Multiview Registration for Large Data Sets

Kari Pulli
Stanford University
Stanford, CA, U.S.A
kapu@graphics.stanford.edu

Abstract

In this paper we present a multiview registration method for aligning range data. We first align scans pairwise with each other and use the pairwise alignments as constraints that the multiview step enforces while evenly diffusing the pairwise registration errors. This approach is especially suitable for registering large data sets, since using constraints from pairwise alignments does not require loading the entire data set into memory to perform the alignment. The alignment method is efficient, and it is less likely to get stuck into a local minimum than previous methods, and can be used in conjunction with any pairwise method based on aligning overlapping surface sections.

1. Previous work

The early approaches to registration of range data were based on matching discrete features [9]. Many of the difficulties inherent with feature-based approaches were overcome by the Iterative Closest Point (ICP) approach, developed by Besl and McKay, Chen and Medioni, and Zhang [3, 6, 24]. Assuming a rough initial registration, ICP registers two meshes by pairing points in one mesh with nearby points in the other, finding a rigid 3D motion that better aligns the paired mesh locations, and iterating these steps as long as the registration improves.

The two main difficulties in ICP, determining the extent of overlap in two scans and extending the method for multiple scans, have been a focus of much further research.

1.1. Matching and alignment methods

There are two basic methods for finding matching points (mates) on other meshes for a given mesh: finding the closest point [3, 24], and extending point's normal vector until it intersects the other mesh [6, 17]. Several additional heuristics have been developed to address the problem of not knowing the extent of overlap and avoiding false point matches. Almost everyone uses the simple heuristic of disallowing pairs that are too far apart. Turk and Levoy [22] disallow mates that are on the mesh boundary (usually due to a silhouette) since otherwise many surface points in one view that are not visible in the other would match to boundary points (see Figure 1). Godin *et al.* [10] match points with closest compatible points, where a point is compatible if the value of some associated feature (color, normal vector, etc.) is within a given threshold. It is also possible to modify the distance function to take into account such features [8, 15]. Dorai *et al.* [7] discard point pairs that are not compatible with the neighboring pairs in the sense that the distance between a point and its mate differs from that of the neighboring pairs.

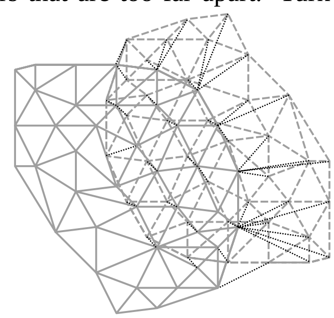

Figure 1: Allowing pairs on mesh boundary causes many spurious pairings.

Another approach for pairing points in a mesh is projecting one mesh onto the other along lines of sight of the scanner. This has typically been used as an acceleration method that avoids the expensive closest point finding [1, 4, 16]. Weik projected individual points onto the other mesh and used intensity and gradient information to find it a better mate nearby [23]. Pulli obtained more consistent point pairs by projecting complete colored meshes against others, performing 2D image alignment, and pairing mesh points ending at the same pixel [18].

There are also two basic approaches for moving views into a better registration: aligning points with their mates (point-point alignment [9, 12]), and aligning points with the tangent planes of their mates (point-plane alignment [6]). Masuda *et al.* [16] used the least median squared estimator to make the point-point alignment method more robust against outliers due to noise or false pairing of non-overlapping surfaces.

1.2. Multiview registration

Chen and Medioni proposed to first register two views, merge the views into a single metaview, and incrementally register and merge the other views into the metaview [6]. This approach was taken also by Masuda *et al.* [16].

Bergevin *et al.* [2] point out a problem in the metaview approach: when more views are added, it is possible that they bring in information that could improve the registration of the previously registered views. Instead, they match points in every view with all the views overlapping with it, and calculate a transformation that registers the first view using all the paired points. This process is iterated over all the views until the registration slowly converges and the registration errors diffuse evenly among all views. Another potential problem with the metaview approach is illustrated in Figure 2, where several scans have already been added to the metaview creating a shell of a finite thickness (Fig. 2(a)). When we register yet another scan with the metaview, ideally the new view would move into the middle of the previous scans (Fig. 2(b)). However, depending on the implementation, it is more likely to stick to the outer or inner shell of the metaview (Fig. 2(c)).

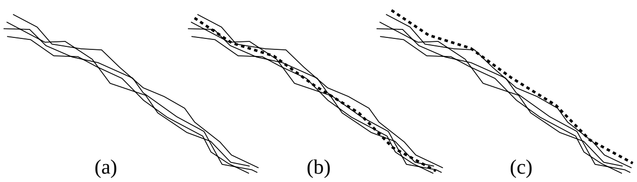

Figure 2. A problem with the metaview approach. (a) Piece of the metaview. (b) Ideal registration for a new scan. (c) Likely new registration.

Jin *et al.*'s approach was to incrementally build a surface model, against which new views can be registered and already registered views can be reregistered [13].

Eggert *et al.* [8] iterate between pairing each point of each scan with exactly one other point and minimizing the total distance between the paired points. This avoids many thresholding issues in point pairing, but can prevent the algorithm from converging to a correct solution. The views may form cliques such that views within a clique are in a good registration, but the cliques themselves are not well registered to each other. If each point is always paired with a point from a mesh within the clique, the inter-clique registration cannot improve. A trivial example of the worst-case behavior is obtained by duplicating each mesh. Another example is shown in Figure 3. Cliques are not likely to form, however, if a point obtains a mate from all the other scans that overlap the same surface location.

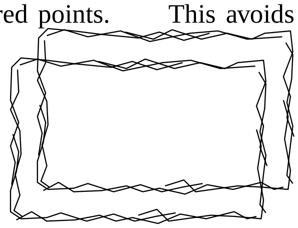

Figure 3: The scans form two cliques such that each scan is well aligned within a clique but not across cliques.

There have been several proposals for the alignment phase of multiview registration. Bergevin *et al.* [2] calculate a transformation for each view separately and then apply them simultaneously before the next round of matchings. Benjemaa and Schmitt [1] accelerate the method by applying the new transformation as soon it is calculated. A quite different approach was taken by Stoddart and Hilton [20] and Eggert *et al.* [8]. They both solve the transformations simultaneously by attaching imaginary springs to the paired points and simulating a simplified version of the resulting dynamic system.

Multiview registration has also been attempted with more general purpose optimization methods such as simulated annealing, but the results do not look very promising in terms of speed or accuracy [4].

2. Pairwise registration

We use a fairly standard ICP derivative to generate the pairwise registrations that our multiview method uses. For the sake of completeness we describe in this section our pairwise registration method. We would like to stress, though, that our multiview method does not rely on using this particular pairwise alignment technique.

2.1. Initial alignment

Most registration methods assume an initial approximate transformation and concentrate on improving it. There have been many proposals for search-based methods that can provide a good starting point for a more fine-grained registration [5, 9, 11, 14, 19]. Unfortunately, such methods are difficult to get to work reliably for arbitrary scan data.

The scanning process can be instrumented by tracking how the scanner (or the object) is moved. If the instrumentation is so accurate that no software alignment is necessary, it can be considered part of the scanner. Otherwise, it can serve as an excellent starting point for registration.

The final alternative is interactive initial alignment. In our work, we have used a mix of instrumentation and interactive alignment.

2.2. Matching heuristics

We follow the ICP tradition of matching points with closest points from other scans, with several additional constraints. First, we have adopted the closest compatible point idea [10], allowing points to match only if their associated normal vectors differ by less than 45 degrees. Another important constraint we use is disallowing a point pair if the closest point on the other scan is on a mesh boundary [22]. Finally, we have two thresholds for the maximum allowed distance between paired points, one dynamic and the other hard. The dynamic threshold is expressed as a percentage: keep the p% closest point pairs of those that survived the first two tests. This threshold effectively removes many spurious point pairs in the early iterations of the pairwise alignment. In absence of other heuristics, the percentage p should reflect the portion of the generated point pairs that are on the overlap of the two meshes (this was the approach taken by Masuda *et al.* [16]). In our case, disallowing point pairs where the closest point is on the mesh boundary already removes most of the spurious matches and p can be in practise quite high, e.g., 90%. The second, hard threshold, disallows a point pair if it is too long. Initially, the threshold should be high enough to allow point pair generation, but should be reduced later to be a small multiple of the scanning accuracy. Conversely, when the hard threshold is reduced, the dynamic threshold can be increased.

In summary, our point matching method is as follows: for a set of points in the first mesh, find the closest point within the hard threshold and with a compatible normal vector in the second mesh. If such a point was found and that point is not on the mesh boundary, create a point pair. Make matching symmetric by matching also points from the second mesh with the first mesh. Finally, keep the p% closest pairs. As the registration gets better, the hard threshold is reduced and the dynamic threshold is increased.

2.3. Alignment methods

We have implemented and experimented with the two major alignment methods. The point-point method [12] produces a closed form least-squares solution for finding a rigid 3D transformation that simultaneously aligns all points in a set with their mates. In a sense, an ideal spring is attached between each point and its mate, and the springs are allowed to pull the sets into better alignment.

The point-plane [6] method incrementally improves the alignment by finding a small rotation and translation that moves points closer to the tangent planes of their mates. In this case one end of a spring is attached to a point, while the other end is free to slide along a plane. Given a fixed pairing of points with tangent planes, the alignment can be improved by iterating a few times.

For pairwise alignment of range scans we strongly prefer the point-plane method over the point-point approach. First, it usually converges an order of magnitude faster. Once the surfaces get fairly close to each other, most points on or near the overlap area can find a nearby compatible mate, even though accurate registration still requires one of the surfaces to slide along the other to a good alignment. In point-point approach (Figure 4(a)), even if several of the

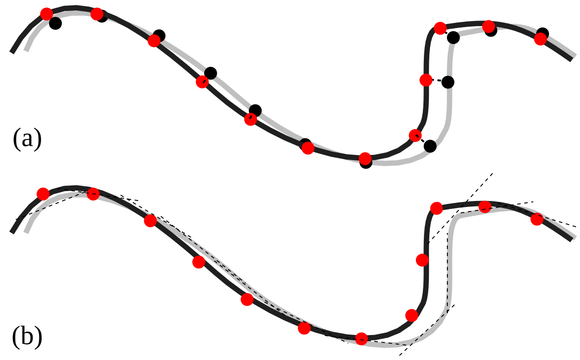

Figure 4. (a) Point-point: points are mapped to discrete points. (b) Point-plane: points are mapped to continuous tangent planes.

points are paired with their ideal mates (a point that corresponds to the same surface location as the first point), every spurious point pair where the points are close to each other resists further motion, preventing the surfaces from moving more than a very small step. Unfortunately, even with a slight misregistration, only a few points are actually matched with their ideal mates. In contrast, the false point pairs do not slow the point-plane method down much, as each point may slide along the tangent plane of its mate (Figure 4(b)). Additionally, the method is much more resistant to wrong point pairs: pairing a point with any point whose tangent plane coincides with that of the ideal mate works as well as using the ideal mate. Another way to think this is that in the point-point approach each point is associated with a *discrete* pointwise approximation of the other surface, while in the point-plane method each point is associated with a *continuous* linear surface approximation. We have observed that the point-point method typically requires at least 10 times as many point matching – alignment iterations as the point-plane method.

The second reason for preferring the point-plane approach is that more accurate results can be obtained with a fast implementation. Finding the closest point in a cloud of points is much faster than finding the closest point on a surface (e.g., a triangle mesh), but unless the closest surface point is found, the point-point method is susceptible to aliasing artifacts due to finite and nonuniform sampling density. The point-plane method, however, is much less affected by aliasing, as a collection of sampled points is aligned with a first order approximation of a continuous surface, i.e., a collection of tangent planes.

However, we cannot unequivocally claim that point-plane method is always superior to point-point method. In situations where a correct registration is inherently difficult because the region of overlap is almost flat or has almost uniform curvature (such as scanning a sphere, torus, or cylinder), the point-plane is more brittle in the sense that the meshes can shoot widely apart starting from a slight misregistration. Other constraints, such as matching color, are more easily dealt with the point-point method, but are not impossible to address with the point-plane approach. Finally, though we prefer point-plane with pairwise alignment, in the following multiview alignment phase we prefer the point-point approach; more of this in the next section.

3. Multiview registration

The basic iterative matching-aligning approach to multiview registration has several problems that motivated our approach.

Computational expense. In the iterative matching–aligning cycle the matching part requires the most computation. Once the matching effort has been expended, one should exploit it as much as possible to bring the views to the best possible alignment. Unfortunately, given some initial registration error, most if not all the points are paired with points that do not correspond to the same surface location, and only modest improvements are possible before the expensive matching part needs to be repeated, eventually dozens of times. Ideally, we would have a really good matching to start with and only one multiview alignment step would be required.

Large memory requirements. If we were to iterate matching and aligning of *n* meshes, we would need to keep complete data of all views (plus additional data structures for, e.g., closest point finding) in memory at the same time. Such approach does not scale to very large data sets.

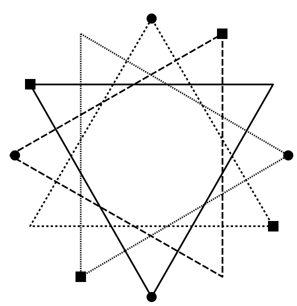

Figure 5. A local minimum configuration for 4 triangles.

Local minima. The iterative matching-aligning cycle is susceptible to getting stuck into a local minimum. Even if we had correct point pairs but incorrect starting positions, we could create situations where each mesh is in a local minimum with respect to their neighboring meshes. Figure 5 shows a simple example of a local minimum starting configuration are four meshes, each consisting of a single triangle and the matching vertices have a matching symbol (dot, square, empty). Every triangle is in a local minimum with respect to its neighbors. We performed some tests on this configuration to find out how stable it is. Each triangle was perturbed by a rotation (uniformly random within $\pm 10^\circ$) around the origin. The matching part was omitted, i.e., each mesh was successively aligned with respect to the other meshes using the correct vertex pairing. If we first calculated the transformations and then simultaneously applied them to each mesh, typically even 1000 iterations was not enough to converge to the correct solutions. When the transformation was applied to a mesh immediately after it was calculated, the process typically required over 20 full iterations to converge.

However, typically the correct point matching is not known beforehand, and such local minima are more likely to occur. Alternatively, the configuration may not lead to an actual local minimum, yet it may radically slow the convergence of the iteration. Additionally, as pointed out in Section 1, depending on the matching order and heuristics there may be a danger of forming cliques of views with good intra-clique registration but poor inter-clique registration.

Our solution. We first perform a pairwise registration between every view and each of its neighboring (overlapping) view once. Though in our implementation of pairwise registration we do iterate matching and aligning, the pairwise registrations converge much faster than the multiview case and provide us with pairwise constraints between views that we later enforce to register all the views simultaneously. The constraints take much less space than the original data and allows registering data sets that are too large to fit in the memory at the same time. Finally, we incrementally enforce these constraints to obtain a global multiview registration using a method that does not depend on the initial registration transformations. The method is less likely to get stuck into a local minimum and it efficiently diffuses the alignment error among the meshes.

Next we explain the constraints obtained from pairwise alignment, followed by a description of our incremental alignment algorithm.

3.1. Constraints

The pairwise registrations provide us with a set of constraints that we should simultaneously satisfy. There are several possible alternatives for the exact formulation of these constraints.

Enforce relative registration transformations. We could try to enforce the pairwise registration constraints by assigning each view a transformation so that the resulting relative transformations between neighboring views would differ as little as possible from the ones obtained through pairwise registrations. However, it is not clear how one should measure the distance between two rigid 3D transformations. For example, the effect of a small rotation can become very large on the overlap area, depending on the center of rotation. Even if a meaningful metric only involving the transformation could be defined, it may not reflect the deviations from relative alignments well.

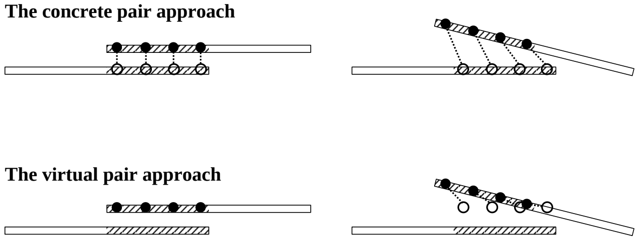

Figure 6. Visualizing concrete pairing (top) and virtual pairing (bottom). The multiview alignment error is the sum of squared lengths of the dotted lines before (left) and after (right) a perturbation.

Align point pairs generated in pairwise registrations. A very straightforward generalization of ICP could just store the point pairs generated from the pairwise registrations, fixing the matching subproblem for once and all, while the only remaining problem would be that of alignment. The alignment could be then solved by minimizing the sum of squared distances of all the point pairs. This *concrete mate* approach (see top row of Figure 6) is a very attractive solution that addresses all the main problems that we enumerated in the beginning of this section. However, it requires that the pairwise registrations are obtained using the same optimization method as the one that is used for multiview alignment. For example, if we first obtained point pairs using point-plane alignment, or performed the pairwise alignment using some non-ICP-based method, using point-point alignment with the same point pairs would yield different relative registrations. A more severe problem is that a least-squares solution for the alignment problem would bias the answer in favor of the worst pairwise registrations, since a small random misregistration is likely to increase the sum of *squared* distances more for the longer point pairs than for the shorter ones. [1]

Enforce the relative alignments of overlapping surface sections. What we really should do is to constrain multiview registration to move each scan relative to its neighbors as little as possible, regardless how the pairwise registrations were obtained. More precisely, if we have registered (and identified) the overlap between two scans, we want to keep the overlapping regions well aligned, whereas other parts of the scan may shift without a penalty (of course while remaining rigidly connected with the overlapping areas). We do this using a generalization of the concrete mate approach that we call the *virtual mate* approach. This approach requires the pairwise registrations to produce a relative transformation between the scans along with a set of points that uniformly subsample the overlapping areas. The relative transformation tells where each subsampled point of one scan should appear in its neighbor's coordinate system, thus creating for the point a virtual mate (see the bottom row of Figure 6). Note that both scans do not need to be subsampled, in principle one would suffice.

The virtual mate approach has two major advantages over the concrete mate approach.

- It provides a clean interface between the results of pairwise registrations and reaching a consensus multiview solution that attempts to maintain the pairwise alignments. In particular, the results of an arbitrary pairwise method can be used, regardless which minimization metric was used, as long as the method is based on aligning overlapping regions of the scans and it is possible to approximate (and subsample) the extent of that overlap.
- The pairwise registrations can be simultaneously enforced using a linear least-squares solution without introducing a bias favoring registrations that have longer point pair distances due to more noise, lower surface sampling density, or inferior registration. The multiview registration metric becomes the sum of squared motions of the subsampled points from their ideal relative positions with respect to the neighboring views. All of the error in the virtual mate approach is entirely due to twisting the pairwise alignments, while in the concrete mate approach the error would additionally contain the pairwise registration error.

3.2. Alignment

In Section 2.3 we said that although we prefer the point-plane method for pairwise registration, in the multiview case we use the point-point method. In pairwise registrations, when most of the point pairs are initially suspect, the ability of the point-plane method to slide along the surface is an asset. However, now we know where each point should appear (with respect to the neighboring view) and we want to minimize all motion, including tangential one.

The order in which the constraints are enforced is not irrelevant. Simultaneously enforcing all the pairwise constraints can lead to a local minimum, as illustrated in Figure 5, or it can make the error landscape locally flat, preventing fast convergence into the global minimum. To avoid these problems we perform the multiview alignment incrementally, reminiscent of the metaview approach of multiview registration (see Section 1.2): we add views into a set

[1] There would not be such bias if we summed the absolute rather than squared distances of the pairs, or if we interpreted the pairs to be ideal springs whose rest lengths are the initial pairwise distances instead of zero. However, we could not solve such a system as a linear least-squares problem.

of consistently aligned views one at the time while keeping the views in that set consistently aligned.

More concretely, we begin the multiview alignment by choosing the view that has the most connections, put that into the active set, and all the other views into the dormant set. We then add the views in the dormant set to the active set one at the time. At each turn, the dormant view with most links into the active set is chosen, and a queue of still moving views is initialized with the current view. This queue is processed until it becomes empty by removing a view from it, aligning it with its neighbors that are in the active set, and merging those neighbors into the queue if the registration error is reduced enough. The registration errors get diffused evenly among the view pairs in this inner loop. Figure 7 gives the pseudocode for this algorithm.

```
dormant_set = views
curr := most_links(dormant_set, dormant_set)
active_set.add(curr)
dormant_set.remove(curr)
WHILE NOT EMPTY dormant_set
    curr := most_links(dormant_set, active_set)
    active_set.add(curr)
    dormant_set.remove(curr)
    queue.push(curr)
    WHILE NOT EMPTY queue
        curr   := queue.pop()
        nbors  := active_set.neighbors(curr)
        relative_change := align(curr, nbors)
        IF relative_change > tolerance
            queue.merge(nbors)
```

Figure 7. Pseudocode for the multiview alignment algorithm.

A view is aligned with a set of neighbors using the constraints obtained from pairwise registration (`align(curr, nbors)` in Figure 7). The constraints take the form of two sets of points sampled from the area of overlap, one set for each scan, and the relative registration transformation. Two lists of 3D points are created: in one list we put the points of the current scan, in the other their ideal mates. The ideal mates are calculated as follows. Let us use $\mathbf{M}_i$ to denote the current registration transformation of view i and $\mathbf{M}_{i,j}$ the relative transformation that takes a point from the local coordinates of view i to the coordinates of view j. The points $\mathbf{p}_k$ of view i from the overlap area with view j are added to the first list as $\mathbf{M}_i\mathbf{p}_k$, to the second list as $\mathbf{M}_j\mathbf{M}_{i,j}\mathbf{p}_k$. Finally, we align the point lists using the point-point method.

The important feature of this algorithm is that it adds views one at a time to a set of views in order to avoid getting stuck to a local minimum due to unfavorable initial configuration. In practise, the alignment is likely to converge faster if we first run the algorithm of Figure 7 with a larger tolerance. Once all the scans are in the active set, a new relaxation process can be started with the desired tolerance threshold. The relaxation of scans could be ordered for example by how much a neighbor of a scan improved its registration at its turn.

4. Results

In order to give some impression of the execution times on our system we tested our system with a data set that has been used in several registration papers before. The data consists of eight views of 256×256 range samples and is shown in Figure 8. An informal test, timed with a stop watch on an interactive run on an SGI O2, starting with identity transformations for each view, gave the following times. Performing eight interactive pairwise alignments and running for each pair our version of ICP, including building the kD-trees for finding the closest point, took altogether 4 minutes. Performing multiview registration took 5 seconds. After that we created a collection of new pairings for overlapping view pairs other than the one interactively created, this took 10 seconds. After running another global registration (5-10 seconds) we studied the registration statistics and interactively improved some of the automatic alignments. Within 10 minutes from the start the alignment was as good as we were able to get it.

Figure 8: The NRC toy soldier.

We have also used our method to register our own scans. Figure 9 shows a heavily subsampled image of the raw registered scans of the David of Michelangelo. Our data set for David, which is 517cm tall, consists of over two billion triangles in almost 500 separate scans, each covering a surface area from a few dozens of square centimeters to over a square meter, at intersample distances of about a third of a millimeter. Doing anything with data sets of this size, even just displaying it, is a severe challenge. Getting the user assisted initial and supervised pairwise registrations is a very laborious process, however, the multiview part using the constraints from pairwise registrations can be run in a few minutes on an SGI Octane. Figure 10 shows a close up of the reconstructed head.

5. Discussion

5.1. Failure modes

There are two principal failure modes for our multiview registration algorithm. First, it may be possible that no set

Figure 9. The David of Michelangelo.

Figure 10. The reconstructed head.

of rigid transformations exists that would simultaneously aling all the scans. The object may have deformed during the scan, either the scanner or the object may have moved during a scan, or the calibration of the sensor is not perfect. A non-rigid registration may sometimes be the only solution (see, e.g. [21]). Otherwise, one can break large scans into partially overlapping smaller scans, thus adding flexibility that may allow data to align better.

The second failure mode is particular to our method. We assume that the pairwise alignments are nearly perfect, and we try to preserve them as well as is possible. If the overlap region between two scans is flat or has nearly constant curvature, it is hard to reliably obtain a good pairwise registration, whereas other methods that reregister a scan with all of its neighbors could overcome a bad initial registration. On the other hand, our approach provides some diagnostic tools that allow us locate false initial registrations and have much more control to avoid accidental sliding across relatively featureless regions.

5.2. Diagnostic tools

There are few diagnostic tools for assessing the quality of registration for traditional multiview registration methods. One can, of course, visually inspect the alignment and perhaps identify obviously misaligned scans. One can also study the overall distance from a scan to other nearby scans, but this distance is not always a good indication of the quality of registration, as scans may drift across relatively featureless regions with little penalty. Yet if a clear misalignment is detected, very little can be done to rectify the problem except move the offending scan, restart the process, and hope it does not shift back where it was previously found.

In our system the multiview registration metric explicitly tracks how well a scan can be simultaneously aligned with all its neighbors. If a scan had been correctly pairwise aligned with every neighbor, and assuming the range data was not warped, it is likely to be able to enforce all the pairwise registrations simultaneously reasonably well. If, however, a scan moved away from its pairwise registration with a neighbor, this is an indication of a pairwise misalignment. In such case we can, for example, first eliminate the pairwise constraint, perform the multiview alignment with all the other constraints, and redo the pairwise registration using the resulting relative pose between the two scans as the starting point. Alternatively, if we *know* that the pairwise registration was very good, yet the multiview phase attempts to move the scans apart, we can assign a higher weight to this particular constraint, either explicitly or by storing more subsampled points from the region of overlap.

5.3. Workflow

When dealing with data sets consisting of hundreds of scans, it is not practical to perform supervised pairwise registrations aligning each scan with other partially overlapping scans. If the scans are already in coarse initial alignment, we create the first pairwise registrations by attempting to automatically align each pair of scans whose bounding boxes intersect. Otherwise we process the scans one at a time by first interactively aligning a scan with one of the previous scans and then automatically creating more pairs.

After the initial pair creation, we perform multiview alignment. We then check the pairs that do not retain their pairwise alignments well. Typically this is due to an incorrect pairwise registration, which is then interactively corrected. Sometimes even a good pairwise alignment cannot be enforced due to mistakes elsewhere, in which case we can increase the relative weight of that pair. This is likely to make the real culprit more obvious by increasing its registration error.

6. Conclusions

In this paper we have presented a new framework for multiview registration. The previous multiview approaches that spread the remaining registration error among all views are direct generalizations of the ICP match–align strategy, and require the expensive matching part repeated hundreds of times while keeping the scan data and supplementary data structures in memory. We perform pairwise registrations once and store from those constraints that are satisfied in the multiview alignment part as well as possible. This yields not only time savings, we can also perform a global registration for data sets that are too large to keep in memory. The method is general in the sense that any pairwise registration method can be used, as long as the relative registration calculation and a subsampling of the overlap can be obtained. The multiview alignment method is independent of the initial registration transformations and is thus less likely to get stuck into a local minimum. We have tested our method with a collection of very large real data sets.

Acknowledgments

Many of the ideas in this paper were developed while the author was at University of Washington and worked with Linda Shapiro, Tom Duchamp, and Werner Stuetzle. The ideas were further developed during the Digital Michelangelo project and benefited from discussions with Sean Anderson, Brian Curless, James Davis, David Koller, Marc Levoy, Lucas Pereira, Domi Piturro, Szymon Rusinkiewicz and others, especially Matt Ginzton who helped a lot in coding. The Digital Michelangelo project was made possible by the financial support of Interval Research, Paul Allen Foundation for Arts, and Stanford University, and the support of Drs. Paolo Liverani and Franca Falletti of Florence, Italy.

References

[1] R. Benjemaa and F. Schmitt. Fast global registration of 3d sampled surfaces using a multi-z-buffer technique. In *Proc. Int. Conf. on Recent Advances in 3-D Digital Imaging and Modeling*, pages 113–120, May 1997.

[2] R. Bergevin, M. Soucy, H. Gagnon, and D. Laurendeau. Towards a general multi-view registration technique. *IEEE Trans. Patt. Anal. Machine Intell.*, 18(5):540–547, May 1996.

[3] P. J. Besl and N. D. McKay. A method for registration of 3-d shapes. *IEEE Trans. Patt. Anal. Machine Intell.*, 14(2):239–256, February 1992.

[4] G. Blais and M. D. Levine. Registering multiview range data to create 3d computer objects. *IEEE Trans. Patt. Anal. Machine Intell.*, 17(8):820–824, August 1995.

[5] C-S Chen, Y-P Hung, and J-B Chung. A fast automatic method for registration of partially-overlapping range images. In *ICCV 98*, pages 242–248. IEEE, January 1998.

[6] Y. Chen and G. Medioni. Object modelling by registration of multiple range images. *Image and Vision Computing*, 10(3):145–155, April 1992.

[7] C. Dorai, G. Wang, A. K. Jain, and C. Mercer. From images to models: Automatic 3D object model construction from multiple views. In *Proceedings of the 13th IAPR International Conference on Pattern Recognition*, pages 770–774, 1996.

[8] D. W. Eggert, A. W. Fitzgibbon, and R. B. Fisher. Simultaneous registration of multiple range views for use in reverse engineering. Technical Report 804, Dept. of Artificial Intelligence, University of Edinburgh, 1996.

[9] O. D. Faugeras and M. Hebert. The representation, recognition, and locating of 3-d objects. *International Journal of Robotic Research*, 5(3):27–52, Fall 1986.

[10] G. Godin, M. Rioux, and R. Baribeau. Three-dimensional registration using range and intensity information. In *Proc. SPIE vol. 2350: Videometrics III*, pages 279–290, 1994.

[11] M. Hebert, K. Ikeuchi, and H. Delingette. A spherical representation for recognition of free-form surfaces. *IEEE Trans. Patt. Anal. Machine Intell.*, 17(7):681–690, July 1995.

[12] B. K. P. Horn. Closed-form solution of absolute orientation using unit quaternions. *Journal of the Optical Society of America A*, 4(4):629–642, April 1987.

[13] H. Jin, T. Duchamp, H. Hoppe, J. A. McDonald, K. Pulli, and W. Stuetzle. Surface reconstruction from misregistered data. In *Proc. SPIE vol. 2573: Vision Geometry IV*, pages 324–328, 1995.

[14] A. Johnson and M. Hebert. Surface registration by matching oriented points. In *Proc. Int. Conf. on Recent Advances in 3-D Digital Imaging and Modeling*, pages 121–128, May 1997.

[15] A. Johnson and S. Kang. Registration and integration of textured 3-d data. In *Proc. Int. Conf. on Recent Advances in 3-D Digital Imaging and Modeling*, pages 234–241, May 1997.

[16] T. Masuda, K. Sakaue, and N. Yokoya. Registration and integration of multiple range images for 3-D model constuction. In *Proc. IEEE Conf. on Computer Vision and Pattern Recognition*, pages 879–883, June 1996.

[17] M. Potmesil. Generating models for solid objects by matching 3d surface segments. In *Proc. Int. Joint Conf. on Artificial Intelligence*, pages 1089–1093, August 1983.

[18] K. Pulli. *Surface reconstruction and display from range and color data*. PhD thesis, Dept. of Computer Science and Engineering, Univ. of Washington, December 1997.

[19] F. Stein and G. Medioni. Toss-a system for efficient three dimensional object recognition. In *Proc. DARPA Image Understanding Workshop*, 1990.

[20] A. J. Stoddart and A. Hilton. Registration of multiple point sets. In *Proceedings of the 13th IAPR International Conference on Pattern Recognition*, pages 40–44, 1996.

[21] R. Szeliski and S. Lavallée. Matching 3-d anatomical surfaces with non-rigid deformations using octree-splines. In *Proc. IEEE Workshop on Biomedical Image Analysis*, pages 144–153, 1994.

[22] G. Turk and M. Levoy. Zippered polygon meshes from range images. In *Proceedings of SIGGRAPH '94*, pages 311–318, July 1994.

[23] S. Weik. Registration of 3-d partial surface models using luminance and depth information. In *Proc. Int. Conf. on Recent Advances in 3-D Digital Imaging and Modeling*, pages 93–100, May 1997.

[24] Z. Zhang. Iterative point matching for registration of free-form curves and surfaces. *International Journal of Computer Vision*, 13(2):119–152, 1994.