Efficient Triangular Surface Approximations using Wavelets and Quadtree Data Structures

Markus H. Gross, *Associate Member, IEEE*, Oliver G. Staadt, and Roger Gatti

Abstract—**We present a new method for adaptive surface meshing and triangulation which controls the local level-of-detail of the surface approximation by local spectral estimates. These estimates are determined by a wavelet representation of the surface data. The basic idea is to decompose the initial data set by means of an orthogonal or semi-orthogonal tensor product wavelet transform (WT) and to analyze the resulting coefficients. In surface regions, where the partial energy of the resulting coefficients is low, the polygonal approximation of the surface can be performed with larger triangles without loosing too much fine grain details. However, since the localization of the WT is bound by the Heisenberg principle the meshing method has to be controlled by the detail signals rather than directly by the coefficients. The dyadic scaling of the WT stimulated us to build an hierarchical meshing algorithm which transforms the initially regular data grid into a quadtree representation by rejection of unimportant mesh vertices. The optimum triangulation of the resulting quadtree cells is carried out by selection from a look-up table. The tree grows recursively as controlled by detail signals which are computed from a modified inverse WT.**
In order to control the local level-of-detail, we introduce a new class of wavelet space filters acting as "magnifying glasses" on the data.
We show that our algorithm performs a low algorithmic complexity, so that surface meshing can be achieved at interactive rates, such as required by flight simulators. However, other applications are possible as well, such as mesh reduction in complex data, FEM or radiosity meshing.
The method is applied on different types of data comprising both digital terrain models and laser range scans. In addition, quantitative investigations on error analysis are carried out.

Index Terms—**Surface Meshing, Triangle Approximations, Level-of-Detail, Quadtrees, Wavelet Transforms, Wavelet Space Filtering, Biorthogonal Wavelets, Mean-Square Error, Digital Terrain Modeling**

I. Introduction

POLYGONAL surface approximations are an essential preprocessing step in scientific visualization [1], since most modern graphics hardware supports the display of shaded and textured triangles. Nevertheless, in order to treat complex data sets efficiently, methods have to be found to reduce the number of triangles representing the data. This problem is not only striking in the field of digital terrain modeling and flight simulation, but also in many other applications, such as finite element, radiosity [2] or parametric surface meshing [3]. Hence, adaptive triangle reduction techniques were established in the past. Most of them try to find mathematical criteria for the importance of a particular mesh vertex, remove it if applicable and perform a local retriangulation of the mesh. [4] for instance analyzes single vertices in the mesh and defines a planarity criterion to decide on the removal of the vertex. In order to avoid cracks in the surface, a local Delaunay triangulation has to be performed. Quadtree-based methods [5] were proposed mostly for radiosity meshing, where the mesh is controlled by the illumination gradient. Other implementations are used for representing rectangular B-spline patches [3].
A lot of work has also been done for the topologically more challenging case of 3D isosurfaces. Based on analysis of topological problems [6], [7] arising with the marching cubes method, different approaches have been published computing adaptive triangulations or reductions of an existing isosurface mesh. Specifically, [8] used octrees to efficiently manage the surface polygons. As a result, the octree is usually built from the volume data set.
Although most of the existing methods work well within the above limitations and can be found in a broad range of applications, the basic issues arising from these approaches are as follows:
1. The criteria employed to thin the triangle mesh are usually based on simple local geometric surface features, such as planarity or Gaussian curvature. It is difficult to quantify global error bounds of the overall approximation.
2. The reduction of the triangle mesh is computationally expensive and once local retriangulations are performed, extensive work on data structures and list management is required.
3. There is no elegant way to focus the level-of-detail locally onto interesting data features — a property of increasing importance in complex data sets.
On the other hand, the wavelet transform, as presented in [9], [10] or [11] has been discovered for computer graphics: [12], [13] and [14] proposed volume rendering techniques, whereas [15] published a volume morphing method. Even approximate solutions of the radiosity equation can be achieved using WTs [16], as well as visualization of multidimensional features, such as in [17]. Others [18], [19] constructed multiresolution curves or surfaces.
All of these approaches employ the WT to expand the data and to control the parameters of the approximation within the mathematical bounds of the L^2-energy norm.
The goal of the following paper is to point out an alternative approach to the adaptive triangulation problem: the usage of the wavelet transform as an overall mathematical framework which controls the data approximation. In some sense, the WT provides a local spectral estimate of the data and describes local variations which can be harvested to govern the courseness of a surface mesh.
As opposed to existing methods [19], [20] our approach extends prior work significantly, since it performs on any type of wavelet, not only in the trivial case of linear splines. Therefore, our criteria for vertex removal are elaborated much differently and employ a modified inverse WT, which carries out the detail signals of each channel. Furthermore, the introduction of a new type of wavelet space filtering enables us to harvest one of the most striking properties of the WT: the localization. This filter helps to define local regions of interest.
Since our method targets at real-time applications, such as flight simulation or virtual reality, we combine the wavelet domain representation with quadtree based meshing. In contrast to existing look-up tables we propose an elaborated algorithm, which provides consistent triangulations, even in the case of level-two

The authors are with the Computer Science Department, Swiss Federal Institute of Technology (ETH), Zürich, Switzerland. E-mail: grossm@inf.ethz.ch.

transitions of adjacent nodes. In particular, it is shown, that the resulting mesh is much more flexible, whereas the look-up table has to be enlarged only moderately.

Some basic ideas of our method have been summarized in [21]. In the current paper, we have extended our investigations in terms of various important aspects. Specifically, we emphasize interpolation and boundary problems arising with the usage of the WT. Furthermore, our algorithm is elaborated, where all required look-up tables are given and explained. We thoroughly investigate the algorithmic complexity of the different processing steps, an aspect of fundamental importance in computer science. Finally, the performance of our triangulation method is analyzed on several data sets, ranging from digital terrain models to laser range scans.

In particular, the triangulation step is embedded in a modified inverse wavelet transform. A quadtree representation grows iteratively as the data is transformed back into the spatial domain. Due to some symmetries within the quadtree cells, the local triangulations can be computed from our look-up table. Thus, we avoid complex list management and perform the triangulation at nearly interactive rates.

The concept of our method is illustrated in Fig. 1. The initially regular surface data grid has to be transformed into a quadtree structure and each quadtree cell has to be triangulated using a look-up table. In order to decide, whether a particular mesh vertex can be removed, we first apply a WT onto the data and then iteratively reconstruct the detail signals. The amplitude of the detail signal is taken as a measure for the local frequency characteristics and decides on the removal of points. The dyadic scale of the standard WT allows to reconstruct the detail signals from the different frequency channels separately. After the first step each second data vertex of the grid is analyzed. Then, as the iteration proceeds the next detail signal is reconstructed and each fourth vertex is analyzed and so forth. This scheme enforces a loop consisting of a modified inverse WT to recover a particular detail signal and an analysis step to label unimportant coefficients. Applying wavelet space filtering allows an elegant control of the local level-of-detail of the triangulation and acts as a local “magnifier”. Furthermore, particular emphasis has to be given to the boundary problems.

Although the scope of our paper is to present a method for 2D surface meshing, it can also be extended to 3D to handle isosurfaces or volumes with tetrahedralizations [22]. Moreover, some of the different ideas encompassed by this method, such as the detail signal criterion and the wavelet space filters can also be used to govern existing meshing methods.

The organization of the paper is as follows: For reasons of readability, we describe the mathematical framework of the 2D wavelet transform for surface approximation and particular emphasis is given to the required extensions, such as modifications of the QM-Filter pyramids to figure out the inverse WT or finite intervals. Furthermore, mathematical formulations of filters in wavelet space are explained and their importance for level-of-detail control is stressed. The next section sheds light on the quadtree-based mesh representation we propose and shows how to derive local optimal cell triangulations from a look-up table. The algorithmic complexity of the method as well as an error analysis is elaborated in section IV. Finally, some examples from a digital terrain model of the Swiss Alps and from various laser range scans illustrate the superiority of the proposed method.

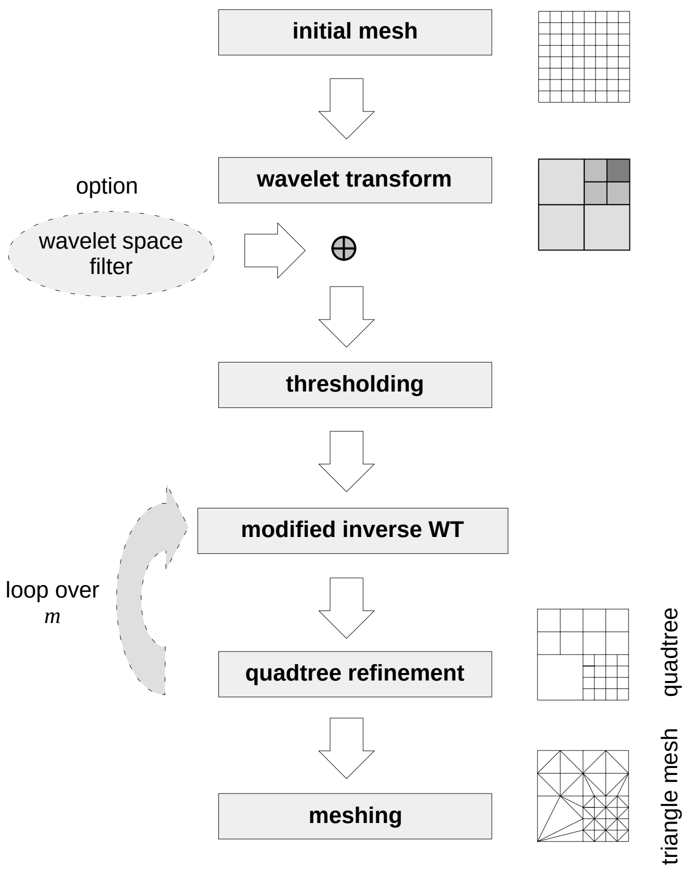


Fig. 1. Illustration of the basic concepts of the method: estimation of the surface parameters useing the local detail signal of the WT, point removal and quadtree based meshing of the remaining suface points.

II. Surface Approximation using Wavelets

A. The 2D Wavelet Transform

The 2D version of the Wavelet Transform (WT) expands any finite energy funtion $f(x,y) \in L^2(\mathbb{R}^2)$ using a set of similar basis functions $\psi_{a,b}(x,y)$. Its generic continuous form description is provided as the following inner product:

$$\mathrm{WT}_{f,\psi}\left(a_x,a_y,b_x,b_y\right) = \langle f,\psi_{a,b}\rangle = \int_{-\infty}^{\infty}\int_{-\infty}^{\infty}\psi_{a,b}(x,y)f^*(x,y)\,dxdy \tag{1}$$

with $a_x,a_y,b_x,b_y \in \mathbb{R}$.

The basis functions are derived from each other by scaling and shifting one prototype function $\psi(x,y)$ controlled by the parameters a_x,a_yandb_x,b_y respectively [9].

$$\psi_{a,b} = \frac{1}{\sqrt{|a_x a_y|}}\psi\left(\frac{x-b_x}{a_x},\frac{y-b_y}{a_y}\right) \tag{2}$$

$$\langle\psi_l,\psi_k\rangle = \delta(l-k), \delta : \text{Kronecker-delta-function}$$

Most discrete formulations of the 2D-WT comprise a tensor product extension along with a dyadic scaling of the bases with $a_x = a_y = 2$ and a unit shift $b_x = b_y = 1$, by which the respective bases are derived:

$$\begin{aligned}
\varphi^2_{mpq}(x,y) &:= 2^{-m}\varphi\left(2^{-m}x-p\right)\varphi\left(2^{-m}y-q\right)\\
\psi^{2,1}_{mpq}(x,y) &:= 2^{-m}\psi\left(2^{-m}x-p\right)\varphi\left(2^{-m}y-q\right)\\
\psi^{2,2}_{mpq}(x,y) &:= 2^{-m}\varphi\left(2^{-m}x-p\right)\psi\left(2^{-m}y-q\right)\\
\psi^{2,3}_{mpq}(x,y) &:= 2^{-m}\psi\left(2^{-m}x-p\right)\psi\left(2^{-m}y-q\right)
\end{aligned} \qquad (3)$$

$m : 1,\ldots,M$ iteration step. φ stands for the so-called scaling function. The upper index 2 denotes the dimension of the bases. Consequently, any finite energy function $f(x,y) \in L^2(\mathbb{R}^2)$ can be approximated by the bases elucidated above.

$$\begin{aligned}
f(x,y) = \sum_p \sum_q \Big(& c^M_{pq}\varphi^2_{Mpq}\\
& + \sum_{m=1}^{M}\left(c^{m,1}_{pq}\psi^{2,1}_{mpq} + c^{m,2}_{pq}\psi^{2,2}_{mpq} + c^{m,3}_{pq}\psi^{2,3}_{mpq}\right)\Big)
\end{aligned} \qquad (4)$$

c^m_{pq} denotes the coordinate of f in functional space with respect to the wavelet ψ^m_{pq}, i. e.

$$c^m_{pq} = \langle f, \psi_{mpq}\rangle \qquad (5)$$

Note, that the previous equation provides a multiresolution hierarchy enabling the control of the bounds of any approximation. For convenience, we will denote the coefficients simply with $\bar{c}_i$.

B. *Biorthogonal Wavelets*

The final design of the wavelet bases is usually figured out by further constraining the function's shape and mathematical properties. In most computer graphics applications [23] and [24] we require strict local support along with an appropriately smooth shape, symmetry and fast decay in frequency domain. Unfortunately, these competing properties cannot be satisfied with orthonormal wavelets. Chui [11] and Unser [25], however, independently developed a class of B-spline wavelets which meet the upper requirements. The bases are not orthogonal to each other, but it is possible to set up a so-called dual frame to perfectly reconstruct the signal from the transform.

Specifically, besides of scaling function φ and wavelet ψ the entire transform is defined by a dual scaling function $\tilde{\varphi}$ and a dual wavelet $\tilde{\psi}$.

The biorthogonal B-spline scaling functions of order j can be defined as a recurrence relation and are assumed to be the cardinal B-spline bases:

$$\varphi_j(x) := \left(\varphi_{j-1} * \varphi_1\right)(x) = \int_0^1 \varphi_{j-1}(x-t)\,dt,\ j \geq 2. \qquad (6)$$

That is, the bases are derived from each other by self-convolution of an initial basis of order 1 where:

$$\varphi_1(x) := \begin{cases} 1 & : \quad 0 \leq x \leq 1 \\ 0 & : \quad \text{else} \end{cases} \qquad (7)$$

Note, that the support of a B-spline basis is always bound by $[0,j]$. Furthermore, the scaling functions are symmetric with respect to the center of support.

Fig. 2 illustrates a set of different B-spline scaling functions of orders $j = 1,2,3,4$.

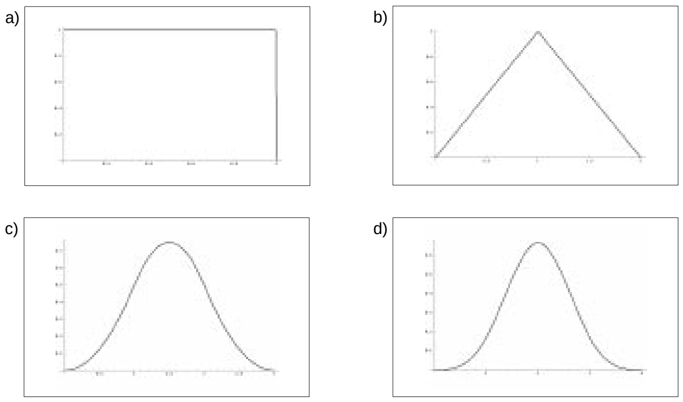


Fig. 2. Cardinal B-Spline scaling functions of increasing order: a) order 1. b) order 2. c) order 3. d) order 4.

The construction of a wavelet, which spans the orthogonal complement space U_m between two approximation spaces of scaling functions V_m and V_{m-1} of different resolution is pointed out in [9]. It starts from the relations of the biorthogonal setting in either spatial or frequency domain and provides piecewise polynomial functions of minimal support. The details of this construction scheme won't be stressed here.

The symmetry of the resulting cardinal B-spline wavelet is restricted to an even order. Fig. 3 shows the functional course of B-spline wavelets of increasing order. The first order type is orthogonal and known as the Haar wavelet.

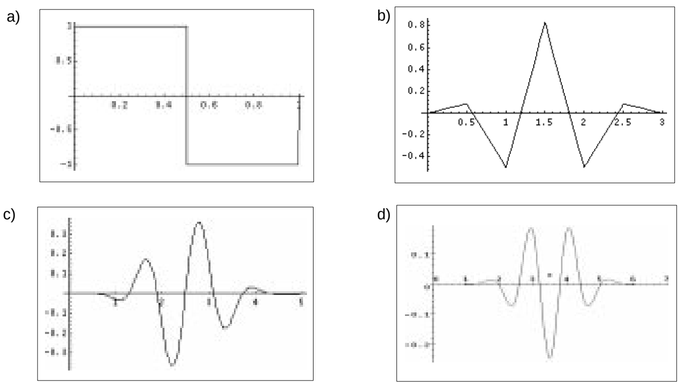


Fig. 3. Cardinal B-spline wavelets of increasing order: a) order 1. b) order 2. c) order 3. d) order 4.

The implementation of biorthogonal wavelet transforms employs the well known QM-Filter pyramids [25], as depicted in Fig. 4.

The dualism of the frames, however, forces us to apply different filter sequences for the decomposition ($H(\omega)$ and $G(\omega)$) and for the reconstruction ($\tilde{H}(\omega)$ and $\tilde{G}(\omega)$). The approximation

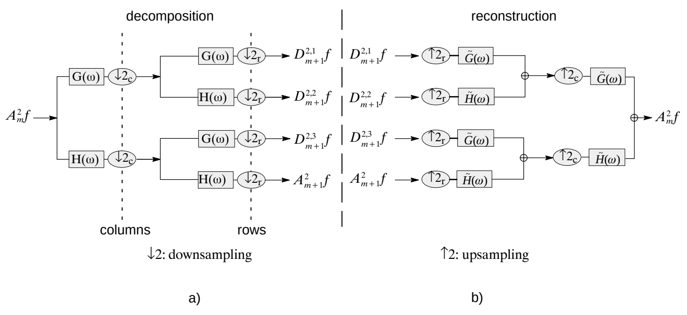


Fig. 4. Filter banks for separable 2D-biorthogonal wavelet transforms: a) Decomposition. b) Reconstruction

$A^2_{m+1}f$ and details $D^{2,1}_{m+1}f$, $D^{2,2}_{m+1}f$ and $D^{2,3}_{m+1}f$, generated by the decomposition path of the filter bank from the higher resolution approximation $A^2_m f$. The reconstruction bank works vice versa. Fast implementations of the WT can be found in [26] and are modified appropriately.

C. How to Recover Detail Signals

One problem arising with the fast QMF implementations of the wavelet transform is, that we need access to the difference signal in each iteration step m of the reconstruction. This is necessary because the detail signal at a particular mesh vertex finally decides whether or not it can be removed. For this purpose, the reconstruction pyramid has to be modified, as indicated in Fig. 5. The procedure recovers the full size detail signals $\Delta_m f$ represented by all wavelets at $m = 1, \ldots, M$ and by the scaling functions f_M. This can be accomplished by reversing the trace of each detail signal from the original down the decomposition pyramid. In other words, any detail signal $\Delta_m f$ at iteration depth m can be obtained from the respective wavelet coefficients by subsequent filtering and upsampling. The final output results from superimposing all detail signals:

$$f(x,y) = \sum_{m=1}^{M} \Delta_m f(x,y) + f_M(x,y). \tag{8}$$

The required extensions of the QM-filterbank are straightforward.

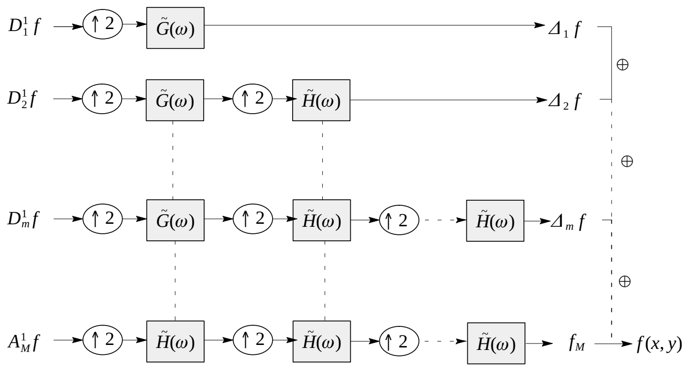


Fig. 5. Modified 1D-version of a QM-filterbank to recover the detail signals.

Note, that this procedure requires additional computation, but although the wavelet coefficients are arranged on a dyadic grid, the Heisenberg principle prevents using them as a direct criterion for vertex removal. We will address this problem again in section III.

D. Interpolation problems with cardinal B-spline wavelets

A problem arising with cardinal B-spline wavelets is the one of interpolating basis functions. For convenience, let's consider again the one-dimensional case of approximating a function $f(x)$. If we assume the initial set of samples $f(x_p) = c^0_p = f_p,\ p : 0, \ldots, N-1$. To represent the wavelet coefficients at iteration $m = 0$, the respective approximation obtained by the corresponding scaling functions $\varphi_{0p} \in V_0$ yields:

$$f^0(x) = \sum_{p=0}^{N-1} c^0_p \varphi^0_p(x) \tag{9}$$

where φ^0_p denote cardinal B-splines of any order. It is clear [25], that this expansion does not reproduce the initial set of coefficients $c^0_p = f_p$ at positions x_p, i. e.

$$f^0(x_p) \neq f_p \tag{10}$$

In order to accomplish interpolation, we have to find a new set of initial coefficients $\hat{c}^0_p$, which hold the interpolation equation below:

$$f(x_p) = f_p = \sum_{p=0}^{N-1} \hat{c}^0_p \varphi^0_p(x_p)\ \forall p \in [0, \ldots, N-1] \tag{11}$$

A further assumption of regular samples and a unit shift of the bases with

$$\varphi^0_p(x_p) = \varphi^0(x - x_p) \tag{12}$$

leads to the following linear system:

$$\mathbf{f} = \mathbf{\Phi} \cdot \hat{\mathbf{c}} \tag{13}$$

where

$$\mathbf{f} = \begin{pmatrix} f(x_0) \\ \vdots \\ \vdots \\ \vdots \\ f(x_{N-1}) \end{pmatrix}$$

$$\mathbf{\Phi} = \begin{pmatrix} \varphi(0) & \varphi(-1) & \varphi(-2) & \cdots \\ \varphi(1) & \varphi(0) & \varphi(-1) & \cdots \\ \vdots & \cdots & \cdots & \cdots \\ \vdots & \cdots & \cdots & \cdots \\ \varphi(N-1) & \varphi(N-2) & \cdots & \cdots \end{pmatrix}$$

$$\hat{\mathbf{c}} = \begin{pmatrix} \hat{c}^0_0 \\ \hat{c}^0_1 \\ \vdots \\ \vdots \\ \hat{c}^0_{N-1} \end{pmatrix}$$

The unknown coefficient vector $\hat{\mathbf{c}}$ is carried out, for instance, by matrix inversion of $\mathbf{\Phi}$ which solves the linear systems of equations.
Note, that the upper linear operator corresponds to a discrete convolution of $\mathbf{f}$ with $\mathbf{\Phi}$. In other words, interpolation can be achieved by inverse low pass B-spline filtering.

E. Boundary Conditions

The wavelet transform and it's fast implementation as a QM-filter bank preassumes a periodic repetition of the samples in the spirit of a torus topology. Consequently, the whole framework is vulnerable to boundary problems on finite intervals. There are several sources in literature (as [27] or [28], which attack this problem and it turns out, that it is possible to construct wavelets on finite intervals. However, this requires to relax the self-similarity of the basis and ends up in different wavelets and scaling functions at the boundaries. In the special case of B-spline wavelets, the construction starts with a set of endpoint-interpolating B-spline bases. The mathematical details should not be explained here in detail, but although the respective projection operators become more complicated, the transform can still be figured out in linear time. Some recipes are given in [12] and [19].
Fig. 19 illustrates the influence of endpoint-interpolating multiresolution B-spline scaling functions as they decompose the control vertices of a B-spline curve. In this example, the scaling functions approximate the curve in different levels of detail.

F. Significance of Wavelet Coefficients

In the upper example, we eroded some curve details by decomposition and rejection of detail coefficients. Consequently, if any data set is transformed into wavelet space, it is necessary to find appropriate criteria to control the accuracy of the surface approximation provided by the wavelet bases. Furthermore, a norm has to be found as a framework for the definition of error bounds. This can be accomplished using the signal energy E_{tot} which is defined by the L^2-norm:

$$E_{\text{tot}} = \|f\|^2 = \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} |f(x,y)|^2 \, dxdy \tag{14}$$

That is, in the discrete case of an $N \times N$ data set with values $(x_1, \ldots, x_{N^2})$ the squared sum of the data values represents the signal energy. Due to the Parseval theorem, it equals the squared sum of the wavelet coefficients $\bar{c_i}$.

$$E_{\text{tot}} = \sum_{i=1}^{N^2} |x_i|^2 = \sum_{i=1}^{N^2} |\bar{c_i}|^2 \tag{15}$$

This relationship, however, is only valid for orthonormal function systems. In the case of biorthogonal wavelets, inner products do not vanish and modify eq. (15). Nevertheless, the upper relations prove that any squared wavelet coefficient is a measure for the fraction of energy provided by its basis functions. It is clear that we can now formulate a simple criterion for the significance of particular wavelet coefficients introducing a threshold τ. Hence, we filter the coefficients according to:

$$\tilde{c_i} := \begin{cases} 0, \ |\bar{c_i}|^2 < \tau \\ \bar{c_i}, \ |\bar{c_i}|^2 \geq \tau \end{cases} \tag{16}$$

Increasing τ will result in increasing the error bounds of the approximation and decreasing τ will also decrease the approximation error. A canonic quantification of the error is given by the ratio of the remaining energy E_r and E_{tot}.
Those coefficients corresponding to the scaling functions should be kept anyway, since they carry the DC part of the data.
The influence of global thresholding on a 2D laser range data set of a human face (Sylvia) is illustrated in Fig. 20. We employed Haar wavelets to enhance the effect. As the number of coefficients is dropped, the details of the approximation get lost.

G. Local Level-of-detail Filtering in Wavelet Space

The introduction of an energy threshold provides a tool for globally influencing the approximation of the wavelets. However, one of the major strengths of the WT has not been harvested so far: *the localization properties*. The local support of the basis functions allows us to localize them both in spatial and in frequency domain and rejecting a particular basis will only affect its area of support. This important property enables an elegant control of the local level-of-detail of the approximation. For this purpose, the coefficients have to be weighted according to the definition of the ROI which corresponds to a filter operation in wavelet space. It can be defined in analogy with the well known filters in spatial or frequency domain. Since the filter affects the local frequency characteristics of the signal, we propose to call it **wavelet space filter**.
Let $g(x,y)$ be a Gaussian weighting function, centered at (x_0, y_0), scaled by (σ_x, σ_y) and rotated by Θ which quantifies the level-of-detail around some location in space (x_0, y_0) and whose elliptical shape is depicted in Fig. 6a.

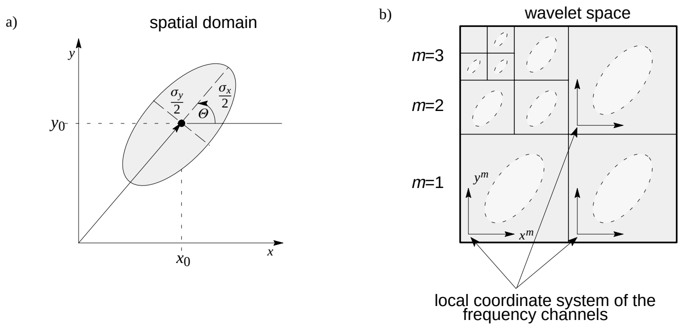


Fig. 6. Filtering in wavelet-space:
a) Rotated, translated and scaled 2D-Gaussian weighting function in spatial domain. b) Transform of the filter into wavelet space results in multiple Gaussians located in each channel.

In order to compute its transformation into wavelet space, we have to note that any point (x_0, y_0) in spatial domain can only be located within the Heisenberg bound in wavelet domain. Furthermore, the spatial localization decreases with increasing iteration depth m.
With the dyadic scale of our 2D-WT, however, the Gaussian splits into all frequency channels and their centers are carried out in wavelet space according to:

$$x_0^m := \frac{x_0}{2^m},\; y_0^m := \frac{y_0}{2^m} \tag{17}$$

and the rotation angle Θ is invariant to the transform.
The set of Gaussian weighting functions $\{g^m(x^m, y^m)\}$ in wavelet space can be elegantly described by using homogeneous coordinates:

$$g^m(x^m, y^m) = \mathrm{e}^{-(\mathbf{R^m}\cdot\mathbf{p^m})^T(\mathbf{R^m}\cdot\mathbf{p^m})+1} \tag{18}$$

The matrix $\mathbf{R^m}$ stands for the affine transform of the Gaussian:

$$\mathbf{R^m} = \begin{pmatrix} \frac{\cos\Theta}{\sigma_x^m} & \frac{\sin\Theta}{\sigma_x^m} & -\frac{x_0\cos\Theta + y_0\sin\Theta}{\sigma_x} \\ \frac{-\sin\Theta}{\sigma_y^m} & \frac{\cos\Theta}{\sigma_y^m} & \frac{x_0\sin\Theta - y_0\cos\Theta}{\sigma_y} \\ 0 & 0 & 1 \end{pmatrix} \tag{19}$$

and $\mathbf{p^m} = (\mathbf{x^m}, \mathbf{y^m}, \mathbf{1})^{\mathbf{T}}$ denotes a position in homogeneous coordinates.
To summarize this section: the control of the local level-of-detail of the wavelet approximation can be accomplished by using one single Gaussian weighting function which surrounds the region of interest in the initial data set. This Gaussian can be interpreted as a filter which is transformed into multiple Gaussians, one in each frequency channel in wavelet space. Premultiplying the coefficients with these Gaussian maps forces any subsequent thresholding to pass only coefficients located within the selected ROI. All others will be removed and hence the reconstructed signal will be most accurate within the ROI along with a Gaussian smoothing of the boundaries. Fig. 21 stresses the effect of level-of-detail filtering. Sylvia's model is decomposed with Haar wavelets and filtered with Gaussians of different locations and parameters. The model is perfectly reconstructed within the focus of the Gaussian, whereas only the scaling functions represent the data outside. In the boundary region, less and less high frequency information is provided and the data becomes more and more "boxlike". Obviously, the proposed wavelet space filter acts as a *magnifying glass* onto the data.
We recommend applying the Gaussian filter and the thresholds only to the wavelets and keeping all coefficients of the scaling function, because they carry the DC fraction of the signal.

III. Quadtree Meshing

A. *Point Removal in Regular Triangle Meshes*

So far, we elaborated some mathematical criteria for approximating a surface data set, sampled on a regular grid, using a multiresolution hierarchy. In order to build an adaptive surface triangulation, however, it is necessary to remove unimportant mesh vertices and to find a triangulation of the remaining ones. The basic criterion, by which a mesh vertex is labeled as unimportant is given by the mathematical framework of the wavelet transform. In contrast to existing methods [18] which base on linear spline wavelets, we aim at generalizing to any type of wavelet. Therefore, our criteria require much more elaboration. Keeping in mind that any triangulation of the surface provides a planar approximation, we only have to bound the error between the original surface function $f(x, y)$ and the bilinear interpolant provided by a triangle. Supposing furthermore that the initial data is expanded by wavelet bases, the detail signal in iteration m helps us to decide whether or not each $2^m + 1$th mesh vertex is necessary for the triangle approximation. First, we visit each second vertex and analyze the value of the detail signal of iteration $m = 1$. If, let's say, the detail signal $\Delta_1 f$ in some neighborhood of vertex n is sufficiently low, then the vertex is not important and the approximation can be accomplished by a linear interpolation between vertex $n-1$ and $n+1$. This scheme can now be applied recursively by subsequent computation of the detail signals $\Delta_m f,\ m = 1, \ldots, M$ and by visiting all dyadic vertices at positions $n = 2^m k + 1$. Once the detail signal is sufficiently small and the adjacent vertices in step $m-1$ are already removed, we are allowed to label the current vertex as well.
As a consequence, our procedure results in recursively building a quadtree representation of the initial mesh by removing dyadic vertices. Fig. 7 again illustrates the thinning method which finally figures out the symbolic quadtree representation of the mesh vertices depicted as an example in Fig. 8. The nodes of the quadtree contain either pointers to some child-nodes, or in case of leaves, point to the entries of a vertex list.

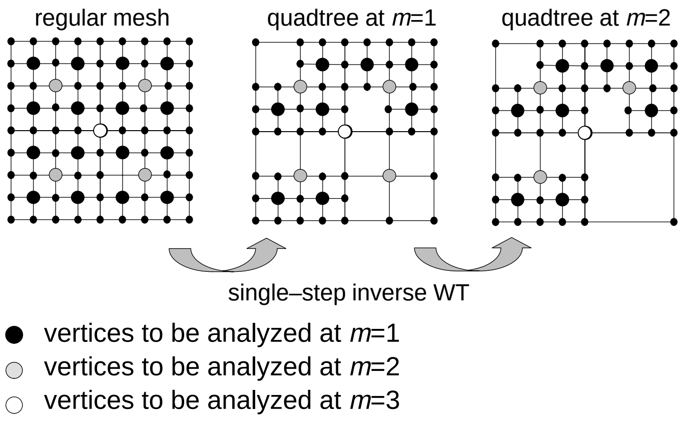

regular mesh
quadtree at m=1
quadtree at m=2
single–step inverse WT
● vertices to be analyzed at m=1
● vertices to be analyzed at m=2
○ vertices to be analyzed at m=3

Fig. 7. Recursive growth of a quadtree from the regular mesh by analysing the detail signals of the WT at each dyadic vertex.

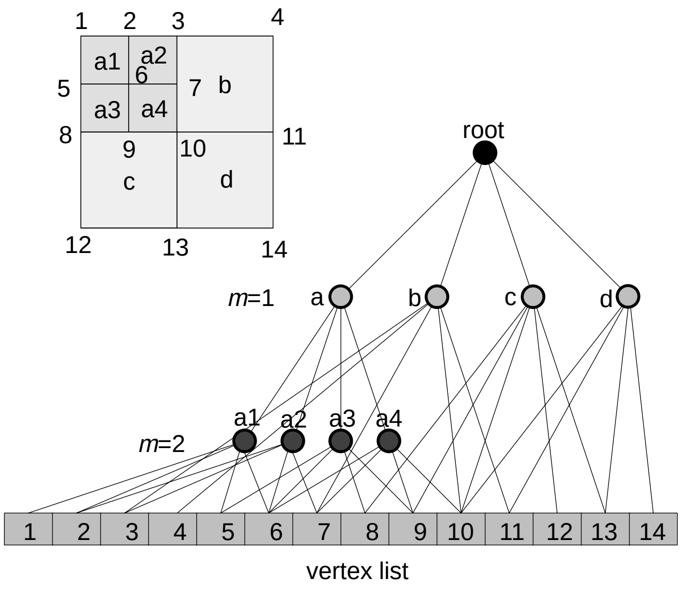


Fig. 8. Symbolic representation of the mesh using a quadtree data structure.

Note again here that the growth of our quadtree is entirely controlled by a single energy threshold τ embedded in the function space of the wavelets. The final maximum depth of the tree depends on the upper decomposition bound M of the WT.

At a first glance, it seems to be natural to take a particular wavelet coefficient to decide whether or not a vertex can be removed. This works out in the trivial case of simple linear splines. Extending the methods to arbitrary types of wavelets, however, requires to elaborate the criteria significantly. This is ultimately founded in Heisenberg's uncertainty principle, determining the lower bound of the spatial frequency localization. If, let's say a wavelet has a particular spatial support, the the inner product with the data exactly represents it's contribution in that region. In the general case, however, it is not possible to recover the contribution of the wavelet at a specific vertex position in that region from the coefficients. Moreover, this would preassume a wavelet, whose spatial localization drops to zero, such as with Delta-distributions. Obviousely, we have to find more appropriate criteria than wavelet coefficients.

Therefore, to finally decide on whether a mesh vertex can be removed or not, we propose the following criteria which also helps to preserve the topology of the tree. Only in cases, where all criteria are TRUE, can the vertex be removed:

1. *Wavelet-criterion*: a vertex at iteration m can be removed, if the sum of the squares of its difference signal and those within a 4-neighborhood at resolution m is less than an upper bound ϵ (Fig. 9a). Upon removal, we preassume our difference signals to be set to zero, but numerical reasons require to set ϵ to a small positive number.

2. *Resolution-criterion*: a vertex at iteration m can be removed, if the four surrounding vertices at resolution $m - 1$ were previously removed (Fig. 9b).

3. *m to m-2-criterion*: a vertex can be removed, if the resulting cell is not adjacent to any cell with higher resolution than $m-2$. Thus, we restrict the growth to cell transitions from m to $m - 2$ which simplifies the triangulation algorithm (Fig. 9c).

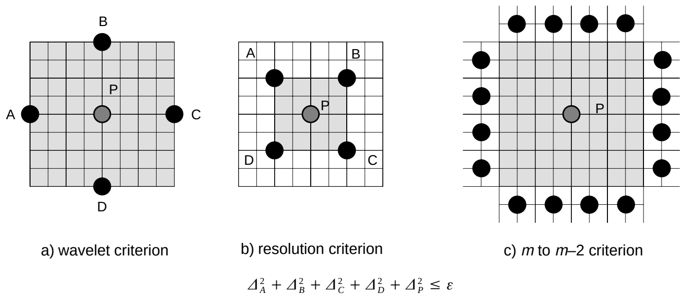

B
P
A
C
D
A
B
P
D
C
P
a) wavelet criterion
b) resolution criterion
c) m to m–2 criterion
$\Delta_A^2 + \Delta_B^2 + \Delta_C^2 + \Delta_D^2 + \Delta_P^2 \le \varepsilon$

Fig. 9. Illustration of the different criteria to decide on the vertex removal.

Another aspect of the method is illustrated in Fig. 10a, where vertex P is analyzed. Suppose P survives all of the above criteria. If we remove P, however and if P_U **and** P_L are already removed, i. e. if two adjacent cells have the same resolution, then we must reject the vertices A and B on the cell boundaries, too. Hence, when traversing the vertex array from upper left to lower right, one has to keep track of upper and left vertices of the same iteration step m as well.

This additional criterion ends up with a partitioning of the initial array into different regions (see Fig. 10b). Within these different regions, we have to check only left, upper or both adjacent vertices.

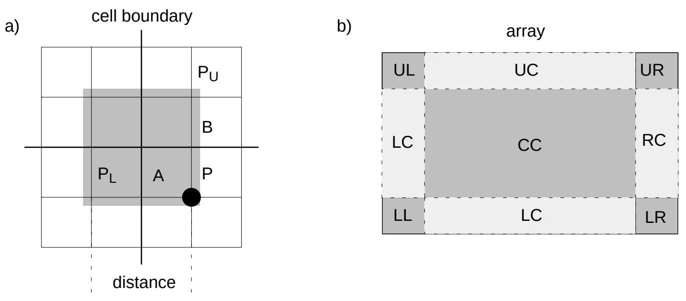


Fig. 10. a) Criteria to remove vertices on the edges of adjacent cells of the same resolution. b) Resulting partitioning of the initial array.

B. Look-up Tables for Local Triangulations

Once the tree is built from the above procedure, the quadtree cells have to be triangulated. A generic problem arising from meshing hierarchies of rectangular surface patches is the occurrence of cracks [2]. A crack occurs if we do not take care of adjacency of quadtree cells of different depth and, hence, different resolution. The surface may break up, holes may appear and any consistency required for normal interpolation gets lost. Fig. 11 shows a crack and also shows how to modify the triangulation to avoid it.

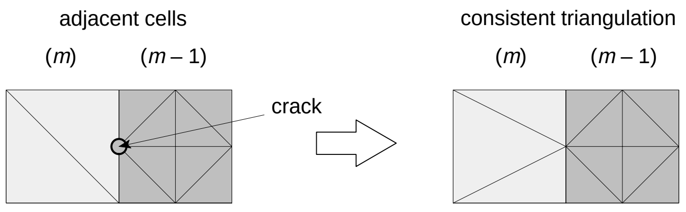


Fig. 11. The occurrence of cracks at the boundaries of adjacent quadtree cells of different resolution.

The scheme we introduce here for fast and consistent cell triangulation is based on the following observation: consider Fig. 12, where two adjacent cells are depicted along with topological arrangements that may occur for transitions from resolution m to $m - 1$ and $m - 2$. There are only 5 cases at the respective cell boundary. Let's presume that we restrict the growth of the quadtree so that only transitions up to $m - 2$ are possible (*resolution criterion*). Consequently, the set of possible arrangements of vertices at the four cell boundaries can now be derived from Fig. 12. Moreover, some look-up tables may be built containing the triangulations as explained below.

For cell transitions from m to $m - 1$ a look-up table with 16 entries is built as presented in Fig. 13. The central idea of the algorithm is to first solve the triangulation within each cell for m to $m - 1$. This is accomplished by analyzing the mesh vertices along each cell edge.

The fast computation of the look up table entry can be accomplished by a binary outcode, generated from bitwise addition of the flags of the respective edge vertices, as indicated in Fig. 13.

Once the corresponding look-up table entry is identified, we then consider mesh vertices which account for the $m - 2$ transitions. This may cause some triangles to be split up into two

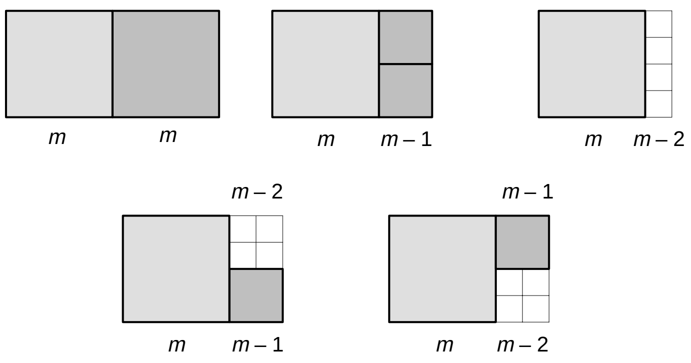


Fig. 12. Topology of mesh nodes of adjacent cells for different resolutions m, $m-1$ and $m-2$.

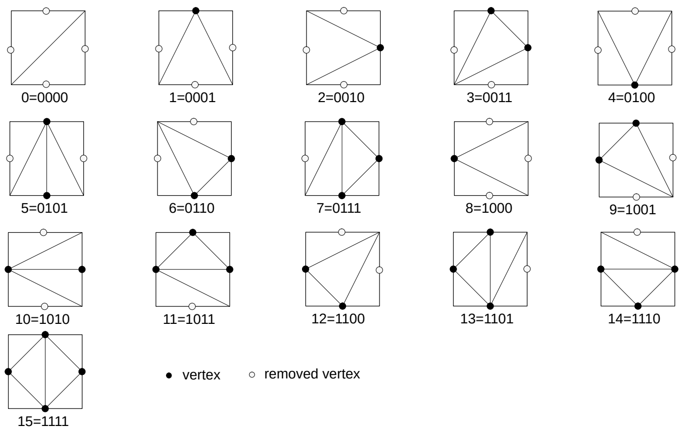


Fig. 13. Look-up table representing the optimal triangulation of all possible cases from m to $m-1$ and corresponding outcode.

pieces, as shown in Fig. 14. Consequently, the algorithm first computes the case for m to $m-1$ and then it decides on the corresponding subcase, by simply analyzing the flags of all intermediate vertices responsible for transitions from m to $m-2$. Although we get 625 possible cases, the total number of triangles required does not exceed 96. They are stored in a look-up table depicted in Fig. 15.

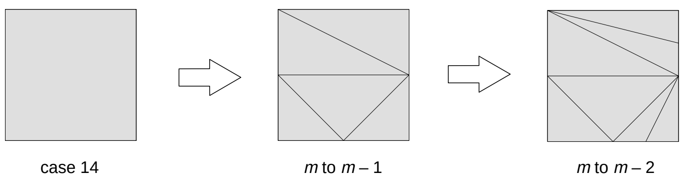


Fig. 14. Cell triangulation for cases from m to $m-2$ as derived from a look-up table entry of m to $m-1$.

All subcases are hardcoded and contain references to these look-up table entries. Note, that although there are 625 cases only one computation of the outcode and at most 8 additional tests are necessary to compute the triangulation. It is clear that we end up with a very efficient algorithm by doing the meshing without any geometric computation but by just checking vertices along the cell edges.

A corresponding pseudocode for the recursive quadtree traversal and meshing is given with:

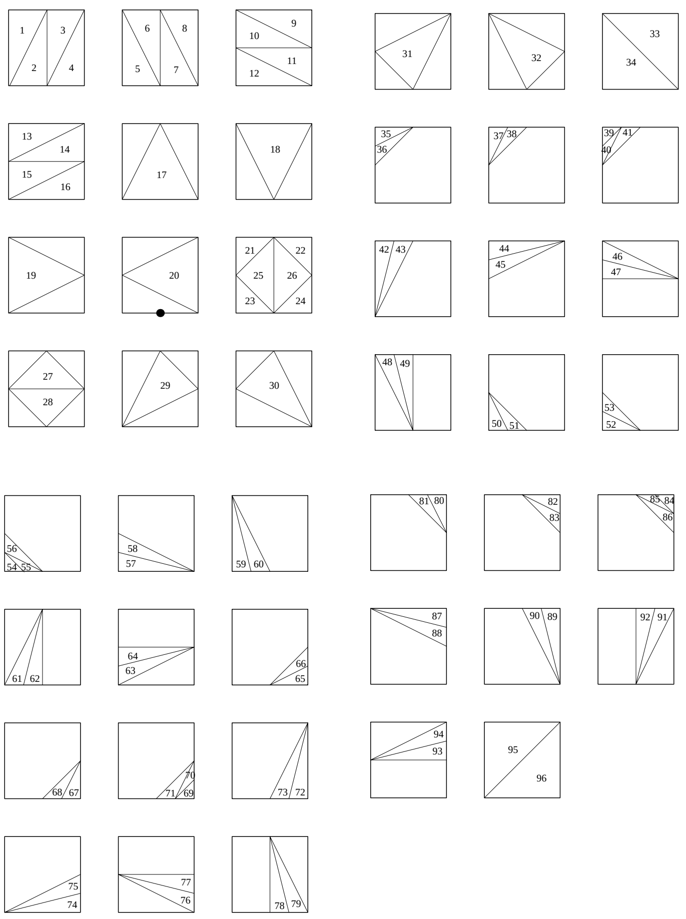


Fig. 15. Look-up table with all 96 possible triangles which are necessary for transitions from m to $m-2$.

```
// The initial array has a size of (N+1)(N+1).
// Let N be a power of 2, N = 2I.
// Each cell is addressed by its upper left corner vertex.
// root cell
x = y = 0;
i = I;
traverse_quadtree(x,y,i);
procedure traverse_quadtree(x,y,i)
{
   // compute center vertex of son cells
   mh = 2i-1;
   xmh = x+mh;
   ymh = y+mh;
   if (i>0) and flag(xmh,ymh)
   {
      i = i-1;
      //analyze the son cells
      traverse_quadtree(x,y,i);
      traverse_quadtree(xmh,y,i);
      traverse_quadtree(x,ymh,i);
      traverse_quadtree(xmh,ymh,i);
   } else
      triangulate(x,y,i);
}
```

IV. Errors and Complexity

A. Error Analysis of Planar Approximations

One important aspect, when dealing with surface approximations is to quantify the error of the method. In our approach, error quantification is figured out by the following mean-square measure. Let $f(x,y)$ be the original surface and $g(x,y)$ be an approximation. We define the mean-square error $\bar{\Delta}^2$, as:

$$\bar{\Delta}^2 = \frac{1}{\Delta x \Delta y} \int\limits_{\Delta x} \int\limits_{\Delta y} |f(x,y) - g(x,y)|^2 \, dxdy \qquad (20)$$

Note, that the error is normalized to the projected surface area $\Delta x \Delta y$. In the discrete case, where K samples $f_i(x_i, y_i)$ of the surface are provided at locations (x_i, y_i) , $i = 1, \ldots, K$ the mean-square error is approximated by the following relation:

$$\bar{\Delta}^2 \approx \frac{1}{K} \sum_{i=1}^{K} \Delta(x_i, y_i)^2 \qquad (21)$$

where $\Delta(x_i, y_i) = f(x_i, y_i) - g(x_i, y_i)$.
Finally, in our triangle meshes the error integral of eq. (20) is evaluated using Monte Carlo methods. For this purpose, we compute a set of K randomized locations within each of our triangles and calculate the surface value $g_i(x_i, y_i)$ by bilinear interpolation. The respective reference value for the surface $f_i(x_i, y_i)$ is obtained by bilinear interpolation of the four mesh vertices in the initial data grid as depicted in Fig. 16. The constant number of samples taken from each triangle forces the overall number of samples for the evaluation to be distributed according to the single triangle surface areas. Due to the adaptive triangulation, we end up with more samples in surface regions of high curvature and accomplish a reasonable distribution. Thus, the local mean square errors of each triangle have to be weighted with their corresponding surface area A_T^{2D} projected into 2D. The final expression of the overall mean square error of the surface yields

$$\bar{\Delta} = \sqrt{\frac{\sum\limits_{\text{all tri}} \left(\frac{1}{K} \sum\limits_{i=1}^{K} \left(f(x_i, y_i) - g(x_i, y_i)\right)^2 \, A_T^{2D} \right)}{\Delta x \, \Delta y}} \qquad (22)$$

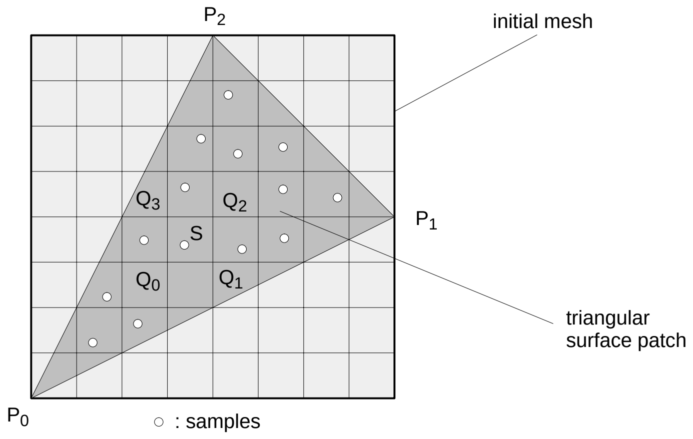


Fig. 16. Computation of the mean square error using a Monte Carlo method.

B. Some remarks on Algorithmic Complexity

One of the very advantages of our method is the low algorithmic complexity for both computation of the respective transforms and for the quadtree meshing. Whereas 2D-FFT based transforms usually require $O\left(N^2 \log_2(N)\right)$ computations, the 2D-WT benefits from dyadic scaling and sparsity and requires only $O\left(N^2\right)$ computations. Although we have to modify the initial QMF-pyramid to compute the detail signal, the complexity still remains $O\left(N^2\right)$. The final expression for the complexity CWT of a D-dimensional WT, however, depends both on the support S of the wavelet and on the iteration depth M:

$$\begin{aligned} C^{WT} &\approx D\, N^D\, S \sum_{m=1}^{M} \frac{1}{2^{D(m-1)}} \\ &< \frac{2^D\, D\, N^D\, S}{2^D - 1} = O\left(N^D\right) \end{aligned} \qquad (23)$$

This is another important reason for the usage of strict compact support wavelets such as the biorthogonal ones we recommend here.
Similar investigations can be carried out for the complexity of building up the quadtree: If, in the worst case, the traversal is done up to the maximum depth of the WT, M, we have to perform at worst 4 energy tests for the wavelet criterion, 4 resolution tests and 16 tests for the m to $m-2$ criterion. Due to the dyadic structure of the vertices to be analyzed, we end up with

$$\begin{aligned} C^A &= 24 \sum_{m=1}^{M} 2^{2I-2m} \\ &= 2^{2I} \left(8 - \frac{8}{4^M} \right) \\ &= O\left(N^2\right) \end{aligned} \qquad (24)$$

to test the importance of vertices, which is still linear with respect to the overall number of mesh vertices $N^2 = 2^{2I}$.
The traversal of the array is carried out by the recursive procedure of depth M given in the pseudocode. In the worst case of none of the vertices to be labeled as being unimportant, it's complexity is quantified by

$$\begin{aligned} C^B &= S^{I-M} \sum_{m=0}^{M-1} 4^m \\ &= \frac{2^{I+M} - 2^{I-M}}{3} \\ &= O\left(2^{2I}\right) = O\left(N^2\right) \end{aligned} \qquad (25)$$

It has to be noted, however, that the worst case analysis provides only a theoretical upper bound. In "real life", vertices labeled as unimportant reduce the computational costs dramatically. A single vertex removal in recursion depth m cuts a whole subtree and saves

$$B = \sum_{k=m}^{M} 4^{k-m} = \frac{4^{M-k+1} - 4}{3} \qquad (26)$$

tests.

V. Applications

A. Mesh Reduction and Error Analysis

For the following investigation, a digital terrain model of the Swiss Alps, Matterhorn/Zermatt DHM 1:25000 was selected.

The initial resolution of the mesh is 256×256. The altitudes range from 1855.1 m (La Monta) to 4431.9 m (Matterhorn). We used cubic B-spline wavelets to decompose the data and the corresponding dual frames to approximate the reconstructions. The iteration depth was $M = 4$, and $K = 3$ samples were taken at each triangle to compute the mean square error. Fig. 17 illustrates, how the ratio of triangle reduction behaves as a function of the threshold τ. Furthermore, the ratio of remaining wavelet coefficients is recorded which can be interpreted as some kind of coding gain. Finally, the root of the mean-square error is plotted as well in meters. Note, that due to the logarithmic scale of the threshold, the functional behavior of both percentage of coefficients and triangles is approximately linear. The relation is further stressed in Fig. 18, where the number of triangles and the mean square error are recorded as a function of the percentage of coefficients employed for the approximation.

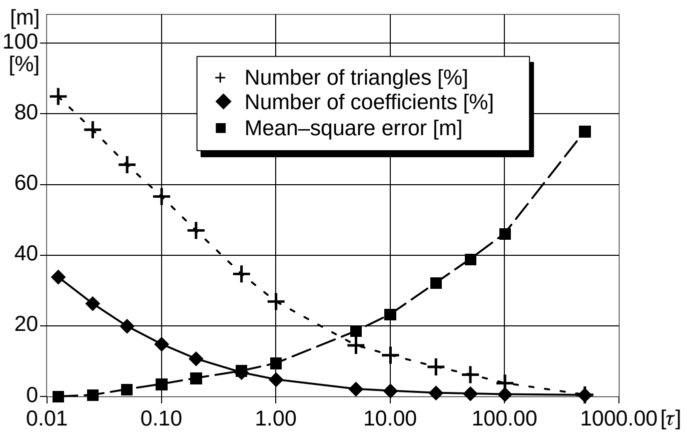


Fig. 17. Number of triangles, wavelet coefficients and mean-square error of digital terrain model as functions of treshold τ.

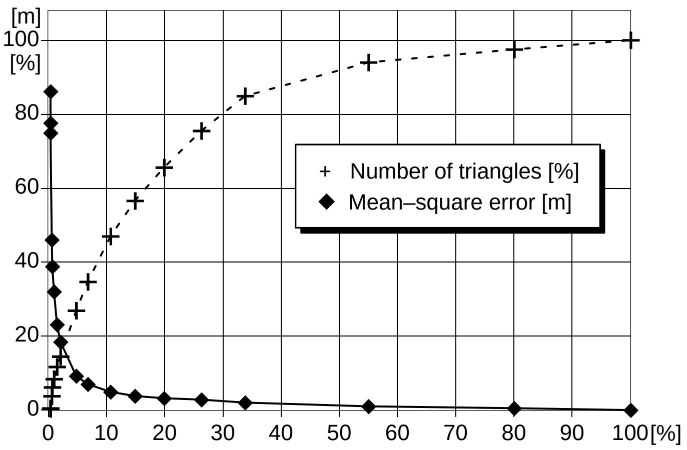


Fig. 18. Number of triangles and mean-square error as function of the coefficients used for the approximation.

Some results of intermediate steps of the triangle reduction are depicted in Fig. 22. The criteria which we defined to reject unimportant mesh vertices thin in particular in those regions of low surface curvature. This is due to the wavelet criterion which provides an estimate of the local spectral energy of the data in different frequency channels. Thus, local high frequency variations in our data force the meshing to be more dense. The corresponding Gouraud-shaded models are also presented in Fig. 22. In addition, depth cueing is used to enhance the presentation.

B. *Level-of-Detail Control*

The effect of wavelet space filtering using the Gaussian is illustrated in the images of Fig. 23 for the same data set. Changing the parameters of the Gaussian ellipse allows us to concentrate on the triangulation of local regions of interest. Hence, for real time animations, such as flight simulation, our method enables us to move the Gaussian for each frame according to the pilots field of vision or line of sight and to adapt the approximation to these parameters. Finally, Fig. 23d presents the Gouraud-shaded image. Obviously, the Gaussian enables the user to interact with a local "magnifying glass"

C. *Laser Range Data*

The following results were obtained on laser range data sets. Fig. 24a and b show a locally enhanced mesh and it's shaded counterpart of a laser range scan of the city of Hannover, Germany.

Another test of our method was carried out in Fig. 25, which presents both globally and locally thresholded images of Sylvia's face.

For the upper investigations, we took an orthogonal projection. In the general case of cylindrical scans, however, we recommend to interpret the angular coordinate immediately in the spirit of a cylindrical projection [17].

Again, the local spectral estimation properties force the algorithm to reduce the mesh either in low frequency regions or where the user has directed the Gaussian filter.

D. *Implementation*

Our current implementation of the method is based on *AVS 5* and was not tuned for real time performance. Nevertheless, we achieve meshing rates of less than 1s for a 256x256 data set on an SGI-Indy, 64 MB, R4400, also depending on the degree of mesh reduction. Actually, the major bottleneck is the *Geometry Viewer* module of *AVS*, rather than any of the described algorithms.

In the previous examples, we compared two polynomial approximations. Therefore, we picked an appropriate tolerance measure τ and computed the resulting error. If, instead we base the error analysis on the wavelet approximation, it is straightforward to predefine a particular error bound to be kept. This requires, however, additional computations. After sorting the wavelet coefficients, the detail signals reconstructed from the recoverd wavelets can be used to compute a mean-square error and can be summed up to a particular upper bound.

VI. Conclusions

We presented a method for fast and efficient surface meshing which benefits from two basic ideas: First, any control of the surface mesh is computed by using an initial wavelet decomposition of the data samples. The mathematical framework of the WT allows us to bound the errors of the approximation and efficient criteria on whether or not single mesh vertices can be removed are provided by analyzing WT outputs. Furthermore, wavelet space filters allow a control of the quality of the surface approximation within local regions of interest and act as local *magnifying glasses*. Secondly, the dyadic structure of the 2D-WT motivated us to build a quadmesh from the initial regular grid. Any triangulation of each quadtree cell is obtained by using a look-up table and hence no additional computation is required for the triangulation, as with standard Delaunay-based methods.

Due to the low complexity of this algorithm, we can achieve retriangulations of the surface at nearly interactive rates in the current AVS implementation on SGI workstations. Thus, we guess that our method is particularly well suited for real-time applications, as virtual reality or flight and driving simulation. Especially, when considering low altitude flights the Gaussian filter could help to control the level-of-detail of the pilot's field of vision. Moreover, any object instance of a geometric data base related to the terrain might also be controlled by the wavelet transform. For this purpose, the actual depth of the quadtree at the object's location on the terrain is used to govern the data base and to select the object instance to be rendered.

Although the method requires an inverse WT with each new triangulation, we have proven the algorithmic complexity is still low. It is clear that we can map the WT onto special purpose hardware, such as signal processors. Currently, the method is implemented in terms of different AVS modules.

Future research has to be conducted towards extensions of the method for 3D isosurfaces in volume data using tetrahedralizations of an octree built from the WT. Additional tuning of the mesh could also be carried out by using the directional selectivity of the WT.

ACKNOWLEDGEMENTS

The authors like to thank the Bundesamt für Landestopographie, Bern, Switzerland for providing the digital terrain model. We are also grateful to Dornier GmbH, Friedrichshafen and ZGDV, Darmstadt, both Germany, for the laser scans.

REFERENCES

[1] G. M. Nielson, "Modelling and visualizing volumetric and surface-on-surface data," in *Focus on Scientific Visualization* (H. Hagen, ed.), pp. 191–242, Springer, 1993.

[2] D. R. Baum, S. Mann, K. P. Smith, and J. M. Winget, "Making radiosity usable: Automatic preprocessing and meshing techniques for the generation of accurate radiosity solutions," in *Computer Graphics (SIGGRAPH '91 Proceedings)* (T. W. Sederberg, ed.), vol. 25, pp. 51–60, July 1991.

[3] G. Farin, *Curves and Surfaces for Computer Aided Geometric Design*. Academic Press, 1990.

[4] W. J. Schroeder, J. A. Zarge, and W. E. Lorensen, "Decimation of triangle meshes," in *Proceedings of SIGGRAPH '92*, pp. 65–70, ACM SIGGRAPH, 1992.

[5] H. Samet, "The quadtree and related hierarchical data structures," *ACM Comp. Surv.*, vol. 16, pp. 187–260, 1984.

[6] A. v. Gelder and J. Wilhelms, "Topological considerations in isosurface generation," Tech. Rep. UCSC-CRL-94-31, Baskin Center for Computer Engeneering and Information Sciences, University of California, Santa Cruz, 1994.

[7] G. M. Nielson and B. Hamann, "The asymptotic decider: Removing the ambiguity in marching cubes," in *Visualization '91*, pp. 83–91, 1991.

[8] P. Ning and L. Hesselink, "Octree pruning for variable-resolution isosurfaces," in *Visual Computing* (T. L. Kunii, ed.), pp. 349–363, Springer, 1992.

[9] I. Daubechies, "The wavelet transform, time-frequency localization and signal analysis," *IEEE Transactions on Information Theory*, vol. 36, pp. 961–1005, 1990.

[10] S. Mallat, "A theory for multiresolution signal decomposition: The wavelet representation," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 11, no. 7, pp. 674–693, 1989.

[11] C. Chui, *An Introduction to Wavelets*. Academic Press, 1992.

[12] E. J. Stollnitz, T. D. DeRose, and D. H. Salesin, "Wavelets for computer graphics: A primer, part1," *IEEE Computer Graphics and Applications*, vol. 15, no. 3, pp. 76–84, 1995.

[13] M. Gross, R. Koch, L. Lippert, and A. Dreger, "A new method to approximate the volume rendering equation using wavelets and piecewise polynomials," *Computers & Graphics*, vol. 19, no. 1, pp. 47–62, 1995.

[14] S. Muraki, "Volume data and wavelet transform," *IEEE Computer Graphics and Applications*, vol. 13, pp. 50–56, July 1993.

[15] T. He, S. Wang, and A. Kaufman, "Wavelet-based volume morphing," in *Proceedings of Visualization '94*, pp. 82–92, IEEE Computer Society, IEEE Computer Society Press, 1994.

[16] S. J. Gortler, P. Schroder, M. F. Cohen, and P. Hanrahan, "Wavelet radiosity," in *Computer Graphics Proceedings, Annual Conference Series, 1993*, pp. 221–230, 1993.

[17] M. Gross and R. Koch, "Visualization of multidimensional shape and texture features in laser range data using complex-valued gabor wavelets," *IEEE Transactions on Visualization and Computer Graphics*, vol. 1, no. 1, pp. 44–49, 1995.

[18] J. M. Lounsbery, *Multiresolution Analysis for Surfaces of Arbitrary Topological Type*. PhD thesis, University of Washington, Seattle, 1994.

[19] A. Finkelstein and D. H. Salesin, "Multiresolution curves," in *Proceedings of SIGGRAPH '94 (Orlando, Florida, July 24–29, 1994)* (A. Glassner, ed.), Computer Graphics Proceedings, Annual Conference Series, pp. 261–268, ACM SIGGRAPH, ACM Press, July 1994. ISBN 0-89791-667-0.

[20] M. Eck, T. DeRose, T. Duchamp, H. Hoppe, M. Lounsbery, and W. Stuetzle, "Multiresolution analysis of arbitrary meshes," in *SIGGRAPH 95 Conference Proceedings* (R. Cook, ed.), Annual Conference Series, pp. 173–182, ACM SIGGRAPH, Addison Wesley, Aug. 1995. held in Los Angeles, California, 06-11 August 1995.

[21] M. H. Gross, R. Gatti, and O. Staadt, "Fast multiresolution surface meshing," in *Proceedings of IEEE Visualization '95*, pp. 135–142, IEEE Computer Society Press, 1995.

[22] J. Bloomenthal, "An implicit surface polygonizer," in *Graphics Gems IV* (P. Heckbert, ed.), pp. 324–349, Boston: Academic Press, 1994.

[23] M. H. Gross, *Visual Computing*. Springer, 1994.

[24] A. Fournier, "Wavelets and their applications in computer graphics." Course Notes SIGGRAPH '94, 1994.

[25] M. Unser, A. Aldroubi, and M. Eden, "A family of polynomial spline wavelet transforms," *Signal Processing*, vol. 30, pp. 141–162, 1993.

[26] W. H. Press, S. A. Teukolsky, W. T. Vetterling, and B. P. Flannery, *Wavelet Transforms*, ch. 13.10, pp. 591–606. Cambridge University Press, 2 ed., 1992.

[27] E. Quak and N. Weyrich, "Decomposition and reconstruction algorithms for spline wavelets on a bounded inverval," *Applied and Computational Harmonic Analysis*, no. 1, pp. 217–231, 1994.

[28] A. Cohen and I. Daubechies, "Wavelets on the interval and fast wavelet transforms," *Applied and Computational Harmonic Analysis*, no. 1, pp. 54–81, 1993.

[29] A. Cohen, I. Daubechies, and J. Feauveau, "Bi-orthogonal bases of compactly supported wavelets," *Comm. Pure and Applied Mathematics 45*, pp. 485–560, 1992.

[30] A. Cohen, "Wavelets and their applications," in *Wavelets and Digital Signal Processing*, pp. 105–121, Hones and Bartlett Publishers, 1992.

[31] L. Lippert and M. Gross, "Fast wavelet based volume rendering by accumulation of tranparent texture maps," in *Proceedings of Eurographics '95*, pp. 431–443, 1995.

[32] S. R. Marschner and R. J. Lobb, "An evaluation of reconstruction filters for volume rendering," in *Proceedings IEEE Visualization '94*, pp. 100–107, IEEE Computer Society, 1994.

Markus H. Gross received his Dipl.–Ing. degree in electrical engeneering in 1986 and his Ph.D. on applications of computer graphics and image analysis in 1989, both from the University of Saarbrücken, Germany. In 1990,twoside,twocolumn he joined the Computer Graphics Center in Darmstadt, where he established and directed the Visual Computing Group. Since July 1994, he is an assistant professor at the Computer Science Department at the ETH Zürich. His research interests include scientific visualization, rendering techniques, wavelets and finite elements.

Oliver G. Staadt received his Dipl.–Inform. degree in computer science from Technical Unversity of Darmstadt, Germany, in 1995. He is currently a research assitent and Ph.D. candidate at ETH Zürich, Switzerland, and a member of the Computer Graphics Research Group of the Computer Science Department. His research interests include geometry compression, adaptive meshing and wavelets.

Roger Gatti received his Dipl.–Informatik–Ing. degree in computer science from ETH Zürich, Switzerland, in 1995. He is currently with r^3 *security engineering ag*, Switzerland.

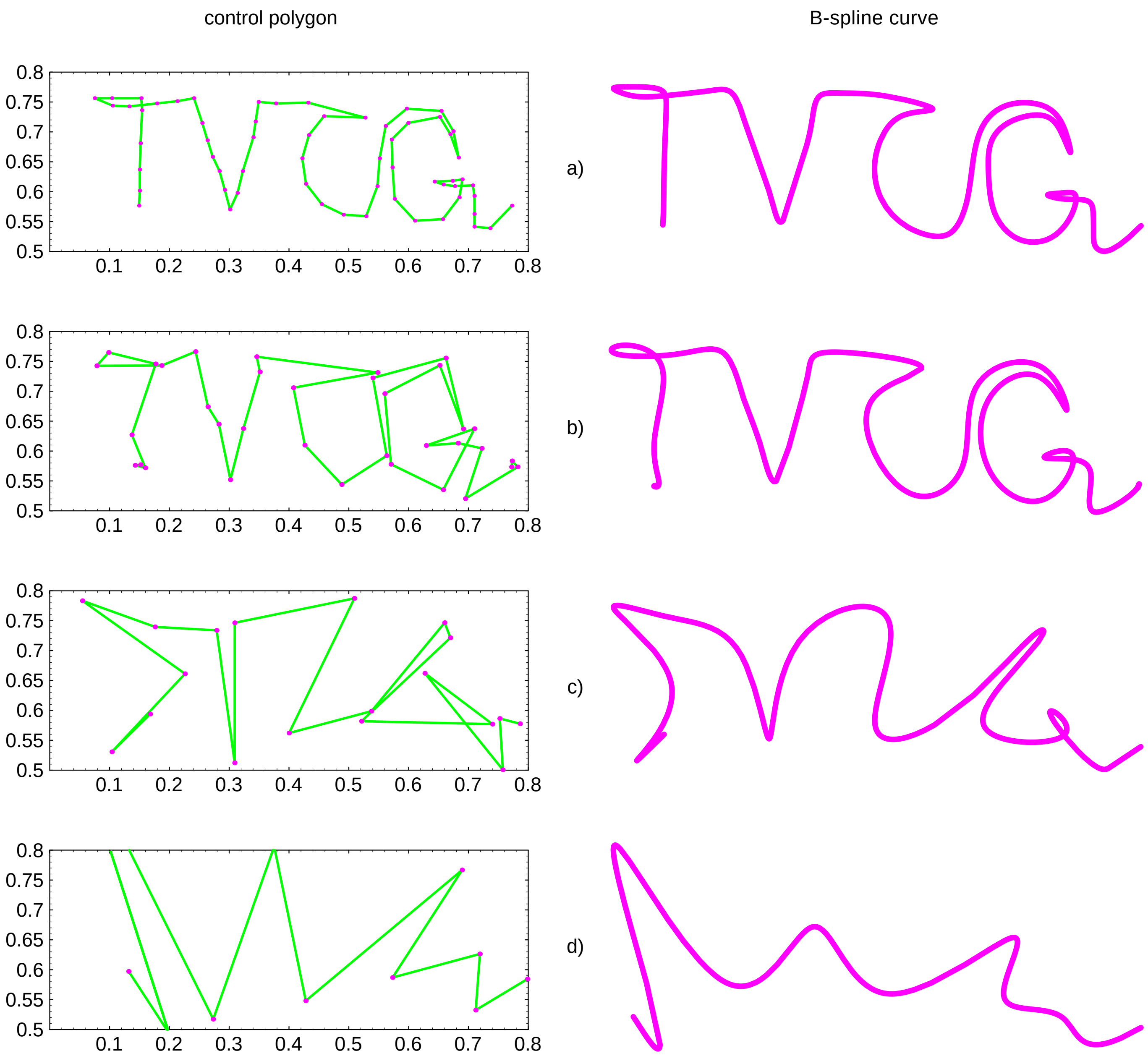
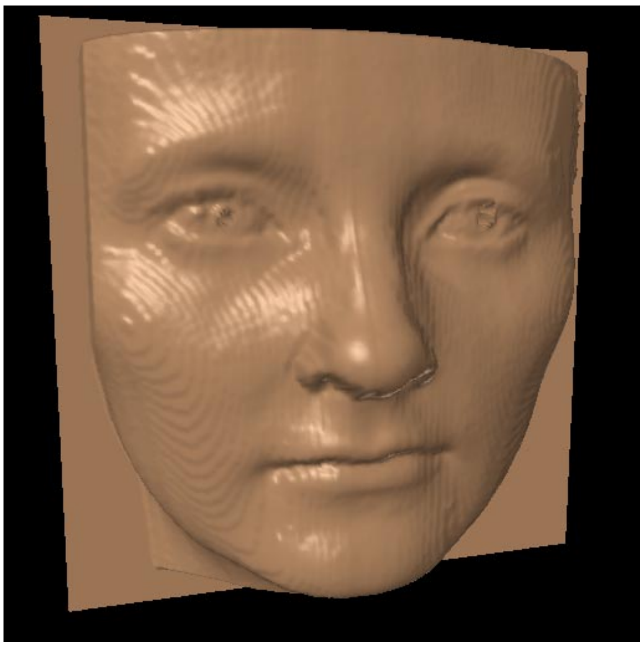
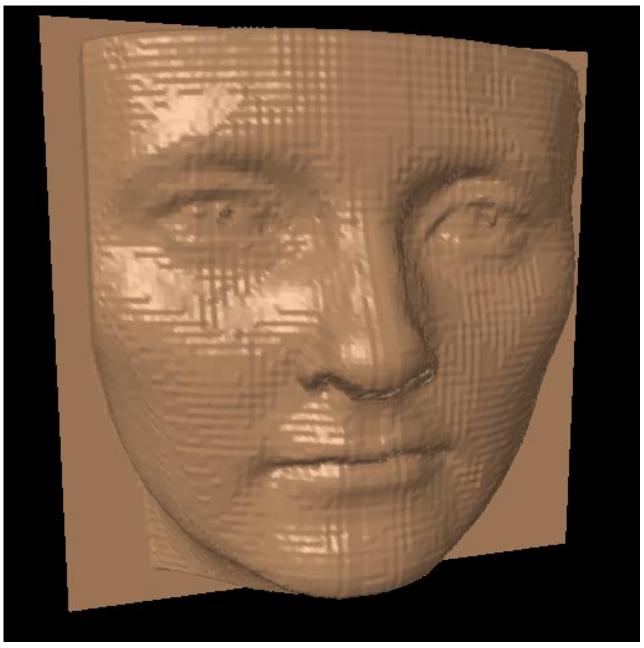


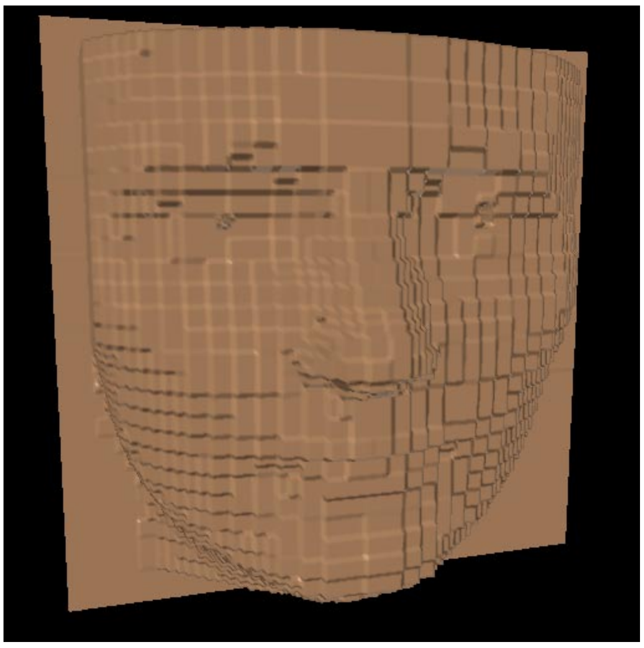
Fig. 19. Decomposition of a B-spline curve using cubig B-spline scaling functions of different resolution M:a) $M = 0$. b) $M = 1$. c) $M = 2$. d) $M = 3$.



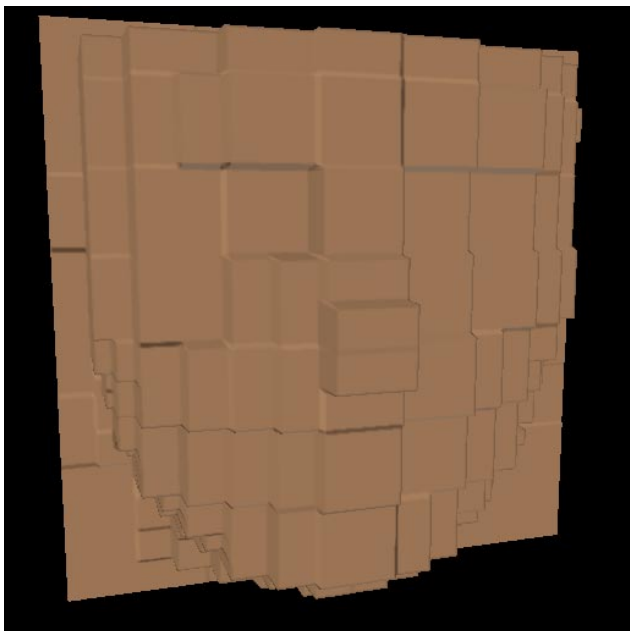
a)



b)



c)



d)

Fig. 20. Different levels of approximation of Sylvia's data set by global thresholding of Haar wavelets. C: percentage of remaining coefficients (data source courtesy ZGDV, Darmstadt).
a) $\tau = 0$, $C = 100\%$. b) $\tau = 1$, $C = 13.7\%$. c) $\tau = 100$, $C = 1.7\%$. d) $\tau = 10000$, $C = 0.2\%$.

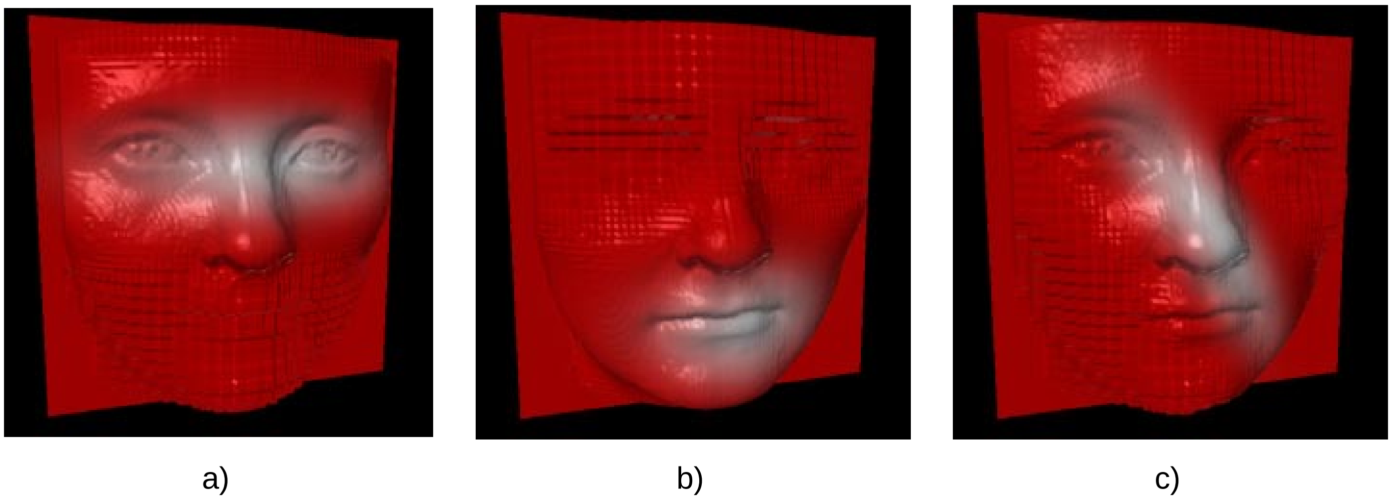
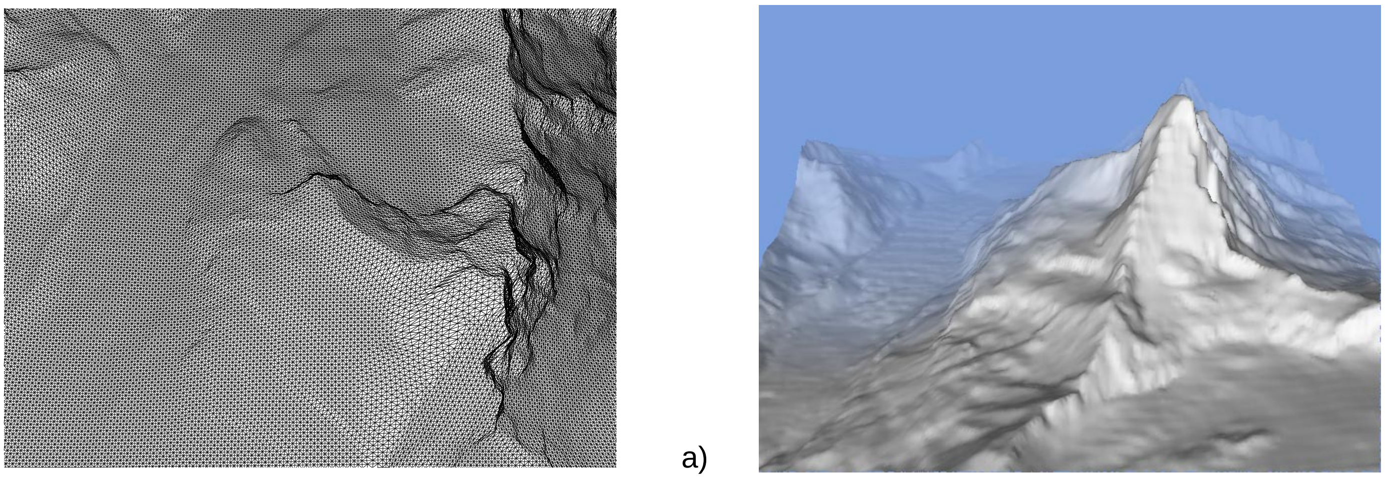
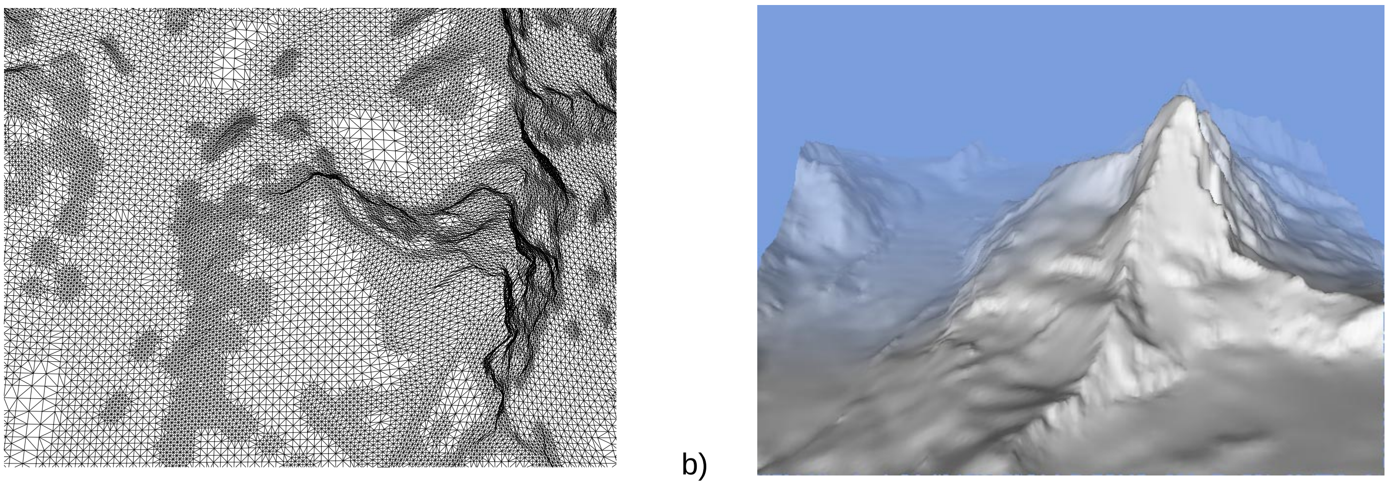


Fig. 21. Example for local level-of-detail filtering in wavelet space: Decomposition of Sylvia's face with Haar wavelets and filtering with different Gaussian space-frequency filters.
a) $\sigma_x = 80$, $\sigma_y = 80$, $\Theta = 0$. b) $\sigma_x = 50$, $\sigma_y = 30$, $\Theta = 0$ c) $\sigma_x = 80$, $\sigma_y = 20$, $\Theta = \pi/4$


a)


b)

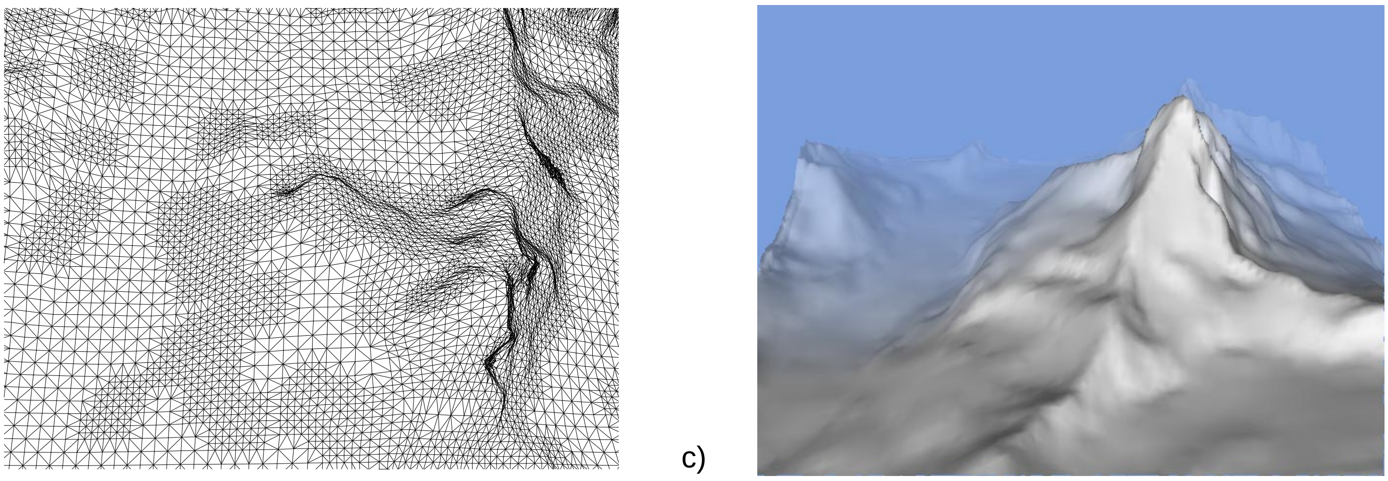

c)

Fig. 22. Adaptive meshing of the digital terrain model: wireframe and Gouraud shaded. τ: threshold, C: remaining coefficients, T: no. of triangles, $\bar{\Delta}$: mean-square error (data source: courtesy provided by Bundesamt für Landestopographie, Bern, Switzerland).
a) $\tau = 0$, $C = 100\%$, $T = 131072$, $\bar{\Delta} = 2.8$. b) $\tau = 0.5$, $C = 18.6\%$, $T = 72386$, $\bar{\Delta} = 3.6$. c) $\tau = 15$, $C = 5.6\%$, $T = 29901$, $\bar{\Delta} = 7.2$.

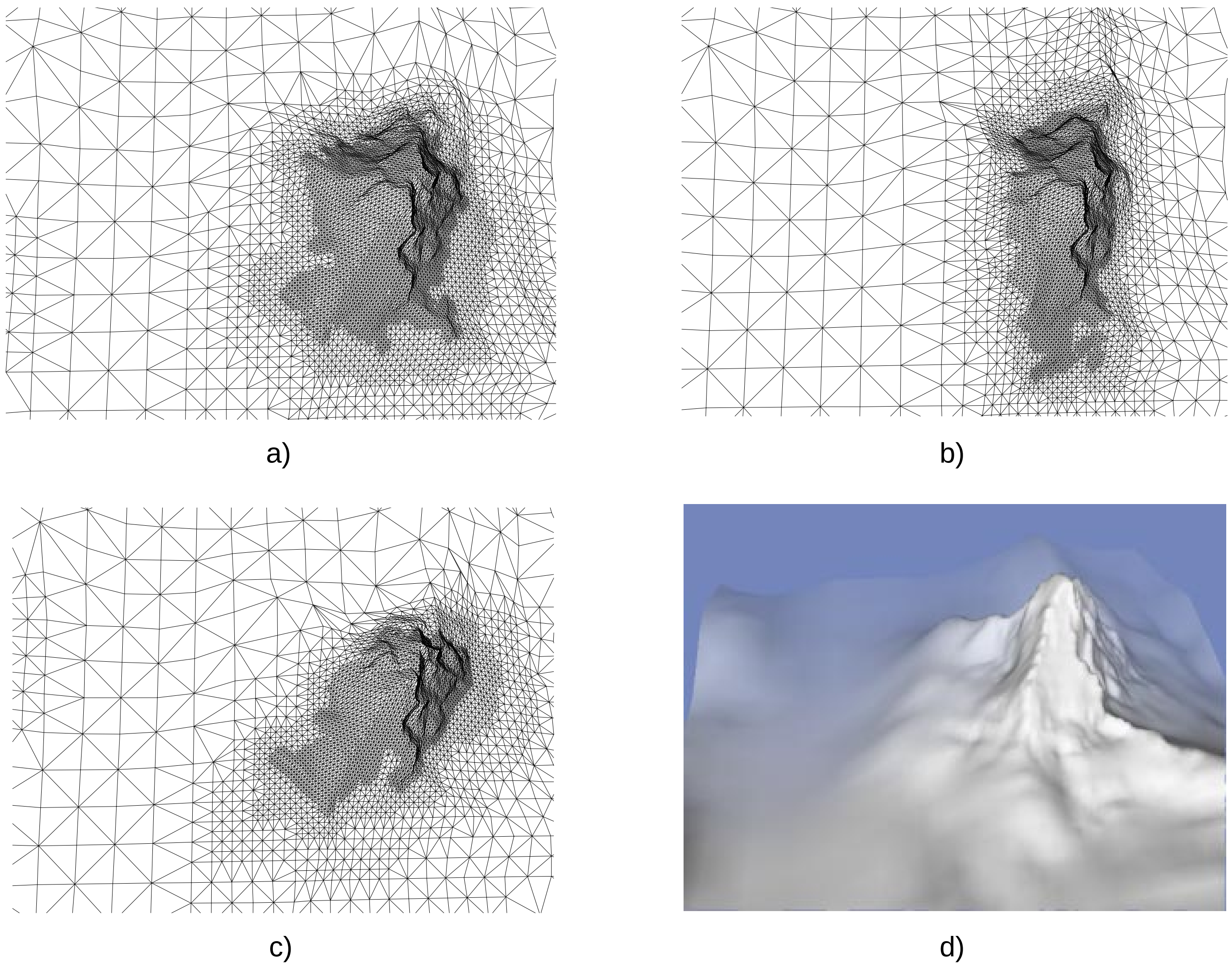


Fig. 23. Level-of-detail meshing using wavelet space filtering (C: remaining coefficients, T: no. of triangles): a) $\sigma_x = 20$, $\sigma_y = 20$, $\Theta = 0$, $C = 5.2\%$, $T = 14753$. b) $\sigma_x = 30$, $\sigma_y = 10$, $\Theta = 0$, $C = 4.2\%$, $T = 11920$. c) $\sigma_x = 20$, $\sigma_y = 10$, $\Theta = \pi/4$, $C = 2.3\%$, $T = 9101$. d) $\sigma_x = 20$, $\sigma_y = 20$, $\Theta = 0$, $C = 5.2\%$, $T = 14753$.

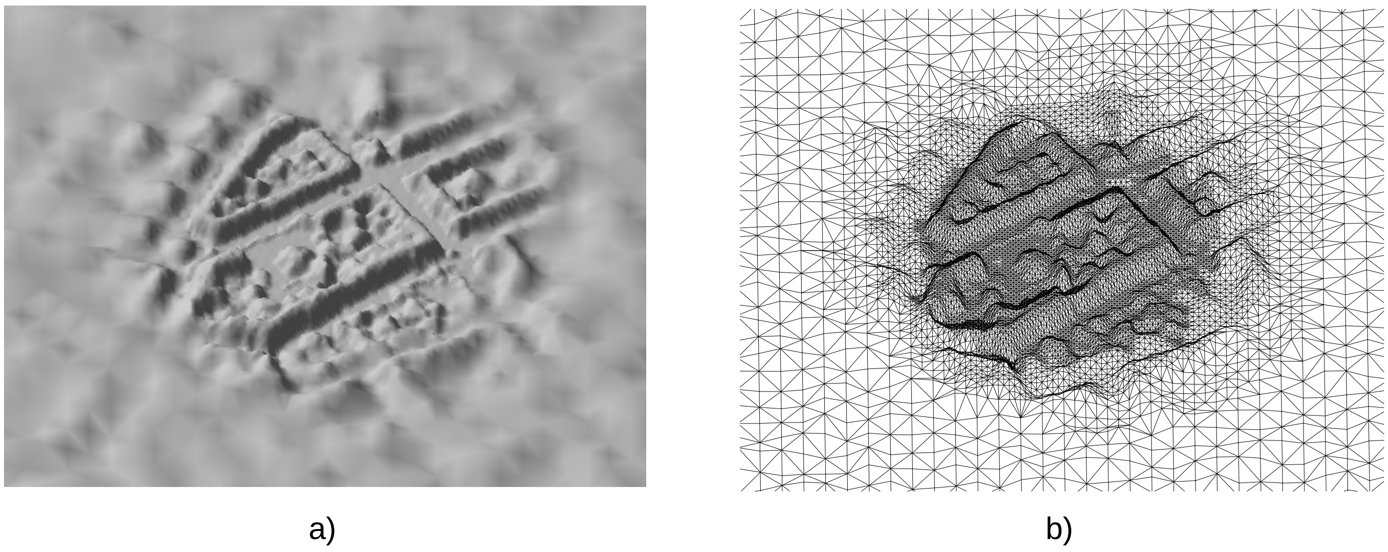


Fig. 24. Locally adaptive mesh of a laser range image of the city of Hannover: $C = 9.0\%$, $T = 19.7\%$:
a) Gouraud shaded image. b) mesh as a wireframe.
(data source: courtesy provided by Dornier GmbH, Germany)

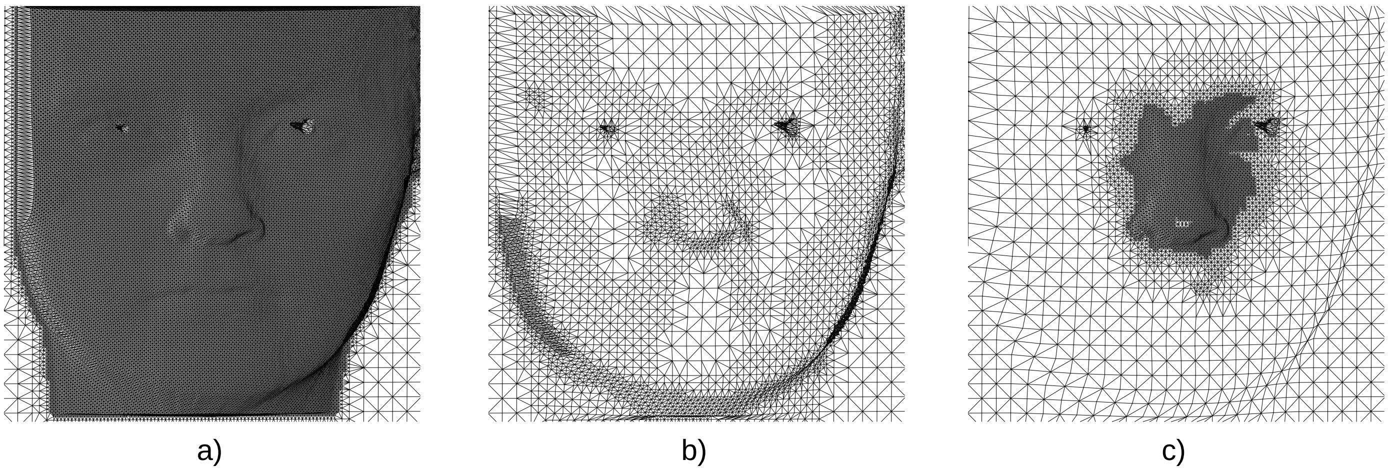


Fig. 25. Globally and locally reduced meshes of Sylvia's image:
a) original: $\tau = 0$, $C = 100\%$, $T = 95142$. b) global threshold: $\tau = 50$, $C = 3.1\%$, $T = 9900$ c) local threshold: $\sigma_x = 20$, $\sigma_y = 25$, $\Theta = 0$, $C = 5.9\%$, $T = 13876$.