Intrinsic Features on Surfaces

Markus Schlattmann*

Institut für Informatik II, Universität Bonn D-53117 Bonn, Germany

Abstract

The detection of stable feature points is an important preprocessing step for many applications in computer graphics. Especially, registration and matching applications often require those points and depend heavily on their quality. In the 2D image case, scale space based feature detection is well established and shows unquestionably good results. For this reason, we generalize a scale space to 3D surfaces and use it for the detection of analog feature points in 3D. Our features are robust to noise and provide a good description of the model.

Keywords: feature detection, intrinsic, surface

1 Introduction and Previous Work

Many 3D applications in computer graphics need features. For instance for morphing applications a feasible mapping between two objects is computed, where salient regions should be mapped on corresponding regions, as for example the eyes of an animal on the eyes of another. In registration and matching applications the features have to be robust and descriptive in order to be able to compute correspondences even if only parts of the objects are similar. In addition to that, the matching procedure can be improved, if the knowledge about the extend of the structure a feature point describes is present. For this reason, we compute feature together with a scale, whereas the scale indicates the extend.

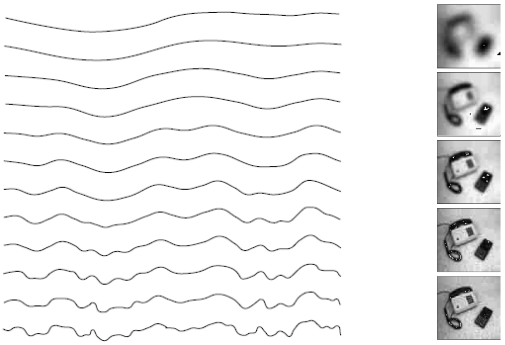
The detection of feature points is well established in 2D image applications. Many feature based matching methods, as surveyed in [16], have shown great applicability. Especially, scale space based techniques [9] are known for their robustness and therefore, are often used in practice. A scale space is computed by smoothing an input signal in an appropriate way, so that the outcome of this is a representation over scales, consisting of the smoothed signals. Thereby, the scale determines the smoothing of the input signal. Figure 1 shows two input signals (bottom), that are iteratively smoothed to obtain a scale space. A scale space for a function $f : \mathbb{R}^D \to \mathbb{R}$ is defined as follows: Given f as a continuous signal, then a scale space $L : \mathbb{R}^D \times \mathbb{R}_+ \to \mathbb{R}$ of f is defined as the solution of the heat diffusion equation

$$\partial_t L = \frac{1}{2}\nabla^2 L = \frac{1}{2}\sum_{i=1}^{D} \partial_{x_i x_i} L, \tag{1}$$

with $L(\cdot,0) = f(\cdot)$. This scale space can be computed by convolution of $f(\cdot)$ with a Gaussian kernel g:

$$L(\cdot,t) = g(\cdot,t) \otimes f(\cdot), \tag{2}$$

with $g : \mathbb{R}^D \times \mathbb{R}_+ \backslash \{0\} \to \mathbb{R}$. Note that the Gaussian kernel is the unique kernel to solve the diffusion equation, as shown in [3, 4].



(a) (b)

Figure 1: An input signal (bottom) is iteratively smoothed to obtain a scale space. a) One dimensional. b) Discrete two dimensional with extrema. [13]

To detect *scale invariant* blob features, Lindeberg [7] used a scale-normalized Laplacian of Gaussian (LOG) function $t\nabla^2 L$ to detect features in the scale space. Scale invariance means, that if an image is scaled with a certain factor, then the features will be detected in a scale, which is multiplied with the same factor. For this normalized LOG Lindeberg has shown scale invariance. In figure 2 two exemplary scale invariant feature points are shown with their signatures, detected in the scale-normalized LOG.

For 2D images Lowe [8] introduced a so called difference of Gaussian representation (DOG) of f, defined as follows:

$$\begin{aligned} DOG(x,t) &= (g(x,kt) - g(x,t)) \otimes f(x) \\ &= L(x,kt) - L(x,t). \end{aligned} \tag{3}$$

The initial image is incrementally convolved with Gaussians to produce images separated by a constant factor k in scale space, shown stacked in figure 3 (Left). Adjacent images are subtracted to produce the difference of Gaussian images in figure 3 (Right). That means for the discrete case, beginning for example with $\sigma_0 = 1$, the σ_i are obtained as follows:

$$\sigma_i = k^i \sigma_0, \tag{4}$$

*markus@cs.uni-bonn.de

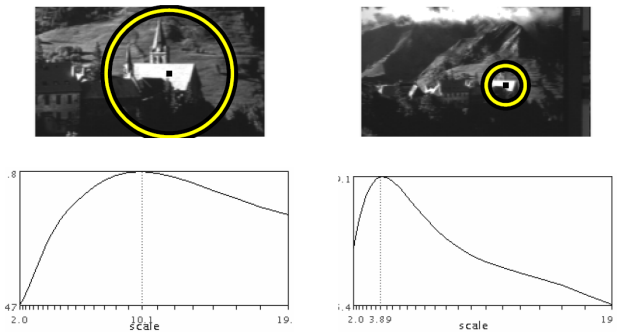


Figure 2: Top row shows images taken with different zoom. Bottom row shows the responses of the Laplacian over scales. The ratio of scales corresponds to the scale factor (2.5) between the two images. The radius of displayed regions in the top row is equal to 3 times the selected scales. [9]

where $t = \sigma^2$. This results in an exponential time step. To be able to find all extrema, the factor k should be small enough. Lowe [8] used values from the interval $(1; \sqrt{2}]$. Depending on the magnitude of σ_0, more or less initially scales of L are excluded for building the *DOG*. Lowe [8] used this representation to approximate the scale normalized Laplacian of Gaussian.

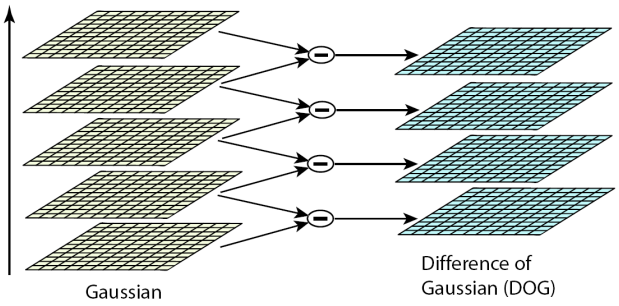


Figure 3: (Left) Gaussian Scale space. (Right) Difference of Gaussian scale space. [8]

A feature point is extracted, if a pixel in a level of the DOG has an extremal value in respect of its neighbors in the same and the neighboring scales, as shown in figure 4. The special advantage of this kind of features is the knowledge about the scale, from which a salient point is extracted. The scale indicates the size of the structure the feature point describes. In addition to that, the feature points of two images of different resolutions can be compared because of the scale invariance.

Following this idea, we generalized the scale space and the feature extraction to triangulated two manifold surfaces in 3D. We use a diffusion flow to derive the sequence of surfaces and use the movements of the vertices as a basis to extract feature points together with a scale, as in 2D is done with the difference of Gaussian representation.

Many other approaches introduced techniques to find features on 3D surfaces, often in the context of matching

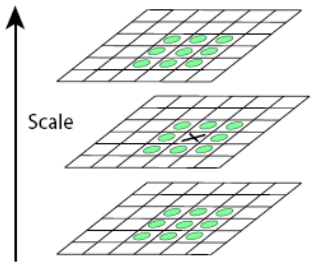


Figure 4: To find maxima and minima in a pixel (marked with X), it is compared to the neighbor pixels (marked with circles). [8]

applications used. In [12] an overview is given of those and other matching procedures. But only a few approaches tackled the problem of finding scale invariant features. The features to extract can be distinguished into two classes: On the one hand, those, that can be described by a line or a curve, and on the other hand, feature points, like our method extracts. Several methods extract multi scale features from the first category, as for instance [10, 15]. For the scale invariant point features only very few methods exists, as for example [1, 6].

An approach to derive and smooth a surface from polygonal data to multiple scales is done in [11]. By using a constrained moving least-squares formulation a surface can be generated, which approximates the input, whereas features with a specified size are smoothed away. Unfortunately, if the surface nearly touch itself, it will accrete at this point, so that marginal differences of the surface could result in a highly different behavior of this smoothing process. For this reason, this formulation of a smoothing of a surface is not usable for our goal.

Lee et al. [5] introduced a method to describe the saliency of every region in a mesh by summing curvatures in all vertices. To this end, a gaussian kernel with iteratively enlarged standard deviation is used to integrate over the mean curvatures of the local neighborhood of a vertex. Whereas nice results are obtained for a simplification based on the saliency, the extremal regions of saliency do not include a scale they are detected in. In particular, to improve the search for partial correspondences, such a scale can be used.

A partial surface matching method based on local descriptors was introduced in [2]. The surface is divided into small regions, whose local shapes can be well approximated by quadrics. These regions are used as descriptors and the most salient ones are chosen for the partial matching process. However, it is unclear whether this method is unsensible to noise, especially because of the dependency of the extracted surface regions on local curvature.

2 General Setup and Notation

Our objective in this paper is to extract scale invariant feature points on a 3D model. These features are intrinsic, because they depend only on the surface. We assume in the following that the model is a closed two manifold surface. In addition to that, we consider only objects with genus zero. The surface is a triangulated mesh M with $M = \{V, E\}$, where $V = \{v_i | v_i \in \mathbb{R}^3, i = 1, ..., |V|\}$ is a set of vertices and $E = \{e_{ij}\}$ the set of edges which connect the vertices. A face is given, if a cycle of three edges e_{ij},e_{jk} and e_{ki} exists. For each vertex v_i, a normal n_i can be computed.

3 Generalization to 3D

To simulate the diffusion equation (see equation 1), we use a surface diffusion flow to iteratively smooth the objects and to obtain a set of smoothed surfaces that constitute our scale space.

In this section we first describe the mean curvature flow and some of its properties. Furthermore, we give the discretisation used in our implementation and in the end the definition of our feature points is introduced.

3.1 Building the Scale Space

In the image case usually a Gaussian kernel is used to generate the representation over scales. That is possible because there exists a global parameterization invariant over all scales. In our case of two manifold surfaces however, such a parameterization is generally not defined. However, a local parameterization for each vertex in each scale is calculable. Therefore an iterative flow is utilizable to simulate a similar diffusion process.

3.1.1 Averaged Mean Curvature Flow

The ordinary mean curvature flow is defined as follows:

$$\frac{\partial v_i}{\partial t} = -H_i n_i, \tag{5}$$

where H_i is the mean curvature at vertex v_i. $\frac{\partial v_i}{\partial t}$ is the position increment vector of vertex v_i so the new position results in $\widetilde{v_i} = v_i + \frac{\partial v_i}{\partial t}$. That means, a vertex v_i is moved in direction of its normal n_i with the magnitude of the mean curvature $H = \frac{1}{2}(\kappa_{min} + \kappa_{max})$, where κ_{min} and κ_{max} denote the principal curvatures. A vertex on a convex region will move inwards, whereas a vertex on a concave region will show an outward movement. At a saddle point, the minimal curvature is negative, while the maximal curvature is positive, so the direction of the movement depends on their magnitudes.

The mean curvature flow is known to shrink volume. Thus, a closed surface with genus zero will evolve into an infinitesimally small sphere (see figure 5).

Figure 5: Ordinary mean curvature flow evolves objects to a infinitesimal small sphere.

A better solution is to use a volume preserving flow like the averaged mean curvature flow. This flow is defined by a modification of the ordinary mean curvature flow as follows:

$$\frac{\partial v_i}{\partial t} = -(H_i - \sum_{v_j \in M} \frac{H_j}{|V|}) n_i. \tag{6}$$

The result is a volume preserving flow as shown in figure 6. Whereas the averaged mean curvature flow is more stable than the ordinary one, it still suffers from one deficiency. If an object has a long thin limb, the flow will trench it after a few steps as shown in figure 7. However, with a little variation in the thickness, it is possible, that the object is not fragmented. This results in big variations of the feature detection, so that the computed features for such objects are not robust. For this reason, it is only useful for restricted types of objects. Therefore, in our work, we use and compare only objects, that do not cause fragmentations.

Figure 6: Averaged mean curvature flow evolves objects to a sphere with the same volume.

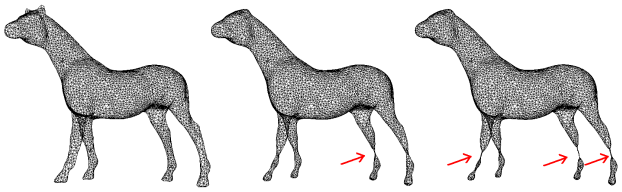


Figure 7: Mean curvature flow trench thin limbs after a few steps.

3.1.2 Discretisation

In the following the implementation details for the iterative computation of the flow are provided. The principal curvatures are computed with a quadratic fitting process to a local sampling of the surface. That means, a quadratic function as shown in figure 8 is computed, and describes the local neighborhood. Hence, the eigenvalues of its hessian correspond to the principal curvatures. The sampling of the local neighborhood is obtained with the Dijkstra-Algorithm, so it consists of the n nearest neighbor vertices v_{i_k} of vertex v_i.

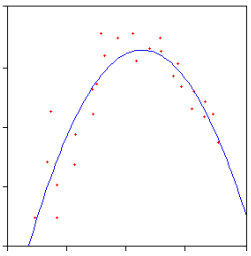


Figure 8: A quadratic function fits a collection of points in 2D.

To fit a quadratic function through the collected points, first the sampled points v_{i_k} have to be transformed onto the tangent plane of v_i. For that purpose two arbitrary orthonormal vectors o_1 and o_2, lying in the plane with normal n_i, are computed. Then the sample points are transformed to points q_k as follows:

$$q_k = ((v_{i_k} - v_i) * o_1, (v_{i_k} - v_i) * o_2). \tag{7}$$

To get the coefficients $c_l \in \mathbb{R}$, the basis $\{B_l(\xi_1,\xi_2)\}_{l=1}^5 = \{\xi_1, \xi_2, \frac{1}{2}\xi_1^2, \xi_1\xi_2, \frac{1}{2}\xi_2^2\}$ of the quadratic functions (without constant coefficient) is used to set up the following system of equations:

$$\sum_{l=1}^{5} c_l B_l(q_k) = (v_{i_k} - v_i)n_i, \qquad k = 1,...,n. \tag{8}$$

With $A = (B_l(q_k))_{k=1,l=1}^{n,5} \in \mathbb{R}^{n\times 5}$ and $C = (A^TA)^{-1}A^T \in \mathbb{R}^{5\times n}$ its pseudo inverse matrix it can be written as

$$[c_1,...,c_5]^T = C[(v_{i_1} - v_i)n_i, ..., (v_{i_n} - v_i)n_i]^T. \tag{9}$$

This way the coefficients can be calculated to a quadratic function $f(x,y) = c_1x + c_2y + c_3x^2 + c_4xy + c_5y^2$ and by computing the eigenvalues of the function's hessian matrix, we get the principal curvatures. This scheme is based on the quadratic fitting technique from Xu [14].

3.1.3 Remeshing

Since geometry changes greatly during smoothing, the mesh has to be adopted, in order to obtain a mesh with neither too large nor too small or narrow triangles. To this end we use flips, collapses and splits. After each smoothing step the following tasks are executed in sequence:

1. Flip all edges e_{ij}, if the resulting edge is shorter than $\|v_i - v_j\|$ and the angle between the normals of the two adjacent facets of e_{ij} is smaller than three degrees. This improves the structure of the mesh without adding or deleting a vertex.

2. Collapse all edges e_{ij}, if their lengths are below one fifth of the average edge length. This avoids too small triangles.

3. Split all edges e_{ij}, if their lengths are above five times of the average edge length or if the roundness of one of the adjacent triangles is above 1.5. The roundness is defined as the ratio between the radius of the circumcircle and the length of the shortest edge of the triangle. This avoids too big or narrow triangles.

The movement of the vertices in one smoothing step is very small, so one iteration after each smoothing is sufficient. Additionally, we assume the initial meshes to have a structure, which does not make such an remeshing operation necessary.

3.2 Scale space signatures

To define the scale space signatures, we first need to formally define our scale space L. Because we are using an explicit scheme, the time step between two scales has to be constant and not too large. If the sample rate is higher, the time step in the smoothing process should be smaller, because we utilize an explicit smoothing scheme and therefore oscillations and other singularities would arise by using too big scale steps. Especially the exponentially enlargement would cause those problems. For this, we first build a discrete scale space as follows:

$$\begin{aligned} L_D(v,j) &= \sum_{i=0}^{j} d_i(v), \quad j \in \mathbb{N}, \\ d_i(v) &= sign(v,i)\|\frac{\partial v^i}{\partial t}\| \\ sign(v,i) &= \begin{cases} -1 & , \text{ if } \langle \frac{\partial v^i}{\partial t}, n^i \rangle < 0 \\ 1 & , \text{ else} \end{cases} \end{aligned} \tag{10}$$

with v^i is the vertex v in scale i ($v^0 = v$) and n^i its normal in this scale. $d_i(v)$ are the signed distances between two scale levels i and $i+1$ of vertex v. To get an approximation to a continuous scale space with scale level σ, we use the discrete values with

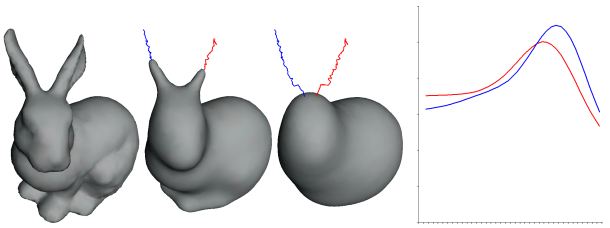
$$L(v,\sigma) = L_D(v,\lfloor\sigma\rfloor) + (\sigma - \lfloor\sigma\rfloor)d_{\lceil\sigma\rceil}(v), \quad \sigma \in \mathbb{R}. \tag{11}$$

Now, we define analogously to the discrete difference of Gaussian representation of Lowe [8]:

$$\begin{aligned} D(v,j) &= L(v,\sigma_{j+1}) - L(v,\sigma_j), \quad j \in \mathbb{N}, \\ \sigma_j &= k^j\sigma_0. \end{aligned} \tag{12}$$

Thereby, σ_0 depends on the constant smoothing step, that is used to smooth the surfaces. If the resolution is high, the step has to be smaller, than for a mesh with a lower resolution. Moreover, in order to subdivide each octave of σ_0 to sixteen steps, we used $k = 2^{\frac{1}{16}}$.

As a signature S of a vertex v we now use the vector $S = \{D(v,0),...,D(v,m-1)\}$, where m denotes the maximal computed scale. In figure 9 the trajectories and signatures of two vertices are shown.



(a) (b)

Figure 9: a) The trajectories of two vertices on the ears of the bunny. b) The scale space signatures (smoothed) of the trajectories.

3.3 Feature Points

In the application of feature detection we need features which provide a sufficient description of the model and which are stable, if an object changes marginally.

In our case, we compute feature points as extrema on *extremum paths* [7]. An extremum path r is a sequence of extremal vertices over the scales. That means, the vertices $r(i)$ of the maximum path r have locally maximal signature values in all scales $i = 1,\ldots,l$:

$$D(r(i),i) \geq \max_{v_k \in N^i(r(i))} (D(v_k,i)), \qquad (13)$$

where v_k are the neighbors of $v = r(i)$ in scale i and l is the length of the path. Note that a vertex v has a different position depending on the scale that is considered. Is $d^i_{geo}(v,w)$ the geodesic distance of two vertices in scale i and the signature values of vertices v_j are maximal in respect to their neighbors in this scale, then the following constraints have to be satisfied:

$$\begin{aligned} \forall v_j: \quad d^i_{geo}(r(i-1),r(i)) &\leq d^i_{geo}(v_j,r(i)) \quad \text{and} \\ d^i_{geo}(r(i-1),r(i)) &\leq d^i_{geo}(v_j,r(i-1)), \\ & i = 1,\ldots,l. \end{aligned} \qquad (14)$$

Thereby, the length l of a path r depends on whether a following maximum exists or not. The computation of the minimum paths is analougly done. An extremum path always begins in the first scale and ends if no following extremum exists.

Now, we detect v as a feature vertex at scale i, if it is included in a maximum/minimum path r with $r(i) = v$ and if the value $D(r(i),i)$ is maximal/minimal in respect to its neighbors $r(i-1)$ and $r(i+1)$.

3.4 Reducing Noise

To reduce noise due to remeshing (because of its local changes in the triangulation), a filtering over the mesh (see figure 10) on the one hand and a filtering over the signatures of the extremum paths (see figure 11) on the other hand is done with Gaussian kernels. The standard deviation σ of the first Gaussian kernel (two dimensional) is set in dependency of the average edge length in the mesh. This is a good choice, because normally the higher the resolution (corresponds to the average edge length) of a mesh, the smaller are the structures in the mesh that can be modeled and the more feature points should and can be extracted. In our application we took a width of twice the average edge length. The standard deviation of the second Gaussian kernel (one dimensional), used to smooth the signatures, is set to four.

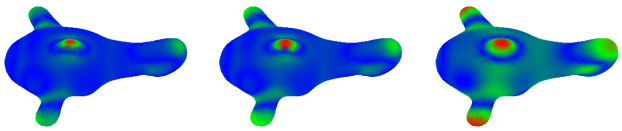


Figure 10: A fish with relatively colorcoded differences. (Left) Unfiltered. (Middle) Filtered with $\sigma = 2$. (Right) Filtered with $\sigma = 4$.

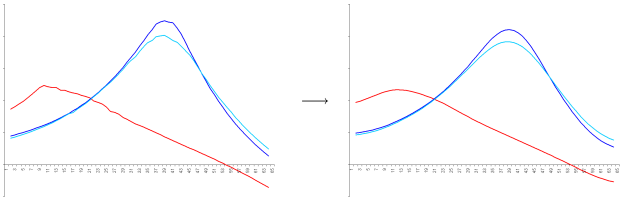


Figure 11: The scale space signatures of three extremum paths of the fish model. (Left) Unfiltered. (Right) Filtered with $\sigma = 8$.

3.5 Eliminating Unstable Features

If a feature point describes a ridge or ravine like structure of the object, often its position is not well determined, because the vertices along this structure have very similar DOG-values. For this reason, Lowe [8] introduced the *hessian condition*. This condition rejects such feature points by thresholding over the ratio of its eigenvalues. That means, the eigenvalues λ_{max} and λ_{min} of the hessian matrix H in respect of the difference of Gaussian values

$$H = \begin{pmatrix} D_{xx} & D_{xy} \\ D_{yx} & D_{yy} \end{pmatrix} \qquad (15)$$

are computed. Now, if the ratio $\frac{\lambda_{min}}{\lambda_{max}}$ is above 0.5, the point is not taken as a feature. Additionally, features are rejected, if their eigenvalues of H have different signs. Because of this threshold all unstable feature points can be removed. Analogously to the image case, we compute the hessian matrix of a feature point in its scale with an radius proportional to its scale. By this, we get a good indicator for figuring out, whether a feature has an unstable position.

4 Results

In this section, several examples of our feature detection method are presented. For the examples, the same thresholds and widths of the Gaussian kernels to smooth the DOG-values are used.

In the following figures, the feature points detected as a maximum are printed in red, while those detected as a minimum are printed in blue. In addition to that, the signatures of the extremum paths are printed in the same colors. A feature point is illustrated as a circle with a radius proportional to the scale the feature was detected in. Thereby, the object is shown in about the scale of the feature points.

4.1 Differently Scanned Objects

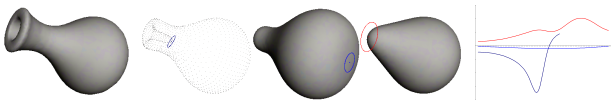
To show the robustness by extracting feature points of differently sampled models, the features of two ants with different resolutions are shown in figure 12. It can be seen, that the same features are extracted, and only the signatures differ marginally.

4.2 Similar Objects

To demonstrate the robustness of our method for pose invariance, we applied our technique on three poses of a hand. In figure 13 the results show a great attitude by using our procedure for those objects.

4.3 Other Examples

The third feature point of the vase in figure 14 shows, that important features are found, which probably would not be found by other methods.

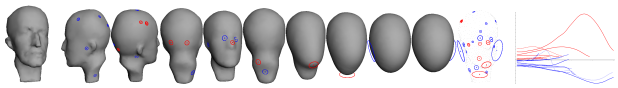


(a) (b) 1501 vertices (c) Signatures

Figure 14: Feature points and signatures of a vase. a) Original model. b) Smoothed object in scales of the features. c) Signatures.

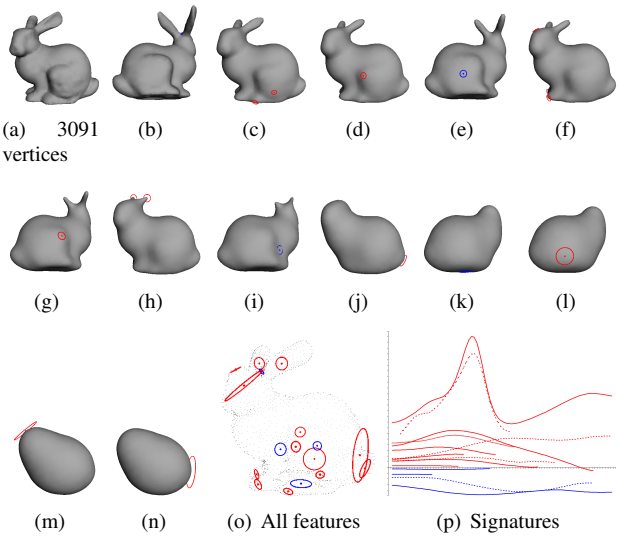
For the features of the Max Planck head in figure 15, a lower threshold (0.35) is used for the hessian condition, because otherwise the nose and the ears would not have been extracted. Unfortunately, the problem of choosing the most appropriate threshold arises, so we think, that in a practical application the ratio of the eigenvalues should better be used as a confidence of a feature than for thresholding.

The last example for our method are the features of the Stanford Bunny in figure 16.



(a) (b) 1643 vertices (c) Signatures

Figure 15: Feature points and signatures of the Max Planck model. a) Original model. b) Smoothed object in scales of the features. c) Signatures.



(a) 3091 vertices (b) (c) (d) (e) (f)

(g) (h) (i) (j) (k) (l)

(m) (n) (o) All features (p) Signatures

Figure 16: Feature points and signatures of the Stanford Bunny. a) Original model. b)-n) Smoothed object in scales of the features. o) All feature points. p) Signatures.

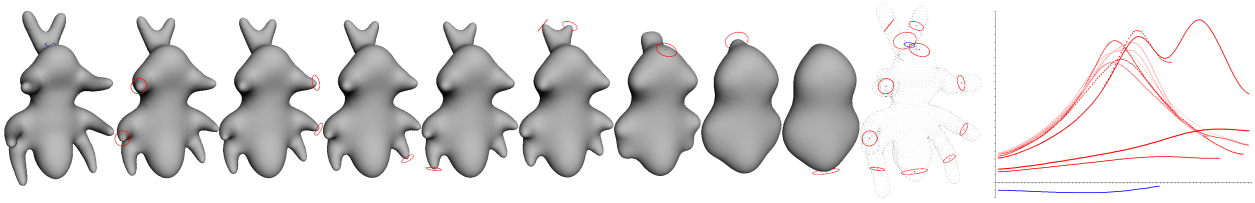
5 Conclusions and Future Work

Robust feature points are needed for many applications, as for instance matching and morphing. Based on approved methods for the image case, we introduced a novel technique for the extraction of feature points on 3D surfaces. Therefore we generalized the scale space method of Lindeberg [7] to 2-manifolds in 3D, and use the averaged mean curvature flow to build an analog representation over scales. We detect a salient point by checking if it is extremal both over the local scales and over the local mesh. The transfer of the hessian condition has shown good results by thresholding unstable features. Lastly, we have shown the robustness with several examples.

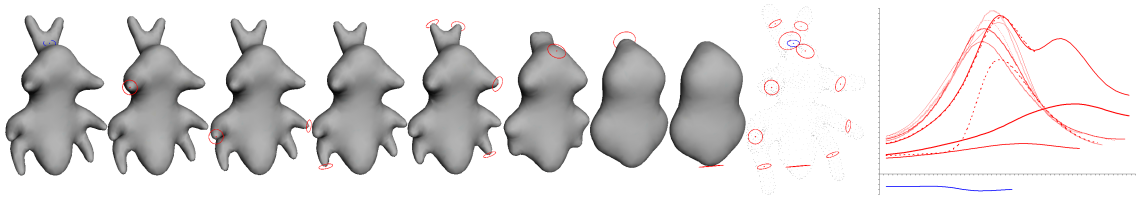
One problem of our approach is the dependency of the used flow. The mean curvature flow is not qualified to be used in a general application, because it tends to fragment specific objects. Because of this, we want to explore different flows and their properties, in order to find a more suitable one for our method.

Due to the fact that of using the scale space for detection, we obtain features, that are robust against noise on the surface. Solely, in the first scales wrong features were found.

In a matching application additionally a descriptor

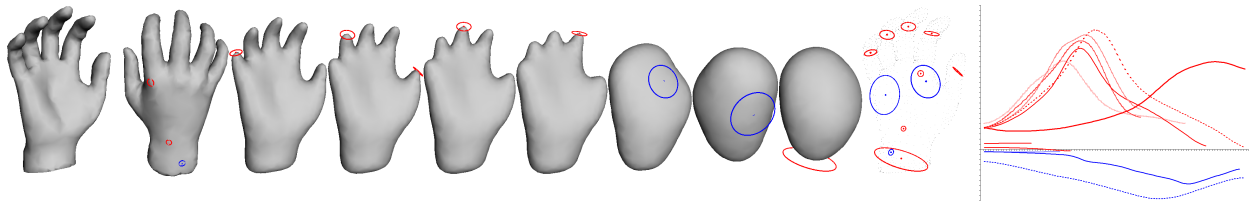


(a) 5087 vertices (b) Signatures of a)

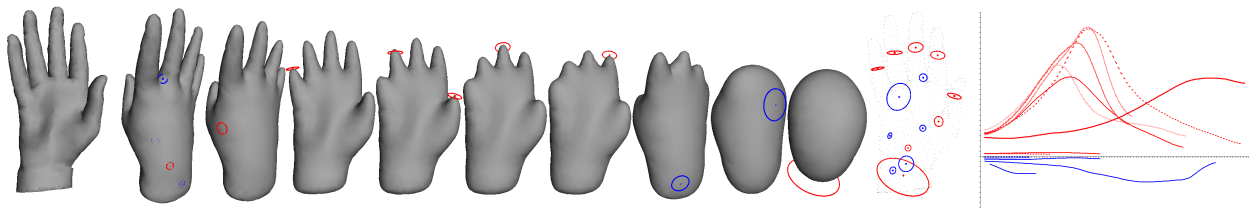


(c) 1793 vertices (d) Signatures of c)

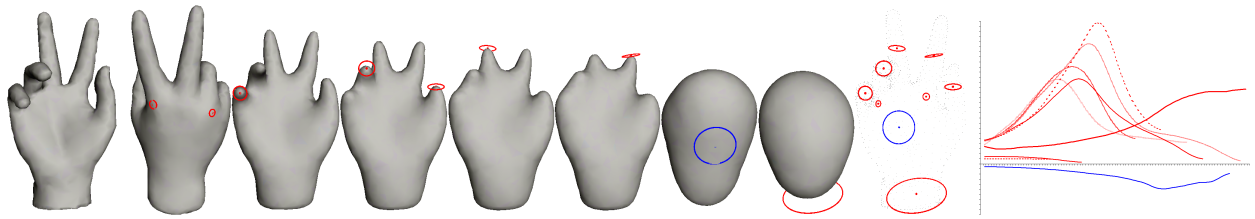
Figure 12: The feature points of an ant model with different sample rates. a) and c) Smoothed models in scales of the features. b) and d) Signatures.



(a) (b) 1404 vertices (c) Signatures of b)



(d) (e) 1420 vertices (f) Signatures of e)



(g) (h) 1419 vertices (i) Signatures of h)

Figure 13: Feature points and signatures of three poses of a hand. a), d) and g) Original models. b), e) and h) Smoothed objects in scales of the features. c), f) and i) Signatures.

could be used to improve the descriptive power of our features. To get scale invariance, this descriptor could work with a radius proportional to the scale of its feature point.

In the future, we would like to modify our method to compute other types of features, as for example line features.

References

[1] Ulrich Clarenz, Martin Rumpf, and Alexandru Telea. Robust feature detection and local classification for surfaces based on moment analysis. *IEEE Transactions on Visualization and Computer Graphics*, 10(5):516–524, 2004.

[2] Ran Gal and Daniel Cohen-Or. Salient geometric features for partial shape matching and similarity. *ACM Transactions on Graphics*, 25(1):to appear, 2006.

[3] Babaud J., A. P. Witkin, M. Baudin, and R. O. Duda. Uniqueness of the gaussian kernel for scale-space filtering. *IEEE Trans. Pattern Anal. Mach. Intell.*, 8(1):26–33, 1986.

[4] J. J. Koenderink. The structure of images. *Biological Cybernetics*, 50:363–370, 1984.

[5] Chang Ha Lee, Amitabh Varshney, and David W. Jacobs. Mesh saliency. *ACM Trans. Graph.*, 24(3):659–666, 2005.

[6] Xinju Li and Igor Guskov. Multiscale features for approximate alignment of point-based surfaces. In *Symposium on Geometry Processing*, pages 217–226, 2005.

[7] Tony Lindeberg. Feature detection with automatic scale selection. *Int. J. Comput. Vision*, 30(2):79–116, 1998.

[8] David G. Lowe. Distinctive image features from scale-invariant keypoints. *Int. J. Comput. Vision*, 60(2):91–110, 2004.

[9] K. Mikolajczyk, T. Tuytelaars, C. Schmid, A. Zisserman, J. Matas, F. Schaffalitzky, T. Kadir, and L. Van Gool. A comparison of affine region detectors. *Int. J. Comput. Vision*, 65(1-2):43–72, 2005.

[10] Mark Pauly, Richard Keiser, and Markus Gross. Multi-scale feature extraction on point-sampled surfaces. 22(3):281–289, 2003.

[11] Chen Shen, James F. O'Brien, and Jonathan R. Shewchuk. Interpolating and approximating implicit surfaces from polygon soup. *ACM Trans. Graph.*, 23(3):896–904, 2004.

[12] Johan W. H. Tangelder and Remco C. Veltkamp. A survey of content based 3D shape retrieval methods. In *Proceedings of the Shape Modeling International*, pages 145–156, 2004.

[13] Andrew P. Witkin. Scale-space filtering. In *8th Int. Joint Conference on Artificial Intelligence*, pages 1019–1022, 1983.

[14] Guoliang Xu. Convergent discrete laplace-beltrami operators over triangular surfaces. In *GMP*, pages 195–204, 2004.

[15] Shin Yoshizawa, Alexander Belyaev, and Hans-Peter Seidel. Fast and robust detection of crest lines on meshes. In *Proceedings of the ACM symposium on Solid and physical modeling*, pages 227–232, 2005.

[16] Barbara Zitová and Jan Flusser. Image registration methods: a survey. *Image Vision Computing*, 21(11):977–1000, 2003.