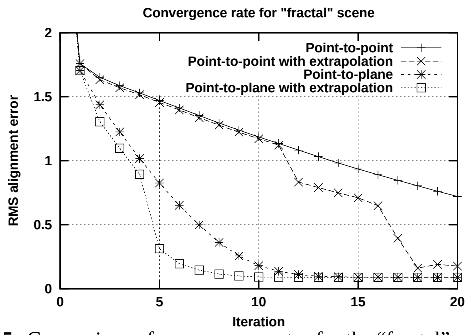


Figure 15: Comparison of convergence rates for the "fractal" meshes, for different error metrics and extrapolation strategies.

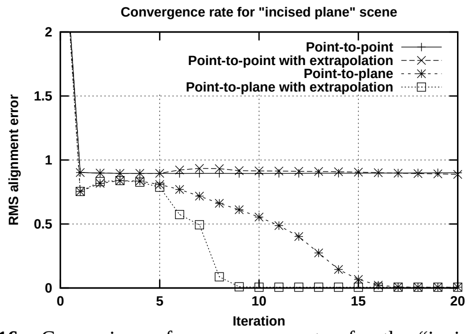


Figure 16: Comparison of convergence rates for the "incised plane" meshes, for different error metrics and extrapolation strategies. Normal-space-directed sampling was used for these measurements.

- Repeatedly generating a set of corresponding points using the current transformation, and finding a new transformation that minimizes the error metric [Chen 91].
- The above iterative minimization, combined with extrapolation in transform space to accelerate convergence [Besl 92].
- Performing the iterative minimization starting with several perturbations in the initial conditions, then selecting the best result [Simon 96]. This avoids spurious local minima in the error function, especially when the point-to-point error metric is used.
- 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 96].
- Stochastic search for the best transform, using simulated annealing [Blais 95].

Since our focus is on convergence speed, and since the latter three approaches tend to be slow, our comparisons will focus on the first two approaches described above (i.e., the "classic" ICP iteration, with or without extrapolation). The extrapolation algorithm we use is based on the one described by Besl and McKay [Besl 92], with two minor changes to improve effectiveness and reduce overshoot:

- When quadratic extrapolation is attempted and the parabola opens downwards, we use the largest *x* intercept instead of the extremum of the parabola.
- We multiply the amount of extrapolation by a dampening factor, arbitrarily set to $^1\!/_2$ in our implementation. We have found that although this occasionally reduces the benefit of

extrapolation, it also increases stability and eliminates many problems with overshoot.

On the "fractal" scene, we see that the point-to-plane error metric performs significantly better than the point-to-point metric, even with the addition of extrapolation (Figure 15). For the "incised plane" scene, the difference is even more dramatic (Figure 16). Here, the point-to-point algorithms are not able to reach the correct solution, since using the point-to-point error metric does not allow the planes to "slide over" each other as easily.

4 High-Speed Variants

The ability to have ICP execute in real time (e.g., at video rates) would permit significant new applications in computer vision and graphics. For example, [Hall-Holt 01] describes an inexpensive structured-light range scanning system capable of returning range images at 60 Hz. If it were possible to align those scans as they are generated, the user could be presented with an up-to-date model in real time, making it easy to see and fill "holes" in the model. Allowing the user to be involved in the scanning process in this way is a powerful alternative to solving the computationally difficult "next best view" problem [Maver 93], at least for small, hand-held objects. As described by [Simon 96], another application for real-time ICP is model-based tracking of a rigid object.

With these goals in mind, we may now construct a high-speed ICP algorithm by combining some of the variants discussed above. Like Blais and Levine, we propose using a projection-based algorithm to generate point correspondences. Like Neugebauer, we combine this matching algorithm with a point-to-plane error metric and the standard "select-match-minimize" ICP iteration. The other stages of the ICP process appear to have little effect on convergence rate, so we choose the simplest ones, namely random sampling, constant weighting, and a distance threshold for rejecting pairs. Also, because of the potential for overshoot, we avoid extrapolation of transforms.

All of the performance measurements presented so far have been made using a generic ICP implementation that includes all of the variants described in this paper. It is, however, possible to make an optimized implementation of the recommended high-speed algorithm, incorporating only the features of the particular variants used. When this algorithm is applied to the "fractal" testcase of Figure 10, it reaches the correct alignment in approximately 30 milliseconds. This is considerably faster than our baseline algorithm (based on [Pulli 99]), which takes over one second to align the same scene. It is also faster than previous systems that used the constant-time projection strategy for generating correspondences; these used computationally expensive simulated annealing [Blais 95] or Levenberg-Marquardt [Neugebauer 97] algorithms, and were not able to take advantage of the speed of projection-based matching. Figure 17 shows an example of the algorithm on real-world data: two scanned meshes of an elephant figurine were aligned in approximately 30 ms.

This paper is not the first to propose a high-speed ICP algorithm suitable for real-time use. David Simon, in his Ph. D. dissertation [Simon 96], demonstrated a system capable of aligning meshes in 100-300 ms. for 256 point pairs (one-eighth of the number of pairs considered throughout this paper). His system used closest-point matching and a point-to-point error metric, and obtained much of its speed from a closest-point cache that reduced the number of necessary *k-d* tree lookups. As we have seen, however, the point-to-point error metric has substantially slower convergence than the point-to-plane metric we use. As a result, our system appears to converge almost an order of magnitude faster, even allowing for increase in processor speeds. In addition, our system does not require preprocessing to generate a *k-d* tree.

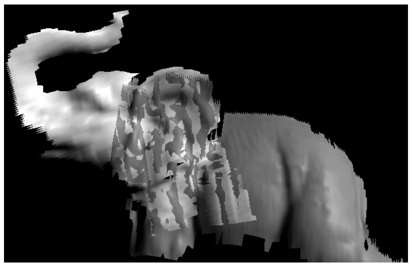


Figure 17: High-speed ICP algorithm applied to scanned data. Two scans of an elephant figurine from a prototype video-rate structured-light range scanner were aligned by the optimized high-speed algorithm in 30 ms. Note the interpenetration of scans, suggesting that a good alignment has been reached.

5 Conclusion

We have classified and compared several ICP variants, focusing on the effect each has on convergence speed. We have introduced a new sampling method that helps convergence for scenes with small, sparse features. Finally, we have presented an optimized ICP algorithm that uses a constant-time variant for finding point pairs, resulting in a method that takes only a few tens of milliseconds to align two meshes.

Because the present comparisons have focused largely on the speed of convergence, we anticipate future surveys that focus on the stability and robustness of ICP variants. In addition, a better analysis of the effects of various kinds of noise and distortion would yield further insights into the best alignment algorithms for real-world, noisy scanned data. Algorithms that switch between variants, depending on the local error landscape and the probable presence of local minima, might also provide increased robustness.

References

[**Arun 87**] Arun, K., Huang, T., and Blostein, S. "Least-Squares Fitting of Two 3-D Point Sets," *Trans. PAMI*, Vol. 9, No. 5, 1987.

[**Benjemaa 97**] Benjemaa, R. and Schmitt, F. "Fast Global Registration of 3D Sampled Surfaces Using a Multi-Z-Buffer Technique," *Proc. 3DIM*, 1997.

[**Bergevin 96**] Bergevin, R., Soucy, M., Gagnon, H., and Laurendeau, D. "Towards a General Multi-View Registration Technique," *Trans. PAMI*, Vol. 18, No. 5, 1996.

[**Besl 92**] Besl, P. and McKay, N. "A Method for Registration of 3-D Shapes," *Trans. PAMI*, Vol. 14, No. 2, 1992.

[**Blais 95**] Blais, G. and Levine, M. "Registering Multiview Range Data to Create 3D Computer Objects," *Trans. PAMI*, Vol. 17, No. 8, 1995.

[**Chen 91**] Chen, Y. and Medioni, G. "Object Modeling by Registration of Multiple Range Images," *Proc. IEEE Conf. on Robotics and Automation*, 1991.

[**Chen 98**] Chen, C., Hung, Y., and Cheng, J. "A Fast Automatic Method for Registration of Partially-Overlapping Range Images," *Proc. ICCV*, 1998.

[**Chen 99**] Chen, C., Hung, Y., and Cheng, J. "RANSAC-Based DARCES: A New Approach to Fast Automatic Registration of Partially Overlapping Range Images," *Trans. PAMI*, Vol. 21, No. 11, 1999.

[**Dorai 97**] Dorai, C., Weng, J., and Jain, A. "Optimal Registration of Object Views Using Range Data," *Trans. PAMI*, Vol. 19, No. 10, 1997.

[**Dorai 98**] Dorai, C., Weng, J., and Jain, A. "Registration and Integration of Multiple Object Views for 3D Model Constrution," *Trans. PAMI*, Vol. 20, No. 1, 1998.

[**Eggert 97**] Eggert, D. W., Lorusso, A., and Fisher, R. B. "Estimating 3-D Rigid Body Transformations: A Comparison of Four Major Algorithms," *MVA*, Vol. 9, No. 5/6, 1997.

[**Faugeras 86**] Faugeras, O. and Hebert, M. "The Representation, Recognition, and Locating of 3-D Objects," *Int. J. Robotic Res.*, Vol. 5, No. 3, 1986.

[**Godin 94**] Godin, G., Rioux, M., and Baribeau, R. "Three-dimensional Registration Using Range and Intensity Information," *Proc. SPIE: Videometrics III*, Vol. 2350, 1994.

[**Hall-Holt 01**] Hall-Holt, O. and Rusinkiewicz, S. "Stripe Boundary Codes for Real-Time Structured-Light Range Scanning of Moving Objects," submitted for publication.

[**Horn 87**] Horn, B. "Closed-Form Solution of Absolute Orientation Using Unit Quaternions," *JOSA A*, Vol. 4, No. 4, 1987.

[**Horn 88**] Horn, B., Hilden, H., and Negahdaripour, S. "Closed-Form Solution of Absolute Orientation Using Orthonormal Matrices," *JOSA A*, Vol. 5, No. 7, 1988.

[**Johnson 97a**] Johnson, A. and Hebert, M. "Surface Registration by Matching Oriented Points," *Proc. 3DIM*, 1997.

[**Johnson 97b**] Johnson, A. and Kang, S. "Registration and Integration of Textured 3-D Data," *Proc. 3DIM*, 1997.

[**Levoy 00**] Levoy, M., Pulli, K., Curless, B., Rusinkiewicz, S., Koller, D., Pereira, L., Ginzton, M., Anderson, S., Davis, J., Ginsberg, J., Shade, J., and Fulk, D. "The Digital Michelangelo Project: 3D Scanning of Large Statues," *Proc. SIGGRAPH*, 2000.

[**Masuda 96**] Masuda, T., Sakaue, K., and Yokoya, N. "Registration and Integration of Multiple Range Images for 3-D Model Construction," *Proc. CVPR*, 1996.

[**Maver 93**] Maver, J. and Bajcsy, R. "Occlusions as a Guide for Planning the Next View," *Trans. PAMI*, Vol. 15, No. 5, 1993.

[**Neugebauer 97**] Neugebauer, P. "Geometrical Cloning of 3D Objects via Simultaneous Registration of Multiple Range Images," *Proc. SMA*, 1997.

[**Pulli 97**] Pulli, K. *Surface Reconstruction and Display from Range and Color Data*, Ph. D. Dissertation, University of Washington, 1997.

[**Pulli 99**] Pulli, K. "Multiview Registration for Large Data Sets," *Proc. 3DIM*, 1999.

[**Simon 96**] *Fast and Accurate Shape-Based Registration*, Ph. D. Dissertation, Carnegie Mellon University, 1996.

[**Stein 92**] Stein, F. and Medioni, G. "Structural Indexing: Efficient 3-D Object Recognition," *Trans. PAMI*, Vol. 14, No. 2, 1992.

[**Stoddart 96**] Stoddart, A. and Hilton, A. "Registration of Multiple Point Sets," *Proc. CVPR*, 1996.

[**Turk 94**] Turk, G. and Levoy, M. "Zippered Polygon Meshes from Range Images," *Proc. SIGGRAPH*, 1994.

[**Walker 91**] Walker, M., Shao, L., and Volz, R. "Estimating 3-D Location Parameters Using Dual Number Quaternions," *CVGIP: Image Understanding*, Vol. 54, No. 3, 1991.

[**Weik 97**] Weik, S. "Registration of 3-D Partial Surface Models Using Luminance and Depth Information," *Proc. 3DIM*, 1997.

Appendix: Scanner Noise and Weighting

During the later stages of ICP, the goal shifts from reducing the error quickly to finding the "correct" transformation as accurately as possible. In order to determine an accurate alignment, it is necessary to take into account the uncertainty in the contribution of each point pair to the error metric. If the weights on point pairs are assigned inversely proportional to the uncertainties, minimizing the weighted error metric will find the transformation that uses the data optimally.

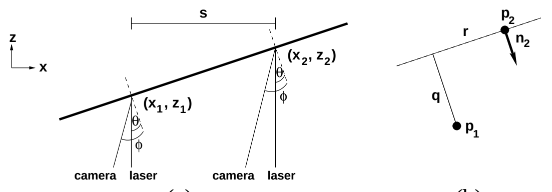
We derive an expression for the uncertainty in point-to-plane distance (see Section 3.5) for the simplified case of a translating laser-plane triangulation scanner. To further simplify the problem, we only consider a single planar surface (Figure 18a).

We begin by considering the width of the laser stripe on the surface of the object. This width varies as

$$W_{surf} = W_0 \sec\theta$$

for some W_0. The width as seen by the camera is then

$$W_{cam} = W_{surf} \cos\phi$$



(a) (b)

Figure 18: (a) Scanner configuration assumed for error analysis. We assume a laser-stripe triangulation scanner with a single camera. The scanner translates a distance s per frame, in a direction perpendicular to the laser plane. The angle between the surface normal and the laser is θ, and the angle between the camera and surface is ϕ. (b) The distance from p_1 to the plane containing p_2 and perpendicular to n_2 is denoted by q.

We now look at the x and z components of the uncertainty in the position of a point on the surface. We assume, as does [Turk 94], that the laser beam has a Gaussian profile, and that the z component of uncertainty is proportional to the uncertainty in finding the peak of the stripe in the camera image; thus, uncertainty in z is proportional to the width of the stripe as seen by the camera. The x component of the uncertainty is a function of scanner calibration, hence is a constant. Thus,

$$\begin{aligned}\Delta z &= a \sec\theta \cos\phi \\ \Delta x &= b\end{aligned}$$

for some constants a and b.

As observed by [Dorai 97], in analyzing scanner errors we must consider not only the uncertainties in position, but also the uncertainty in computing surface normals:

$$\tan\theta = \frac{z_2 - z_1}{x_2 - x_1}$$

Differentiating,

$$\begin{aligned}(\sec^2\theta)\Delta\theta &= \frac{\Delta(z_2 - z_1)}{x_2 - x_1} + \frac{z_2 - z_1}{x_2 - x_1}\frac{\Delta(x_2 - x_1)}{x_2 - x_1} \\ \Delta\theta &= \left(\frac{a\sec\theta\cos\phi}{s} + \frac{b\ \tan\theta}{s}\right)\cos^2\theta \\ &= \frac{a}{s}\cos\theta\cos\phi + \frac{b}{s}\cos\theta\sin\theta\end{aligned}$$

Thus, we see that the uncertainty in surface normals is actually highest when the surface faces the camera and lowest when it is oblique to the camera.

We may now consider the uncertainty in the point-to-plane error (see Figure 18b):

$$\begin{aligned}\Delta q &= r\Delta\theta + \cos\theta(\Delta z_1 + \Delta z_2) + \sin\theta(\Delta x_1 + \Delta x_2) \\ &= \frac{r}{s}\cos\theta\big(a\cos\phi + b\sin\theta\big) + 2a\cos\phi + 2b\sin\theta\end{aligned}$$

This expression is a function of r, which is the point-to-point distance along the normal plane. When the two scans are close together, we expect r to be on the order of $s \cdot \sec\theta$, where s is the spacing in x of range samples. Substituting, we obtain

$$\Delta q = 3a\cos\phi + 3b\sin\theta$$

For most range scanners, the uncertainty along the line of sight (which is proportional to the constant a) will dominate the uncertainty in scanner position (given by b). In this case, the error in point-to-plane distance is just proportional to $\cos\phi$. In summary, the optimal weighting of point pairs for the point-to-plane algorithm is proportional to the secant of ϕ, the angle between the surface normal and the line of sight to the camera.