Eurographics Symposium on Geometry Processing (2005)
M. Desbrun, H. Pottmann (Editors)

Robust Global Registration

Natasha Gelfand † Niloy J. Mitra † Leonidas J. Guibas† Helmut Pottmann ‡

† Computer Graphics Laboratory, Stanford University
‡ Geometric Modeling and Industrial Geometry, Vienna University of Technology

Abstract

We present an algorithm for the automatic alignment of two 3D shapes (data and model), without any assumptions about their initial positions. The algorithm computes for each surface point a descriptor based on local geometry that is robust to noise. A small number of feature points are automatically picked from the data shape according to the uniqueness of the descriptor value at the point. For each feature point on the data, we use the descriptor values of the model to find potential corresponding points. We then develop a fast branch-and-bound algorithm based on distance matrix comparisons to select the optimal correspondence set and bring the two shapes into a coarse alignment. The result of our alignment algorithm is used as the initialization to ICP (iterative closest point) and its variants for fine registration of the data to the model. Our algorithm can be used for matching shapes that overlap only over parts of their extent, for building models from partial range scans, as well as for simple symmetry detection, and for matching shapes undergoing articulated motion.

1. Introduction and Background

Global registration, or optimal alignment of two three-dimensional shapes in arbitrary initial positions, is a fundamental problem in shape acquisition and shape modeling. Given two shapes, often called the model and data, the goal is to find a rigid transform that optimally positions, or registers, the data with respect to the model. This process is part of most 3D shape acquisition pipelines, where self-occlusions and scanner limitations usually require the acquisition of multiple partial scans that overlap. To build a complete model, the partial scans need to be brought into a common coordinate system (Figure 1), which is usually done by pairwise registration. This problem is particularly hard when no information is available about the initial position of the model and data shapes, the inputs contain noise, and the shapes overlap only over parts of their extent (and the overlaps may not be known in advance).

Solutions to the registration problem fall into two general classes. One class, known as voting methods, makes use of the fact that the rigid transform is low-dimensional and exhaustively searches for the small number of parameters needed to specify the optimal transform. Generalized Hough transform [HB94], geometric hashing [WR97], and pose clustering [Sto87] quantize the transformation space into a

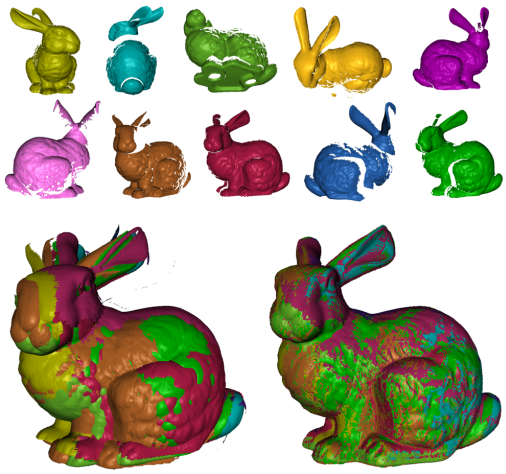


Figure 1: *Automatic registration of range data. Top: 10 input scans (shown here in good position for visualization, the actual input positions are arbitrary). Bottom left: Registration after applying our algorithm to overlapping pairs of scans. Bottom right: Registration after applying ICP and error relaxation to the initial pose produced by our algorithm.*

† This research was supported in part by NSF grants CARGO-0138456 and FRG-0454543, ARO grant DAAD19-03-1-033, a Max Planck Institut fellowship and a Stanford Graduate Fellowship.
‡ This research was supported by the Austrian Science Fund under grant P16002-N05.

six dimensional table. For each triplet of points in the model shape and each triplet in the data shape, the transformation between the triplets is computed and a vote is recorded in the corresponding cell of the table. The entry with the most votes gives the optimal aligning transform. Another variant of this scheme, the alignment method [HU90], tallies for each transform proposed by two triplets of points how many points of the data are brought by the transform close to a point in the model. The transform which brings the most data points within a threshold of a point in the model is chosen as the optimal aligning transform. Voting methods are guaranteed to find the optimal alignment between the data and model shapes and are independent of the initial pose of the input shapes. However, since these methods tend to be costly, they are usually not used for global registration of scan data.

The second class of approaches to the registration problem tries to solve the underlying correspondence problem: for each point on the data shape, the goal is to find its corresponding point on the model. Given a set of corresponding point-pairs, a rigid transform can be computed that best positions the two shapes so that the distance between corresponding points is minimized [ELF97]. When the initial positions of the model and data are close, the correspondences and the transform are usually found using a variant of the Iterated Closest Point algorithm (ICP) [RL01, MGPG04]. The algorithm assigns to each point in the data its closest point in the model as a correspondence, computes the aligning transform, and applies it to the data shape. This process is iterated until some convergence criterion is reached. The main limitation of ICP and its variants is that, as a local optimization method, it is not guaranteed to find the globally optimal alignment, and therefore is only effective when the initial position of the input shapes is close to the correct alignment [MGPG04]. In shape acquisition systems, the input scans are usually positioned manually, and then registered using ICP.

Both the voting schemes and the correspondence search can be improved by using geometric descriptors. A geometric descriptor is a quantity computed for each point of the model and the data, based on the shape of the local neighborhood around the point. Points whose descriptors are similar potentially correspond. High-dimensional, or rich, descriptors such as spin images [JH99] and shape contexts [BMP02] provide a fairly detailed description of the shape around each point in transformation-independent manner. The advantage of rich descriptors is that given a point in the data shape, it is likely that only a few points in the model shape will have a similar descriptor, and the point with the best-matching descriptor is likely to be the correct corresponding point. Incorrect correspondences are few and can be removed using outlier detection methods [FB81], which means that rich descriptors can be used to directly solve the correspondence problem. Huber [HH03] uses spin images computed from subsampled input data for automatic global registration of range data. Rich descriptors are particularly popular for object recognition and shape retrieval, where the computation of descriptors can be amortized over large number of comparison queries [BMP02, FMK*03].

Low-dimensional descriptors, on the other hand, usually compute only a few values per point. Examples of such descriptors include curvature and various curvature-based quantities such as shape index [Koe90] and curvedness [KD92]. Low-dimensional descriptors are typically much easier to compute, store, and compare than high-dimensional rich descriptors. However, for a given point in the data shape, there may be many points in the model shape with the same descriptor value. Therefore, low-dimensional descriptors are usually used in conjunction with a voting scheme [BS97] to reduce the size of the search space or with an iterative alignment scheme to improve the funnel of convergence (set of starting positions which result in correct alignment) [SLW02, GLB99].

Since the inputs to the registration algorithm are usually large, a common speedup technique is to pick a set of feature points on the model and data based on the computed descriptor values [MKY01]. The registration is then performed only with respect to the feature points, which results in significant reduction of the size of the search space. Feature extraction methods, however, can suffer from the problem of picking inconsistent points on the model and data, since the two shapes are processed separately. The resulting set of feature points, therefore, may not have a good alignment. Because of possible errors in feature selection, correspondence assignment techniques based on geometric descriptors usually build large correspondence sets to increase robustness to incorrect features and pairings. Therefore, these methods, unlike the voting schemes, do not make use of the low-dimensionality of the aligning rigid transform.

Contributions. In this paper, we develop a new global registration algorithm, based on robust feature identification and correspondence search using geometric descriptors. The main contributions of our method are as follows:

- Our algorithm makes use of the fact that the aligning transform is low-dimensional to robustly find a small set of matching point-pairs that specify the optimal alignment.
- We focus on identifying a small number of feature points on the data shape, and then searching the entire model shape for correspondences. This approach avoids the problem of selecting incompatible features that is common in other feature-based registration methods.
- We use a novel shape descriptor, based on performing integral operations on the underlying shape, for identifying features in the data and selecting potential correspondence points in the model. Our feature selection algorithm picks points on the data shape which have uncommon descriptor values across a range of scales.
- For each feature point, the correspondence search algo-

rithm examines the entire model shape to identify the optimal corresponding point. The search is made efficient by using a measure of quality of correspondences based on computing only intrinsic quantities of the model and data shapes. This allows us to avoid computing an aligning transformation, and results in an efficient branch-and-bound algorithm. Additionally, we use the rigid transform constraints for efficient pruning of the search space.

- Since our algorithm only uses descriptor values, which are invariant under rigid transforms, and intrinsic geometric properties of the input shapes, we are able to align the model and data shapes without any assumptions about their initial position. The pose produced by our algorithm is further refined by ICP, producing an automatic global registration pipeline.

2. Integral Volume Descriptor

Let $\mathbf{P}$ be the input shape, consisting of N points $\mathbf{p}_1 \dots \mathbf{p}_N$. The input can be specified as a mesh or as a point cloud. An m-dimensional geometric descriptor is a function that assigns to each point $\mathbf{p} \in \mathbf{P}$ a vector $f(\mathbf{p}) \in \mathbb{R}^m$. To be useful in registration algorithms, a descriptor should be invariant under rigid transformations, robust to noise, and based on local geometry around $\mathbf{p}$ (since the input shapes may be only partially overlapping). We will restrict our attention to low-dimensional descriptors, since they are cheaper to compute, store, and compare than rich descriptors.

Most of the common low-dimensional shape descriptors are based on differential quantities of the shape, since they are invariant under rigid transformations. The main limitation of differential descriptors, which has made them unpopular in registration algorithms, is that any noise present in the input gets amplified when derivatives are computed. As a result, algorithms that rely on differential descriptors need to perform careful smoothing of both data and model shapes.

An alternative approach, that has yielded promising results in object recognition and feature classification, is to use local shape invariants that are based on integration instead of differentiation [MHYS04, CRT04]. Integral descriptors retain the desirable properties of differential invariants such as locality and invariance under rigid transformations, but are more robust to noise. Manay et al. [MHYS04] showed that integral invariants have descriptive power comparable to curvature-based descriptors, but are more effective in 2D object recognition in the presence of noise. In this section, we extend the integral invariants of [MHYS04] to 3D.

2.1. Definition of Descriptor

We develop a 3D integral invariant, called the *integral volume descriptor*. This invariant is defined at each vertex $\mathbf{p}$ of the input shape as follows,

$$V_r(\mathbf{p}) = \int_{B_r(\mathbf{p}) \cap \overline{S}} dx. \tag{1}$$

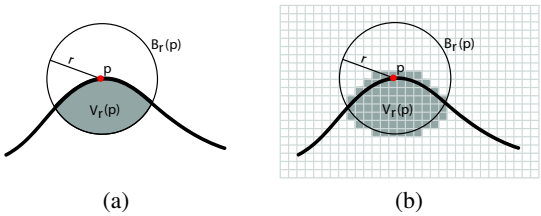


Figure 2: *Illustration of the volume integral descriptor in 2D. (a) We take the intersection of a ball of radius r centered at point* $\mathbf{p}$ *with the interior of the surface. (b) Discretization of the volume descriptor as computed by our algorithm. The cell size of the grid is* ρ.

Here the integration kernel $B_r(\mathbf{p})$ is a ball of radius r centered at the point $\mathbf{p}$, and $\overline{S}$ is the interior of the surface represented by $\mathbf{P}$. The quantity $V_r(\mathbf{p})$ is the volume of the intersection of the ball $B_r(\mathbf{p})$ with the interior of the object defined by the input mesh. The invariant is illustrated in 2D in Figure 2(a). Assuming the intersection of S and $B_r(\mathbf{p})$ is simply-connected, the volume descriptor is related to mean curvature at $\mathbf{p}$ as follows,

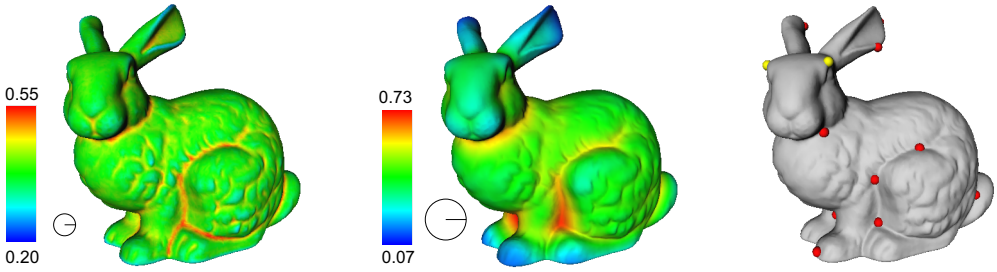
$$V_r(\mathbf{p}) = \frac{2\pi}{3} r^3 - \frac{\pi H}{4} r^4 + O(r^5). \tag{2}$$

The leading term is the volume of the half-ball of radius r, and the correction term involves the mean curvature H at the point $\mathbf{p}$.

To show that this descriptor is robust to noise, let $\mathcal{P}$ be the patch that bounds the surface where it intersects the ball $B_r(\mathbf{p})$. Noise causes a perturbation that moves $\mathbf{p}$ to $\mathbf{p}'$ and thus the kernel ball $B_r(\mathbf{p})$ undergoes a translation to $B_r(\mathbf{p}')$. The latter intersects the perturbed surface in a patch $\mathcal{P}'$. Translating $B_r(\mathbf{p}')$ back to $B_r(\mathbf{p})$ moves $\mathcal{P}'$ to a patch $\mathcal{P}^*$. Apart from a negligible part along the intersection with the ball $B_r(\mathbf{p})$, the change of the value of the volume descriptor is given by the oriented volume ΔV between $\mathcal{P}$ and $\mathcal{P}^*$. Let us assume that $\mathcal{P}$ can be expressed as a parametric surface $\mathbf{s}(u,v)$. We express the perturbation towards $\mathcal{P}^*$ using a displacement field $\tau(u,v)$ in normal direction of each point of $\mathcal{P}$. Then, the change in volume descriptor at $\mathbf{p}$ due to the perturbation can be shown to be

$$\Delta V = \int_{\mathcal{P}} \tau(u,v) dA - \int_{\mathcal{P}} \tau^2(u,v) H \, dA + \frac{1}{3} \int_{\mathcal{P}} \tau^3(u,v) K \, dA, \tag{3}$$

where K is the Gaussian curvature at $\mathbf{p}$. We assume the perturbation noise is independently, identically distributed with mean zero and variance σ^2. Let H_{max} be the maximum mean curvature over the patch $\mathcal{P}$, and $\mathcal{A}$ be the area of the patch, then the expected change in the volume descriptor can be bounded by $|E[\Delta V]| \leq H_{max}\sigma^2\mathcal{A}$. The change in descriptor value relative to the volume of the integration kernel is proportional to σ^2/r^2, which shows the robustness of the descriptor to noise.



(a) (b) (c)

Figure 3: *Values of the normalized volume descriptor computed for the bunny model (a) for small and (b) large integration kernels. We normalize the descriptor value with respect to the volume of the ball B_r to ease visualization (c) Feature points picked by our algorithm on the bunny model. Small-scale features, shown as yellow spheres, are persistent for small values of the integration kernel. Features that are persistent over large kernel radii are shown in red.*

2.2. Implementation

The integral volume descriptor can be computed efficiently by a convolution of the occupancy voxel grid G_O with the ball grid G_B (Figure 2(b)). $G_O(c) = 1$ if the voxel c lies inside the shape or intersects the boundary, and is zero otherwise. The ball grid G_B contains the scan-converted ball of radius r placed at the center of the grid. The value of the volume descriptor at each cell c is given by $V(c) = (G_B * G_O)(c)$. The convolution can be computed efficiently by using the Fourier transform of the ball grid and the occupancy grid.

The occupancy grid G_O can be computed using scan conversion algorithms for meshes [NT03] or ray shooting algorithms for point clouds [AA03]. We set the grid size ρ to be large enough to account for the perturbation of the vertices due to noise. Once the convolution is computed, the value of the volume descriptor at each vertex of the input shape is given by the value of the descriptor at the voxel containing the vertex.

Holes in the input surfaces affect the value of the volume descriptor of all cells that lie within distance r of the hole, since they result in gaps in the occupancy grid. Therefore, we modify our descriptor computation to be robust to gaps in the surface. If the hole is small, we fill it by performing a dilation of the occupancy grid by a few voxels, followed by a contraction. If desired, other more expensive hole-filling methods [DMGL02, NT03] can be used, however, the dilation method is fast and effective in filling small holes which are common in range data due to noise in the scanning.

Integral descriptors are particularly suited for multiscale representation since the scale is given by the radius of the kernel B_r. Figures 3 (a) and (b) show the volume descriptor computed for the Stanford Bunny model for two different ball radii. In Section 3.2 we describe an algorithm that uses the multiscale representation of the volume descriptor to robustly identify persistent features.

3. Feature Point Selection

Our registration algorithm is based on finding correspondences in the model $\mathbf{Q}$ for a small number of feature points picked from the data shape $\mathbf{P}$. The features are selected in such a way that makes the search for correspondences particularly simple. The key property of our feature selection algorithm is that feature points should come from regions with rare descriptor values. Since the data and model shapes are similar over the matching region, and we use descriptor values to select potential corresponding points in the model for each feature point in the data, points with rare descriptor values are likely to have only a few potential correspondence points. Thus, our feature selection algorithm specifically picks points such that the resulting search for correspondences will be fast. Additionally, we do not need to select many points as features, since a rigid transform can be specified using only a small number of points. Selecting a small number of feature points, such that each will have only a small number of potential correspondences results in a tractable correspondence search problem.

We first describe an algorithm for a general descriptor, and then show how we use the scale-space representation of the volume descriptor together with a persistence [CZCG04] algorithm to robustly select feature points at multiple scales.

3.1. Basic Algorithm

Let f be the geometric descriptor which associates with each point $\mathbf{p}_i$ a value $f(\mathbf{p}_i)$. The descriptor can be of any dimension, in this section we assume that the descriptors are one-dimensional $f(\mathbf{p}_i) \in \mathbb{R}$. A point $\mathbf{p}$ is defined to be a feature if its descriptor value is rare among all descriptors computed for the data shape $\mathbf{P}$. The feature point selection proceeds as follows:

1. Compute a histogram of descriptor values, $f(\mathbf{p}_i)$ for all points in $\mathbf{P}$. The number of bins b in the histogram is computed using Scott's rule, $b = 3.49\sigma_f N^{-\frac{1}{3}}$, where σ_f is the standard deviation of the N descriptor values [Sco79].

2. To select feature points, we identify the k least populated bins such that the total number of points in these bins is smaller than some maximum threshold s. The points that belong to these bins are the potential features. Intuitively, features are those points which are dissimilar from the rest of the shape, which is captured by the low occurrence of their descriptor values. The parameter s controls the tradeoff between accuracy of the transform (more correspondences) and running time of the algorithm. In our implementation, we set $s = 0.01N$.
3. Since nearby points are likely to belong to the same feature, we want to prevent the algorithm from picking points that are too close to each other. We also want the points to cover the whole shape since in case of partial matching we do not know a priori which part of the data shape will overlap with the model. When a point $\mathbf{p}_i$ is picked, we mark all points that fall into a ball of radius R_e around $\mathbf{p}_i$ as unavailable for selection. Enforcing the minimal separation distance between the feature points also results is more stable configurations in the correspondence search stage of the algorithm (Section 5).

Notice that this process is not invariant to the order of points in $\mathbf{P}$. This means that it cannot be used to pick canonical points on the data and model shapes. As mentioned before, we do not rely on feature points being canonical, since we will search the entire model shape for correspondences, instead of trying to match up canonical points. This means, as long as a feature point lies in the overlap region between the model and data, it will have a correspondence assigned to it by the matching stage of our registration algorithm.

The above algorithm works with any kind of descriptor which can be represented as a vector in $\mathbb{R}^m$. Since we are picking as features those points of $\mathbf{P}$ that have uncommon descriptor values, we need a descriptor that is robust to noise, making integral descriptors particularly well suited for this kind of approach. Figure 5(b) shows the feature points selected on the dragon model corrupted with zero-mean Gaussian noise.

3.2. Scale-space Representation and Persistent Features

We further improve the robustness of the feature selection algorithm by considering the scale-space representation of the descriptor. Integral descriptors are particularly well suited for such approach, since the scale is naturally controlled by the radius of the integration kernel B_r, but other scale-space representations, such as curvature scale space [MKY01, PKG03] can also be used.

If a point $\mathbf{p}$ is an actual feature point, it should be present over a set of consecutive scales of the descriptor. A point that is an outlier, on the other hand, is likely to disappear as a feature as the scale is varied. Therefore, we use the persistence [CZCG04] of a feature point in the scale-space representation to identify true features and discard outliers.

Most shapes contain features at different scales, so we do not expect a point to be a feature over the entire scale-space of the descriptor. Instead, we define as a feature a point whose descriptor value is rare over a set of consecutive kernel radii of the volume descriptor. Small scale features will be persistent for small radii of the descriptor and large-scale features over large radii, and outliers may look like features for some radii but are not persistent. In addition to identifying a point as a feature, our algorithm automatically identifies the scale of the feature.

To implement the persistence algorithm, we sample the scales of the volume descriptor at discrete intervals. We divide the range $r_{min} \leq r \leq r_{max}$ of possible ball radii into k intervals ($k = 5$ in our implementation) and convolve the occupancy grid with the different ball grids. To avoid discretization errors, r_{min} is set to 10ρ, where ρ is the resolution of the voxel grid. We also limit r_{max} to some fraction (usually set to 0.1) of the diameter of the input shape to preserve the locality property of the shape descriptor. We also normalize the magnitude of the volume descriptor for each radius r by the volume of the ball B_r. For a point $\mathbf{p}$ to be a feature, it should be selected as a feature for a set of continuous scales. We use the basic algorithm described in Section 3.1 to identify feature points for each radius of the volume descriptor, and then select only those points that are a feature for at least two consecutive radii.

Figure 3 shows feature points selected for the bunny model using the scale-space algorithm. Most of the features are persistent over large radii in the scale-space representation, except for features near the eyes, which are persistent only over small scales.

The result of the feature selection stage is a set of feature points $\mathbf{P}'$ selected from the data shape. For each feature, we are given the coordinates of the point, scale-space representation of the volume descriptor, and the range of radii for which this point was classified as a feature. We now develop an algorithm that finds, for each feature point on the data, a corresponding point on the model.

4. Distance Metrics

Given a set of n feature points $\mathbf{P}'$ selected from data shape $\mathbf{P}$, the goal of the correspondence search algorithm is to find, for each $\mathbf{p}_i \in \mathbf{P}'$, a point $\mathbf{q}_i \in \mathbf{Q}$, that is the best match to $\mathbf{p}_i$. Let $\mathbf{P}'$ and $\mathbf{Q}'$ be two sets of points with correspondences given as $(\mathbf{p}_i, \mathbf{q}_i)$. We present two ways of evaluating the quality of the correspondence based on the coordinates of the points $(\mathbf{p}_i, \mathbf{q}_i)$.

The standard measure of distance between two point sets with known correspondences is the *coordinate root mean squared error*, or cRMS. This error measures how close each point $\mathbf{p}_i$ comes to each corresponding point $\mathbf{q}_i$ after an optimal rigid aligning transform is computed for the entire set of

corresponding points.

$$\text{cRMS}^2(\mathbf{P}', \mathbf{Q}') = \min_{\mathbf{R},\mathbf{t}} \frac{1}{n} \sum_{i=1}^{n} ||\mathbf{R}\mathbf{p}_i + \mathbf{t} - \mathbf{q}_i||^2, \quad (4)$$

where $\mathbf{R}$ is a rotation matrix and $\mathbf{t}$ is a translation vector. The optimal aligning rigid transformation needs to be computed before the error can be evaluated, which can be done using one of the methods described in [ELF97]. The cRMS distance metric is the most frequently used measure of residual error in registration algorithms.

An alternative metric of distance between two point sets with known correspondences is the *distance root mean squared error*, or dRMS. This metric is commonly used in computational molecular biology for comparing the similarity of two protein shapes [Koe01]. The dRMS error is computed by comparing all internal pairwise distances of the two point sets, and is defined as

$$\text{dRMS}^2(\mathbf{P}', \mathbf{Q}') = \frac{1}{n^2} \sum_{i=1}^{n} \sum_{j=1}^{n} (||\mathbf{p}_i - \mathbf{p}_j|| - ||\mathbf{q}_i - \mathbf{q}_j||)^2. \quad (5)$$

The triangle inequality and the property that the optimal transform aligns the centroids of $\mathbf{P}'$ and $\mathbf{Q}'$ allows us to upper bound dRMS using cRMS as follows,

$$\text{dRMS}(\mathbf{P}', \mathbf{Q}') \leq \sqrt{2}\ \text{cRMS}(\mathbf{P}', \mathbf{Q}'). \quad (6)$$

To compute the lower bound, we need to examine both $\mathbf{Q}'$ and its reflection around any arbitrary plane $\overline{\mathbf{Q}'}$ (since dRMS is invariant under reflection, but cRMS is not). The lower bound can be shown to be

$$\frac{1}{k\sqrt{n}} \min(\text{cRMS}(\mathbf{P}', \mathbf{Q}'), \text{cRMS}(\mathbf{P}', \overline{\mathbf{Q}'})) \leq \text{dRMS}(\mathbf{P}', \mathbf{Q}'). \quad (7)$$

Here n is the number of corresponding point pairs and k is a small constant, depending on ratio of the diameter of the data shape to the feature exclusion radius used in Section 3. These bounds mean than when the dRMS of two point sets is small, their cRMS will also be small (when there is no reflection), indicating that the point sets are in good alignment. Therefore, we can use dRMS instead of cRMS to evaluate how well two point sets correspond.

dRMS has the advantage that it does not require computation of the aligning transform before the quality of the correspondence can be evaluated. It is, in fact, only comparing intrinsic properties of the two sets of corresponding points, namely the internal pairwise distances of each pointset, as opposed to comparing the distances between the two point sets. This means that, given the set of feature points $\mathbf{P}'$, its pairwise distance matrix needs to be computed only once, and then compared to pairwise distance matrices of the potential correspondence sets $\mathbf{Q}'$. Additionally, since only intrinsic properties of the point sets are examined in dRMS computation, we will be able to efficiently prune correspondence sets that contain wrong matches without having to compare the entire sets $\mathbf{P}'$ and $\mathbf{Q}'$. This will allow us to develop an efficient branch-and-bound algorithm described in the next section.

5. Correspondence Search

5.1. Computing Potential Correspondences

Let $\mathbf{P}'$ be the set of n points picked by the feature selection algorithm. For each feature point $\mathbf{p}_i$ we also have the scale-space representation of the volume descriptor $(V_{r_1}(\mathbf{p}_i), \ldots, V_{r_k}(\mathbf{p}_i))$, and the values r_a^i, r_b^i, which are the minimum and maximum radii of the kernel of the volume descriptor for which $\mathbf{p}_i$ is a persistent feature. We now use the descriptor values to select potential corresponding points in the model shape $\mathbf{Q}$ for each feature point.

For our descriptor-based matching algorithm, we first compute the same scale-space representation of the volume descriptor on the model shape. That is, we compute volume descriptors for radii $r_1, r_2, \ldots, r_k$ for each point on the model shape. Let $\mathbf{p}$ be a feature point selected from the data shape, and let r_b be the largest feature radius. We perform a range query in the model, and select all points $\mathbf{q}$ such that $|V_{r_b}(\mathbf{p}) - V_{r_b}(\mathbf{q})| < \varepsilon$. We can also perform the range query for any radius between r_a and r_b of $\mathbf{p}$, however we prefer the largest possible radius since it gives the most stable descriptor. The variation of the descriptor values ε can be related to the grid size ρ and the radius of the volume descriptor r as $\varepsilon \approx \frac{3\rho}{4r}$. This accounts for the variation in the value of the volume descriptor due to discretization using the voxel grid. Since we pick ρ to be large enough to account for noise in the data, therefore ε also absorbs the noise term in Equation 3.

The range query results in the set of points $C_{initial}(\mathbf{p})$ whose volume descriptor for the given radius is similar to the descriptor value at $\mathbf{p}$. Similar to the approach in the feature selection algorithm, we want to pick a set of points that represent distinct areas of the model. We cluster all points in $C_{initial}(\mathbf{p})$ into clusters of radius R_c and pick from each cluster the point $\mathbf{q}$ that minimizes $|V_{r_b}(\mathbf{p}) - V_{r_b}(\mathbf{q})|$. This gives the final set of correspondences for $\mathbf{p}$, $C(\mathbf{p})$. We repeat this procedure for each point in the feature set.

Using a range search instead of exact match of the descriptor values ensures that the correct correspondence of $\mathbf{p}$ is included in the set $C_{initial}(\mathbf{p})$ (under a reasonable noise model). After clustering, we are guaranteed that the correct correspondence is within R_c of a point in $C(\mathbf{p})$. It follows that the correct set of corresponding points of $\mathbf{P}'$ has cRMS at most R_c, and dRMS is bounded by $\sqrt{2}R_c$. The value of R_c, therefore, is a knob that controls the quality of the resulting registration.

5.2. Matching Algorithm

Even though we have a comparatively small number of feature points, and each feature point has a small number of potential correspondences, exhaustive exploration of the space

of all correspondences can still be prohibitively expensive. The key observation that will allow us to develop a fast algorithm is that we can use the rigidity constraints of the aligning transform to efficiently eliminate a large set of potential correspondences.

Given a set of feature points $\mathbf{P}' = (\mathbf{p}_1, \ldots, \mathbf{p}_n)$ selected from the data shape, and a set of potential correspondences for each point in the model shape $(C(\mathbf{p}_1), \ldots, C(\mathbf{p}_n))$, we want to select a set of points $\mathbf{Q}'$ such that $\mathbf{q}_i \in C(\mathbf{p}_i)$ and the error metric of Equation 5 is minimized over all sets of such correspondences. Since we will only be considering points in $\mathbf{Q}$ that belong to some potential correspondence set, we will change the notation slightly in this section to simplify the explanation of the algorithm. Given a feature point $\mathbf{p}_i$, we will designate the j-th member of the potential correspondence set $C(\mathbf{p}_i)$ as $\mathbf{q}_i^j$.

Consider a pair of feature points $(\mathbf{p}_i, \mathbf{p}_j)$. According to their descriptor values, any pair of points $(\mathbf{q}_i^k, \mathbf{q}_j^l)$ can be used as corresponding points. Rigid transform constraints tell us that the distance between $\mathbf{p}_i$ and $\mathbf{p}_j$ needs to be the same as the distance between their correspondences in the model. Since we are using correspondences that are only approximate within the clustering radius R_c, the correspondence pairs need to satisfy the relationship

$$\left| ||\mathbf{p}_i - \mathbf{p}_j|| - ||\mathbf{q}_i^k - \mathbf{q}_j^l|| \right| < 2R_c. \tag{8}$$

We apply this thresholding rule in a branch-and-bound algorithm for finding the best set of correspondences. Let $\mathbf{Q}' = (\mathbf{q}_1^*, \ldots, \mathbf{q}_n^*)$ be the current best set of correspondences for the set of feature points $\mathbf{P}'$, and let $E_{min} = \text{dRMS}(\mathbf{P}', \mathbf{Q}')$ be the error of the current best correspondence set. We initialize the set of correspondences using a greedy algorithm described in Section 5.3. The branch-and-bound correspondence search proceeds as follows:

1. Assume corresponding points have been assigned for the first $k-1$ feature points, which gives us a partial correspondence set $(\mathbf{q}_1^{c_1}, \ldots, \mathbf{q}_{k-1}^{c_{k-1}})$. We are looking for the correspondence for the k-th feature point.
2. **Threshold:** For each potential correspondence of $\mathbf{p}_k$, apply the thresholding test of Equation 8 with respect to all previously selected points. That is we verify that Equation 8 holds for all pairs $(\mathbf{p}_i, \mathbf{p}_k), (\mathbf{q}_i^{c_i}, \mathbf{q}_k^j)$ for $i = 1, \ldots, k-1$. If one of the tests fails, we can prune the branch that includes the correspondence pair $(\mathbf{p}_k, \mathbf{q}_k^j)$.
3. **Prune:** For each $\mathbf{q}_k^j$ that passes the thresholding test, form the partial correspondence $(\mathbf{q}_1^{c_1}, \ldots, \mathbf{q}_{k-1}^{c_{k-1}}, \mathbf{q}_k^j)$ and evaluate the dRMS error of this partial correspondence. If the partial error is greater than the error of the current best estimate E_{min}, discard $\mathbf{q}_k^j$ as a correspondence.
4. **Branch:** For each of the remaining $\mathbf{q}_k^j$ that pass both the thresholding and the pruning tests, assign $c_k = j$, and recursively invoke Step 1. Once all correspondences for $\mathbf{p}_k$ have been examined, we backtrack and assign the next correspondence to the previous point $\mathbf{p}_{k-1}$.
5. **Bound:** If all feature points have been assigned correspondences, compute the error of the match E. If the dRMS error is less than E_{min}, we potentially have a better correspondence set, and a new bound, unless the current assignment is actually a reflection. We can rule out reflection by making sure the cRMS error of the current correspondence set is also small. If the cRMS error check passes, we assign $E_{min} = E$ and $\mathbf{Q}' = (\mathbf{q}_1^{c_1}, \ldots, \mathbf{q}_n^{c_n})$.

The branch-and-bound algorithm is possible because we are using the dRMS error metric, which can be computed for partial correspondences without the need for the optimal aligning transform. The only time when the aligning transform in computed is in the last step, and only if we need to update the bound.

5.3. Greedy Bound

The initial correspondence and error bound is established using a hierarchical greedy algorithm. The algorithm first finds the best correspondences for each pair of feature points. Then it combines the pairs to form best corresponding sets of four points, then combines fours into eights and so on.

1. **Form pairs:** For each pair of feature points $(\mathbf{p}_i, \mathbf{p}_j) \in \mathbf{P}'$, choose the best pair of corresponding points $(\mathbf{q}_i^k, \mathbf{q}_j^l)$ in their associated potential correspondence sets. The best matching pair of correspondences is one that minimizes the distance metric penalty $\left| ||\mathbf{p}_i - \mathbf{p}_j|| - ||\mathbf{q}_i^k - \mathbf{q}_j^l|| \right|$. This gives us the set E_2 of $O(n^2)$ two-point correspondences. We sort E_2 in order of increasing distance discrepancy.
2. **Combine pairs:** Combine two-point correspondences into four-point correspondences. Given a two-point correspondence $e \in E_2$, find the two-point correspondence in E_2 that does not contain any of the points of e, and that minimizes the dRMS error of the resulting four-point correspondence. Remove from E_2 all correspondences that have the same endpoints as the new four-point correspondence, and continue until the set E_2 becomes empty. Call this set E_4, and again sort it by increasing dRMS error.
3. **Build hierarchy:** We continue merging in this manner, merging pairs of elements of a set E_k to form the set E_{2k}. We typically stop at either E_8 or E_{16}.
4. **Assign the rest of the points:** We pick the correspondence from the resulting set E_k that has the smallest dRMS error. We use this partial (8 or 16 point) correspondence to compute the rigid transform $(\mathbf{R}, \mathbf{t})$ that minimizes the cRMS error (Equation 4) and apply it to the entire feature point set $\mathbf{P}'$. For all points in $\mathbf{p}_i \in \mathbf{P}'$ that do not yet have correspondences, we assign the point $\mathbf{q}_i^j \in C(\mathbf{p}_i)$ that is closest to $\mathbf{R}(\mathbf{p}_i) + \mathbf{t}$. We use this as the initial correspondence $(\mathbf{P}', \mathbf{Q}')$ and initialize E_{min} to $\text{dRMS}(\mathbf{P}', \mathbf{Q}')$ in the algorithm described in Section 5.2.

This approach is greedy because each step picks the best correspondences to merge together and never backtracks. Therefore is it possible that an incorrect correspondence is found for $\mathbf{P}'$. However, as long as some points are matched to their correct corresponding points in Step 1, the algorithm tends to produce a tight bound that greatly speeds up the basic branch-and-bound algorithm. In practice, this approach often results in a very good guess of the correct alignment, resulting in effective pruning in the branch-and-bound algorithm.

5.4. Partial Matching

When the model and data shapes overlap only over part of their extent, not all the feature points picked on the data will have corresponding points in the model. Therefore, we modify our matching algorithm to handle such partial matches.

In addition to performing the search over all correspondences, we also need to find the subset of the feature points that are the same in the model and data. We augment the set of potential correspondences for each point, $C(\mathbf{p}_i)$, with the not present value $\emptyset$. When a point is assigned $\emptyset$ as a correspondence, it does not contribute to the computed dRMS error. We want to maximize the number of feature points that get assigned a valid corresponding point in the model, while still keeping the dRMS error of the correspondence set low.

Suppose we know that k feature points are missing from the model, but do not know which k. We can run our correspondence search algorithm, but prune away any branch that has more than k points assigned the $\emptyset$ correspondence. This will select the best $n-k$ feature points that have the best correspondences. Since we do not know k, we can run the same algorithm for k ranging from 0 to $n-3$ (since only three points are needed to specify a rigid transform). For robustness, we actually require at least 5 points to have a valid correspondence. We can detect the maximum k since the error will sharply decrease once $n-k$ reaches the correct number of common feature points. Figure 6(d) shows the dRMS error vs. the number of matched feature points for the David model.

6. Results

6.1. Object Registration

We applied our algorithm to a number of registration problems. Although in the examples the model and data shapes are shown in similar positions, the reader should keep in mind that our algorithm does not depend on any assumptions about the initial positions of the input shapes, and the input shapes were given to our algorithm in arbitrary positions. Timing results for the experiments are given in Figure 4.

In the first example, we use the algorithm for whole object alignment in the presence of significant noise. We align the dragon model to a copy of itself corrupted by zero-mean Gaussian noise. Figure 5 shows the results. Our alignment brings the data (noisy) shape close enough to the model (smooth) shape that applying one iteration of standard ICP with point-point error metric [RL01] brings the shapes into exact alignment.

	model size	selection time	num features	corr time	num corr
Dragon	29,455	6.3	38	2.2	9
David	68,480	84.5	15	35.7	6
Bunny	35,000	21.8	11	13.9	4
Part	20,002	5.9	13	15.7	8
Hinge	45,311	19.0	30	1.2	12

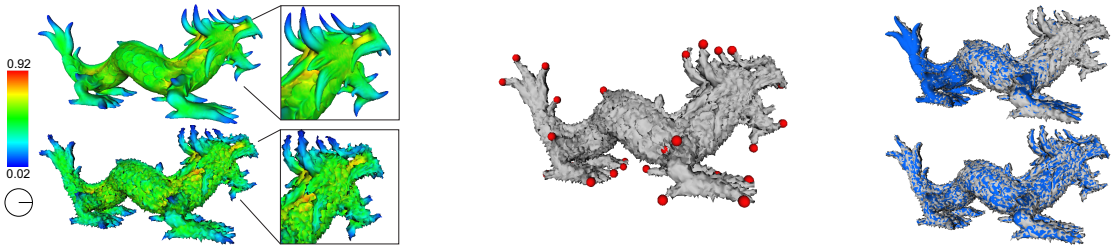
Figure 4: *Input size, running time (in sec), and number of feature points for the registration experiments. In all cases the model size and data size are similar, so we only give the size of the model. The feature selection time includes descriptor computation for both data and model. We also indicate the number of selected feature points and average number of potential correspondences ($|C(\mathbf{p})|$) for each point.*

Figure 6 shows the results of applying our algorithm to register partially overlapping range data. We take two raw scans of the David's face, subsample them, and convert to a mesh representation. We do not perform any other smoothing or surface reconstruction. The scans are given in arbitrary initial positions (scanner coordinates) and brought into close alignment by our algorithm. The pose computed by our algorithm is refined by running three iterations of ICP. Fifteen feature points were picked on the data shape, eight of which were assigned correspondences and used to compute the alignment.

Finally, we use our algorithm to build a complete model out of constituent range scans. Given as input ten range scans of the Stanford bunny taken from different view points, we bring all scans to a common coordinate frame using our algorithm. The rough alignment accumulates errors since we align each scan only to one other, and do not perform any bundle adjustment. However, the scans are now close enough to refine the pairwise matches using ICP, and diffuse the accumulated error over all scans using a global adjustment algorithm [Pul99]. This gives us a completely automatic model construction pipeline. The result is shown in Figure 1.

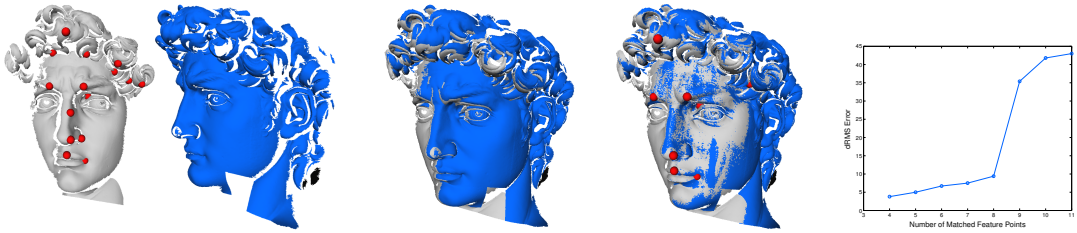
6.2. Symmetry Detection

Our registration algorithm can be trivially extended to detect symmetry in objects by matching an object to a copy of itself. Instead of returning the best matching orientation, we return all matches with small error. Since the feature points picked by our algorithm are spaced far apart, the difference between the symmetry configurations and other matches will be large. Figure 7 shows the results of detecting symmetries



(a) (b) (c)

Figure 5: *Dragon example. (a) Input to the matching algorithm: Smooth dragon (the model) and noisy dragon (the data) with descriptor values shown at each point. Even under noise the descriptor values at feature points look similar. (b) Feature points picked on the data shape. (c) Top: Registration after applying our algorithm. Bottom: Registration after refinement by ICP.*



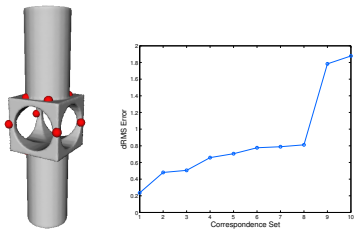
(a) (b) (c) (d)

Figure 6: *(a) Two scans of the David's face. Feature points picked on the data shape are shown in red.(b) Registration after applying our algorithm. (c) Registration after refinement by ICP. Points actually used to compute alignment in (b) are shown in red. (d) Graph of dRMS error as the function of the number of matched features. Notice the significant increase in error for more than 8 points, which is the correct number of common features.*

of a mechanical part. Notice that the graph of error in Figure 7 shows eight configurations with small error, which corresponds to the eight-way symmetry of the model. We expect that this method can be extended to be able to detect partial symmetries in the shape, which is not possible using existing methods for symmetry detection [KFR04].

6.3. Articulated Matching

Our global registration algorithm can be used to discover rigid parts in objects that undergo articulated deformation.



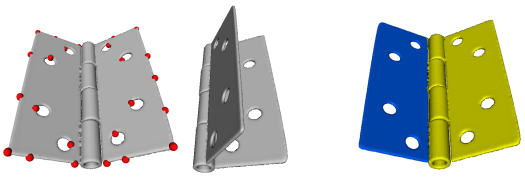
(a) (b)

Figure 7: *Symmetry detection using registration. (a) Feature points picked by our algorithm, when the shape is aligned to a copy of itself. (b) Graph of the error for different correspondence sets. The eight correspondences with small error indicate the eight-way symmetry of the shape.*

In this case, $\mathbf{P}$ and $\mathbf{Q}$ are two positions of the object. We want to decompose the shape $\mathbf{P}$ into the minimum set of parts $\mathbf{P}_1 \ldots \mathbf{P}_k$, such that each $\mathbf{P}_i$ can be aligned to a part of $\mathbf{Q}$ using a rigid transform. Here, we present a simple proof of concept implementation.

We perform articulated decomposition by partial matching of $\mathbf{P}$ and $\mathbf{Q}$. This gives the transform $\mathbf{R}_1, \mathbf{t}_1$. We apply the transform to the data shape, and classify all points of the data that fall within a threshold of the model as belonging to component $\mathbf{P}_1$. We then separate $\mathbf{P}_1$ and the corresponding $\mathbf{Q}_1$ from the input shapes and repeat the partial matching algorithm with $\mathbf{P} - \mathbf{P}_1$ and $\mathbf{Q} - \mathbf{Q}_1$. We repeat the process until the size of the residual set becomes too small. Figure 8 shows the result of segmenting a shape into rigid components using this algorithm.

The features picked on the data shape in Figure 8 also point one of the advantages of the non-canonical nature of our feature selection and correspondence search. If a linear feature is present in the input, such as the long edge of the hinge model, our feature selection algorithm discretely samples the edge at intervals given by the exclusion radius R_e. If we were picking and matching features on both data and model shapes, this discrete sampling could potentially result in two sets of points which do not match each other. However, since we only pick features on one shape, the data, and then search the entire model, we always find a compatible set



(a) (b)

Figure 8: *Simple articulated matching. (a) Two input positions of the shape. Feature points picked by our algorithm are shown in red. (b) Using repeated partial matching, the algorithm discovers two rigid components.*

of points (to within the error given by the clustering radius E_c) with which to align the features.

7. Conclusions and Future Work

We presented a global registration algorithm that aligns two three-dimensional shapes without any assumptions about their initial positions. Our algorithm is able to align whole and partially overlapping shapes, and is robust to noisy data. The algorithm works well in the presence of strong point-like features in the input data. In the future, we would like to extend the algorithm to align linear features directly instead of performing point sampling. Additionally, when the input shapes do not have strong features, the correspondence search space examined by the algorithm becomes quite large. However, in this case, the shape is relatively simple, and ICP-like approaches should have large convergence funnels. We would like to study the exact relationship between the size of the features, the performance of global registration, and the performance of ICP to develop an even more complete automatic registration system that works for arbitrary input data with almost no restrictions.

References

[AA03] ADAMSON A., ALEXA M.: Ray tracing point set surfaces. In *SMI* (2003), pp. 272–279. 4

[BMP02] BELONGIE S., MALIK J., PUZICHA J.: Shape matching and object recognition using shape contexts. *PAMI* (2002). 2

[BS97] BAREQUET G., SHARIR M.: Partial surface and volume matching in three dimensions. *PAMI* (1997), 929–948. 2

[CRT04] CLARENZ U., RUMPF M., TELEA A.: Robust feature detection and local classification for surfaces based on moment analysis. In *IEEE Trans. Vis. Comp. Graphics* (2004). 3

[CZCG04] CARLSSON G., ZOMORODIAN A., COLLINS A., GUIBAS L.: Persistence barcodes for shapes. In *Geometry Processing* (2004), pp. 127–138. 4, 5

[DMGL02] DAVIS J., MARSCHNER S., GARR M., LEVOY M.: Filling holes in complex surfaces using volumetric diffusion. In *3D Data Processing, Visualization, and Transmission* (2002). 4

[ELF97] EGGERT D., LORUSSO A., FISHER R.: Estimating 3D rigid body transformations: a comparison of four major algorithms. In *Mach. Vision and Appl.* (1997), pp. 272–290. 2, 6

[FB81] FISCHLER M., BOLLES R.: Random sample consensus: A paradigm for model fitting with appl. to image analysis and automated cartography. *Comm. of the ACM* (1981), 381–395. 2

[FMK*03] FUNKHOUSER T., MIN P., KAZHDAN M., CHEN J., HALDERMAN A., DOBKIN D., JACOBS D.: A search engine for 3d models. *TOG* (2003), 83–105. 2

[GLB99] GODIN G., LAURENDEAU D., BERGEVIN R.: A method for regist. of attributed range images. In *3DIM* (1999). 2

[HB94] HECKER Y., BOLLE R.: On geometric hashing and the generalized hough transform. In *IEEE SMC* (1994), vol. 24. 1

[HH03] HUBER D., HEBERT M.: Fully automatic registration of multiple 3d data sets. *Image and Vision Computing 21*, 7 (2003). 2

[HU90] HUTTENLOCHER D., ULLMAN S.: Recognizing solid objects by alignment with an image. In *IJCV* (1990). 2

[JH99] JOHNSON A., HEBERT M.: Using spin-images for efficient multiple model recognition in cluttered 3-d scenes. In *PAMI* (1999), pp. 433–449. 2

[KD92] KOENDERINK J., DOORN A.: Surface shape and curvature scales. In *Image and Vision Computing* (1992). 2

[KFR04] KAZHDAN M., .FUNKHOUSER T., RUSINKIEWICZ S.: Symmetry descriptors and 3D shape matching. In *Geometry Processing* (2004). 9

[Koe90] KOENDERINK J.: *Solid shape*. MIT Press, 1990. 2

[Koe01] KOEHL P.: Protein structure similarity. In *Current Opinion in Structural Biology* (2001), pp. 348–353. 6

[MGPG04] MITRA N. J., GELFAND N., POTTMANN H., GUIBAS L.: Regist. of point cloud data from a geometric opt. perspective. In *Geometry Processing* (2004), pp. 23–32. 2

[MHYS04] MANAY S., HONG B., YEZZI A., SOATTO S.: Integral invariant signatures. In *ECCV* (2004), pp. 87–99. 3

[MKY01] MOKHTARIAN F., KHALILI N., YUEN P.: Multi-scale free-form 3-d object recognition using 3-d models. *Image and Vision Computing 19*, 5 (2001), 271–281. 2, 5

[NT03] NOORUDDIN F., TURK G.: Simplification and repair of polygonal models using volumetric techniques. *IEEE Trans. on Vis. and Computer Graphics* (2003), 191–205. 4

[PKG03] PAULY M., KEISER R., GROSS M.: Multi-scale feature extraction on point-sampled surfaces. *Computer Graphics Forum 22*, 3 (2003). 5

[Pul99] PULLI K.: Multiview registration for large datasets. In *Proc. 3DIM* (1999). 8

[RL01] RUSINKIEWICZ S., LEVOY M.: Efficient variants of the ICP algorithm. In *3DIM* (2001). 2, 8

[Sco79] SCOTT D.: On optimal and data-based histograms. *Biometrika 66* (1979), 605–610. 4

[SLW02] SHARP G., LEE S., WEHE D.: Icp registration using invariant features. *PAMI 24*, 1 (2002), 90–102. 2

[Sto87] STOCKMAN G.: Object recognition and localization via pose clustering. *CVGI* (1987), 361–387. 1

[WR97] WOLFSON H., RIGOUTSOS I.: Geometric hashing: an overview. In *IEEE Comp. Sci. and Eng.* (1997), pp. 10–21. 1