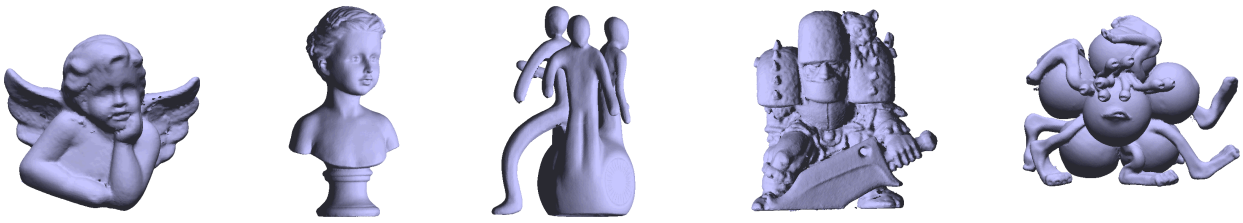
Automatic multiview quadruple alignment of unordered range scans

Frank B. ter Haar and Remco C. Veltkamp
Department of Information and Computing Sciences
Utrecht University, the Netherlands



C2 angel (16/16) C2 buste (16/16) C3 memento (32/36) C3 warrior (24/24) C2 bozbezbozzel (44/44)

Abstract

This paper presents a new multiview alignment algorithm that performs both the coarse and fine alignment of unordered sets of range scans. Our algorithm selects quadruples of range scans, which have feature points of their 2D projections in common. These quadruples are then verified using an Iterative Closest Point (ICP) algorithm. The accepted quadruples form incomplete models of an object that can be aligned using isometries of the Principal Component Analysis (PCA) in combination with an ICP algorithm. The output of our method is a set of finely aligned meshes. Our method was applied to range scans of different clusters of objects varying in the type of acquisition system, the number of range scans, the scan resolution, and scan accuracy. Results show that our method is both effective and efficient for the alignment of meshes: it is capable of aligning object meshes with various properties, aligns the majority of meshes without a priori knowledge, and doesn't require a multiview ICP algorithm to improve the final alignment.

Keywords: Range scanning, Registration, Alignment, Merging

1. Introduction

To construct a 3D model out of a real world object, the object's surface can be digitized with the use of a laser range scanner. Such a scanner measures the distance from the scanner to the object. The most common laser range scanners can only scan one side of an object at a time, and only those parts of the object's surface that are not occluded and are within the scanner's range. To capture more data of the object's surface, the object has to be scanned from many different sides and for several poses. Because each scan is generated with respect to the scanner's coordinate system, they need to be transformed into a common coordinate system in which they are aligned to each other. A common way to obtain such an alignment of range scans is to interactively align pairs of overlapping meshes. This results in a coarse alignment which is usually refined by the application of a multiview ICP algorithm [2, 16, 17, 20]. Finally, the 3D model can be constructed by merging the aligned meshes into one single surface mesh [7, 18, 22].

An *interactive pairwise* alignment of meshes is performed by a user, who decides which pairs of meshes have parts of their surface in common. He or she then either selects a few corresponding points on the common surface of two meshes, or manually rotates and translates one mesh towards the other, to bring them into alignment.

An *automatic pairwise* approach will try to find a set of corresponding points on two meshes automatically. When enough correspondences are found, the two meshes are brought into alignment. Several techniques have been developed to perform the pairwise alignment of meshes automatically, including: the exhaustive search for corresponding points [3, 4] and the use of surface signatures such as spin-images [12], point signatures [5], bitangent curves [23], spherical attribute images [9], and 3D tensors [15].

A problem with the automatic pairwise approach occurs when incorrect correspondences are selected to align meshes, with an unsuccessful alignment as a result. An *automatic multiview* approach will try to solve this with the use of a global consistency check for pairs of meshes with

high correspondence. Methods to perform the alignment according to the multiview approach include the work of Huber et al. [10] and Mian et al. [14]. Huber describes a framework in which spin-images are applied on all possible pairs of meshes followed by the construction of a minimal spanning graph. This spanning graph determines which pairs of meshes should be aligned to construct a globally correct alignment. Mian extracts 3D tensors (grid representations of local surface) from all meshes and constructs a correspondence tree with the mesh having the most tensors as root. The remaining meshes may only be added to the leafs of the tree, and only when the mesh has enough corresponding tensors and passes global verification.

For a set of N meshes, the approaches of Huber and Mian involve a number of N^2 and $O(N^2)$ mesh-to-mesh comparisons respectively. More importantly they are both using large sets of complex features (i.e. spin-images and 3D tensors). Afterwards, these methods apply a multiview ICP algorithm to refine their coarse alignment of meshes, which has a worst case complexity of $O(N^2)$ as well.

Contribution

In this work we propose a new method to align unordered sets of meshes. The main idea of our method is to select small groups of meshes, which can be aligned using Principal Component Analysis (PCA) in combination with an Iterative Closest Point (ICP) algorithm. Our method selects groups of four meshes (quadruples) that represent the front, right, back, and left views of an object's pose. To select the optimal quadruples, sets of 2D silhouette features are extracted from all meshes, and matched.

This new method has a number of advantages. Firstly, we are able to align mesh sets of objects scanned using different acquisition systems, objects with few and many protrusions, objects with smooth and rough surfaces, and mesh sets with little and much noise. Secondly, our method is able to automatically align all meshes from 12 out of 25 datasets and 80% of the total amount of meshes, without a priori knowledge. Thirdly, our approach applies $O(N^2)$ mesh-to-mesh comparisons using low cost operations only. The rest of the algorithm, which is the most time consuming part, requires $O(N)$ mesh-to-mesh comparisons and obtains a high quality alignment similar to an $O(N^2)$ multiview ICP algorithm in less time.

2. Method

2.1. Overview

This paper describes an efficient way to align a set of 3D meshes $\mathbf{M_{i\in(1\ldots N)}}$. The approach is to find quadruples of meshes first, that represent the object's front, right, back, and left side of a certain pose. Therefore, we center all meshes and extract features from the boundary of projection silhouettes (Section 2.2) and match them to the features of other meshes (Section 2.3). In Section 2.4 we describe how quadruples are selected and how the four meshes are coarsely aligned within such quadruples. The coarse alignment of meshes in a quadruple is refined and the quadruple itself is verified (Section 2.5). During the verification of a quadruple, the quadruple is rejected or accepted, or a subgroup of three of its meshes is accepted. The accepted groups of either three of four meshes are aligned in Section 2.6 which returns the final alignment of meshes.

Before starting the feature extraction, we resample the meshes for efficiency reasons. The meshes $\mathbf{M_i}$ are resampled towards both fine meshes $\mathbf{FM_i}$ and coarse meshes $\mathbf{CM_i}$. See Section 2.7 for implementation details.

2.2. Feature extraction

For each of the fine meshes $\mathbf{FM_i}$, a set of features are extracted from two of its projections: one for the mesh mapped to the XY-plane ($\mathbf{FM_i^{xy}}$), and one for the mesh mapped to the YZ-plane ($\mathbf{FM_i^{yz}}$). We use no projection of a mesh onto the XZ-plane. This projection could be used to find a top or bottom view. However, to scan a top or bottom view the object has to be toppled over towards a new stable pose, which should result in a new group of different side views. So, we use only the XY- and YZ-projections. For mesh $\mathbf{FM_i^{xy}}$ we extract the vertices on the boundary that two opposite meshes (i.e. opposite model views) most likely have in common, which are the right-, top-, left-, and bottommost set of silhouette vertices of the projected mesh (see Figure 1). We refer to this set of vertices as the XY-features. For mesh $\mathbf{FM_i^{yz}}$ only the frontmost silhouette vertices are selected as its YZ-features, which may correspond to a subset of the XY-features as described in the next Section.

2.3. Feature matching

For each fine mesh $\mathbf{FM_i}$, we try to find a left, right and back side by comparing the mesh to all other meshes using the extracted sets of features. Mesh $\mathbf{FM_i}$ is selected as the front mesh of a newly created quadruple and all other meshes $\mathbf{FM_j}$ are compared as being a potential right, back, or left side of this quadruple. For a back to front comparison the mirrored set of XY-features of the potential back side is compared to the XY-features of the front. For a side to front comparison the YZ-features are matched to the front's XY-features. The distance measure used in these comparisons is the Minimum Least Squares (MLS) distance after i_{max} iterations of the point-to-point ICP algorithm (see Algorithm 1 applied to two point sets). When a side to front comparison is performed on an actual side-front pair, the side's

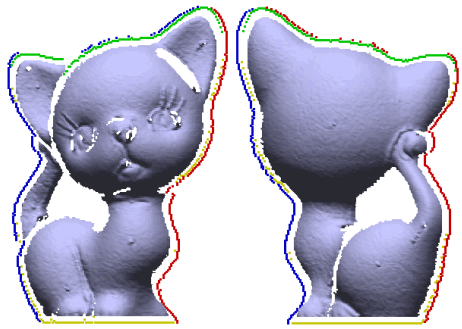


Figure 1. The extracted features (in color) from two opposite meshes shown with an offset. Vertices from holes and cavities are not extracted, which makes the two sets of features comparable.

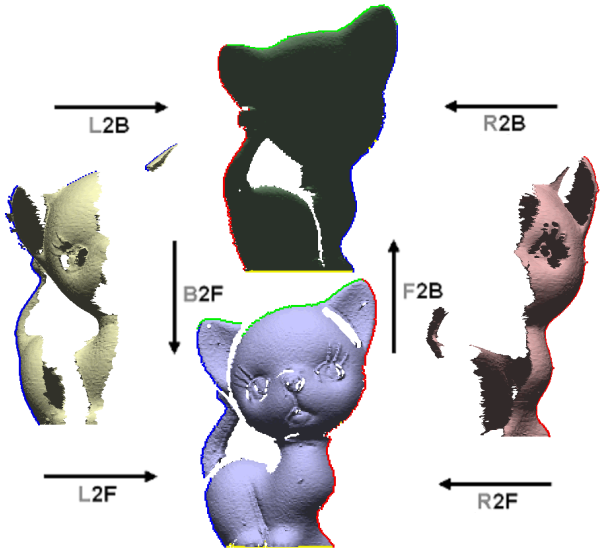


Figure 2. Features (in color) from the left, right and back view are matched to the features of the front view. Note that the features from the left view correspond to the leftmost features of the front view.

YZ-features form a subset of the front's XY-features, that is, when we place the feature sets in one plane. Moreover, these features correspond either to the leftmost XY-features in case of a left side, or to the rightmost XY-features in case of a right side (see Figure 2).

Because the point-to-point ICP algorithm requires an initial alignment of point sets, the centers of mass of either the entire sets of features or only the specific subsets (in case of a side view) are aligned. For a perfect match the MLS distance after the point-to-point ICP algorithm is zero. For the less perfect matches, the small number of ICP iterations allow little correction. Because each mesh is compared to all other meshes as its potential right, back, and left side, this process is quadratic in the number of meshes. However, we have used only low cost operations, that is, the ICP algorithm is applied to silhouette points only.

Algorithm 1 point-to-point ICP (PS_1, PS_2)

1: $dist_{min} \leftarrow \infty$
2: **for** $i \leftarrow 1$ **to** i_{max} **if** no convergence **do**
3: $pairs \leftarrow$ for all points in PS_1 find closest point in PS_2
4: $dist \leftarrow$ minimize the sum of squared distances between the points in $pairs$
5: **update** $dist_{min}$
6: **end for**
7: **return** $dist_{min}$ **and** 2D rigid transformation

2.4. Quadruple selection

For each fine mesh $\mathbf{FM_i}$ a quadruple is constructed of those four meshes that have the minimal total sum of MLS distances of the following twelve feature set comparisons: front to back, left to right, right to front, back to right, left to back, front to left, and vice versa (see Figure 3). As a result, *N* quadruples are selected. Since the same quadruple of four meshes may be selected up to four times, we retain only the unique $N' \leq N$ quadruples.

During the matching of feature sets we obtain not only a distance value for two sets of features, but a 2D rigid transformation to align the feature sets as well. Each 2D transformation consists of a translation and a rotation in the plane to which the features were mapped. The twelve rigid transformations within a quadruple are employed to obtain a coarse alignment of the quadruple's meshes.

2.5. Quadruple verification

In the previous section our method selects quadruples of four meshes that represent the object's front, right, back, and left side of a certain pose. So, the selection of an incorrect side view will result in a (partly) incorrect quadruple. Furthermore, the initial alignment obtained so far is not precise enough for general sets of meshes, because it assumes 90 degree rotations between consecutive meshes. Therefore, we need to improve the initial coarse alignment and verify which groups are correct and which are not. To finely align the meshes within a quadruple, we employ Rusinkiewicz's implementation of the point-to-plane ICP algorithm [19] (see Algorithm 2 applied to two meshes M_1 and M_2). This algorithm establishes point-to-plane correspondences from vertices of one mesh to the nearest facet

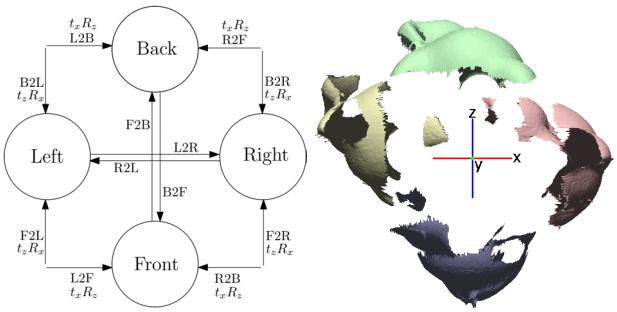


Figure 3. The relative transformation of each of the four views consist of: a translation (t) along either the x- or z-axis and a rotation (θ) around the z- or x-axis. F2R is the 2D transformation of the front's YZ-features towards the right side's XY-features rotated 90° around the y-axis for global alignment.

on the other mesh and vice versa. Point-to-plane pairs with a distance larger than 2.5 times the standard deviations (σ) and without normal compatibility (angle between points normals $> 45°$) are rejected. When either convergence or a stopping criterion is reached the MLS distance is returned. By changing the maximum number of iterations i_{max}, the target number of pairs p_{target}, and the minimum number of pairs p_{min} the outcome of the algorithm can be influenced.

To improve each of the (N') unique quadruples, the point-to-plane ICP algorithm is applied four times, once for each pair of neighboring meshes in a quadruple. This gives us 3D rigid transformations to transform each mesh towards its two neighbors and a MLS distance for each transformation. The minimum of all ($4N'$) MLS distances is used to accept or reject each 3D transformation. This minimal distance represents an optimal and reachable distance for the alignment of a pair of meshes, that is, under the assumption that all meshes $\mathbf{M_i}$ were scanned using the same laser range scanner and the fact that they were resampled and cleaned in a similar way. 3D transformations with a MLS distance larger than t_{dist} times the minimal distance are rejected. When three or four meshes remain linked by their 3D transformations, then these three or four meshes are finely aligned using the transformations and the *group* is accepted as such. See Figure 4 for several accepted and thus finely aligned groups of an object.

In rare situations, the ICP algorithm might reach the minimum number of corresponding point-pairs with a MLS distance that satisfies the selected t_{dist}, while the alignment is not correct. Because correct groups of meshes have similar volumes for their tight bounding box, we reject groups for which the tight bounding box volume is larger than t_{bb} times the median tight bounding box volume of all groups. Since the bounding box of a PCA normalized object is an approximation of its tight bounding box, we apply this check after PCA normalization in Section 2.6. See Section 2.7 for the values for i_{max}, p_{target}, p_{min}, t_{dist} and t_{bb}.

Algorithm 2 point-to-plane ICP (M_1, M_2)

1: $dist_{min} \leftarrow \infty$
2: **for** $i \leftarrow 1$ **to** i_{max} **if** no convergence **do**
3: $pairs \leftarrow$ closest point to plane pairs of p_{target} randomly selected vertices from M_1 to M_2 and M_2 to M_1
4: $pairs \leftarrow pairs$ with distance $< 2.5\sigma$ **and** with normal compatibility
5: **if** $\#pairs < p_{min}$ **return failure**
6: $dist \leftarrow$ minimize the sum of squared distances between the points in $pairs$
7: **update** $dist_{min}$
8: **end for**
9: **return** $dist_{min}$ **and** 3D rigid transformation

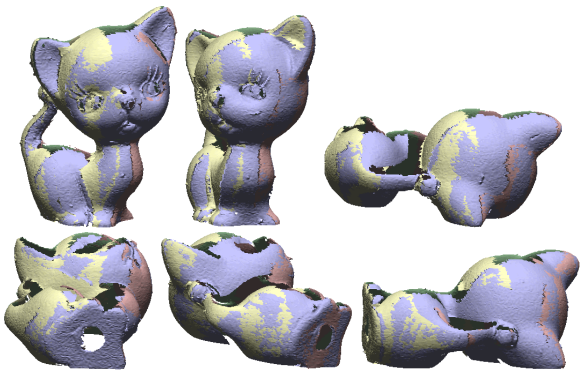


Figure 4. Six different, accepted, and finely aligned groups obtained after quadruple verification.

2.6. Group alignment

At this stage we have verified and finely aligned groups of either three or four meshes for different poses of the object. These groups cover large parts of the object's surface, which enables us to employ a technique to align incomplete models. To do so, we merge the data of each group's meshes into one (incomplete) model, and then pose normalize the models and obtain their correct alignment using the point-to-plane ICP algorithm as explained below. Because an incomplete model consists of approximately three or four times the number of vertices compared to one fine mesh $\mathbf{FM_i}$, we merge the coarse meshes $\mathbf{CM_i}$ instead.

An incomplete model is PCA normalized by using its center of mass as the coordinate system's origin and its principal axes of variation as the coordinate system's axes. A

well known problem of the PCA normalization is the existence of eight possible isometries to align the three principal axes to the x-, y- and z-axis. However, four of those involve mirroring, which ought to be excluded. The incomplete model with most vertices is selected to align the other models to, using our PCA/ICP algorithm (see Algorithm 3 applied to two models/meshes). This algorithm employs the point-to-plane ICP algorithm to each of the valid PCA isometries to find the optimal alignment of two models. In case of an object with three distinct eigenvalues, this algorithm has to try four different isometries. When an object has two similar eigenvalues, then swapping the two corresponding eigenvectors might be a necessity to find the optimal alignment. In case of even three similar eigenvalues the algorithm requires a maximum of 24 valid PCA isometries. Since the point-to-plane ICP algorithm with a large number of target pairs returns only a low MLS distance for the correct alignment of two models, a cut-off criterion can be used to limit the number of tried PCA isometries. This is accomplished by returning the minimal distance and the corresponding 3D rigid transformation once the algorithm reaches a state in which the minimal distance is sufficiently smaller than the maximal encountered ICP distance, determined by threshold t_{dist} (see Section 2.7). As a result the algorithm requires at least two and at most 24 applied point-to-plane ICP algorithms. With the alignment of the accepted groups the final alignment of meshes is completed. When a single mesh is included in multiple groups, its 3D transformation with the lowest ICP distance is retained.

Algorithm 3 PCA/ICP (M_1, M_2)

$dist_{min} \leftarrow \infty$, $dist_{max} \leftarrow 0$
for $i \leftarrow 1$ **to** 6 **do**
  $M_0 \leftarrow M_1$
  $dist \leftarrow$ MLS from point-to-plane ICP(M_0,M_2)
  update $dist_{min}$ **and** $dist_{max}$
  if $t_{dist} \times dist_{min} < dist_{max}$ **break**
  $M_z \leftarrow$ rotate M_1 180° around the z-axis
  $dist \leftarrow$ MLS from point-to-plane ICP(M_z,M_2)
  update $dist_{min}$ **and** $dist_{max}$
  if $t_{dist} \times dist_{min} < dist_{max}$ **break**
  $M_y \leftarrow$ rotate M_1 180° around the y-axis
  $dist \leftarrow$ MLS from point-to-plane ICP(M_y,M_2)
  update $dist_{min}$ **and** $dist_{max}$
  if $t_{dist} \times dist_{min} < dist_{max}$ **break**
  $M_x \leftarrow$ rotate M_1 180° around the x-axis
  $dist \leftarrow$ MLS from point-to-plane ICP(M_x,M_2)
  update $dist_{min}$ **and** $dist_{max}$
  if $t_{dist} \times dist_{min} < dist_{max}$ **break**
  if i = 1 **then** swap most similar eigenvectors
  else rotate M_1 90° to a new unique pose.
end for
return $dist_{min}$ **and** 3D rigid transformation

2.7. Implementation

Meshes obtained by a laser range scanner can be highly over sampled. For a time efficient method, we size normalized and resampled the meshes $\mathbf{M_i}$. The resampling was done by extracting a synthetic range image from all meshes using a fixed resolution (i.e. storing its z-buffer). This resolution was automatically selected, such that the largest mesh was resampled using 20,000 sample points. These synthetic range images were converted into our fine meshes $\mathbf{FM_i}$ by connecting adjacent range samples. Similarly we used only $\frac{1}{16}$-th of the extracted range samples to obtain our coarse meshes $\mathbf{CM_i}$. The meshes $\mathbf{FM_i}$ and $\mathbf{CM_i}$ were cleaned by removing faces with an edge longer than t_e ($t_e = 3.8$) times their resolution and removing connected components with less than t_f ($t_f = 20$) faces divided by the mesh resolution. The advantage of this remeshing process is that each vertex in a coarser mesh still represents a point on the originally constructed surface, and that the extraction of right-, top-, left- and bottommost sets of vertices becomes a trivial process. A disadvantage is that very slender object parts may not be sampled and become excluded from a coarser mesh. See Figure 5 for some resampling results.

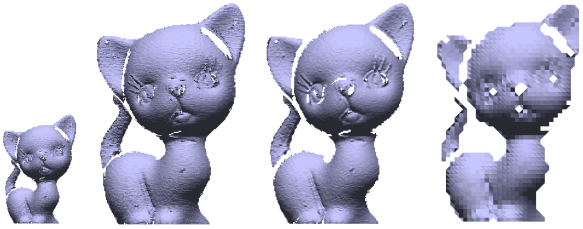


Figure 5. From left to right: the original, the resized, the finely resampled, and the coarsely resampled meshes of the *C2 kitten* object.

To match sets of features with the use of Algorithm 1 a maximum number of four iterations was sufficient. For Algorithm 2 we used 30 iterations for i_{max}, and a target number of pairs (p_{target}) and a minimum number of pairs (p_{min}) equal to, respectively, 80% and 6% of the maximum amount of vertices of the two meshes M_1 and M_2. During the verification of quadruples we removed 3D transformations with a distance $t_{dist} = 2.5$ times the minimal MLS distance, and rejected groups with a tight bounding box volume larger than $t_{bb} = 1.25$ times the median tight bounding box volume of all groups. Finally, we have used a cut-off criterion of $t_{dist} = 3.8$ in Algorithm 3.

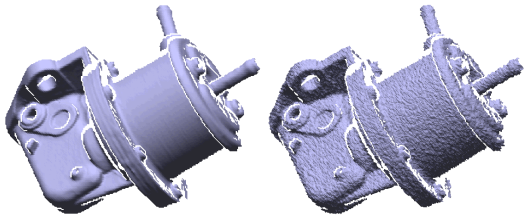


Figure 6. Adding synthetic noise to a clean range scan.

Object	Our method		MeshAlign	
	MLS(mm)	time(sec)	MLS(mm)	time(sec)
C1 bimba	1.333	93	1.332	340
C1 elephant	2.683	96	2.645	408
C1 greek	3.112	99	3.095	439
C1 oil-pump	2.833	98	2.811	363

Table 1. The quality of the alignments and the time to obtain them. Our method was applied to unaligned scans and MeshAlign to our alignment. The quality was measured using the MLS distance from aligned vertices to ground truth models.

3. Datasets

To test the effectiveness of our method we applied it to five classes (C1-C5) of increasing difficulty, with a total number of 25 objects. The first cluster consists of four existing 3D models from which 64 meshes were generated as follows. All models were scaled to a 200 mm sized model and transformed to eight unique poses. For each model's pose a range image (i.e. z-buffer) was generated for eight different sides, rotating the model in steps of 45° around the Y-axis. The range images were converted to meshes by connecting adjacent vertices and the meshes were cleaned by removing faces with a normal almost perpendicular ($\alpha_\Delta < 10^\circ$) to the scan direction. To model scanning noise (Figure 6), we randomly displaced each vertex within range $[-\frac{1}{2}\eta, \frac{1}{2}\eta]$ in its normal direction, with η equal to the average edge length in the mesh (we used Trimesh2 for this [19]). The original cluster models are used as ground truth models, to compare the quality of our final alignment to that of a multiview ICP algorithm.

The second cluster consists of seven objects scanned using the Roland LPX-250 laser scanner. This device scans objects orthogonally in the scan direction and has a rotation table, which we employed to obtain sets of either four or eight range scans per object's pose.

The third cluster contains a subset of three objects from the first cluster. These objects were scanned with a Minolta vi-910 for four, eight or twelve views per pose with approximately the same angle between sequentially scanned views.

The fourth cluster consists of Mian's four objects [14]. It is used to show the applicability of our method to sets of range scans without an exact number of views per pose.

To compare our results with Mian's alignment results [15] we included a fifth cluster from the Stuttgart Range Image Database [21] that he used in his work: a cluster of seven 3D models from which per model 66 range images were synthetically generated. Again the range images were converted into meshes and cleaned, and noise was added.

The models from cluster C1 and most of the models and scans from clusters C2 and C3 are available in the AIM@SHAPE shape repository [1].

4. Results

In Section 4.1 we evaluate the fine alignments obtained by our method, and we show that the quality of our alignments is comparable to the alignment obtained by a multiview ICP algorithm. Since our final alignments are of high quality, the meshes are directly merged towards a single surface mesh (Section 4.2). In Sections 4.3 and 4.4 we describe the effectiveness and efficiency of our method.

4.1. Fine alignment quality

For each cluster an object's final alignment (of meshes $\mathbf{FM_i}$) is shown in Figure 7, the other alignments are of similar quality. Because the alignment of grouped meshes were refined during quadruple verification and all accepted groups were finely aligned using a PCA/ICP algorithm, this Figure shows qualitative good final alignments.

For the models in cluster C1, we compared the quality of our alignments to the results of a multiview ICP algorithm. For this comparison we applied MeshAlign (v.2) [11] to our alignment to see if the alignment improves. MeshAlign applies Pulli's [17] multiview ICP algorithm to improve the initial alignment of large datasets. The quality of an alignment is quantified using Metro [6], which in our case computed the MLS distance of all vertices of the aligned meshes towards the ground truth model. Results from Table 1 show that MeshAlign's multiview ICP algorithm barely improves (less than 1.5%) the alignment we obtained using our method. In other words, our method returns a highly accurate alignment similar to the result of a multiview ICP algorithm.

4.2. Merged model quality

When an accurate fine alignment of meshes is obtained, a merge method can construct a single surface out of the

(a) C1 oil-pump (64/64) (b) C2 kitten (24/24) (c) C3 memento (32/36) (d) C4 dino2 (16/21) (e) C5 pit-bull (42/66)

Figure 7. One final alignment of each cluster C1-C5. The fraction of successfully aligned meshes $\mathbf{FM_i}$ is shown.

partially overlapping meshes. We used MeshMerge [11] to merge our sets of meshes. MeshMerge builds a carefully weighted distance field for each mesh, and blends all the distance fields together in a seamless way in a single volumetric representation. The final surface is reconstructed through the standard Marching Cubes algorithm [13].

The merged models in Figure 8 show qualitative good models, even though some models show holes due to missing scan data or to the lack of aligned meshes (Figure 8(l)). Notice that the models from clusters C1 and C5 are still noisy due to the added noise. Results for the *biplane* are based on the alignment of the original meshes, rather than resampled data, to avoid data loss near slender parts (such as the wings in side views). To inspect the full models we refer to [8].

4.3. Effectiveness

The results from Figure 8 and Table 2 show that our method is able to align mesh sets of objects scanned using different acquisition systems (different clusters), objects with few and many protrusions (e.g. *C2 buste* vs. *C2 memento*), objects with smooth and rough surfaces (e.g. *C3 pierrot* vs. *C3 warrior*), and mesh sets with little and much noise (e.g. *C3 memento* vs. *C5 pit-bull*). Furthermore, we see that our method aligns all meshes from C1 and C2, which are the meshes from the optimal acquisition process. For the datasets in clusters C3 and C4 the majority of meshes are aligned. The difficulty with the meshes in C3, is that the acquisition system's software generated rigorously cleaned meshes, which eliminated useful data. Our method automatically aligns a significant number of meshes (upto 51) for the noisy datasets in cluster C5. Note that our method performs better on the scan reconstructed models *bone*, *dino3*, *dragon*, *frog* and *pit-bull*, than on the highly symmetric CAD models *porsche* and *biplane*. In total our method aligns all meshes from 12 out of 25 datasets and 80% of the total amount of meshes.

The results of Mian's method shown in Table 2 are based on his automatic pairwise registration algorithm [15]. To obtain the correct alignment of these meshes, he had to select an *ordered* subset of (18 to 26) meshes for which a priori knowledge had to be defined about the overlap of views. Knowing which view pairs had overlap, his method was able to align all selected (noiseless) meshes. To align all 66 meshes, he would have to order all meshes while our method is able to automatically align large numbers of meshes (with noise) without a priori knowledge.

4.4. Efficiency

Our method aligns datasets of N meshes $\mathbf{M_{i\in(1\ldots N)}}$. For each mesh we synthetically generate a range image from which we construct a fine mesh $\mathbf{FM_i}$, a coarser mesh $\mathbf{CM_i}$ and two sets of 2D features. Then each fine mesh is matched to all other fine meshes, which is quadratic in the number of meshes. It takes constant time to generate quadruples in which four meshes are coarsely aligned to each other. Similar quadruples are removed, leaving N' quadruples of meshes to be verified. During the verification of all quadruples, four mesh-to-mesh comparisons are applied involving a point-to-plane ICP algorithm, which is of $O(N')$. After verification there are N'' accepted finely aligned groups of either three or four meshes left. The coarse meshes $\mathbf{CM_i}$ are "merged", that is, considered to be the surfaces of one mesh without additional computations. Each merged group has fewer vertices than a single fine mesh, because a group consists of three or four coarse meshes and each coarse mesh $\mathbf{CM_i}$ has approximately $\frac{1}{16}$ of the fine mesh's vertices. Finally, we apply N'' times the PCA/ICP algorithm for which we use at most 24 times the point-to-plane ICP algorithm, in other words, we apply at most $24N''$ mesh-to-mesh comparisons, which is of $O(N'')$. Since $N'' \leq N' \leq N$ we have a total number of $O(N^2)$ mesh-to-mesh comparisons.

The N^2 mesh-to-mesh comparisons during the matching of features are based on low cost operations only. Experiments show that the time to match the sets of 2D features

is much less than the time to verify the quadruples, even for the datasets with 66 meshes. Thus, up to 66 meshes the group verification with a number of $O(N')$ comparisons is the bottleneck. However, the group verification is also the main reason why we don't need to apply a multiview ICP algorithm, which requires $O(N^2)$ mesh comparisons. For comparison, the time that MeshAlign's multiview ICP algorithm required to "improve" the alignment of the 64 meshes of the *C1 bimba* model was 340 seconds, which is even more than the total time of 93 seconds (see Table 2) our algorithm required to obtain an alignment of similar quality starting from the *bimba*'s unaligned meshes.

Our method's feature extraction requires less than one second, the verification of a group two to three seconds, and the alignment of groups nearly a second per accepted group. For the alignment of the majority of 66 noisy scans, our method needed two to five minutes in total. Sets of 64 meshes from cluster C1 were even aligned in less than two minutes. All timings are based on a Pentium IV 2,8 GHz with 520 MB internal memory.

Obviously, the total number of applied point-to-plane ICPs (Algorithm 3) shown in Table 2 is related to the number of accepted groups. In general two to five point-to-plane ICPs were required to align an accepted group. In Section 2.6 we already stated that this number could increase when the object has similar eigenvalues. This is what happened for *C3 pierrot*. The effect on *C2 pierrot* is however much less, because its data is more complete.

5. Discussion

A limitation of our method is that each quadruple (and thus each scan) should cover more or less the entire object, because accepted quadruples are aligned using their center of masses and principal axes. Scans that cover only a small part of an object's view, or scans without proper left and right neighbors are most likely not aligned. Although the matching of features can correct for small rotations between neighbors, the method performs best on horizontally scanned meshes. This also explains the difference in performance for clusters C1 and C5: the latter consists of scans without any pose assumptions, while the first has for each pose eight scans (i.e. two quadruples) with the same bottom-up direction. Nevertheless, our method performs very well on the scans from cluster C5.

6. Concluding remarks

In this paper, we propose a new and efficient method to perform both the coarse and fine alignment of meshes. Our method selects for each mesh three other meshes that represents its left, right, and back side neighbors. Such quadruples are then verified and the accepted groups of three or four meshes form incomplete 3D models of an object pose. These incomplete models are pose normalized and with the selection of the optimal PCA isometry (based on its ICP-error) these models can be correctly aligned. With the alignment of the incomplete models we obtain our final alignment.

Our method is capable of aligning various sets of meshes both effectively and efficiently. Effectively, because 80% of the total amount of meshes were automatically aligned and even all meshes of 12 out of 25 datasets were completely aligned, without a priori knowledge. Efficiently, because our method requires $O(N^2)$ mesh-to-mesh comparisons using low cost 2D silhouette feature matching. The quadruple verification and group alignment use high cost operations, but require only $O(N)$ mesh comparisons. In contrast, other coarse alignment systems apply a multiview ICP algorithm to refine the obtained coarse alignment, which requires $O(N^2)$ high cost mesh comparisons with a final alignment of similar quality as a result.

Future work includes the development of an efficient algorithm that can be applied to align the remaining unaligned meshes to the set of aligned meshes. To accomplish this, an efficient mesh selection strategy is required to select a subset of aligned meshes that represent various object views and to find overlap between these meshes and the unaligned ones.

Acknowledgements

This research was supported by the FP6 IST Network of Excellence 506766 AIM@SHAPE. We acknowledge Stuttgart University and A.S. Mian for their models, and the Visual Computing Lab of ISTI-CNR for their software. The *memento* statue is copyrighted by Artihove Art Centre and Maarten Benschop. The *bozbezbozzel* is made by Marian van der Steenhoven.

References

[1] AIM@SHAPE. Shape Repository at `http://www.aimatshape.net/resources`, Nov 2006.

[2] R. Bergevin, M. Soucy, H. Gagnon, and D. Laurendeau. Towards a general multiview registration technique. *IEEE Trans. PAMI*, 18(5):540–547, 1996.

[3] C.-S. Chen, Y.-P. Hung, and J.-B. Cheng. RANSAC-based DARCES: a new approach to fast automatic registration of partially overlapping range images. *IEEE Trans. PAMI*, 21(11):1229–1234, 1999.

[4] J. Cheng and H. Don. A graph matching approach to 3-D point correspondences. *Int. Journal of Pattern Recognition and Artificial Intelligence*, 5(3):399–412, 1991.

[5] C. S. Chua and R. Jarvis. Point signatures: A new representation for 3D object recognition. *Int. Journal of Computer Vision*, 25(1):63–85, 1997.

Object	#Meshes (1041)	#Unique Quadruples	#Accepted Groups	#Point-to-plane ICPs during PCA/ICP	Total time in seconds	#Aligned Meshes Our method(828)	#Aligned Meshes Mian's method
C1 bimba	64	16	16	44	93	64	-
C1 oilpump	64	16	16	39	96	64	-
C1 greek	64	16	16	43	99	64	-
C1 elephant	64	16	16	38	98	64	-
C2 angel	16	4	4	16	16	16	-
C2 buste	16	4	4	6	15	16	-
C2 kitten	24	6	6	14	22	24	-
C2 pierrot	20	5	5	14	23	20	-
C2 memento	20	5	5	12	19	20	-
C2 warrior	24	6	6	16	23	24	-
C2 bozbezbozzel	44	15	15	52	73	44	-
C3 pierrot	24	12	10	77	67	15	-
C3 memento	36	11	9	23	44	32	-
C3 warrior	24	10	8	22	33	24	-
C4 chef	22	14	14	30	72	21	-
C4 chicken	16	6	5	10	31	12	-
C4 dino1	16	13	13	40	66	13	-
C4 dino2	21	16	13	37	71	16	-
C5 bone	66	52	50	119	253	51	23
C5 dino3	66	48	42	100	221	44	21
C5 dragon	66	51	46	113	248	47	21
C5 frog	66	51	46	113	248	47	19
C5 pit-bull	66	51	42	105	224	42	22
C5 porsche	66	36	12	37	137	29	18
C5 biplane*	66	40	9	18	265	15	26

Table 2. Details of our alignment method. *The original meshes were used.

[6] P. Cignoni, C. Rocchini, and R. Scopigno. Metro: measuring error on simplified surfaces. *Computer Graphics Forum*, 17:167–174, 1998.

[7] B. Curless and M. Levoy. A volumetric method for building complex models from range images. In *Proc. SIGGRAPH 1996*, volume 30, pages 303–312, 1996.

[8] F. B. ter Haar. Quadruple alignment at `http://give-lab.cs.uu.nl/quadruple-alignment/`, Nov 2006.

[9] K. Higuchi, M. Hebert, and K. Ikeuchi. Building 3-D models from unregistered range images. *CVGIP-GMIP*, 57(4):315–333, 1995.

[10] D. Huber and M. Hebert. 3-D modeling using a statistical sensor model and stochastic search. In *IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, pages 858–865, 2003.

[11] ISTI-CNR Visual Computing Laboratory. MeshAlign v.2, MeshMerge and Metro at `http://www.aimatshape.net/resources`, Nov 2006.

[12] A. E. Johnson. *Spin Images: A Representation for 3-D Surface Matching*. PhD thesis, Carnegie Mellon University, Pittsburgh, 1997.

[13] W. E. Lorensen and H. Cline. Marching cubes: a high resolution 3D surface construction algorithm. In *Proc. SIGGRAPH 1987*, volume 21, pages 303–312, 1987.

[14] A. S. Mian, M. Bennamoun, and R. A. Owens. 3D model-based object recognition and segmentation in cluttered scenes. *IEEE Trans. PAMI*, 2006.

[15] A. S. Mian, M. Bennamoun, and R. A. Owens. A novel representation and feature matching algorithm for automatic pairwise registration of range images. *International Journal of Computer Vision (IJCV)*, 66(1):19–40, 2006.

[16] P. J. Neugebauer. Reconstruction of real-world objects via simultaneous registration and robust combination of multiple range images. *Int. Journal of Shape Modeling*, 3(1-2):71–90, 1997.

[17] K. Pulli. Multiview registration for large data sets. In *Proc. 2nd Int. Conf. on 3D Digital Imaging and Modeling (3DIM)*, pages 160–168, 1999.

[18] C. Rocchini, P. Cignoni, F. Ganovelli, C. Montani, P. Pingi, and R. Scopigno. The marching intersections algorithm for merging range images. *The Visual Computer*, 20:149–164, 2004.

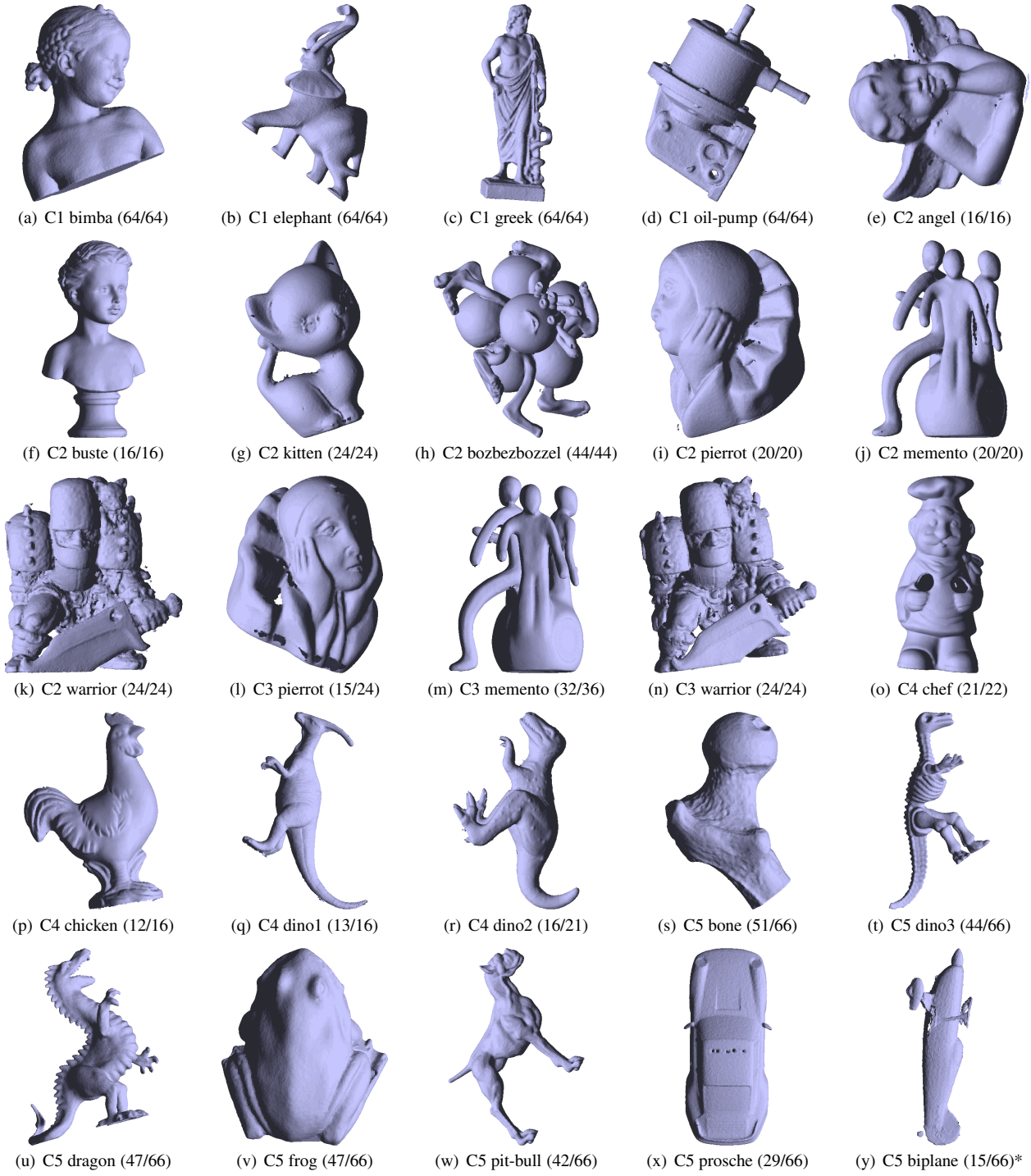
[19] S. Rusinkiewicz. Trimesh2 at `http://www.cs.princeton.edu/gfx/proj/trimesh2/`, Nov 2006.

[20] A. J. Stoddart and A. Hilton. Registration of multiple point sets. In *Proc. 13th Int. Conf. on Pattern Recognition*, volume A, pages 40–44, 1996.

[21] Stuttgart. Range image database at `http://range.informatik.uni-stuttgart.de/htdocs/html/`, Nov 2006.

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

[23] J. V. Wyngaerd, L. V. Gool, and M. Proesmans. Invariant-based registration of surface patches. In *IEEE Int. Conf. on Computer Vision*, pages 301–306, 1999.



(a) C1 bimba (64/64) (b) C1 elephant (64/64) (c) C1 greek (64/64) (d) C1 oil-pump (64/64) (e) C2 angel (16/16)

(f) C2 buste (16/16) (g) C2 kitten (24/24) (h) C2 bozbezbozzel (44/44) (i) C2 pierrot (20/20) (j) C2 memento (20/20)

(k) C2 warrior (24/24) (l) C3 pierrot (15/24) (m) C3 memento (32/36) (n) C3 warrior (24/24) (o) C4 chef (21/22)

(p) C4 chicken (12/16) (q) C4 dino1 (13/16) (r) C4 dino2 (16/21) (s) C5 bone (51/66) (t) C5 dino3 (44/66)

(u) C5 dragon (47/66) (v) C5 frog (47/66) (w) C5 pit-bull (42/66) (x) C5 prosche (29/66) (y) C5 biplane (15/66)*

Figure 8. The final reconstructed 3D models of clusters C1-C5. The surface merging was performed with MeshMerge on the fine meshes $\mathrm{FM_i}$. The fraction of successfully aligned meshes used to reconstruct a model is shown. *The original meshes were used.