Marc Fournier · Jean-Michel Dischler · Dominique Bechmann

A new vector field distance transform and its application to mesh processing from 3D scanned data

Abstract In this paper we define a new 3D vector field distance transform to implicitly represent a mesh surface. We show this new representation is more accurate than the classic scalar field distance transform by comparing both representations with an error metric evaluation. Widely used Marching Cube triangulation algorithm is adapted to the new vector field distance transform to correctly reconstruct the resulting explicit surface. In the reconstruction process of 3D scanned data, useful mesh denoising operation is extended to the new vector field representation which enables adaptive and selective filtering features. Results show mesh processing with this new vector field representation is more accurate than with the scalar field distance transform and it outperforms previous mesh filtering algorithms. Future works are discussed to extend this new vector field representation to other mesh useful operations and applications.

Keywords 3D vector field distance transform · Volumetric mesh representation · Marching Cube triangulation algorithm · Scanned data mesh reconstruction · Triangle mesh denoising

1 Introduction

In the past decade a lot of improvements have been made in the field of 3D scanners to acquire and to digitize real world objects. The advancement in computer technologies have made possible the design of 3D scanners to respond to the increasing needs in many fields such as digitizing precious cultural heritage [1]. The most common output data structure produced by 3D scanners is range images. From these raw data, many operations are needed in order to produce the final mesh which represents the real object geometry. All these operations are achieved in the scalar field distance transform (SFDT) domain which produces good results for each step of the reconstruction procedure. An initial mesh registration procedure [2] is performed to express all range images in the same coordinates system. The first operation in the SFDT domain is mesh fusion [3] to integrate all range images into a unique representation, followed by mesh repair [4] to fill holes in the model, and then mesh smoothing [5] to remove acquisition noise introduced by the scanner and finally mesh simplification [6] to produce a more compact model without loss of details. The distance transform is an implicit representation of a surface and after the processing is done in this domain a triangulation algorithm such as Marching Cube [7] or Marching Triangle [8] is used to produce the final result.

In this paper we present a new and more accurate vector field distance transform (VFDT) to implicitly represent a range image or a previously reconstructed 3D mesh. Such as for the classic SFDT, we use a volumetric voxel grid to compute the implicit representation. Then we adapt Marching Cube triangulation algorithm used with SFDT to correctly reconstruct the resulting mesh from our new VFDT. Finally, we adapt a mesh adaptive filtering algorithm to the VFDT to remove acquisition noise and we propose a selective filtering feature based on this new VFDT representation. The remaining parts of the paper are organized as follows: Section 2 overviews related works. Section 3 presents our new VFDT and the adapted triangulation algorithm to this VFDT representation. Section 4 describes an error metric to evaluate and quantify the accuracy of the VFDT over the SFDT representation and other mesh processing algorithms. Section 5 presents the adaptive and selective filtering algorithm design based on the VFDT representation. Finally in Section 6, before concluding we propose some results of mesh denoising with our new VFDT compared with the SFDT representation and previous mesh smoothing algorithms.

2 Related work

The classic SFDT is an important implicit representation of a mesh. It is widely used in the final mesh reconstruction procedure from 3D scanned data. It has the advantage of being able to process every step of the reconstruction procedure in the same data structure and it produces good results at each step with relatively simple algorithms. The first mesh fusion step is mandatory to integrate all range images into a unique representation. Some volume-based methods [9, 10] use a voxel grid to organize the datasets and perform mesh fusion. Surface-based methods [11, 12] which work locally on the overlap region have been proposed. Most of these methods require range images taken from a unique scanner point of view as input data. Recent portable and hand-held scanners [13] produce full 3D datasets and only few fusion methods such as the distance transform method can handle these datasets.

Mesh smoothing and denoising is another important and major task in mesh processing to reduce acquisition noise introduced by the scanner and to correct previous steps reconstruction errors from scanned data. Many methods inspired by signal processing theory have been proposed to achieve this task. They are based on different kinds of low-pass filter designs and they work locally and iteratively on the data structure of the surface. Some of them [14, 15, 16] filter and modify directly the vertices positions. Others [17, 18, 19] work on the normal vector of triangles and they are adaptive according to one or more local surface parameters to preserve geometric features of the surface. Relying on different strategies, some methods [20, 21, 22] evaluate higher level information of the surface such as curvature or

energy parameters and the filtering algorithms are based on these higher level information. Other methods such as the level set [23] have introduced some alternative surface representations and they work on these representations instead of working directly on the mesh to perform surface operations including mesh smoothing.

The SFDT is also an alternative mesh representation useful for many mesh processing operations as previously introduced in the context of reconstruction procedure from scanned data. It is also useful for other mesh operations such as morphing [24] and skeletonization [25]. In this paper we introduce a new VFDT alternative representation. We show this new representation itself is more accurate than the classic SFDT. This VFDT representation is suitable for mesh processing operations such as mesh fusion and mesh smoothing performed on the SFDT representation. Since the VFDT is more accurate compared to the SFDT representation, it increases the results quality of the mesh processing operations on distance transforms. In this paper we show its application and performances to adaptive and selective mesh denoising.

Previously introduced mesh smoothing [5] by applying a 3D convolution to its SFDT is extended to the VFDT representation. We propose a generalization of a 2D image processing mean adaptive filter presented in [26] and we apply it to the VFDT. We also introduce an additional selective feature filter based on the VFDT representation. Other recent works that show good results have extended 2D image processing filters for denoising 3D meshes. For example in [27] the angle between triangles normal vector is used as an analogy to 2D image parameters to determine the filter area. Many mesh smoothing methods diffuse information along the surface and cannot effectively smooth regions that are not manifold. We propose a new method that is suitable with scanned data which often contain discontinuities and non-manifold regions.

3 Vector field distance transform definition

The VFDT of a range image or a previously reconstructed 3D mesh is defined with a 3D cubic grid created inside the bounding box of the mesh. Each single cube or cell of the grid is called a voxel. For each voxel the closest point on the mesh surface is found. In the classic SFDT definition only the distance between the voxel and that closest point found on the surface is saved in each voxel. In the VFDT, the 3D vector starting from the voxel and pointing to that closest point found on the surface is saved in each voxel. Two Booleans information are also saved in each voxel to complete the VFDT definition. We need to know if the voxel is in front or behind the surface according to the surface normal. This information is useful for the Marching Cube triangulation algorithm. A simple dot product between the surface normal at the closest point found on the surface and the previously saved vector information gives us the answer, a positive result means the voxel is behind the surface and a negative one means it is in front. In cases of open meshes or meshes with holes, we also need to know if the closest point found on the surface is on the boundary of the surface. This information is useful for the mesh fusion algorithm. If the closest point found on the surface is on a boundary edge or vertex then the boundary information is true for that voxel.

This new VFDT is more precise and complete than the SFDT representation because at each voxel we know exactly the distance and the orientation of the closest point on the surface. With the SFDT definition only the distance to the surface is known at each voxel. In theory the zero-set $f(x,y,z) = 0$ of the distance transform defines the mesh surface. This is true for both distance transforms, when the distance equal zero in the SFDT and when the vector length is zero in the VFDT representation. In practice with a discrete and finite grid resolution we need to fix a threshold to determine if a voxel is on the surface. With the SFDT representation it introduces an approximation error on the surface vertices. In the VFDT representation we use the small residual vector of the surface voxels to correct the vertices coordinates and retrieve the exact surface position.

3.1 Marching Cube adaptation

The Marching Cube is the most widely used triangulation algorithm to reconstruct a mesh from a SFDT. We adapt this algorithm to correctly reconstruct the mesh from our VFDT representation. The overall algorithm is the same with the VFDT except for the interpolation part to find the new triangle vertices on the cubes edges. To determine whether a cube needs to be triangulated or not and which edges to interpolate, we use the Boolean information of neighbor voxels being in front or behind the surface. This is equivalent as using the distances sign in the SFDT to determine the same parameters. Then the SFDT interpolation method is replaced with the new VFDT method. Fig. 1 is a 2D example which shows the advantage of the VFDT over the SFDT representation in terms of the interpolation method.

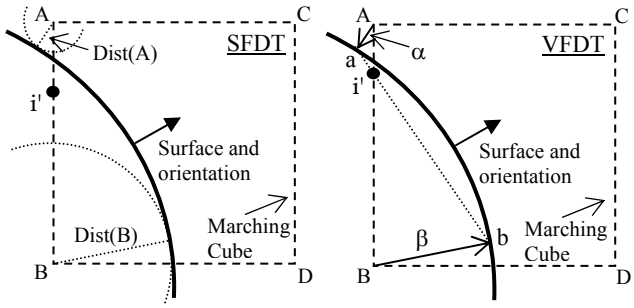

Fig. 1 SFDT and VFDT surface interpolation

In Fig. 1 example, A, B, C and D are neighbor voxels of the distance transforms and the doted square represents the "marching cube". In the SFDT representation, Dist(A) and Dist(B) are the scalar minimal distances to the surface from voxels A and B. In the VFDT representation, α and β are the vectors from voxels A and B to the closest points a and b on the original surface. At reconstruction step with the Marching Cube algorithm, we need to interpolate a new surface vertex i' on edge AB. With the SFDT representation, adding the two distances Dist(A) and Dist(B) leads to a first incoherence according to edge AB length. In general cases, either there is no solution or two solutions (infinity of solutions in 3D) for the intersection of the two circles of radius Dist(A) and Dist(B) which are centered on A and B. So the rough interpolation is made

with the ratio of distances Dist(A) and Dist(B) transposed on edge AB length which leads to a greater error.

With the VFDT representation, a and b original surface points are used to interpolate vertex i' on edge AB at intersection with interpolated surface segment ab. In general 3D cases, edge AB and interpolation segment ab will not intersect so the closest point to segment ab on edge AB is found and it is the new surface interpolated vertex i'. This interpolation method gives a better approximation of the original surface at same grid resolution compared to the SFDT representation. We define vector u = AB and vector v = ab. Then the interpolated vertex i' = λu and Equation 1 shows how to find λ.

$$\lambda = \frac{(u \circ v)(v \circ \alpha) - (v \circ v)(u \circ \alpha)}{(u \circ u)(v \circ v) - (u \circ v)^2} \qquad (1)$$

In Equation 1, λ is the factor which resize vector u to obtain the interpolation vertex i' on edge AB. The dot (◦) operator defines a dot product between vectors. At closest points in 3D between the two infinite lines defined by AB and ab, the vector joining these points is perpendicular to both lines resulting in dot products equal to zero between this vector and each vector u and v. This condition is used to solve a linear system which expresses the two infinite lines defined by AB and ab in terms of parametric equations and it leads to Equation 1 described in [28]. There is more than one method to obtain interpolation vertex i' but this one is the most efficient in terms of implementation. It uses only the two vectors u and v with the additional vector α between them and it uses only simple dot products compared to other methods which use projection planes and cross products as well. With this formulation it is also easy to isolate particular cases such as when the two vectors u and v are parallel, i.e. the denominator of Equation 1 is zero. If λ is outside interval [0,1] then the interpolation vertex i' coordinates are either voxel A or B coordinates.

Fig. 2 shows an example of Marching Cube triangulation over a SFDT and a VFDT of the same object. In Fig. 2 example, the SFDT and VFDT of a mesh which defines a 3D step function have been computed. Then the Marching Cube algorithm has been applied to triangulate both of them, using the standard algorithm over the SFDT and the modified one over the VFDT representation. The same coarse grid resolution at same spatial position has been defined for both distance transforms to highlight the differences and advantages of the VFDT representation. With the SFDT sharp edge A has been respected, there are unwanted inflection points at B and E and sharp edges at corners C, D, and F have been truncated. With the VFDT only sharp edge at corner C has been truncated because of the very low grid resolution used in this example. At same grid resolution, the VFDT has a more accurate representation of the underlying mesh. If we refine the grid resolution, the VFDT representation is still more accurate but the difference between both representations gets smaller. There is another advantage of the VFDT at mesh boundary and around holes. It represents exactly the same original mesh boundary whereas the SFDT representation can shift the boundary slightly inside or outside the original one depending on the grid position and resolution.

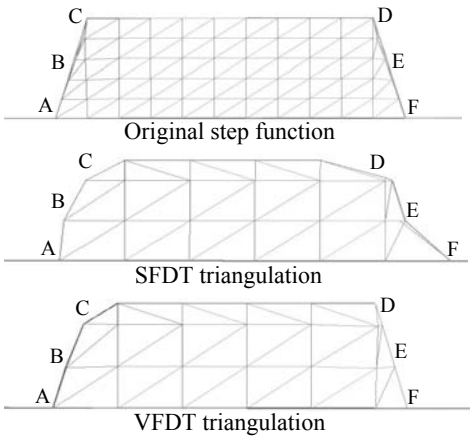

Fig. 2 SFDT and VFDT step function model triangulation

Fig. 3 shows another advantage of the VFDT representation in terms of the triangles size and distribution produced by the Marching Cube algorithm on a half sphere model. In Fig. 3 example both raw triangulations are shown without any mesh simplification algorithm. Both results have the exact same amount of triangles and vertices. On the SFDT triangulation, we see the usual Marching Cube elevation lines produced with unwanted small degenerated triangles. The VFDT representation produces a more uniform triangulation without any small degenerated triangle. The VFDT solid shaded model is therefore visually more uniform than the SFDT model. This VFDT representation advantage can save a simplification step for specific applications.

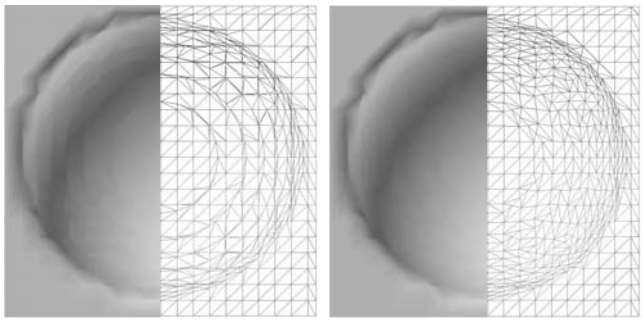

Fig. 3 SFDT and VFDT half sphere model triangulation

3.2 Efficient implementation

In this paper we compute the VFDT of a triangle mesh, we perform denoising algorithms on the implicit representation and we triangulate a final resulting mesh. In practice we do not need to evaluate the VFDT for each grid voxel inside the mesh bounding box. Only the voxels on and around the surface in a local neighborhood are of interest for most applications which determine the useful thickness of the narrow band around the surface to consider. For example in the denoising process based on 3D convolution filters, it is the convolution kernel size which determines the useful narrow band thickness to consider. In the fusion process it is the maximum distance between two overlapping surfaces which are considered to be the same model part. This parameter is directly related to the scanner precision. In computing the VFDT from a mesh, we need to find the closest point on the surface for each voxel and it is a time consuming task. Instead of finding the closest point on each

triangle to find the overall closest point, we use a projection plane described in [3] to limit the search to a few wisely chosen triangles.

The original Marching Cube algorithm does not preserve surface topology in particular cases and does not represent sharp features well. Many Marching Cube extensions have been proposed to improve the basic algorithm and our new VFDT representation is suitable with the major extensions introduced. We have implemented a topology preserving simplification extension [29] and a feature sensitive extension [30] to correctly reconstruct preserved topology and sharp featured surfaces from the VFDT. Our Marching Cube implementation uses two lookup tables, one for every cube configuration and one for edges interpolation. Optimizations such as saving pre-calculated edges interpolation are integrated in the algorithm to increase its performances.

4 Vertex to surface error metric evaluation

To evaluate and quantify the difference on larger meshes between the SFDT and VFDT representations, we define a vertex to surface error metric. This error metric will also be useful to compare the VFDT adaptive filtering algorithm to previous mesh smoothing algorithms. The vertex to surface error metric computes the difference between a mesh to evaluate and a reference mesh. It is based on the vertex to vertex error metric introduced in [19] which computes the minimal Hausdorff distance. In the vertex to vertex error metric, for each vertex of the mesh to evaluate, the distance to the closest vertex in the reference mesh is computed. In our vertex to surface error metric, for each vertex of the mesh to evaluate, instead of considering only the vertices of the reference mesh, we consider the whole surface and we evaluate the shortest distance to the reference mesh. This gives a more accurate evaluation of the error between two meshes. The vertex to surface error metric ε_v is defined by Equation 2.

$$\varepsilon_v = \sqrt{\frac{1}{3A(M')}\sum_i A(v'_i)\,dist(v'_i, M)^2} \qquad (2)$$

In Equation 2, M is the reference mesh and M' is the mesh to evaluate. Vertices v'_i are the mesh to evaluate vertices. A(M') is the total area of M'. $A(v'_i)$ is the area of all triangles incident to the vertex v'_i. $dist(v'_i, M)$ is the minimal distance between the vertex v'_i and the reference mesh M. For each vertex of the mesh to evaluate, the shortest distance to the reference mesh is computed and this distance is weighted by the area of all triangles incident to the vertex. This gives a local difference between the meshes. We add this parameter for all vertices and then we divide the result by three times the area of the mesh to evaluate because the area of each triangle has been considered three times, one for each triangle vertex.

In Fig. 2 example, the error metric of the triangulated mesh from the SFDT compared to the original step function mesh is 0.433 and the VFDT error metric compared to the same reference is 0.360. According to that error metric, the VFDT is approximately 16.9% more accurate compared to the SFDT representation. This is without considering the boundary errors of the SFDT because on such small meshes it would have largely distorted the result in favor of the VFDT representation.

5 Vector field distance transform adaptive filtering

Smoothing a mesh by applying a 3D digital convolution to its distance transform [5] is a linear operation which can be extended to the VFDT representation. The convolutions of 3D kernels which weights define low pass filters are applied independently on each vector coordinate of the VFDT. The result of this operation is a smoothed distance transform leading to a smoothed mesh after triangulation as shown in Fig. 4 with the 3D step function, before and after smoothing.

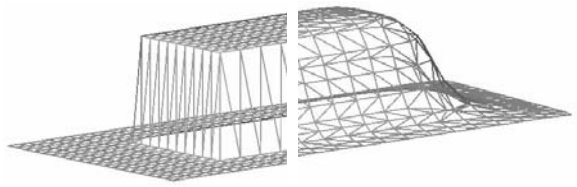

Fig. 4 Step function VFDT 3D convolution smoothing

In Fig. 4 example a mean kernel of 3x3x3 voxels was used to smooth the VFDT. This filtering concept is very flexible, only by changing the kernel size and weights distribution we obtain new filter designs. However this basic operation does not preserve the mesh features. We propose the generalization of a 2D image processing mean adaptive filter presented in [26] and we extend its application to our VFDT representation to preserve the model sharp edges and the mesh geometric features. This adaptive filtering algorithm is based on the VFDT local variance which is computed independently for each vector coordinate and compared to a specified noise variance threshold. If the VFDT local variance in a specific direction is smaller than the noise variance then the output of the adaptive filter in that direction is the basic 3D convolution computation. If the local variance is greater than the noise variance then the output is an adaptive combination of the initial VFDT value and the 3D convolution result according to the noise variance on local variance ratio. This adaptive filter g(X) is defined by Equation 3.

$$g(X) = \begin{cases} F_r & if \quad \sigma_n^2 > \sigma_r^2 \\ \left[1 - \left(\frac{\sigma_n^2}{\sigma_r^2}\right)\right] f(X) + \left(\frac{\sigma_n^2}{\sigma_r^2}\right) F_r & if \quad \sigma_n^2 \le \sigma_r^2 \end{cases} \qquad (3)$$

In Equation 3 the output g(X) is the VFDT filtered value at X=(x,y,z) in 3D space. F_r is the output of the local region basic 3D convolution filter defined with the kernel weights distribution. f(X) is the initial VFDT value at X. σ_n^2 is the noise variance which depends on the scanner specifications. If this parameter is unknown it can be specified manually or automatically selected to optimize the visual results. σ_r^2 is the local region variance considered by the convolution kernel and it is defined by Equation 4. If the noise variance is smaller or equal to the local region variance, the first summation term to evaluate g(X) is the feature preserving part and the second term is the equation filtering part. The noise on local region variance ratio determines the equation preserving and filtering weights.

$$\sigma^2_{r\,(x,y,z)} = \frac{\sum_i [(d_{i(x,y,z)}) - m_{(x,y,z)}]^2}{N} \quad (4)$$

In Equation 4 the local variance σ_r^2 is computed independently for each vector coordinate. It is a summation on all voxels considered by the filter kernel. d_i are the vector distance values to the surface considered independently for each coordinate. N is the total number of voxels of the local region. m is the average vector distance value of the local region. The resulting adaptive filter output vector g(X) is compared to the initial VFDT value f(X) with a dot product and if the result is negative then the VFDT Boolean information of being in front or behind the surface is switched. This adaptive filter operation does not modify the VFDT boundary information. Fig. 5 shows an example of the adaptive VFDT filtering on the step function which has been corrupted with a randomly distributed white noise on the left and after the filtering process on the right.

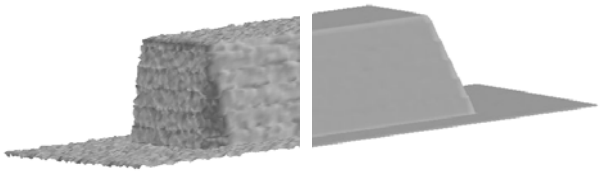

Fig. 5 Noisy step function VFDT adaptive filtering

In Fig. 5 a low-pass linear weights distribution kernel of 3x3x3 voxels has been used with a 0.006 noise variance and the filter has been applied iteratively seven times on the previous iteration result to filter the VFDT progressively until the best result was obtained according to the error metric comparison with the initial mesh without noise. Applying the filter more than seven iterations does not lead to a visually better result. According to its simple shape, over seven iterations the mesh becomes stable and there are no significant changes with the previous result. The model sharp edges are well preserved by the adaptive filtering algorithm which produces results without shrinkage of the models.

We applied the equivalent adaptive filtering to the SFDT of the same model and we repeated this procedure on other models. This algorithm leads to better results with the VFDT representation because the local variance is computed and compared to the noise variance independently for each vector coordinate. Moreover the VFDT representation allows specifying a vector noise variance which can be compared independently for each vector coordinate to the local region variance in Equation 3. A different noise variance in each axis can depend on static characteristics of the scanner according to the scanning technology implementation. It can also be influenced by dynamic parameters at each acquisition such as the angles between the surface normal vector and the scanner optical head. This dynamic parameter which leads to confidence evaluation of measurements can be encoded with the dataset at each acquisition during the scanning session and restored at the filtering process to evaluate the local noise variance. The VFDT adaptive filter can be configured to remove noise induced by specific scanning hardware. Fig. 6 shows this VFDT selective noise variance filtering feature advantage over the SFDT adaptive filtering algorithm. Noise has been added to the half sphere model only in Z axis.

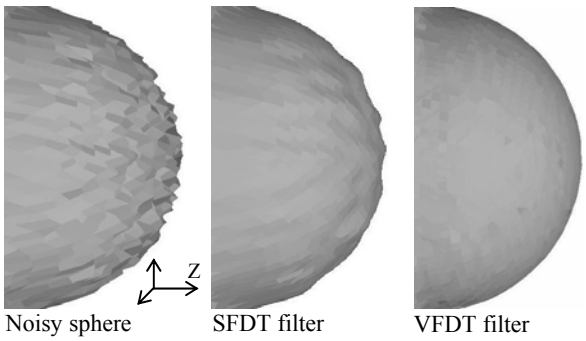

Fig. 6 Noisy sphere SFDT and VFDT adaptive filtering

In Fig. 6 example the best results according to the error metric are shown for both filters. The VFDT noise variance has been set to (0, 0, 0.01) and for the SFDT it has been set to 0.01. For both filters a 3x3x5 mean kernel has been used to accentuate the filtering effect in Z axis direction. The best results are reached after 6 iterations for the VFDT filter and after 8 iterations for the SFDT filter. At best results, the error metrics compared to the initial sphere without noise are 0.182 for the SFDT filter and 0.147 for the VFDT filter. The VFDT filter is approximately 19.2% better, it converges to its best result in less iterations and it removes specific directional noise more accurately without altering the global shape in other directions. Applying the filter a few more than 8 iterations on the SFDT produces a more uniform result but the mesh is considered over-smoothed because the overall shape is altered compared to the initial sphere without noise. The error metric is then greater than the one at 8 iterations. This is caused by the simultaneous SFDT filtering action in all directions including X and Y axis in which no noise was added.

In Fig. 6 example we chose the noise variance value according to the average noise amplitude added to the initial sphere model. With noisy scanned data this parameter should be fixed according to the scanner specifications. If this information is unavailable, the noise variance is automatically selected. The whole dynamic range of noise variances is tested to make sure the result is the best of all after comparison between them. The noise variances dynamic range is defined by a minimum and a maximum value ($\sigma_n^2{}_{min} \leq \sigma_n^2 \leq \sigma_n^2{}_{max}$). Since our VFDT filter is an adaptive algorithm, it should have both adaptive cases (feature preserving) and non-adaptive cases (noise filtering) during its process otherwise the algorithm has no more purpose. To find the noise variances dynamic range boundaries, the percentage of adaptive cases ($\sigma_n^2 \leq \sigma_r^2$) according to Equation 3 is computed during the filtering process. Under $\sigma_n^2{}_{min}$ value there are 100% of adaptive cases and over $\sigma_n^2{}_{max}$ value there are 0% of adaptive cases.

6 Results

This section presents mesh processing results with the VFDT new representation. First the VFDT and the SFDT of a mesh are computed and back-triangulated to evaluate and compare both results according to the error metric presented in section 4. Then we apply the SFDT and VFDT filters to 3D models and we compare their results. We also

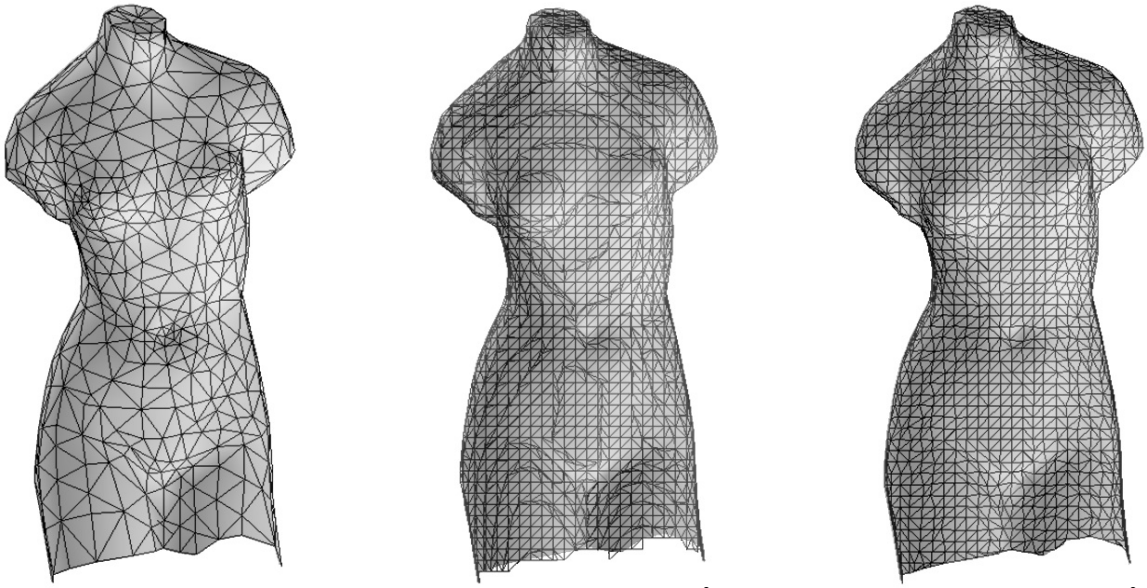

Fig. 7 SFDT and VFDT triangulation of the Venus model

apply other mesh denoising methods to the same models to compare our VFDT filter to existing filtering algorithms.

Fig 7 shows the triangulation results of the Venus model with the SFDT and VFDT representations. In both cases, the distance transforms have been computed and then they have been triangulated without any other processing operation. This procedure is used to compare both representations and their triangulation algorithm. The same grid resolution has been selected for both results and this parameter has been chosen to have a fine description according to the initial model resolution. The SFDT result has these usual Marching Cube elevation lines produced with small degenerated triangles. The VFDT representation produces a more uniform triangulation. At the model bottom end, we see that the SFDT result do not match exactly the initial model underneath the mesh. The VFDT do not have this problem at same grid resolution. The results error metrics compared to the initial model are shown in Fig. 7 and the VFDT result is 11.2% better than the SFDT result.

Fig. 8 shows filtering results on the head of a noisy camel model. The original clean model is used as reference mesh and we added artificial randomly distributed white noise to this model. This procedure is useful to compare the VFDT filter result to others based on the vertex to surface error metric. Results for the SFDT, VFDT and MMSE filters are shown. We decided to compare our filter design to the MMSE filter [17] because it is a recent adaptive algorithm which has been found more efficient than other algorithms such as the Laplacian, mean and median filters. This filter is an adaptive minimum mean squared error filter applied on the mesh triangles normal vector which also uses a local region variance parameter compared to a noise variance to determine the adaptive cases. Fig. 8 also shows the filters parameters used to produce these results. The three filters behave in a similar iterative progression until the best result is obtained according to the error metric.

For all three filters, the noise variance has been automatically selected from the whole dynamic range to be sure the results shown are the best of all. The optimal noise

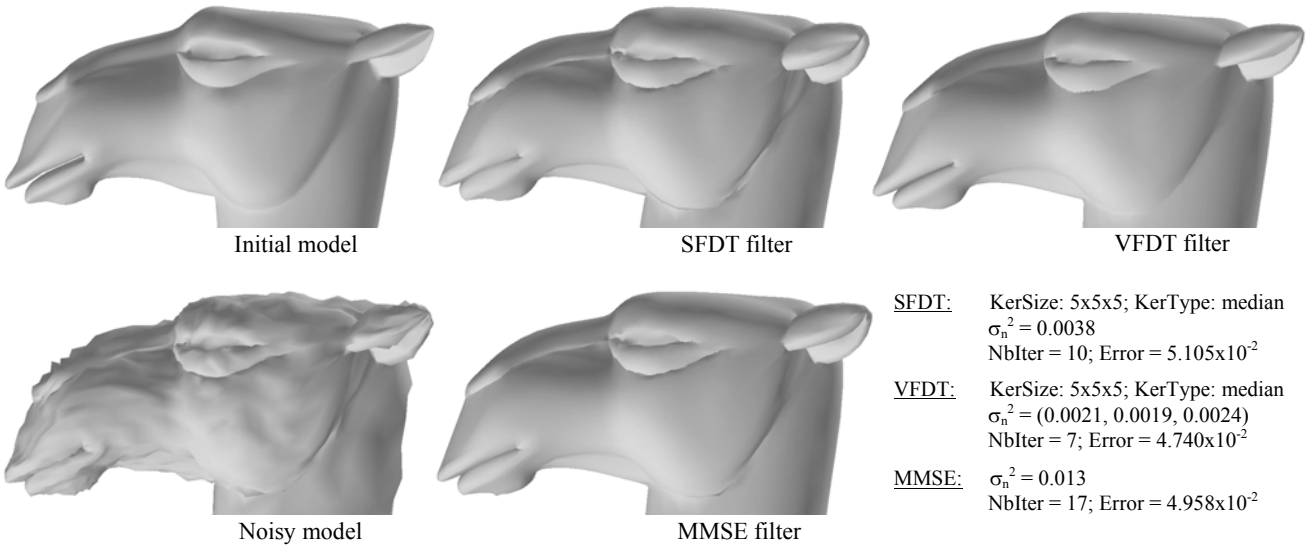

Fig. 8 SFDT, VFDT and MMSE filter results on the camel model

variance is not the same for all filters and there is a difference of almost an order of magnitude between MMSE filter noise variance and the distance transforms filters noise variances because these thresholds are used to compare local region variances which are computed from different data types for each filter. The three results are very good compared to the highly noisy input model. All filters are able to retrieve the original shape in the mesh more uniform regions. At sharp edges such as along the nose and around the eye the SFDT and MMSE filters results are less accurate compared to the VFDT filter. At these sharps edges and particularly along the jaw, the MMSE filter is a bit more accurate than the SFDT filter.

Fig. 9 shows the vertex to surface error metric evolution in the iterative filtering process for the three filters. During the first iterations the results get better until the best results are reached at minimal errors for each filter. Beyond these best results the errors are growing and the meshes are over-smoothed in all cases. The best results are reached at a relatively low number of iterations for all filters. The VFDT filter converges more quickly than the others, its best result is 7.15% better than the SFDT filter and 4.4% better than the MMSE filter best results but it also over-smoothes the mesh at a greater rate than the other filters.

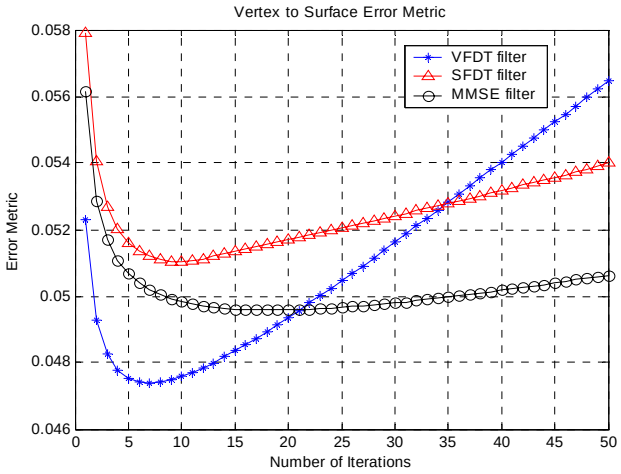

Fig. 9 Error metric for the camel model results

We have tested these filtering algorithms on a total of eight different models which have a broad range of mesh features and their compared behaviors are similar to the camel model results. The VFDT filter produces more accurate results than the others in all cases ranging from 3.1% to 11.6% better results than the MMSE filter. Results with less significant difference between VFDT and MMSE filters are obtained with more rounded shape meshes and more significant differences between these two filters are observed on models which have more sharp edges. Since the VFDT filter works with a noise variance threshold, its limitation is reached when the noise variance has the same magnitude or is greater than the local mesh features variance resulting in smoothed mesh features and lost of details in the filtering process. In practice, with real noisy data coming from 3D scanners, these cases are rarely encountered since we usually choose a scanner which has a noise induction magnitude smaller than the finest features to be scanned in order to obtain an accurate result. The VFDT filter performances are closely related to its noise variance threshold. If this important parameter is wrongly selected, the resulting mesh features could be smoothed or some significant noise could be left after the mesh filtering process. The automatic noise variance selection has been introduced to overcome this eventuality in cases which the noise variance parameter is unknown.

The camel model is composed of 69 092 triangles and one VFDT filter iteration on this model took 91ms with a Pentium 4 CPU computer at 3.03GHz clock compared to 134ms for a MMSE filter iteration. A VFDT filter iteration takes less time because of the algorithm simplicity which consists mainly in successive multiplication and addition operations to compute the convolution. The MMSE filter consists in many parameters computation before updating the mesh vertices position. In the context of mesh reconstruction from scanned data the distance transform is already computed to perform previous step operations. In other cases such as the present algorithm evaluation, we need to consider the overhead of distance transform computing, which is a very time consuming task, and final model triangulation. For the camel model it took 3.76s to compute the distance transform and it took only 2ms to triangulate the resulting model.

Fig. 10 shows filtering results on the face of a scanned kitten sculpture model. This model is a single raw range image scan which is used to demonstrate denoising on real noisy data with noise introduced by the scanner at the acquisition step. This model is also used to compare our VFDT filter with the mean and median iterative mesh denoising algorithms introduced in [19]. A 3x3x3 voxels Gaussian kernel ($\alpha = 3$) and a scalar noise variance of 4.6×10^{-4} have been used with the VFDT filter. The number of iterations to obtain the visually best results for each filter is shown in Fig. 10.

In cases of real noisy data without any clean reference mesh, it is not possible to use an error metric to stop the iterative process at best result. This is a known problem and other iterative filtering methods [17, 19] did not provide any solution yet. They subjectively estimate the best result visually to stop their iterative processes. However we do use our surface to vertex error metric to estimate the range of probably best results. At each iteration, we compute the difference between the current result with the previous one and when this difference becomes less significant it means we are getting closer to the best result. As for the other methods, the choice of the best result remains a subjective visual criterion.

In Fig. 10 results, low curvature mesh regions such as the kitten forehead show the VFDT filter removes acquisition noise better than other filters. The mean filter also removes much noise in low curvature regions but it is unable to preserve the mesh features, the right eye and eyelash region is blurred. The median filter preserves the mesh features much better but it also preserves a significant part of the noise. Our VFDT filter preserves the mesh features as well and its adaptive algorithm produces the overall best result.

7 Conclusion and future work

A new VFDT representation for meshes has been introduced in this paper. According to a reliable error metric evaluation, the new representation was found more

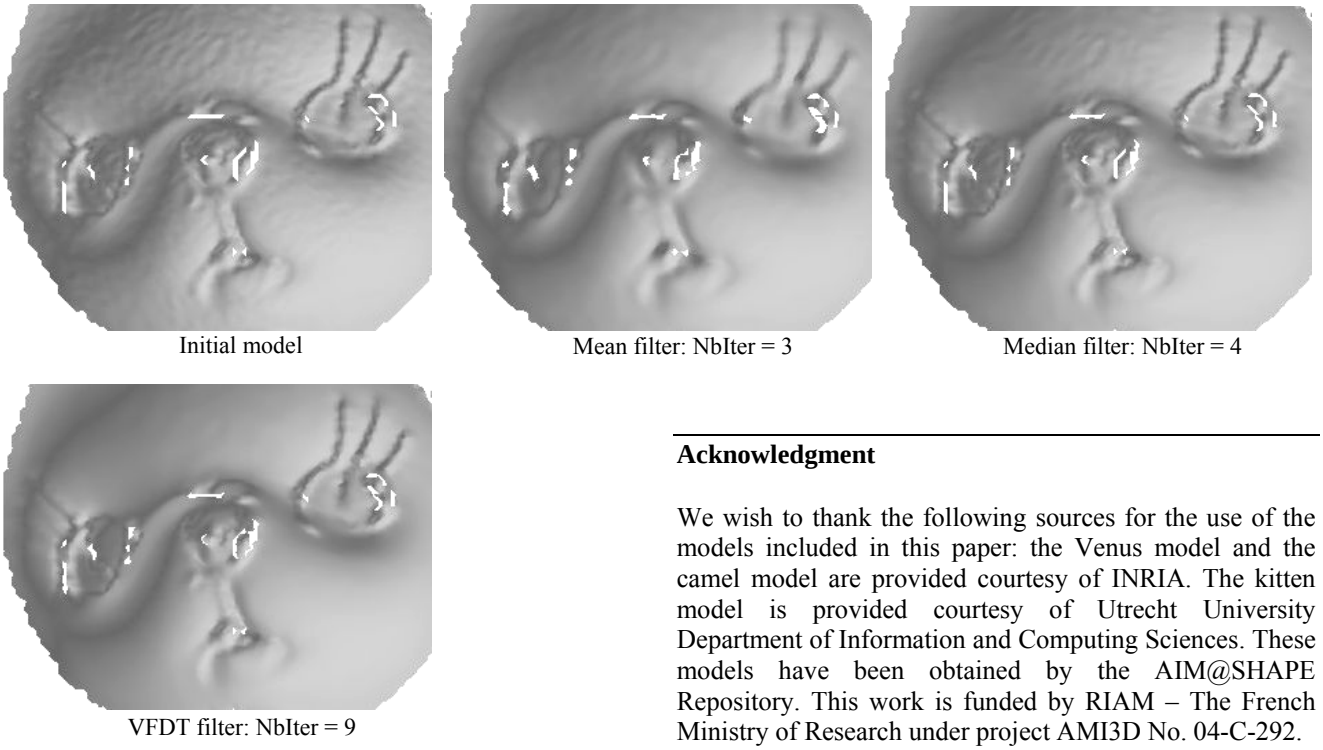

Fig. 10 Mean, median, VFDT results on the kitten model

accurate than the classic SFDT. We adapted the Marching Cube triangulation algorithm to our new representation to correctly reconstruct the final model. Compared to the SFDT, it produces a more uniform triangulation with a better boundary representation. A new method for mesh denoising has been introduced and adapted to the VFDT. It is an adaptive algorithm which removes the noise and preserves geometry features of the mesh. The new filter design is more versatile than previous ones. The size, symmetry and type of filter are fully customizable to adapt the mesh smoothing to different kinds of noises. If the noise characteristics are known according to the scanner, even if they are directional or dynamically evolving, the filter is able to consider these detailed specifications to efficiently remove specific noise on triangle meshes. Results show the new VFDT adaptive filtering is more accurate than the equivalent algorithm on the SFDT and it outperforms previous mesh denoising algorithms in terms of an error metric comparison.

Future works based on previous fast algorithms which compute regular voxel grid SFDT and its triangulation with graphics hardware will be investigated to adapt these algorithms to the VFDT representation and its modified triangulation. Other mesh operations usually performed on SFDT will be extended to the VFDT to obtain more accurate results. With this new and improved VFDT definition, we will investigate the design of new useful mesh operations which are not applicable to the SFDT representation. The effect of new filter types such as high-pass and band-pass will be tested on the VFDT. Future works will also include the design of an algorithm to compute the VFDT directly from a point cloud. Finally we will investigate the unsolved problem of finding the best result with iterative filtering methods.

Acknowledgment

We wish to thank the following sources for the use of the models included in this paper: the Venus model and the camel model are provided courtesy of INRIA. The kitten model is provided courtesy of Utrecht University Department of Information and Computing Sciences. These models have been obtained by the AIM@SHAPE Repository. This work is funded by RIAM – The French Ministry of Research under project AMI3D No. 04-C-292.

References

1. Rocchini C., Cignoni P., Montani C., Pingi P., Scopigno R.: A Low Cost 3D Scanner Based on Structured Light. EUROGRAPHICS 2001, Manchester, UK, Computer Graphics Forum, Vol. 20, No. 3. (2001)
2. Besl P.J., McKay N.D.: A Method of Registration of 3D Shapes. IEEE Transactions on Pattern Analysis and Machine Intelligence, Vol. 14, No. 2, 239-256. (1992)
3. Hilton A., Stoddart A.J., Illingworth J., Windeatt T.: Implicit Surface-Based Geometric Fusion. Computer Vision and Image Understanding, Vol. 69, No. 3, 273-291. (1998)
4. Davis J., Marschner S., Garr M., Levoy M.: Filling Holes in Complex Surfaces Using Volumetric Diffusion. First International Symposium on 3D Data Processing, Visualization, and Transmission, Padua, Italy. (2002)
5. Sealy G., Wyvill G.: Smoothing of Three Dimensional Models by Convolution. The 14th International Conference of the Computer Graphics Society, Pohang, South Korea, 184-190. (1996)
6. Nooruddin F.S., Turk G.: Simplification and Repair of Polygonal Models Using Volumetric Techniques. IEEE Transactions on Visualization and Computer Graphics, Vol. 9, No. 2, 191-205. (2003)
7. Lorensen W.E., Cline H.E.: Marching Cubes: A High Resolution 3D Surface Reconstruction Algorithm. ACM SIGGRAPH 1987, Anaheim, USA, Computer Graphics, 163-169. (1987)
8. Hilton A., Stoddart A.J., Illingworth J., Windeatt T.: Marching Triangles: Range Image Fusion for Complex Object Modelling. International Conference on Image Processing, Lausanne, Switzerland, Vol. 1. (1996)
9. Curless B., Levoy M.: A Volumetric Method for Building Complex Models from Range Images. SIGGRAPH 1996, New Orleans, USA, Computer Graphics, 221-227. (1996)

10. Hoppe H., DeRose T., Duchamp T., McDonald J., Stuetzle W.: Surface reconstruction from unorganized data points. SIGGRAPH 1992, Chicago, USA, Computer Graphics, 71-78. (1992)
11. Turk G., Levoy M.: Zippered Polygon Meshes from Range Images. SIGGRAPH 1994, Orlando, USA, Computer Graphics, 311-318. (1994)
12. Rutishauser M., Stricker M., Trobina M.: Merging Range Images of Arbitrary Shaped Objects. IEEE Conference on Computer Vision and Pattern Recognition, Seattle, USA, 573-580. (1994)
13. Harvey E., Arsenault M., Lavoie J-F., Belanger B., Boucher M-A..: Compact and Portable 3D Camera for Space Applications. Third International Conference on 3D Digital Imaging and Modeling, Quebec, Canada, 31-37. (2001)
14. Taubin G.: A Signal Processing Approach to Fair Surface Design. ACM SIGGRAPH 1995, Los Angeles, USA, Computer Graphics, 351-358. (1995)
15. Levy B., Mallet J.-L.: Constrained Discrete Fairing for Arbitrary Meshes. Tech. Report, ISA-GOCAD (Inria Lorraine/CNRS) ENSG, Vandoeuvre, France. (1999)
16. Fleishman S., Drori I., Cohen-Or D.: Bilateral Mesh Denoising. ACM SIGGRAPH 2003, San Diego, USA, ACM Transactions on Graphics, Vol. 22, No. 3. (2003)
17. Mashiko T., Yagou H., Wei D., Ding Y., Wu G.: 3D Triangle Mesh Smoothing via Adaptive MMSE Filtering. Fourth International Conference on Computer and Information Technology, Wuhan, China. (2004)
18. Ohtake Y., Belyaev A.G., Seidel H.-P.: Mesh Smoothing by Adaptive and Anisotropic Gaussian Filter Applied to Mesh Normals. Vision, Modeling, and Visualization 2002, Erlangen, Germany. (2002)
19. Yagou H., Ohtake Y., Belyaev A.G.: Mesh Smoothing via Mean and Median Filtering Applied to Face Normals. Geometric Modeling and Processing Theory and Applications, Wako, Japan, 124-131. (2002)
20. Bobenko A.I., Schröder P.: Discrete Willmore Flow. Third Eurographics Symposium on Geometry Processing, Vienna, Austria, 101-110. (2005)
21. Desbrun M., Meyer M., Schröder P., Barr A.H.: Implicit Fairing of Arbitrary Meshes Using Diffusion and Curvature Flow. ACM SIGGRAPH 1999, Los Angeles, USA, 317-324. (1999)
22. Hildebrandt K., Polthier K.: Anisotropic Filtering of Non-Linear Surface Features. EUROGRAPHICS 2004, Grenoble, France, Computer Graphics Forum, Vol. 23, No. 3, 391-400. (2004)
23. Sethian J.A.: Level Set Methods and Fast Marching Methods. Cambridge University Press. (1999)
24. Cohen-Or D., Levin D., Solomovici A.: Three-Dimensional Distance Field Metamorphosis. ACM Transactions on Graphics, Vol. 17, No. 2. (1998)
25. Nikolaidis N., Pitas I.: 3D image processing algorithms. John Wiley & Sons, New York, USA. (2001)
26. Gonzalez R.C., Woods R.E.: Digital Image Processing (2nd ed.). Prentice Hall. (2002)
27. Mao Z., Ma L., Zhao M., Xiao X.: SUSAN Structure Preserving Filtering for Mesh Denoising. Visual Computer, Vol. 22, No. 4, 276-284. (2006)
28. Gellert W., Gottwald S., Hellwich M., Kästner H., Künstner H.: Concise Encyclopedia of Mathematics (2nd ed.). Van Nostrand Reinhold, New York. (1989)
29. Zhang N., Hong W., Kaufman A.: Dual Contouring with Topology-Preserving Simplification Using Enhanced Cell Representation. IEEE Visualization, 505-512. (2004)
30. Kobbelt L.P., Botsch M., Schwanecke U., Seidel H.P.: Feature Sensitive Surface Extraction from Volume Data. ACM SIGGRAPH 2001, Los Angeles, USA, Computer Graphics, 57-66. (2001)

Marc Fournier • Jean-Michel Dischler • Dominique Bechmann
LSIIT – UMR 7005 – CNRS – Louis Pasteur University
Image Sciences, Computer Sciences and Remote Sensing Laboratory; Strasbourg, France
Tel.: +33 390.244.553
Fax: +33 390.244.455
E-mail: {fournier, dischler, bechmann}@lsiit.u-strasbg.fr

Marc Fournier is a PhD candidate in computer science at the Strasbourg University, France and a member of the LSIIT CNRS Lab since 2004. Before starting his PhD, he did his undergraduate bachelor degree in electrical engineering and his graduate master degree in computer vision, both at Superior technology school in Montreal, Canada within the Quebec university network. Its past master thesis and actual PhD thesis researches both address 3D scanner data processing and reconstruction algorithms to correctly reconstruct 3D objects from raw scanned data and to improve the final model quality.

Jean-Michel Dischler is a professor of computer science at the Strasbourg University and a researcher at the LSIIT CNRS Lab since 2001. He belongs to the computer graphics team (http://lsiit.u-strasbg.fr/) where he is supervising the realistic rendering and scientific visualization research activities. He is also member of the recently created INRIA project CALVI (http://math.u-strasbg.fr/calvi/). His research interests include texturing, texture synthesis, natural phenomena, real-time rendering, cultural heritage and volume rendering.

Dominique Bechmann is a professor of computer science at the Strasbourg University and a researcher at the LSIIT CNRS Lab since 1996. For 10 years now, she has also been head of the computer graphics research team in Strasbourg, France. From 1986 to 1989, she did her PhD at the IBM scientific center (Paris, France) on geometric modeling of anatomic organs. In 1990, she did a postdoc at the T.J. Watson research center (Yorktown Heigths, New York, USA) on deformation models. In 1991, she became an assistant professor and since then she is working on geometric modeling and virtual reality.