Pacific Graphics (2015)
N. J. Mitra, J. Stam, and K. Xu (Editors)

Short Paper

Pairwise Surface Registration Using Local Voxelizer

Peng Song and Xiaoping Chen

School of Computer Science and Technology, University of Science and Technology of China

Abstract

Surface registration is the process that brings scans into a common coordinate system by aligning their overlapping components, which can be achieved by finding a few pairs of matched points on each scan pair using shape descriptors and employing the matches to compute an alignment transformation. This paper proposes a local voxelizer *descriptor, and the key idea is to define a unique local reference frame (LRF) using the local shape around a basis point, perform voxlization for the local shape within a cubical volume aligned with the LRF, and concatenate local features extracted from each voxel to construct the descriptor. A pairwise registration algorithm is developed by choosing a single pair of matched points using the local voxelizer descriptor, and computing a rigid transformation based on aligning the corresponding LRFs. Quantitative experiments show that our algorithm can register scan pairs with small overlap, while maintaining acceptable registration accuracy.*

Categories and Subject Descriptors (according to ACM CCS): I.3.5 [Computer Graphics]: Computational Geometry and Object Modeling—Surfaces and object representations

1. Introduction

Surface registration is essential for 3D shape acquisition and modeling, which brings scans or partial surfaces captured from different views into a common coordinate system. Given a pair of scans, the goal of rigid surface registration is to find a rigid transform that optimally positions one scan relative to the other by aligning their overlapping components. When the pose of two scans are close to each other, Iterative Closest Points (ICP) [BM92] [CM92] can be used to register the scans. However, this usually requires manual effort to position the scans for a coarse alignment. To register two scans with arbitrary initial poses, a few pairs of matched points are required to be established for estimating an aligning transform [CHC99] [GMGP05] [MBO06] [AMCo08] [MAM14].

The point matching problem can be solved by using local shape descriptors, which are quantities computed for each basis point on a scan surface by using the local shape (i.e., support) around the point. Points with similar descriptors potentially correspond. Normally, a few (at least 3) corresponding points found by matching descriptors are required to compute a rigid transform. By defining a unique LRF for each point using its support and attaching the LRF with descriptors, using one pair of matched points is able to compute a rigid transform based on aligning the three corresponding axes of their LRFs [TSS10] [GSB*13]. This drastically reduces the space of searching corresponding points (i.e., from at least 3 pairs to 1 pair), and thus increases the chance to find correct aligning transforms for the scans.

Taking advantage of a uniquely defined LRF [GSB*13], we propose a new shape descriptor, called *local voxelizer*. The key idea is to firstly define a unique LRF using the support around a basis point and then perform voxlization for the local shape within a cubical volume aligned with the LRF. The descriptor is constructed by concatenating shape features extracted from each voxel. To find out local shape feature that ensures high discriminative power of the descriptor, we propose a set of feature candidates and perform quantitative comparison among them.

A pairwise registration algorithm is developed by using local voxelizer to represent scans and find corresponding scan points. First, a number of matches of scan points are established by comparing their descriptors. Then, aligning transform candidates are generated by aligning the two LRFs of each matched point pair and further ranked according to their descriptor distances. Lastly, the transform that ensures the scan pair to have maximal surface overlapping area is selected. Experiments on scans of several 3D models show that our algorithm can register scan pairs with small overlap, while maintaining acceptable registration accuracy.

DOI: 10.2312/pg.20151272

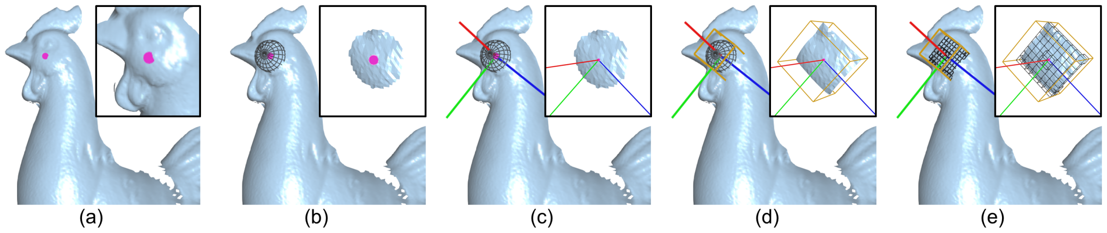


Figure 1: *Constructing local voxelizer. (a) Select a basis point; (b) define support using a sphere centered at the basis point; (c) construct LRF; (d) define a larger local shape using a cube aligned with the LRF; (e) voxelize the local shape.*

2. Related Work

Rigid surface registration can be achieved either by using ICP or matching local shape descriptors. We mainly review methods that use descriptors since they are more related.

Low-dimensional shape descriptors, such as integral volume [GMGP05] [PWHY09], surface curvature [GCO06], and surface hash [ART10], are easy to compute and compare. Yet, they have low discriminative ability since different points on the same scan surface could have very close descriptor values. Thus, multiple ambiguous matches can be resulted and a further disambiguating process is required.

High-dimensional descriptors provide a fairly detailed description of the shape around a surface point, and thus can be directly used to solve the correspondence problem. Johnson et al. [JH99] proposed a spin image representation by spinning a 2D image about the normal of a feature point and summing up the number of points falls into the bins of that image. Huber and Hebert [HH03] further applied the spin images for automatic surface registration. Frome et al. [FHK*04] proposed 3D shape context descriptor by accumulating 3D histograms of points within a partitioned sphere centered at a feature point. Mian et al. [MBO06] proposed a 3D tensor representation by constructing an LRF from a pair of oriented points and encoding the intersected surface area into a multidimensional table.

More recently, Zhong [Zho09] proposed intrinsic shape signatures by improving [FHK*04] based on a different partitioning of the 3D spherical volume and a new definition of LRF with ambiguity (four variants). Tombari et al. [TSS10] proposed the signature of histograms of orientations (SHOT) by constructing a unique LRF for a feature point and concatenating local histograms defined on each bin within a 3D spherical volume. Guo et al. [GSB*13] constructed a unique and more robust LRF, and then extracted a rotational project statistics (RoPS) descriptor for a feature point.

Comparing with [TSS10] [GSB*13] that also attach a unique LRF to shape descriptors, local voxelizer is based on the voxelization of local shape within a cubical volume (rather than a spherical volume) aligned with the LRF, and partitions the volume into uniform bins (i.e., voxel) such that local feature inside each bin can be equally weighted and extracted more easily, e.g., surface area feature that requires mesh clipping. Moreover, we propose a set of feature candidates that can be extracted from each bin, and perform a quantitative comparison among them to find out the best one to construct the descriptor. Comparing with [MBO06] that also uses voxelization to construct 3D tensors, local voxelizer performs voxelization within a uniquely defined LRF, and thus enables scan alignment using a single pair of matched points based on aligning the LRFs. As a result, local voxelizer requires less amount of overlap for aligning scans.

3. Local Voxelizer Shape Descriptor

We take a surface mesh S as input. If a 3D point cloud is given, we first convert it into a mesh [HDD*92]. A local voxelizer descriptor is a function that assigns to each point $p \in S$ a vector $f(p) \in \mathbb{R}^m$ by analyzing the support around p, where m is the length of the vector.

3.1. Local Voxelizer Construction

Given a basis point p and a support radius r, we construct local voxelizer by defining a unique LRF using the support around p and performing local voxlization within the LRF, see Figure 1. The descriptor vector is calculated by concatenating value(s) computed from shape features (e.g., point, normal, curvature) of local mesh triangles within each voxel.

Constructing LRF. We define the support around p by intersecting the input mesh S with a sphere of radius r centered at p, see Figure 1(b). Taking the support as input, we construct an LRF using the method in [GSB*13] with two steps: 1) construct three orthogonal directions based on principal component analysis (PCA) of triangles in the support; 2) disambiguate the sign of each orthogonal direction to obtain three unique coordinate axes of the LRF, see Figure 1(c). Note that the sign disambiguation method in [GSB*13] could fail for local symmetrical surfaces, e.g., flat or spherical surfaces. Thus, we adopt the surface normal of p to assist the disambiguation such that the principal axis associated with the smallest eigenvalue (i.e., the red axis in Figure 1(c)) is consistent with the normal of p.

Local Shape Voxelization. Once the LRF is constructed, we can define a cubical volume centered at p, whose edges are aligned with the LRF and have length of $2r$, see Figure 1(d). Note that such cubical volume is the smallest one that contains the support used to construct the LRF. We intersect the cubical volume with the input mesh S, obtaining a local surface patch $\mathrm{S_p}$.

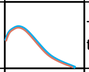
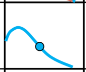
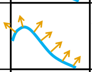
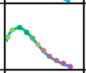
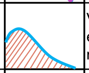
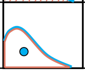
No.	Feature	Illustration	Description	# Value
F1	Surface Area		Total area of clipped mesh triangles in the voxel	1
F2	Surface Centroid		Average position for local mesh vertices in the voxel	3
F3	Average Normal		Angle between basis point normal and average normal of local mesh triangles	1
F4	Average Curvature		Average of surface curvatures for local mesh vertices	1
F5	Shape Volume		Volume of the local shape enclosed by local mesh and related voxel faces	1
F6	Shape Centroid		Centroid of the local shape enclosed by local mesh and related voxel faces	3

Table 1: *Feature candidates extracted from local shape within each voxel. Note that $F2$ and $F6$ use centroid position relative to the minimum point of the voxel cube.*

Taking $\mathrm{S_p}$ as input, we perform local voxelization for it by partitioning the cubical volume into $K \times K \times K$ grids (i.e., voxels), see Figure 1(e). For each voxel $\mathrm{V_i}$, we find the intersection between $\mathrm{V_i}$ and $\mathrm{S_p}$ by clipping $\mathrm{S_p}$ using the six planes of $\mathrm{V_i}$. The resulting triangles in $\mathrm{V_i}$ are denoted as $\mathrm{S_p^i}$. For each non-empty voxel, we compute value(s) based on the shape features of $\mathrm{S_p^i}$, while empty voxels are assigned default zero value(s). We calculate the descriptor vector by concatenating the value(s) assigned for each voxel. Since most of the voxels in the voxelization are likely to be empty, the resulting descriptor will have many zero elements.

Extracting Local Feature Candidates. As described above, for each non-empty voxel $\mathrm{V_i}$, one or a few values need to be computed from $\mathrm{S_p^i}$ for representing the local shape in $\mathrm{V_i}$. In Table 1, we propose a set of feature candidates that can be extracted, as well as the number of output values. For each candidate, we normalize its value(s), e.g., $F1$, $F2$, $F5$, and $F6$ are normalized relative to the voxel dimension.

Most of the proposed feature candidates are straightforward to calculate except $F5$ and $F6$, and we estimate them using a uniform sampling approach [SFLF15]. For a local voxelization with $K \times K \times K$ voxels, we first build a $(b \times K + 1)^3$ uniform 3D point grid within it, where we have $b+1$ sample points along each edge of each voxel. Then we cast $(b \times K + 1)^2$ rays through the local mesh $\mathrm{S_p}$, where each ray passes through $(b \times K + 1)$ sample points. We compute intersecting points between each ray and $\mathrm{S_p}$, and identify each intersecting point as inner or outer based on the angle between the normal of the point and the ray direction (we set the threshold as 90 degree). We classify each sample point as interior or exterior by checking if it locates between inner intersecting point (or the voxel boundary) and outer intersecting point along the ray direction, see Figure 2. After that, $F5$ is estimated by counting the number of interior sample points and then computing their coverage percentage within the voxel, while $F6$ is estimated by averaging the positions

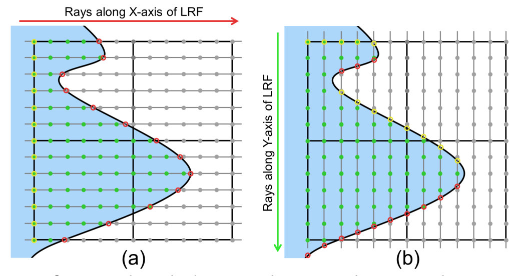


Figure 2: *Our local shape volume and centroid estimation method for ray directions along (a) x-axis and (b) y-axis of LRF, where $b = 6$. Inner and outer ray-mesh intersecting points are marked as yellow and red circles, while interior and exterior sample points are marked in green and gray.*

of all interior sample points. In our experiments, we select performing the ray casting along x-axis of the LRF such that most rays will intersect $\mathrm{S_p}$ only once, see Figure 1(e) and Figure 2(a). Note that the concept of $F5$ is similar to the integral volume feature employed in [GMGP05], while using a cubical instead of spherical bounding volume.

3.2. Local Voxelizer Generation Parameters

The local voxelizer descriptor has three parameters: (i) the support radius r; (ii) the voxel grid resolution K; and (iii) the local feature f extracted for each voxel.

We conducted experiments for different settings of parameters on the UWA dataset [MBO06] using the criterion of recall versus 1-precision curve (RP Curve) [MS05], which is calculated as follows. Given two scans $\mathrm{S_i}$ and $\mathrm{S_j}$ and the ground truth transformation, a point on $\mathrm{S_i}$ is matched against each point on $\mathrm{S_j}$ to find the closest match by using the Euclidean distance of two points' descriptor vectors. If the ratio between the smallest distance and the second smallest one is less than a threshold τ, then the point on $\mathrm{S_i}$ and the closest one on $\mathrm{S_j}$ are considered a match. A match is considered as a true positive only if the distance between the physical locations of the two points is sufficiently small, otherwise it is considered as a false positive. Thus, recall is the number of true positives relative to the total number of corresponding points, and 1-precision is the number of false positives relative to the total number of matches. An RP curve can be further generated by varying the threshold τ.

Support Radius. Support radius r determines the amount of surface that is encoded by local voxelizer. A large support radius enables the descriptor to encapsulate more information of scan surface and therefore provides more descriptiveness, yet it also makes the descriptor more sensitive to the overlapping size of input scans since the overlapping region of two scans need to be large enough to contain supports with radius r for at least one pair of matched points. We tested the performance of the descriptor on the UWA dataset with respect to a number of support radii. The other two parameters were set constant as $K = 12$ and $f = F1$.

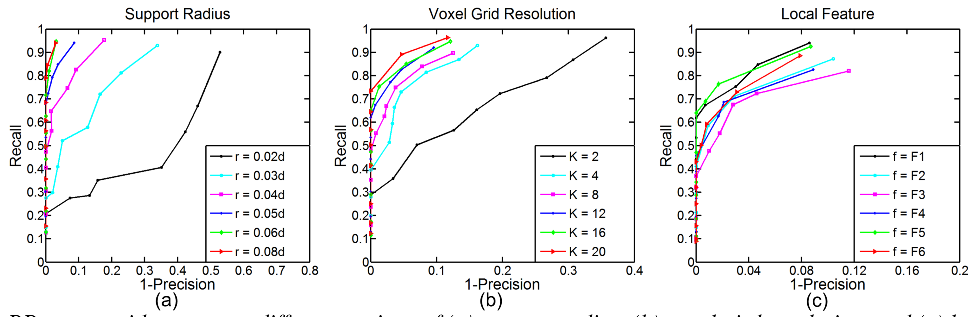


Figure 3: *RP curves with respect to different settings of (a) support radius, (b) voxel gird resolution, and (c) local feature.*

Figure 3(a) presents the generated RP curves. The plot shows that the descriptor performance improved significantly when r was increased from $0.02d$ to $0.05d$, and improved slightly when r was further increased from $0.05d$ to $0.08d$, where d is the average diagonal size of test scans. Therefore, we select $r = 0.05d$ as a tradeoff between the descriptiveness and robustness to scan overlap size.

Voxel Grid Resolution. Voxel grid resolution K determines the descriptiveness of the descriptor since a dense partition offers more details about the local shape. However, a dense partition also requires more computation cost for both descriptor construction and comparison. We tested the performance of the descriptor with respect to a number of different partitions. The other two parameters were set constant as $r = 0.05d$ and $f = F1$.

Figure 3(b) presents the generated RP curves. The plot shows that the descriptor performance kept improving when K was increased from 2 to 20. However, the improvement was not that obvious after K has reached 12. Therefore, we select $K = 12$ as a tradeoff between the descriptiveness and computation cost.

Selecting Local Features. Selection of local features plays an important role in generating the descriptor. It determines not only the descriptor's ability to encapsulate local shape information but also the size of the descriptor vector. We tested the performance of the descriptor with respect to the feature candidates presented in Table 1. The other two parameters were set constant as $r = 0.05d$ and $K = 12$.

Figure 3(c) presents the generated RP curves. The plot shows that integral features (i.e., $F1$ and $F5$) achieved the best performance, and $F5$ is a bit better when 1-precision is close to zero. The performance of two centroid features (i.e., $F2$ and $F6$) is very close to each other, yet still not comparable with that of integral features. The differential features (i.e., $F3$ and $F4$) obtained the worst performance. One reason is that their values are close to zero when the local surface is flat, which is the same as the default value assigned for empty voxels, making discriminating such surfaces difficult. Moreover, differential features are sensitive to mesh noise and varying resolution, thus we suggest avoiding using them for constructing local voxelizer.

4. Pairwise Surface Registration using Local Voxelizer

So far we have developed a novel shape descriptor based on local voxelization within a unique LRF. In this section, we apply the descriptor for pairwise surface registration. Given a data scan $\mathrm{S_d}$ and a fixed reference scan $\mathrm{S_r}$, our pairwise registration algorithm consists of three key steps to align $\mathrm{S_d}$ with $\mathrm{S_r}$, i.e., scan representation, generating scan alignment candidates, and selecting best scan alignment.

Step 1: Scan Representation. Given a scan, we first select N seed points from the scan point cloud. To represent the scan more closely, we want the seed points to cover the whole scan surface and to avoid picking points that are too close to each other. Thus, we randomly sample the point cloud and enforce minimal separation distance among the samples to obtain N seed points. For each seed point, the corresponding LRF and local voxelizer are constructed, and stored in a library. We select $N = 2000$ in experiments as a tradeoff of computation cost and sampling performance.

To align a pair of scans $\mathrm{S_d}$ and $\mathrm{S_r}$, we simply can match the descriptors of their sampled seed points. However, since the seeds points cover the whole scan surface evenly, one randomly picked point on $\mathrm{S_d}$ could match one seed point on $\mathrm{S_r}$ correctly, giving that the physical distance between the seed point and the real match on $\mathrm{S_r}$ is small. In our experiment, we find that sampling $M = 200$ feature points on S_d and matching their descriptors with those of N seed points on $\mathrm{S_r}$ can achieve good matching result, and vice versa. Thus, for each scan, we further sample M feature descriptors from the original N seed descriptors, and store them in the library.

Step 2: Generating Scan Alignment Candidates. To align $\mathrm{S_d}$ with $\mathrm{S_r}$, each feature descriptor of $\mathrm{S_d}$ is matched against all seed descriptors of $\mathrm{S_r}$. If the Euclidean distance between the two descriptor vectors is less than a threshold, the feature point on $\mathrm{S_d}$ and its closest seed point on $\mathrm{S_r}$ are considered a match. Note that the match is not guaranteed to be correct since: 1) there could be no or very small overlap between $\mathrm{S_d}$ and $\mathrm{S_r}$; 2) the local shape around the feature point is not discriminative, e.g., flat or spherical; and 3) there exist similar or symmetrical shape features on $\mathrm{S_r}$. Each generated match creates a scan alignment candidate (i.e., a 4×4 transformation matrix) by aligning the three axes of the uniquely defined LRFs, see Figure 4 for two examples.

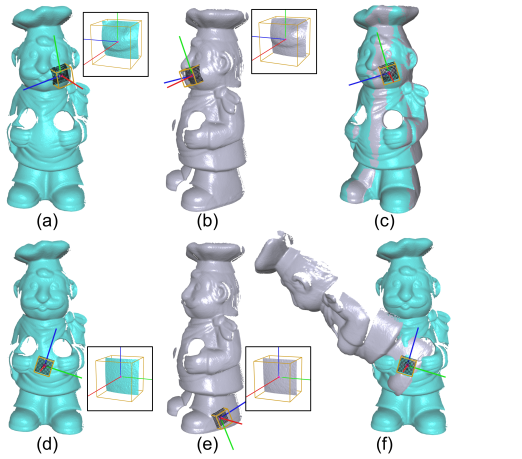


Figure 4: *Align two scans of* CHEF *by matching local voxelizer. (a&b) A pair of correctly matched descriptors (zoom-in views show the local shapes); (c) align the two scans based on the LRFs; (d&e) another pair of matched descriptors; (f) the two scans are not aligned properly since the selected local shape is not discriminative (i.e., flat).*

Step 3: Selecting Best Scan Alignment. By matching the descriptors of S_d and S_r, we can obtain around M alignment candidates. We first sort these candidates based on the descriptor distance and then pick the top five candidates with the smallest distance. To find the (real) best one from the five candidates, we evaluate each candidate by transforming S_d into S_r and estimating the normalized overlapping area between the transformed S_d (denoted as S_d^t) and S_r. In detail, we first build a kd-three for the point cloud of S_r. Then for each point in S_d^t, we find its closest point in S_r. We consider a point in S_d^t and its closest point in S_r as overlapping if their distance is smaller than a threshold. The score of the alignment candidate is calculated as the overlap area divided by the surface area of S_d. Lastly, we select the one with the highest score from the five candidates as the output.

By the above procedure, a pair of scans with certain amount of overlap can always be properly aligned, which can be further refined by using ICP [BM92]. Since the initial transformation calculated by aligning LRFs is very close to the real one, near perfect alignment can be achieved after only one or two ICP iterations. Figure 5 shows two scan alignments before and after using ICP. Their accuracy (without using ICP) are reported in Section 5.

5. Results and Quantitative Analysis

We implemented our method in C++ and executed it on a desktop PC with an Intel i7-3770 CPU (3.4GHz, 4 cores) and 8GB memory. In general, our method took around 20 seconds to register a pair of scans in the UWA dataset [MBO06], where the number of triangles of scans ranges from $50k$ to $100k$, see Figure 4 and 5 for examples. We further evaluate our pairwise algorithm according to two criteria: 1) accuracy of alignment; and 2) required amount of scan overlap.

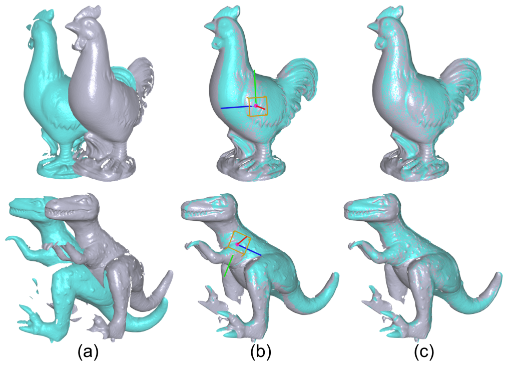


Figure 5: *Pairwise alignment for* CHICKEN *and* T-REX. *(a) Input scans; aligned scans (b) before and (c) after using ICP. (b) shows also the matched local voxelizer for the alignment.*

(1) Accuracy of Alignment. To evaluate accuracy of our pairwise registration algorithm quantitatively, we compare the transformation generated by our algorithm (without ICP refinement) with the ground truth transformation. In detail, for each scan pair S_i and S_j, the ground truth rotation matrix (R_{iGT}) and translation vector (t_{iGT}) of S_i with relative to S_j were computed by manually positioning the scans for a coarse alignment and then refining it using ICP. Next, the transformation (i.e., R_i and t_i) resulting from our pairwise algorithm is compared to the ground truth transformation. Using the similar criterion as in [MBO06], the error in the two rotation matrices was calculated using Equation 1,

$$\theta_{ie} = arccos\left(\frac{trace(R_i R_{iGT}^{-1}) - 1}{2}\right)\frac{180}{\pi} \tag{1}$$

where θ_{ie} is zero in the case of no rotational error. Similarly, the translational error t_{ie} was calculated using Equation 2.

$$t_{ie} = \|t_i - t_{iGT}\| \tag{2}$$

We performed this experiment on the scans of four objects in the UWA dataset. Figure 6(a) shows that all rotational errors are less than 2 degree, and most of them are around 0.5 degree. Figure 6(b) shows that the translational errors are all less than 0.01d, and most of them are around 0.003d, where d is the average diagonal size of the scans. We also computed accuracy for alignment examples of CHICKEN and T-REX shown in Figure 5, which have rotational errors of 0.28 and 0.54 degree, and translational errors of 0.001d and 0.003d, respectively.

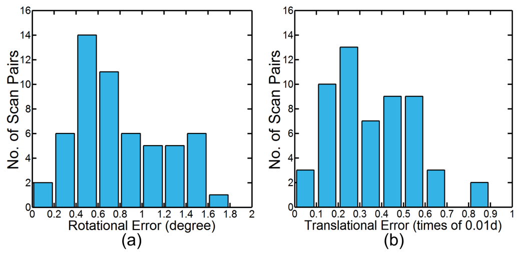


Figure 6: *Histograms of (a) rotational and (b) translational errors of our pairwise registration algorithm on four models.*

(2) Required Amount of Scan Overlap. We also evaluated the algorithm performance against varying amounts of overlap between each pair of scans. We define the amount of overlap between two scan meshes S_i and S_j using Equation 3,

$$overlap = \frac{\mathrm{C}(\mathrm{S_i}, \mathrm{S_j})}{min(\mathrm{V}(\mathrm{S_i}), \mathrm{V}(\mathrm{S_j}))} \tag{3}$$

where $\mathrm{V}(\mathrm{S})$ denotes the number of vertices of scan S, and $\mathrm{C}(\mathrm{S_i}, \mathrm{S_j})$ denotes the number of corresponding vertices of $\mathrm{S_i}$ and $\mathrm{S_j}$. We calculated overlap between all possible $L(L-1)/2$ pairs of scans, where L is the total number of scans per object. Next, we used our pairwise algorithm to align each scan pair. We categorized each alignment result as correct or incorrect by measuring its accuracy, i.e., rotational error less than 5 degree and translational error less than $0.02d$.

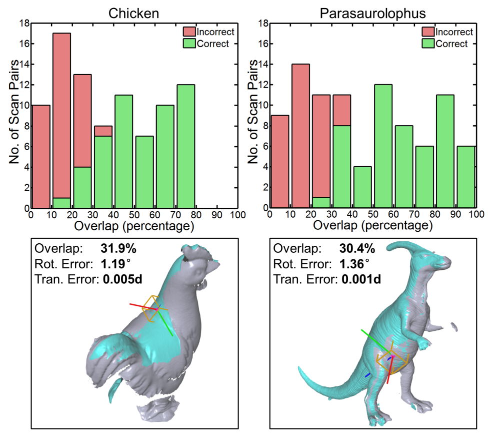


Figure 7: *Top: Histograms of correctly and incorrectly aligned scan pairs with respect to the amount of overlap. Bottom: two example alignments when the overlap is small.*

We performed this experiment on two objects in the UWA dataset. Figure 7(top) shows the histograms of correctly and incorrectly aligned scan pairs, in which our method achieved remarkable performance when the overlap is above 30%.

6. Conclusion

This paper proposes a novel *local voxelizer* descriptor for pairwise surface registration. Local voxelizer is constructed by defining a unique LRF for the support around a basis point and performing local voxelization within the LRF. Two scans can be aligned by finding a single pair of matched points using the descriptor and aligning the corresponding LRFs. Experiments show that our algorithm can align scan pairs with 30% overlap or above, with acceptable accuracy.

Limitations. First, our algorithm may not be suitable for registering very noisy scans since it requires surface normal. Second, our algorithm needs to covert an input point cloud into a mesh, in which useful information could be lost.

Acknowledgments

This work was supported by the Fundamental Research Funds for the Central Universities (WK0110000044) and the National Natural Science Foundation of China (61403357).

References

[AMCo08] Aiger D., Mitra N. J., Cohen-or D.: 4-points congruent sets for robust pairwise surface registration. *ACM Trans. on Graphics (SIGGRAPH) 27*, 3 (2008). Article 85. 1

[ART10] Albarelli A., Rodolà E., Torsello A.: Loosely distinctive features for robust surface alignment. In *European Conference on Computer Vision* (2010), pp. 519–532. 2

[BM92] Besl P. J., McKay N. D.: A method for registration of 3-D shapes. *IEEE Trans. on PAMI 14*, 2 (1992), 239–256. 1, 5

[CHC99] Chen C.-S., Hung Y.-P., Cheng J.-B.: RANSAC-based DARCES: a new approach to fast automatic registration of partially overlapping range images. *IEEE Trans. on PAMI 21*, 11 (1999), 1229–1234. 1

[CM92] Chen Y., Medioni G.: Object modelling by registration of multiple range images. *Image and Vision Computing 10*, 3 (1992), 145–155. 1

[FHK*04] Frome A., Huber D., Kolluri R., Bülow T., Malik J.: Recognizing objects in range data using regional point descriptors. In *European Conference on Computer Vision* (2004), pp. 224–237. 2

[GCO06] Gal R., Cohen-Or D.: Salient geometric features for partial shape matching and similarity. *ACM Trans. on Graphics 25*, 1 (2006), 130–150. 2

[GMGP05] Gelfand N., Mitra N. J., Guibas L. J., Pottmann H.: Robust global registration. In *Symposium on Geometry Processing* (2005), pp. 197–206. 1, 2, 3

[GSB*13] Guo Y., Sohel F., Bennamoun M., Lu M., Wan J.: Rotational projection statistics for 3D local surface description and object recognition. *International Journal of Computer Vision 105*, 1 (2013), 63–86. 1, 2

[HDD*92] Hoppe H., DeRose T., Duchamp T., McDonald J., Stuetzle W.: Surface reconstruction from unorganized points. In *SIGGRAPH* (1992), pp. 71–78. 2

[HH03] Huber D. F., Hebert M.: Fully automatic registration of multiple 3D data sets. *Image and Vision Computing 21*, 7 (2003), 637–650. 2

[JH99] Johnson A. E., Hebert M.: Using spin images for efficient object recognition in cluttered 3D scenes. *IEEE Trans. on PAMI 21*, 5 (1999), 433–449. 2

[MAM14] Mellado N., Aiger D., Mitra N. J.: Super 4PCS fast global pointcloud registration via smart indexing. *Computer Graphics Forum (Proc. of SGP) 33*, 5 (2014), 205–215. 1

[MBO06] Mian A. S., Bennamoun M., Owens R. A.: A novel representation and feature matching algorithm for automatic pairwise registration of range images. *International Journal of Computer Vision 66*, 1 (2006), 19–40. 1, 2, 3, 5

[MS05] Mikolajczyk K., Schmid C.: A performance evaluation of local descriptors. *IEEE Trans. on PAMI 27*, 10 (2005), 1615–1630. 3

[PWHY09] Pottmann H., Wallner J., Huang Q.-X., Yang Y.-L.: Integral invariants for robust geometry processing. *Computer Aided Geometric Design 26*, 1 (2009), 37–60. 2

[SFLF15] Song P., Fu Z., Liu L., Fu C.-W.: Printing 3D objects with interlocking parts. *Computer Aided Geometric design (Proc. of GMP) 35-36* (2015), 137–148. 3

[TSS10] Tombari F., Salti S., Stefano L. D.: Unique signatures of histograms for local surface description. In *European Conference on Computer Vision* (2010), pp. 356–369. 1, 2

[Zho09] Zhong Y.: Intrinsic shape signatures: A shape descriptor for 3D object recognition. In *12th International Conference on Computer Vision Workshops* (2009), pp. 689–696. 2