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

Multi-scale Features for Approximate Alignment of Point-based Surfaces

Xinju Li, Igor Guskov

University of Michigan, Ann Arbor, USA

Abstract

We introduce a novel method for approximate alignment of point-based surfaces. Our approach is based on detecting a set of salient feature points using a scale-space representation. For each feature point we compute a signature vector that is approximately invariant under rigid transformations. We use the extracted signed feature set in order to obtain approximate alignment of two surfaces. We apply our method for the automatic alignment of multiple scans using both scan-to-scan and scan-to-model matching capabilities.

1. Introduction

Surface acquisition methods are becoming popular for many practical applications in manufacturing, art, and design. Often, multiple acquired scans need to be brought into correspondence in order to be combined or compared. The ICP method [CM92, BM92] and its recent modifications [RL01, GIRL03] are very useful for the final high-precision alignment of two surface patches. In order for ICP to converge, the initial relative position of two surfaces needs to be sufficiently close to the truth.

Several methods have been proposed for approximate alignment of partially overlapping surfaces [JH97, YF02, WG02, FHK*04, SLW02, HH03, CHC99]. A popular approach is to compute local shape descriptors at a number of points sampled from each scan, match up points with similar signatures, and use these matches to choose the best alignment transformation. For instance, the spin-images method [JH97] uses a random set of basis points at which a local rotation-invariant point distribution signature is computed. As the complexity of scanned point clouds increases it becomes important to have a principled way to sample a representative set of feature points from the scanned shape. Surfaces acquired with modern scanning techniques can be very complex and can contain distinguishing features at different scales. *The main contribution of this paper is to introduce a multi-scale salient feature extraction algorithm and analyze its performance in the context of approximate surface alignment of point-based surfaces.*

Our approach is inspired by the SIFT method of Lowe [Low04] which uses multi-scale image features in order to perform image matching with excellent results. We follow the pipeline similar to SIFT: first, a set of reliable salient feature points is found by considering the set of extrema for a scale-space representation of a point-based input surface. This produces a set of features with position, normal, and scale attributed to each of them. Using these attributes we compute a signature for each salient feature. Using features whose signatures match, a voting procedure is invoked to perform the surface matching (that is, to find a spatial transformation that brings a set of features of the first object into correspondence with the set of features of the second object).

Figure 11 shows an example of features extracted by our algorithm for two overlapping scans from the Arrigo dataset. The color of the surface is related to the sign of difference between levels at each point, and circles represent features found in each scan. The yellow circles with letters show some features that can be matched for this pair of scans. Scale of each feature is shown by the radius of the circle.

Our method is able to deal both with coarser noisy scans used in the recent work of Huber and Hebert [HH03] and with finely detailed scans. The main emphasis of our work is on creating good quality feature sets, and our pairwise and multiview matching approaches are much simpler than the ones considered in [HH03]. Our matching procedure can potentially deal with changes in scale, however most evaluation was performed with a known scale rigid transform alignment.

Our paper is organized as follows: the following section describes related work. We proceed to define our scale-space feature extraction procedure in Section 3.1. Section 3.3 explores repeatability of extracted features under various data transformations such as resampling of the surface and introduction of noise. We define local surface signatures in Section 4, and use them for surface matching algorithm of Section 5. Our surface alignment results are presented in Section 6.

2. Related work

The idea of using scale-space processing for extracting salient features is well established in image processing [Wit83] [Lin94] [Low04]. There has also been applications of multi-scale analysis of surface curvature distribution for 3D face recognition [HGY*99]. In computer graphics, multiresolution processing of point-based surfaces is an active area of research: robust modeling, rendering, and processing approaches for point and surfel clouds have been recently developed [PKKG03][ABCO*03][PZvBG00]. A recent work of Pauly et al. [PKG03] extracts line features using a scale-space representation similar to ours. The method of [GWM01] extracts meaningful features from point clouds. In our application, we are not concerned that our feature points represent visually important portions of the model. Rather, our interest is in detecting a stable set of feature points that can be used for surface matching. Additionally, the features extracted by our approach are of the radial nature (we do not extract ridge-like structures).

The ability to extract a small but representative set of keypoints is important for efficient surface matching. To be useful the extracted features should be stable (or repeatable) under resampling and rigid transformations. Therefore, invariant characteristics of the surface such as curvature are often used for feature detection. For instance, in Yamany and Farag [YF02], only a set of high curvature points is considered for signature computation. Another approach used in [WG02] is to consider a set of bitangent points. An alternative simple method often used in practice is to uniformly distribute the basis points over the shape [JH97] [FHK*04].

There has been a variety of surface descriptors used for computing feature point signatures [JH97, YF02, WG02, FHK*04, SM92, SLW02]. Frome et al. [FHK*04] evaluate various techniques in the context of recognizing noisy range scans. It is observed that signatures based on point distribution such as spin-images and 3D shape contexts are robust to outlier points and can handle noisy scans. This paper is oriented towards automatic reconstruction of good quality 3D scans that contain little noise. For such scans, the spin-image signatures become less expressive, and may not capture fine variations of the surface detail especially for small scales. Curvature-based signatures are better equipped to handle fine surface detail and can use normal data. Our local signatures are based on the normal field near the surface and are approximately invariant to rigid transformations.

The goal of our work is very similar to the objective in the spin-image based work of Huber and Hebert [HH03] on fully automatic registration of 3D scans. We evaluate our matching method on some of the data sets used in that paper. The results are given in Section 6.

3. Multi-scale surface features

We build a scale-space representation of the input shape, and use the locations of level difference extrema as the salient feature points. This process is similar to finding extrema of the scale-normalized Laplacian of Gaussian in the 2D image case [Low04], and is easy to implement for point-based surfaces. In this section, we start by defining the scale-space for surfel clouds, then introduce our feature point extraction procedure and explore its stability under noise introduction and resampling.

3.1. Scale-space representation for surfel clouds

Our smoothing procedure is based on the point-set surface projection, as described in [AK04]. Given a surfel cloud $\mathcal{P} = \{(\mathbf{p}_i, \mathbf{n}_i)\}_{i\in\mathcal{I}}$, we define the projection operator $Proj_{[\mathcal{P},h]}$ that can map a 3D point $\mathbf{x}$ onto the smoothed version of the shape defined by the surfel cloud $\mathcal{P}$ and the scale parameter $h > 0$. The projection is found by iterative minimization of the following MLS error:

$$E_{[\mathcal{P},h]}(\mathbf{x}) := \sum_{i\in\mathcal{I}} \theta_i^h(\mathbf{x})\langle \mathbf{n}_{[\mathcal{P},h]}(\mathbf{x}), \mathbf{x}-\mathbf{p}_i\rangle^2,$$

where $\mathbf{n}_{[\mathcal{P},h]}(\mathbf{x})$ is the smoothed normal defined based on the input cloud normals:

$$\mathbf{n}_{[\mathcal{P},h]}(\mathbf{x}) := \sum_{i\in\mathcal{I}} \theta_i^h(\mathbf{x})\mathbf{n}_i$$

and the weights θ^h are normalized Gaussian weights

$$\theta_i^h(\mathbf{x}) := \frac{A_i e^{-d^2(\mathbf{x},\mathbf{p}_i)/h^2}}{\sum_{j\in\mathcal{I}} A_j e^{-d^2(\mathbf{x},\mathbf{p}_j)/h^2}}.$$

We introduce the *area weight* A_i for each point: in the case where mesh connectivity information is available, we assign A_i to be the area associated to the vertex (one third of the area of adjacent faces). If no information on the vertex areas is given, we assign $A_i = 1$ for all the points. Introduction of the area weight is important for stable computation of the scale-space for point clouds with non-uniform point distributions (such as the ones produced by an error-based mesh simplification such as [GH97]), this is shown by our experiments in Section 3.3.

Given a particular scale h, we define the smoothed version of the surfel cloud $\mathcal{P}$ by mapping each point $\mathbf{x}_i^h := Proj_{[\mathcal{P},h]}(\mathbf{x}_i)$, and computing its normal as $\mathbf{x}_i^h := \mathbf{n}_{[\mathcal{P},h]}(\mathbf{x}_i)$. We consider a sequence of scales $h_j = h_0 F^j$ for some $F > 1$.

We use $F = 2^{1/4}$ in this paper. The smoothed version of $\mathcal{P}$ at the scale h_j is denoted as $\mathcal{P}_j := \{(\mathbf{p}_i^{h_j}, \mathbf{n}_i^{h_j})\}_{i \in \mathcal{I}}$.

For the efficient computation of $Proj_{[P,h]}$, we perform spatial queries on the surfel cloud data using the KD-tree library of [AMN*98]. We define $b(\mathbf{q},R) \subset \mathcal{I}$ to be the set of indices such that the corresponding points lie within distance R from the point $\mathbf{q}$. Formally, $b(\mathbf{q},R) := \{k \in \mathcal{I} : d(\mathbf{p}_k, \mathbf{q}) < R\}$. This query is always run within the original point cloud. The found index set can then be used to access the smoothed versions of the point cloud.

3.2. Features

We can now describe the salient feature detection procedure. We define the normal difference between two adjacent levels of our multi-scale representation: $d_j(i) := \langle \mathbf{n}_i^j, \mathbf{p}_i^j - \mathbf{p}_i^{j-1} \rangle$, for $j \geq 1$ and $i \in \mathcal{I}$. We call a point i on level j to be a *neighborhood maximum* of d, if

$$\begin{aligned} d_j(i) &> d_{j-1}(i') \quad \text{for all } i' \in b(p_i, Ch_{j-1}), \\ d_j(i) &> d_j(i') \quad \text{for all } i' \in b(p_i, Ch_j) \setminus \{i\}, \\ d_j(i) &> d_{j+1}(i') \quad \text{for all } i' \in b(p_i, Ch_{j+1}). \end{aligned}$$

We use $C = 1/2$ in this paper for smaller scans. The *neighborhood minimum* of d is defined analogously.

In other words, we compare the value of the normal difference at a point to its neighbors within the level as well as to its neighbors on the two neighboring levels of the scale-space hierarchy. If the difference is larger (smaller) than for all of its neighbors within thus constructed neighborhood then the neighborhood maximum (minimum) is detected. The size of the neighborhood grows as we proceed to the coarser scales.

We use the set of neighborhood minima and maxima for a given shape as its salient feature set.

The intuition behind using the extrema of differences between consecutive levels of the scale-space is that the smoothing with a filter of effective radius R will not have much effect on features whose scale is larger than R but will eliminate most of the features at the scales smaller than R. As we consider differences between two versions of the surface smoothed at two scales R_1 and R_2, the most change will happen in the regions that contain features of scales between R_1 and R_2. For more information, see [Lin94]. In Figure 12 we show a simple example of a scale-space for a sphere with two differently sized bumps. The surface at each level is colored green or red depending on the sign of the normal difference between the current and the previous level. The intensity of the color corresponds to the absolute value of change. The circles denote detected extrema of the scale-space. We see that on coarser levels two detected features correspond to the surface bumps. Each feature gets a scale associated with it that is represented by the radius of the circle in Figure 12.

The scale-space construction starts with a scale that is a multiple of the median distance between points and their closest neighbors (this can be easily found for a given point cloud). The largest scale is specified by the user and is typically comparable to the model size. When several models are given (and no rescaling is required) we use a fixed smallest scale for all of them.

There are two sets of features: one corresponding to the minima and one corresponding to the maxima of the normal difference. These two sets are treated separately in our matching application (we assume that the surface corresponds to the boundary of a solid object, thus the outside normal direction is always well defined).

In order to get a more robust feature matching we exclude features within flat noisy regions from consideration. This filtering procedure is based on the signature computation and is described in Section 4.3. We also perform a simple boundary detection procedure on the input point cloud and exclude the boundary feature points from the consideration.

3.3. Repeatability of features

Given two finely sampled surfel clouds representing the same surface, we can hope that their scale-space representations will be similar and will result in a similar set of feature points on scales that are sufficiently large in comparison to the sampling density. We have explored the validity of this assumption by applying several types of "attacks" to input point clouds, and comparing the resulting sets of feature positions.

In order to evaluate the repeatability of our salient feature point set extraction procedure we applied the following transformations to a set of 3D models:

- Uniform resampling: we subdivide the input mesh models and then randomly remove a set of points until a desired number of points is reached (see Figure 2 for an example).
- Noise addition: we displace the input position in the normal direction by 0.5% of the bounding box. For the Venus model *we recompute all the vertex normals* using the resulting mesh (see Figure 2 for an example). For Bunny and Buddha models, each component of the normal vectors is modified by noise uniformly distributed between -0.05 and 0.05.
- Mesh simplification with the quadrics algorithm [GH97].

In our experiments, we extract the set of features before and after a particular modification and compare the results as follows: for a feature f with position $\mathbf{x}$ and scale h in one model we see if there exist a feature f' in the other feature set that is on the same or a neighboring level in the scale-space representation (that means that the scale h' of f' is between $2^{-1/4}h$ and $2^{1/4}h$), and whose position x' is within distance $h/2$ from $\mathbf{x}$ (we consider the set of minima and maxima features separately). If such a similar feature exists, we say that the feature f found a correspondence. We then count

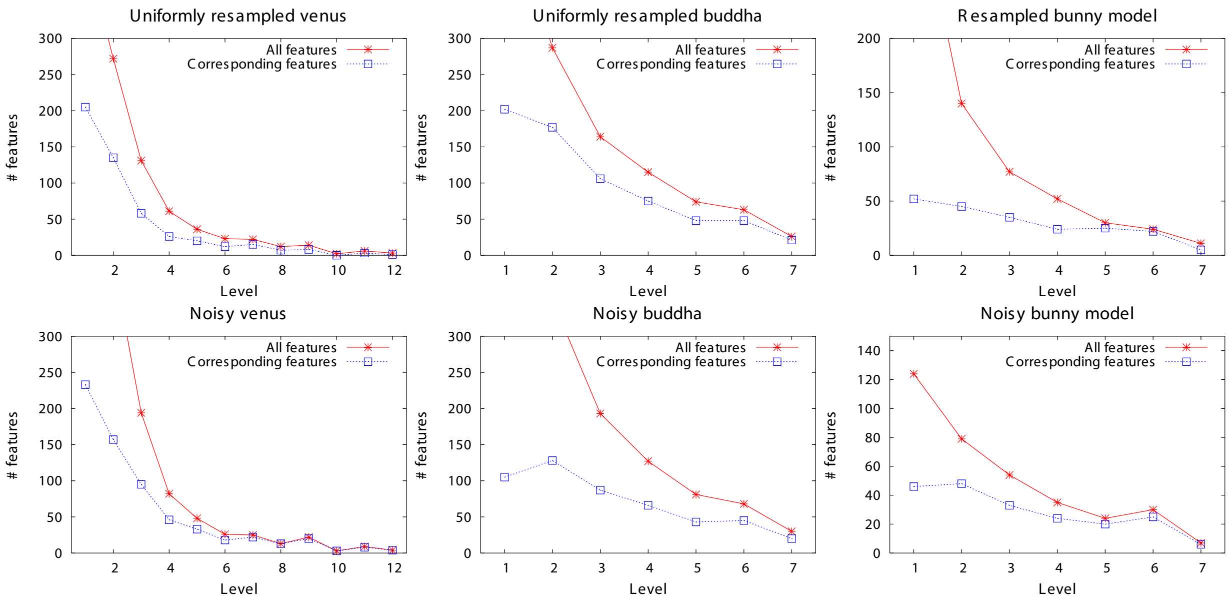


Figure 1: *Feature set repeatability for the uniform resampling (top) and noise addition (bottom) transformation applied to Venus, Buddha, and Bunny models. See text for more details.*

the number of features in both models that have correspondence and compare it to the overall number of features in both models. We would like to have 100% of all the features find their correspondences after the transformation.

Figure 1 shows the results of comparing feature sets of original and uniformly resampled models for the venus head, happy buddha, and Stanford bunny datasets. We see that on coarser levels there is a good correspondence between the original and resampled feature sets. Note that the original venus and buddha models we used are already the result of quadrics simplification procedure, hence the distribution of samples in them is rather non-uniform. In this experiment, we did not use any vertex area information so that all area weights were set to 1 (see below for the area-corrected comparisons). The number of vertices in the original and resampled models were roughly the same.

Figure 1 shows the comparison between the original and noisy models for the same three datasets. We see that again the percentage of corresponding features rises from about 50% for fine scales to almost 100% for the coarser scales.

Since many scans come with some mesh connectivity information, it is possible to use that mesh information to compute area weights for every vertex. The error-based simplification typically results in very non-uniform surface sampling which negatively affects the repeatability of extracted feature sets. Using area weights however improves the situation significantly, as shown in Figure 3 where the original Venus mesh with 50K vertices and quadrics-simplified mesh with 30K vertices are compared first without and then with

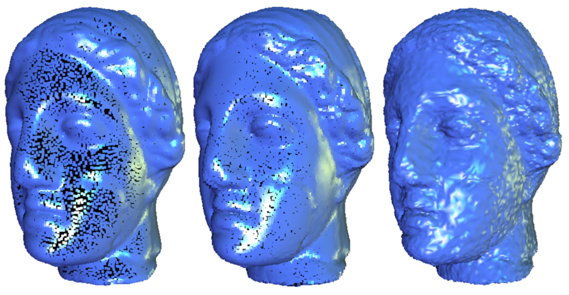


Figure 2: *Original, resampled, and noisy datasets for the evaluation of feature point detection stability.*

area weights. The plot on the right shows the percentage of corresponding features. We see that with area correction the percentage of corresponding features improves and gets to almost 100% for coarse scales. See also the sets of extracted features in Figure 13 where the corresponding features are shown as yellow circles and non-corresponding features are shown in magenta (note that one has to look not only on the same but also on neighboring scale for finding correspondences).

In our experience, the coarse features hold most significant semantic shape information (see for example the nose and eyes features in Figure 11). However, for matching applications, these are not the most useful features since there are not so many of them. Rather, intermediate scale features (of which there are hundreds) play more significant role.

We have also evaluated the area-corrected smoothing in the case when only one model has the connectivity information while the other uses unit area weights. This works well when the distribution of the non-weighted model samples is uniform. Figure 4 shows the comparison of area corrected and non-area corrected feature extraction for the resampled happy buddha model.

For the surface scans used in our matching experiments, the distribution of samples is rather uniform, hence we did not use any area correction. For an example of extracted features in that case see Figure 11.

4. Local surface signatures

Feature points from the preceding section have several attributes (the position, the normal, and the scale) that can be used to perform matching between partial scan of the same shape. However, these attributes are not invariant under rigid transformation or scaling. Such transformation can usually relate the coordinate systems for a pair of scans. In order to enable efficient matching between shapes given in different coordinate systems, we endow each feature with a signature that is invariant to such spatial transformation. Given a surfel cloud P and a nearby point $\mathbf{x} \in \mathbf{R}^3$ together with a normal direction $\mathbf{a}$ and a scale $h > 0$, we define a local signature vector $\sigma_P(\mathbf{x}, \mathbf{a}, h)$ as is described below.

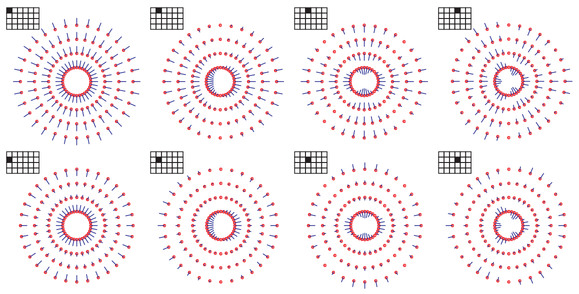


Figure 5: *Projected normal patterns for eight "basis" signatures (the black box shows which entry of a 6×4 signature matrix is equal to one, white boxes are zeros).*

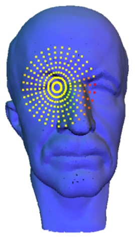
4.1. Definition of signatures

Given a normal direction $\mathbf{a}$ we define an orthogonal local frame $(\mathbf{u}, \mathbf{v}, \mathbf{a})$. The choice of $\mathbf{u}$ and $\mathbf{v}$ is not unique as they can rotate around the axis defined by $\mathbf{a}$ (we address this ambiguity below by performing Fourier transform to obtain a signature approximately invariant to the choice of $\mathbf{u}$ and $\mathbf{v}$).

Our signature computation starts by defining a $N \times M$ array of 3D points (ξ_{kl}) that sample a disc around point $\mathbf{x}$:

$$\xi_{kl} := \mathbf{x} + \frac{2lh}{M}\left(\cos(\frac{2\pi k}{N})\mathbf{u} + \sin(\frac{2\pi k}{N})\mathbf{v}\right);$$

where $k = 1..N$ and $l = 1..M$. Next, we define the corresponding array of normals as $\nu_{kl} = \mathbf{n}_{[P,h]}(\xi_{kl})$.



We determine a $N \times M$ array of real numbers by projecting the normals onto the direction connecting the signature center and the current sample location as follows (see the figure on the left for an example of signature samples with colors corresponding to the value of s_{kl}):

$$s_{kl} = \frac{\langle \xi_{kl} - \mathbf{x}, \nu_{kl} \rangle}{\| \xi_{kl} - \mathbf{x} \|} \qquad (1)$$

We then apply a Discrete Cosine Transform to s_{kl} in the radial l-direction, followed by Discrete Fourier Transform in the k direction. We record the resulting magnitudes into the array $(\tilde{s}_{kl}), k = 1..N, l = 1..M$. The upper left corner of this array contains the filtered values that are approximately invariant to the choice of tangent vectors $\mathbf{u}$ and $\mathbf{v}$ if the number of samples N around the disk is sufficiently large. We use $N = 32$ and $M = 8$.

We can now define signature vector by extracting the upper-left corner of the array $\tilde{s}$ so that

$$\sigma_P(\mathbf{x}, \mathbf{a}, h) := (\tilde{s}_{kl})_{k=1...N', l=1...M'}.$$

In most of our applications we use 24-dimensional signature vector by choosing $N' = 6$ and $M' = 4$. Our signature contains positive numbers and is invariant to rigid transformations. For the scaling transform, the invariance requires multiplying the point coordinates and the scale of the signature by the same number. Since the absolute value of the Fourier coefficients is taken, the signatures will also be invariant under reflection transformation (for instance, the signatures of features located on the left and the right ears would be similar).

Figure 5 visualizes eight "basis" normal vector projection patterns for the 2×4 top-left corner of the signature matrix. We can interpret the four columns of Figure 5 as follows:

- The first column corresponds to a local spherical point, either convex or concave.
- The second column represents a non-symmetric component of the signature – the numbers in this column will be often close to zero.
- The third column shows a saddle-like behavior, near a hyperbolic point. It can also be used in a combination with non-trivial first column in order to get an elliptic (non-spherical) point.
- The fourth column would have non-zero entries near a corner of a cube where three creases come together in a symmetric fashion.

We can see that the first and the third columns have some qualitative relation to the curvature at a point (although it is hard to quantify that relation).

Also, note that while the input normals may be noisy, for the signature computation we sample a smoothed normal field. Additional denoising happens when we exclude the high frequency components of the Fourier transform.

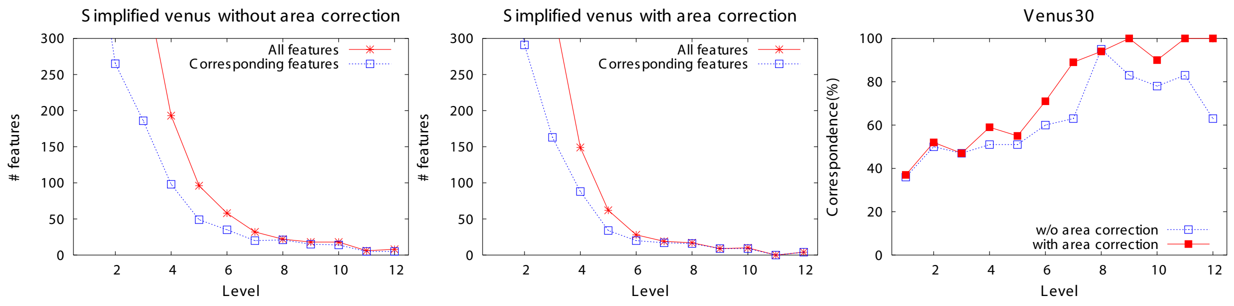


Figure 3: *Feature set repeatability for the quadric simplification applied to the Venus model. The performance for the area corrected version is evaluated.*

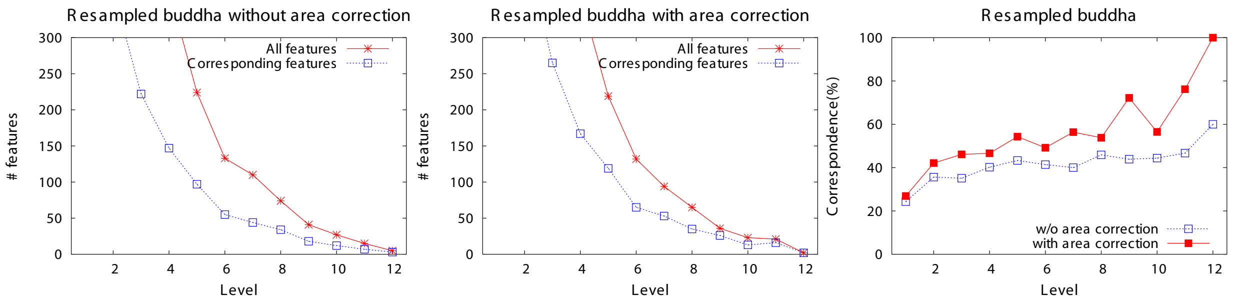


Figure 4: *Feature set repeatability for the resampled Buddha model. The performance for the area corrected version is evaluated. Only one of the models have the area information.*

The local signatures introduced in this section can be found for any position, normal, and scale values. In our application, we shall only use signatures computed at the feature points. For a feature $f = (\mathbf{x}(f), \mathbf{n}(f), h(f))$ we denote its signature as $\sigma(f) := \sigma_P(\mathbf{x}(f), \mathbf{n}(f), h(f))$.

4.2. Signature stability

In our signature computation we use the normals evaluated on the tangent disk to the surface, hence the actual locations where the normal field is sampled are not on the surface. We have experimented with computing normals at projected surface locations but found that the surface projection operation can be very discontinuous in that small noise in either the tangent plane orientation or the underlying model sampling can introduce huge changes in the computed signature. Figure 6 illustrates this by comparing the errors of signatures extracted from the original Buddha model with the signatures computed at the same locations with the same normal and scale but on a resampled version of the model. We see that the projection operation introduces a lot of outliers. Therefore, in our matching application we chose to use signatures computed as described above in Section 4.1.

There should be more experiments comparing our signature vectors with other possible signature definitions used in the literature. We leave this as future work.

4.3. Signature-filtered feature points

The feature-finding procedure of Section 3.2 does not perform any filtering on the found extrema of the scale-space difference function. Thus, in an almost flat region with very small amount of noisy displacement, there will appear a lot of unreliable features. We notice that these “flat” feature points will typically produce local signature vector of a small magnitude. Therefore, we eliminate all the features with signature vector magnitudes below a user-specified threshold. For all the datasets of this paper we remove all the features whose signature vector has length less than 0.2 (note that the signatures are based on a fixed number of unit normals thus no additional normalization is required for this threshold).

5. Surface matching

We use our multi-scale surface features together with their signatures as defined in the previous section to perform approximate matching of surfel clouds. While our method can potentially handle change of scale, most of the scanned data we obtained did not require estimating the scale. Thus, in this section we will describe matching procedures for scans and models that are related by rigid transformations.

Using the terminology from Huber and Hebert’s paper [HH03], we start by describing *pairwise matching* procedure, and then use its results for *multiview matching*. Be-

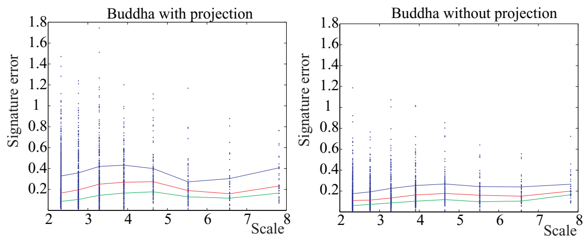


Figure 6: *Error of signatures computed at the same locations on resampled Buddha model. On the left, the signature normals are computed at the projected surface locations, on the right the signatures are computed off the surface (as defined in Section 4.1). Each feature point is plotted with its scale on the horizontal axis and its error on the vertical axis. The three color curves show the 25%, 50%, and 75% error percentiles for each scale.*

fore performing the matching we initialize the set of features with signatures for each input point-based surface as was described in the preceding sections.

5.1. Pairwise surface matching

The input to our pairwise matching algorithm is a pair of surfaces given in different coordinate systems related by an unknown rigid transformation. These could be either two scans of roughly the same size, or a scan and a (partially) reconstructed surface model which can be much bigger. The result of matching is the best candidate for the rigid trasformation (R,t) and the estimated percentage of overlap between two surfaces.

We designate one of the surfaces as the *model* M and the other one as the *scan* S. We compute the feature sets as Ω_M and Ω_S for both surfaces, and compute the signatures at every feature point (also using its normal and scale). Then, for each feature $s \in \Omega_S$ of the scan we find the set $\omega_M(s)$ of $K(s)$ features whose signature vectors are the closest to $\sigma(s)$. The signatures of the model surface $\sigma(\Omega_M)$ are placed in a KD-tree data structure to facilitate fast spatial querying.

The number of model feature points $K(s)$ that are included in the set $\omega_M(s)$ is determined as follows: we first find a fixed constant number K' of neighboring signatures from $\sigma(\Omega_M)$. Denote the feature points corresponding to the found signatures as $m_1,\dots,m_{K'}$. We compute the distances from each found signature to $\sigma(s)$: $\Delta_k = \| \sigma(m_k) - \sigma(s) \|$, and order the features in the order of increasing Δ_k. Thus, $\Delta_1 \le \Delta_2 \le \dots \le \Delta_{K'}$ and we find the smallest value of $k > 0$ such that $\Delta_{k+1} > 1.2\Delta_1$, and discard all the features that follow this index k. Thus, at least one match is found for every feature (we use $K' = 10$ so at most ten matches can be found).

The matching problem consists of finding a rotation matrix R and a translation vector t such that after the transformation the two surfaces overlap within some common region. In order to find the transformation, we consider pairs of *scan features*, and for each pair (s,s') we consider all the corresponding pairs of *model features* (m,m') such that $m \in \omega_M(s)$ and $m' \in \omega_M(s')$, and for each match $\mu : (s,s') \mapsto (m,m')$ we estimate the rigid transformation that maps the positions and normals of the scan feature pair onto the corresponding positions and normals of the model feature pair. Thus, a rigid transform (R_μ,t_μ) is found using the normal direction and position correspondences at the matched pair of points by a method similar to unit quaternion method of [HHN88]. Now, we need to quantify whether this found transform gives a good overlap between the scan and the model. Since this needs to happen for a large number of matches μ, we cannot use an exact error evaluation. Therefore, we shall use an approximate voting procedure that takes into account the correspondences established by signature matching.

The vote is computed for a rigid transform (R,t) by summing the individual contributions $V(s)$ of all the scan features $s \in \Omega_S$. The contribution $V(s)$ is computed as the sum of unit votes for each $m \in \omega_M(s)$ which satisfies $\| Rx(s) + t - x(m) \| < d_1$. Here, d_1 is the threshold that is set to a multiple of the median distance between closest neighbor points in the input data. Thus, only the features that are brought close together by the transformation contribute to the vote, and their contribution is proportional to the quality of their match.

The search for the best rigid transform proceeds until all the feature pairs are considered. The rigid transform with the largest vote is returned as the found solution to the approximate matching problem.

Given the best transform candidate for two surfaces, we compute an approximate overlap percentage $OV(S,M)$ by counting the percentage of points in S that are within a threshold distance d_2 from the model surface M. We choose d_2 to be the smallest scale of the scale-space (two times the median distance between points in the input point cloud). We only consider point matches that have consistent normals (with angle difference less than ten degrees). This simple estimate works well for uniformly sampled surfel clouds. We do *not* run ICP before computing the overlap estimate.

5.2. Multiview matching

In this section we describe a procedure that automatically performs multiview alignment of a set of scans for an object. Given a set of K scans, we run $K(K-1)$ pairwise approximate matching as described in the preceding section and order the resulting pairs of scans by their estimated overlap percentage. Each pair has an estimated (R,t) whose quality will deteriorate as we proceed down this ordered list to the pairs with low overlap.

After the pairs are ordered, we start merging them proceeding from the top of the list and performing ICP alignment for pairs of scans starting from the estimated approximate (R,t) (we use ICP implementation of [RL01]). We keep track of who merged with whom and maintain a disjoint set collection of scan indices (see Figure 10 for the illustration). We skip all the pairs that already belong to the same merged component. Each merged component has scans aligned and specified in the same coordinate system. Effectively, we perform a greedy construction of a spanning tree for the set of scans. This is a very simple procedure and we do not perform cross-validation or discrete search as described in [HH03]. The fact that this simple merging algorithm works well attests to the quality of our pairwise matching.

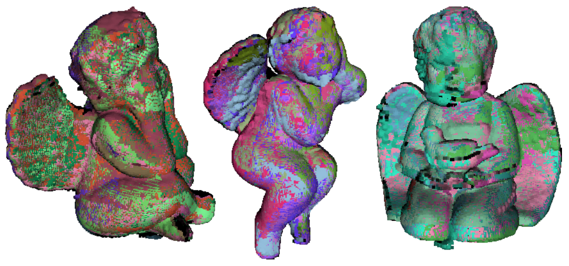


Figure 7: *Result of automatic alignment of scans of the models Angel1, Angel2, and Angel3.*

6. Results

We evaluated our approximate matching procedure on several sets of laser scans. These included scan sets for three angel models from the work of Huber and Hebert [HH03], the scans of bunny and buddha model from Stanford repository, and the Arrigo data set which includes 177 fine scans of a basrelief. The following table enumerates data sets with the number of scans, average number of points per scan, and matching times. The following discussion will give more detail.

Dataset	Num.of scans	Points per scan	Init time	Matching time
Angel1	17	5K	3m28s	50s
Angel2	20	5K	4m24s	63s
Angel3	18	5K	3m41s	74s
Bunny	10	20K	10m	94s
Buddha	15	35-70K	31m	7m35s
Arrigo(I)	30	50-70K	90m	32m
Arrigo(II)	147	30-70K	see	text

6.1. Angels, bunny, and buddha datasets

We reconstruct the three angel models, the bunny model, and the buddha model using the multiview matching procedure described above without any user input. The timing results are given in the table above and the reconstruction results can be seen in Figures 7 and 8. Initialization time includes computing the scale-space, extracting features, and computing signatures. The matching time includes all other operations for the multiview matching including ICP alignment. We believe that our method is competitive with the method of [HH03].

6.2. Arrigo reconstruction

The above multiview matching procedure requires quadratic time with respect to the number of scans. For the datasets that cover the same area multiple times it makes more sense to perform two-stage processing. The first stage merges a set of scans that cover most of the surface into a single *combined* point cloud model using the multiview matching from Section 5.2, and the set of features and signatures can be computed for this combined model. In the second stage, all the remaining non-merged scans can be matched to this combined model one-by-one. Since the combined model covers most of the surface, it will have non-trivial overlap with each scan and the pairwise matching should work.

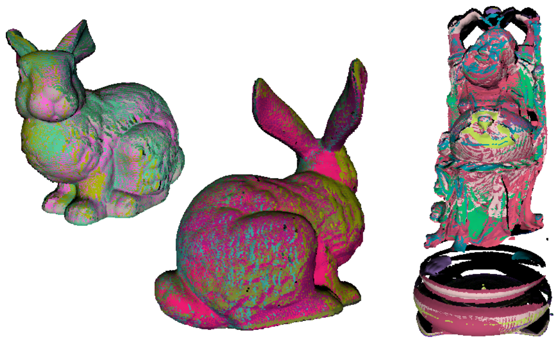


Figure 8: *Result of automatic multiview matching and alignment of scans of the bunny and buddha models.*

The data set of Arrigo basrelief consists of 177 scans separated into four parts each corresponding to a single pass over the whole surface. We took the smallest pass containing 30 scans and run the first stage of our multiview matching that created a combined model consisting of 700K points shown in Figure 10.

In the second stage of Arrigo reconstruction we performed pairwise matching of each of the remaining 147 scans to the combined model. As the result, 144 scans were matched successfully and 3 scans could not be matched automatically. Note that this second stage matches a small scan to a big model: this would be impossible to do with a random search initialization. See the final reconstruction result in Figure 10.

The three non-matched scans were easy to detect by looking at the maximal error between the initialized point cloud and its points after ICP transformation. All three bad matches had error above 50mm, whereas all the good

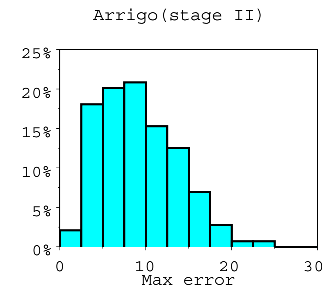


Figure 9: *Histogram of maximal difference between the initialized pointcloud and the exactly aligned version after ICP. The histogram is given for 144 scans matched in the second stage to the combined data from the first stage. Horizontal axis is in millimeters (the statue height is 600mm).*

matches had their error below 25mm as shown in histogram of Figure 9.

The processing for the Arrigo scans were split into two stages: the preprocessing stage and the actual matching stage. In a preprocessing stage each scan was first simplified to approximatedly 50K points, and all of the processing happened with these simplified surfel clouds. For each scan, a scale-space representation was built by smoothing the model for 10 levels. This was the most time consuming of the preprocessing steps and took about two minutes (all the times will be given for a 50K point scan running on a 2.8GHz Pentium 4 PC). In the remaining two steps, the extrema of the inter-scale difference function was found (5 seconds), and the signatures were computed for each extracted feature point. The number of features varied greatly depending on the complexity of any given scan. For a 50K point model, the number of found features were on the order of 1% (or 500). We used the parameter value $C = 1/\sqrt{2}$ for the feature set extraction of Section 3.2 (this reduces the number of features and makes the matching perform faster). The signature computation time for these feature points was about 30 seconds. The multiview matching of the first stage was performed in about 32 minutes. The resulting combined pointcloud was simplified to 700K points and the features and signatures were computed for it in about 35 minutes. The second stage processing for matching each of the 147 scans to the combined model took about 16 seconds per scan (4 seconds for approximate matching and 12 seconds for ICP final alignment). The current bottleneck of our processing is the smoothing stage, this time should be significantly improved by using fast multipole-like methods.

7. Conclusions

We have introduced an approach for approximate alignment of point-based surfaces. Our method is based on detection of salient features that persist under resampling, and on signatures that are approximately invariant under spatial transformations. Future work should include other applications of the proposed multi-scale signed features such as recognition and search of a small surface patch in a database of multiple surface models.

Acknowledgements This work was supported in part by NSF CAREER award (CCR-0133554) and by the NSF ERC RMS (EEC-9529125). The authors would like to thank Jacob Barhak and Lee Markosian for useful comments. The Arrigo dataset are courtesy of the Visual Computing Lab of CNR-ISTI, Pisa, Italy and Roberto Scopigno; the buddha and bunny scans are courtesy of the Stanford 3D scanning repository. The angels datasets were provided by Daniel Huber.

References

[ABCO*03] ALEXA M., BEHR J., COHEN-OR D., FLEISHMAN S., LEVIN D., SILVA C. T.: Computing and rendering point set surfaces. *IEEE TVCG 9*, 1 (2003), 3–15. 2

[AK04] AMENTA N., KIL Y. J.: Defining point-set surfaces. *ACM Trans. Graph. 23*, 3 (2004), 264–270. 2

[AMN*98] ARYA S., MOUNT D. M., NETANYAHU N. S., SILVERMAN R., WU A. Y.: An optimal algorithm for approximate nearest neighbor searching fixed dimensions. *J. ACM 45*, 6 (1998), 891–923. 3

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

[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. PAMI 21*, 11 (1999), 1229–1234. 1

[CM92] CHEN Y., MEDIONI G.: Object modelling by registration of multiple range images. *Image Vision Comput. 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 *Proc. of ECCV* (2004), pp. 224–237. 1, 2

[GH97] GARLAND M., HECKBERT P. S.: Surface simplification using quadric error metrics. *Proceedings of SIGGRAPH* (1997), 209–216. 2, 3

[GIRL03] GELFAND N., IKEMOTO L., RUSINKIEWICZ S., LEVOY M.: Geometrically stable sampling for the ICP algorithm. In *Proc. 4th International Conference on 3D Digital Imaging and Modeling (3DIM)* (2003), pp. 260–267. 1

[GWM01] GUMHOLD S., WANG X., MACLEOD R.: Feature extraction from point clouds. In *Proceedings of the 10th IMR* (Oct. 2001), pp. 293–305. 2

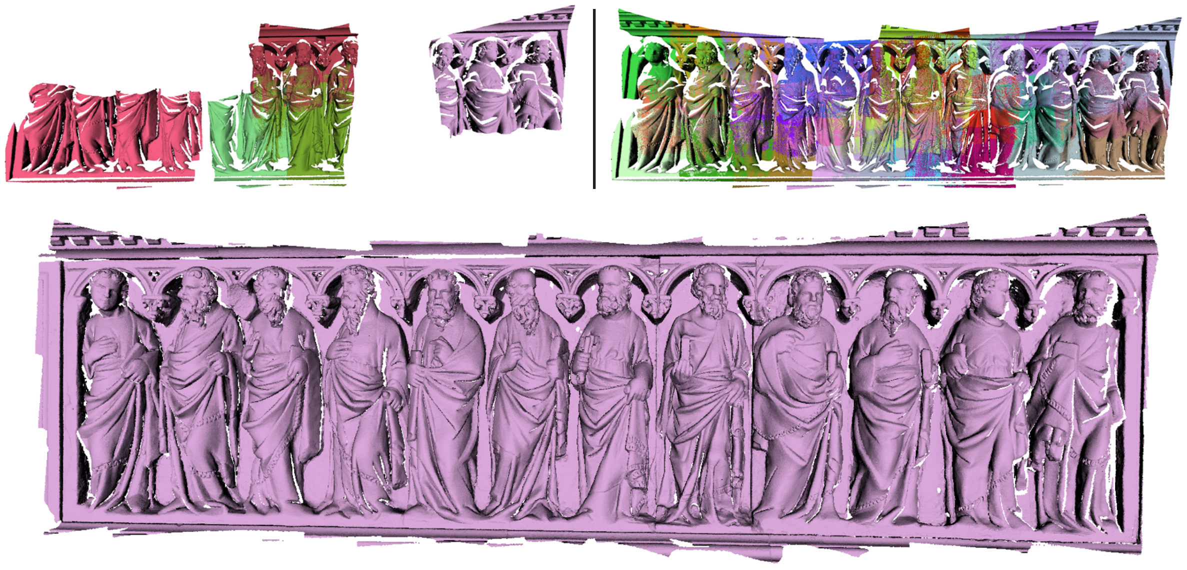


Figure 10: *Top left: during the merging of the first stage. First 10 pairs merge and form five connected components (visualized here in their final positions, each component in one color). Top right: the combined result of the first stage of the Arrigo basrelief alignment. 30 scans are aligned automatically. Bottom: final reconstruction from 174 scans.*

[HGY*99] HALLIMAN P. L., GORDON G. G., YUILLE A. L., GIBLIN P., MUMFORD D.: *Two- and Three-Dimensional Patterns of the Face*. A. K. Peters, 1999. 2

[HH03] HUBER D., HEBERT M.: Fully automatic registration of multiple 3D data sets. *Image and Vision Computing 21*, 7 (July 2003), 637–650. 1, 2, 7, 8

[HHN88] HORN B., HILDEN H., NEGAHDARIPOUR S.: Closed-form solution of absolute orientation using orthonormal matrices. *J. Opt. Soc. Am. 5* (1988), 1127–1135. 7

[JH97] JOHNSON A. E., HEBERT M.: Surface registration by matching oriented points. In *Proceedings 3DIM* (1997), IEEE Computer Society. 1, 2

[Lin94] LINDEBERG T.: *Scale-Space Theory In Computer Vision*. Kluwer, Dordrecht, 1994. 2, 3

[Low04] LOWE D. G.: Distinctive image features from scale-invariant keypoints. *International Journal of Computer Vision 60*, 2 (2004), 91–110. 1, 2

[PKG03] PAULY M., KEISER R., GROSS M.: Multi-scale feature extraction on point-sampled models. In *Proceedings of Eurographics* (2003). 2

[PKKG03] PAULY M., KEISER R., KOBBELT L. P., GROSS M.: Shape modeling with point-sampled geometry. *ACM Trans. Graph. 22*, 3 (2003), 641–650. 2

[PZvBG00] PFISTER H., ZWICKER M., VAN BAAR J., GROSS M.: Surfels: surface elements as rendering primitives. In *SIGGRAPH '00* (2000), pp. 335–342. 2

[RL01] RUSINKIEWICZ S., LEVOY M.: Efficient variants of the icp algorithm. In *Proc. 3DIM 2001.* (2001). 1, 8

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

[SM92] STEIN F., MEDIONI G.: Structural indexing: Efficient 3-D object recognition. *IEEE Trans. Pattern Anal. Mach. Intell. 14*, 2 (1992), 125–145. 2

[WG02] WYNGAERD J. V., GOOL L. V.: Automatic crude patch registration: toward automatic 3D model building. *Comput. Vis. Image Underst. 87*, 1-3 (2002), 8–26. 1, 2

[Wit83] WITKIN A. P.: Scale-space filtering. In *Proc. of 8th IJCAI* (1983), pp. 1019–1022. 2

[YF02] YAMANY S. M., FARAG A. A.: Surfacing signatures: An orientation independent free-form surface representation scheme for the purpose of objects registration and matching. *IEEE Trans. Pattern Anal. Mach. Intell. 24*, 8 (2002), 1105–1120. 1, 2

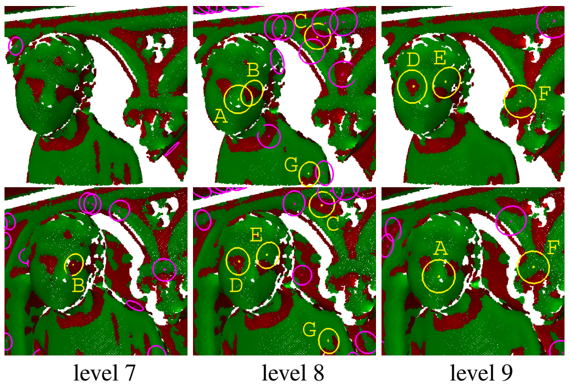


Figure 11: *Seven features shown with lettered yellow circles are shared within levels 7,8,and 9 of scale-space representation for two overlapping Arrigo scans.*

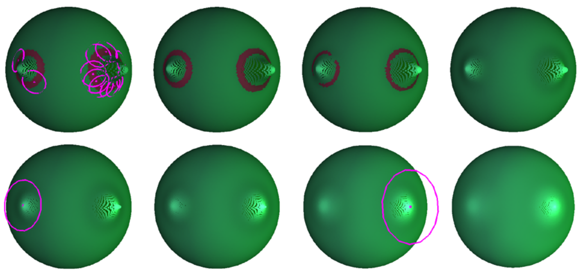


Figure 12: *Simple example: while many features are detected on the fine levels, on the coarser scales we detect exactly two features with the scales corresponding to the sizes of two bumps. Point colors correspond to the value of* d_j.

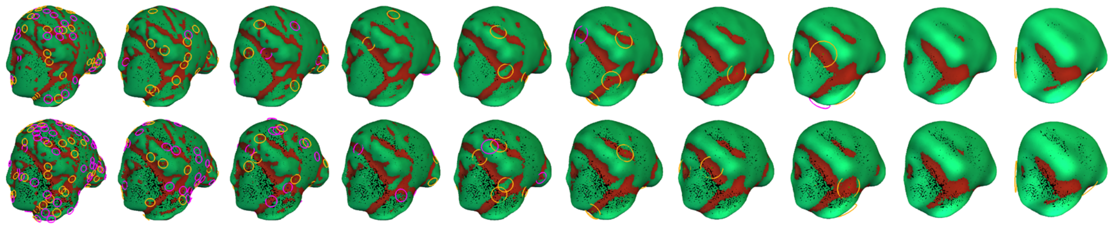


Figure 13: *Comparison of scale-space features from two differently sampled surfel clouds representing the Venus head (top: original, bottom: quadrics-simplified to 30K vertices). One needs to also look at the neighboring levels to find common features (shown by yellow circles). These two models use the area correction. See also the central plot of Figure 3.*