ShapeVis

Analyzing Shapes of Segmented 3D Objects

Project Report

Christian Hirsch
e0825187@student.tuwien.ac.at

Vienna University of Technology
Institute of Computer Graphics and Algorithms
Computer Graphics Group
Prof. Dr. Eduard Gröller

University of California, Davis
Department of Computer Science
Visualization and Interface Design Innovation Research Group
Prof. Kwan-Liu Ma, PhD

1 Problem Definition

Surface measurements of a mesh are an interesting feature in various fields in computer graphics and visualization. Such a measurement can for example be used in mesh retrieval. In that case, the surface mesh is used to search for similar meshes in a database of meshes [1]. Our work focuses on quantitatively measuring the surface of meshes to classify them. The resulting measurements are evaluated on the use-case of the classification of benign and malignant breast cancer in Computed Tomography Mammography (CTM) data sets.

In cancer treatment several factors play an important role on how a physician's therapy plan looks like. One of these factors is the shape of the tumor itself. This shape information can also give information about the progress of the disease and the expected healing chances. Traditionally, specialists analyze the CTM data slice-by-slice to gain this information. However, this procedure is time-consuming and difficult to deduce the shape of a three-dimensional object by using only two-dimensional images.

The use-case evaluated in our work, the classification of benign vs. malignant breast cancer, is a shape-based classification problem. The shape of a benign tumor is typically known as a smooth, spherical shape, whereas a malignant tumor's shape has a more rough and irregular structure. Hence, we investigate and compare the surface structure of segmented tumors. Additionally, we measure the evolution of the tumor's surface over time in order to provide insight into the efficiency of a certain therapy.

2 State of the Art

This section describes several methods that can be used for classifying smooth and irregular surfaces. One example are fractal dimensions, which make use of a combination of fractal objects to describe natural objects. Spectral-based methods compute a frequency spectrum, like in audio signal processing, to process and analyze data. The last example described here, are graph-based methods, which build a graph to describe and analyze the data.

Fractal Dimension

Regular surfaces can be expressed by the combination of several basic surfaces. But, natural shapes are often not based on regularity. They are irregular and hard to describe with the combination of basic structures. To overcome this, Nguyen and Rangayyan use the fractal dimension for shape analysis of breast masses in two-dimensional mammograms [2]. As fractals are highly irregular objects they are suitable to describe natural objects. In shape analysis two fractal descriptors are widely used: the box and the length descriptor.

The basis of the box descriptor is a regular split space. Each box can either be a part of the object or it is not. The process of deriving the boxes that belong to the shape, is redone with a successively smaller grid size. Each time the number of boxes that belong to the object are counted. The logarithm of the number of the boxes with the decreasing box sizes is counted and this number is used to describe the shape of the object [2].

Another way to get a quantitative value for the fractal dimension, is the length descriptor. A scale of a fixed length is used to measure the circumference of the object. Again, the process is repeated several times with decreasing length of the scale. With a large scale the circumference of the object is small, but it is getting bigger as the length of the scale decreases. These measures are used to calculate the log over the different scale sizes and, again, this gives a quantitative information of the object's shape [2].

Spectral-Based Shape Analysis

Spectral-based shape analysis is often used in combination with graph-based systems. In spectral mesh processing, the mesh is transformed from the spatial domain to the frequency domain. This is especially done in signal/audio processing. The frequency domain gives information about the structure of a signal. Especially how the signal is built, e.g., how much influence a certain frequency has to the signal. With special filters it is possible to change the signal by, e.g., reducing or amplifying certain frequencies. In this special case, the frequency domain should help to get information about the shape of a certain cancer. Highly irregular surfaces contain a lot of high frequencies and this information is of interest for our breast cancer analysis.

The frequency domain can be calculated by several different mathematical bases. Here two certain types, which are used in mesh processing, are described.

Wavelets The main difference between wavelets and the Fourier transform is that with the Fourier transform, the signal is decomposed in its sine and cosines at different frequencies. In order to compute a wavelet transform a basis function is defined and different scales of this function are used instead.

In order to analyze a shape, a suitable basis function has to be determined. Spherical wavelets are one example where spheres at different scales build the basis. This is, for example described by Nain et al. for statistical shape analysis of brain structures [3]. Yu et al. use spherical wavelets for cortical shape analysis [4]. Despite the spherical wavelets, Antoine and Barache presented the continuous wavelet transform to do multiscale shape analysis [5].

Manifold harmonics The manifold harmonics are built upon the eigenfunctions of the Laplace-Beltrami operator. This operator is known to be a function basis for computing Fourier transform of surface meshes. The work of Vallet and Levy shows how to compute a Fourier like spectrum using the manifold harmonics to apply filters on surface meshes to transform them [6], e.g., to smooth them by filtering out the high frequencies. This work was adapted by Lewiner et al. to make a stereo music visualization [7] where a song's frequency information is used to transform the underlaying 3D model.

Graph-Based Shape Analysis

Graph-based shape analysis is often based on simple shape analysis methods, like those described above, which are used to build up a graph for a certain mesh. This graph is then used as a feature that represents the surface mesh. Several simple shape analysis methods can be combined to create a graph. This helps to describe more complex meshes that are build upon a combination of several other surfaces. Berger et al. uses the first derivative Laplace-Beltrami operator to build a graph, called the Fiedler tree, to do multiscale surface analysis [8]. The method creates surface patches which can be used for irregular multiresolution analysis. Torres et al. compare multiscale fractal dimension and contour saliences, using

a graph-based approach, in two-dimensional images [9]. The methods are tested by classifying 11.000 objects of 1100 distinct classes. Their work shows, that the graph-based contour saliences approach achieves the best separating curve by testing it on the database.

3 Program Design

This section describes the design and the aim of our program that is being developed. One aim of the program is to visualize volumetric data sets, acquired by, e.g., computed tomography or magnetic resonance imaging. A three-dimensional rendering with a state-of-the-art transfer function has been implemented to render the volumetric data. Beside this view, a two-dimensional slice view is used to show slices of the data set. The slice view can change between an axial, sagittal and coronal view. Currently, the segmentation is done within the slice view. Our program is capable of segmenting objects in a volume, as well as converting these objects and exporting them as a mesh. As we aim to analyze the spectra of segmented objects, it needs to be able to compute and show the frequency spectrum. The subsequent goals have been reached within our work so far:

- Two-dimensional slice and three-dimensional renderings of volumetric data sets
- Segmentation functionality with a simple, hence powerful, user interface
- Computation as well as visualization of frequency spectra of surface meshes

3.1 Libraries

In order to achieve our goals within the given time constraints, the use of different libraries is necessary. A library is collection of methods and functions that can be used by different programs. Such libraries help to reduce the implementation work by providing solutions to common problems. In the following, the main libraries are described. More information of the libraries described can be found either in books, publications or on their websites. To load, export and segment data sets the Medical Imaging Toolkit (MITK), which includes the Insight Segmentation and Registration Toolkit (ITK) and Visualization Toolkit (VTK), is used. Qt, a open-source graphical user interface framwork, is used to create the user interface. OpenGL provides functionality to do the two- and three-dimensional rendering. To calculate the frequency spectrum of a mesh, the C++ port of the Arnoldi Package, ARPACK++[1], as well as the linear solver for sparse matrices, TAUCS[2], are used.

[1] `http://www.ime.unicamp.br/~chico/arpack++/`
[2] `http://www.tau.ac.il/~stoledo/taucs/`

3.1.1 VTK

VTK is an open-source toolkit tailored for scientific as well as information visualization, rendering of 3D graphics and image processing. It also provides methods to show three-dimensional widgets in renderings, and two- and three-dimensional annotations.

VTK was started as part of the textbook *The Visualization Toolkit - An Object-Oriented Approach to 3D Graphics* in 1998 [10]. Many contributions came from big medical imaging companies, like GE Healthcare, researchers around the world, as well as software developers who still develop and improve it.

3.1.2 ITK

ITK is an open-source C++ toolkit crafted for segmentation and registration of volume data. Segmentation is the progress of defining certain entities with similar properties to belong to the same object, like a cancer. The toolkit provides a lot of different methods to achieve this in various application fields/scenarios (like medical brain imaging, vessels visualization etc.). It can also be used to export the segmented objects as a mesh. Registration is the method of bringing, or aligning, different data sets that are either measured at different times or with a different method. An example is to align a Computed Tomography (CT) scan of certain part of, e.g. a human body, with the corresponding magnetic resonance imaging scan of the same body part. However, we do not utilize such feature in the current version of our software.

The ITK project started in 1999 by the US National Library of Medicine of the National Institutes of Health. Since then a lot of different companies, as well as software developers and researcher, contributed to the development of this framework [11].

3.1.3 MITK

MITK is an C++ toolkit that combines ITK and VTK. It provides wrapper objects and functions to easily access the functionality provided by those two toolkits. Furthermore, it extends the functionality for the special medical scenarios. MITK can be used as an standalone program which can be extended by

plug-ins as well as a framework which offers all the functionality for self-developed software projects [12].

3.1.4 ARPACK / ARPACK++

ARPACK++ is a C++ wrapper for the Arnoldi Package, ARPACK, which is written in Fortran. It provides methods to calculate eigenvalues and eigenvectors of large sparse matrices using the Arnoldi method [13].

3.1.5 TAUCS

TAUCS is a library to solve linear systems of large sparse matrices. Beside the in memory computation, the current development version also provides an out-of-core algorithm to compute a Cholevsky factorization and linear equations using this factorization [14]. This was one of our main reasons for using this library. Computing eigenvalues of huge matrices, which may be possible due to geometry meshes with several thousands of vertices, may not fit in the computers memory. Out-of-core algorithms store parts of the results on the hard disks. This makes the program independent from the computers amount of memory.

4 Implementation Details

The implementation details and algorithms used in our program are described in this section. Basically, this section focuses on the two main parts: How to segment parts of volumetric data, and how to compute the frequency spectrum of a geometry mesh using manifold harmonics.

4.1 Segmentation

As we focus on analyzing the surface of segmented objects, but the focus is on analyzing meshes, not on segmentation, only a simple and easy to use segmentation is implemented.

The algorithm's input are seed points that are selected by the user in the slice view. The user has to pick at least one point that correspond to the desired object in the volume data. With more than one seed points selected a mean value and a variance of the density values is computed, which are used to calculate an upper and lower threshold for the region. If the user selects just a single seed point, the 26 connected neighbors of the voxel are used to calculate the density mean value and variance.

Beginning from the seed points, a region growing algorithm, provided by ITK, adds all the voxels to the segmentation which are between the calculated lower and upper thresholds. To store the segmented result, each volume has a masking volume at it's disposal. The masking volume has the same size as the original volume with a voxel being 1 byte in size. Segmented voxels in the volume that remain together get the same mask value. So it is possible to store 255 different segmentations. One value is used to specify non segmented parts of the volume.

4.2 Mesh Generation

The export function is provided by ITK. After segmenting parts in the volume data a marching cubes algorithm is used to generate a surface mesh out of the segmented voxels. Generally, it would be possible to use a marching cubes algorithm to create a mesh out of the volume data without segmentation. In this case a threshold needs to be specified which is used to decide if there is a mesh boundary or not. However, this might lead to surfaces with a lot of holes and

also structures that do not remain to a certain part as this is a global operation. The creation of a surface with segmented voxels creates what the segmentation represents.

4.3 Manifold Harmonics

The idea behind manifold harmonics is to make a Fourier-like transformation of geometry data to a frequency spectrum. But because Fourier transform uses the combination of sines and cosines, a different approach needs to be found that can be used to find bases which can describe the mesh information. Vallet and Levy [6] use the Laplace-de-Rahm operator from the Discrete Exterior Calculus of the mesh.

This operator can be expressed as a $n \times n$ matrix Δ, where n is the number of vertices. The coefficients Δ_{ij} are 0 if the vertices i and j are not adjacent and otherwise they are calculated by

$$\Delta_{ij} = -\frac{\cot \beta_{ij} + \cot \beta'_{ij}}{\sqrt{|v_i^*||v_j^*|}},$$

$$\Delta_{ii} = -\sum_j \Delta_{ij}.$$

β_{ij} and β'_{ij} are the angles of the opposite vertices of vertex i and vertex j. $|v_i^*|$ respectively $|v_j^*|$ are the areas of the Voronoi region of the two vertices i and j.

The bases that are used for the frequency transform, the Manifold Harmonics Basis (MHB), can now be computed by calculating the Eigenpairs of Δ. The Eigenpair computation is achieved using the reverse communication interface of ARPACK++ in combination with the TAUCS library.

Finally, after the MHB is calculated, the last part is to obtain the frequency spectrum. This is done by calculating $\tilde{x}_k$ which is the amplitude of the x coordinate values of the vertices at frequency k. To do this the following equation is used:

$$\tilde{x}_k = \sum_{i=1}^{n} x_i H_i^k$$

where H_i^k is the i^{th} value of the k^{th} basis of the MHB. This is also done for the y and z coordinate values. For detailed information on the implementation please refer to the original publication of Vallet and Levy [6].

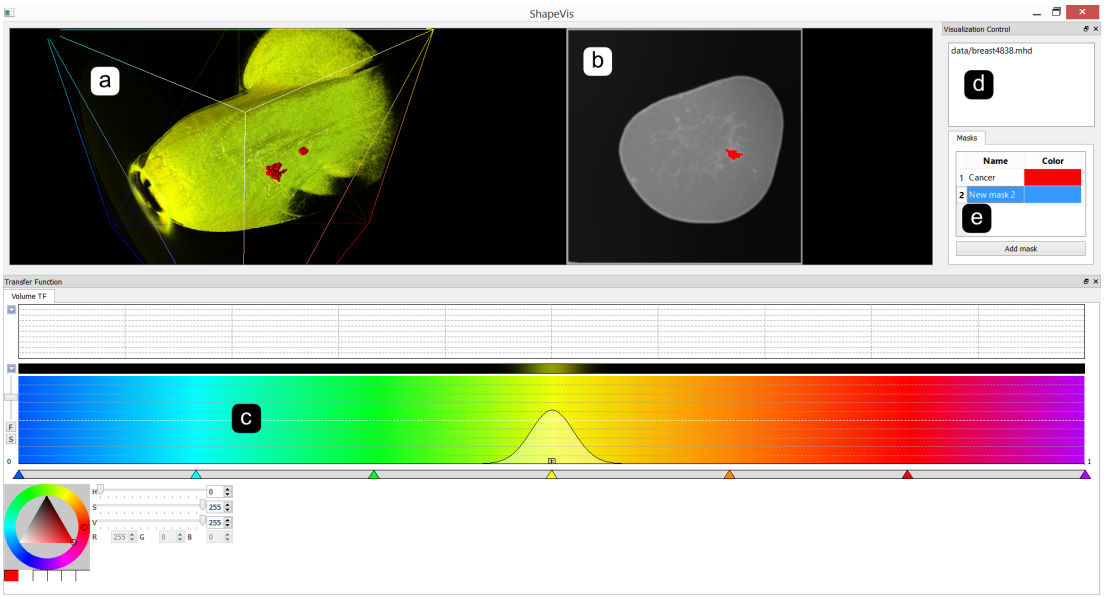

Figure 1: Screenshot of the program with the different user interface and visualization parts.

5 Graphical User Interface

This section describes the current Graphical User Interface (GUI) of our implemented software prototype. The basic user interface of the program is shown in Figure 1. In this screenshot a breast data set is loaded and two objects in it are segmented (shown in red). Figure 1(a) shows the three-dimensional rendering view. Figure 1(b) shows the interactive slice renderer which can render axial, coronal and sagittal slices. The slice view is also used for the segmentation. The transfer function, Figure 1(c), is used to change the appearance of the rendered volume data set. The listing in Figure 1(d) shows which volumetric data sets and geometry meshes are loaded. It can also be used to interact with the meshes. Figure 1(e) is used create and organize masks and their corresponding colors.

5.1 3D View

The three-dimensional view, Figure 1(a) can render volumetric data, as well as geometry meshes. In the figure a breast data set is loaded and rendered. Also a segmentation result is rendered in this example. The data in the 3D view can be rotated and scaled.

To render a volumetric data set, a state-of-the-art volume ray casting method is used. As described above, the mask is another volumetric data set. The loaded data set is rendered with the transfer function specified in Figure 1(c), then the corresponding mask is rendered with the colors specified in the mask user interface, see Figure 1(e).

5.2 2D/Slice View

The slice view, Figure 1(b), renders the volumetric data set in two-dimensional slices. The user is able to chose among three different slice orientations by clicking the corresponding key on the keyboard. E.g., if the user wants to view the YZ planes, i.e. looking along the X direction, the "X" key on the keyboard has to be pressed. To change the displayed slice, the mouse wheel is used. Further interactions are planned, however, due to our time constraints, they are not implemented yet.

As described above, the two-dimensional view is also used for the segmentation. By right-clicking into the slice view, seed points are selected and those are used to create segmentations.

5.3 Transfer Function

Figure 1(c) shows the transfer function. A transfer function is a mapping of a voxel's density value, X-axis, to a specified color. The Y-axis is used to specify an opacity that is used for the specified color. The X-axis ranges from a density value of 0.0 to 1.0. As the volumetric data sets are normalized, this is sufficient. The Y-axis ranges from fully transparent(0.0) to fully opaque (1.0).

The transfer function is used to change the appearance of the volumetric rendering. The transfer function can be adjusted by clicking on the color band, as shown in Figure 1(c). Left clicking creates a mapping, whereas right clicking (swiping over) it deletes the mapping. The user can specify the colors that are used for certain density values below the color band. Clicking with the left mouse button inserts a new color or an already created color can be changed. Right clicking onto a color point destroys the color value. The gaps between the user-specified colors are linearly interpolated and displayed in the color band.

6 Manual

This section is a manual of the developed program. Every functionality will be described with step-by-step instructions.

6.1 Loading Data

There are two ways how volumetric data sets or meshes can be loaded:

Passing arguments: When the program "ShapeVis.exe" is started from the command line, it is possible to pass arguments to it: By adding "volume *filename*" the program loads the volumetric data set *filename*. By appending "geometry *filename*" the program loads the geometry mesh *filename*. It is also possible to combine the arguments and load many different geometry meshes and volumetric data sets at once.

Using the GUI: By right clicking into the file list a context menu opens up with the possibility to either load a geometry, or volumetric data set, see Figure 2(a). By clicking on the menu item, a open-file dialog opens up which can be used to locate the file.

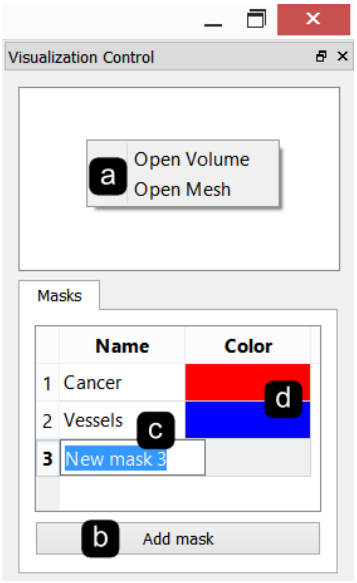

Figure 2: The visualization control allows the user to load geometry meshes and volume data, as well as creating and organizing masks.

6.2 Segmenting Data

First of all, a mask needs to be created. In the provided user interface, see Figure 2, this can be done by clicking "Add mask" (Figure 2(b)). Then a name can be chosen for the mask by double clicking onto the name (Figure 2(c)). But this is not necessary for just segmenting data. It should just help to keep the segmented objects organized. The color, that will be used for rendering the segmented object, can be changed by double clicking onto the displayed color (Figure 2(d)). A color dialog opens up that is used to select a color.

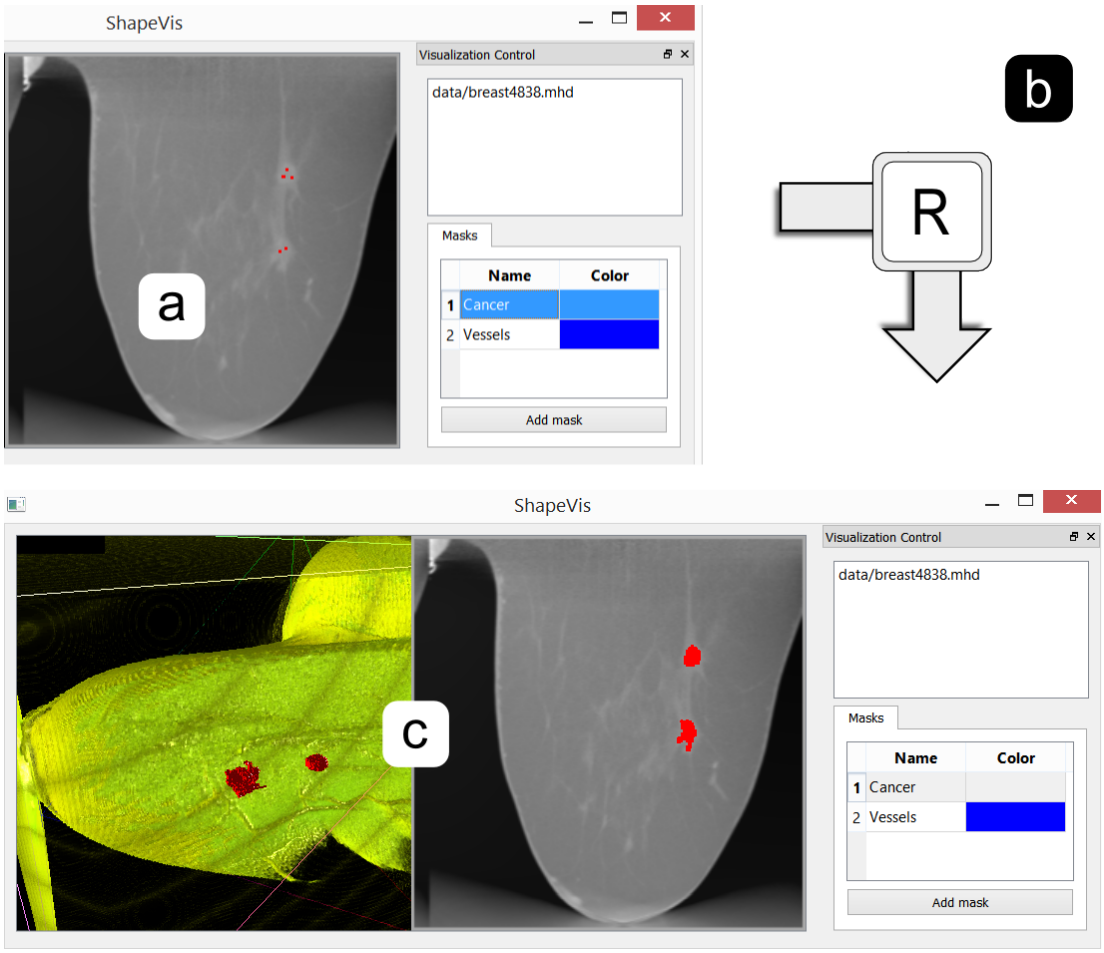

Figure 3: The basic segmentation tasks: (a) select seed points; (b) press "R" key on the keyboard; (c) view the result.

After a name and a color are chosen, the mask needs to be selected by clicking onto it. When the mask is selected, it will get highlighted. The user needs to keep in mind, that all the following steps for segmenting data will use the selected mask.

The next step is to select seed points in the slice view by right clicking onto the view (Figure 3(a)). The user can browse through the slices and select the voxels that remain to the region. When a voxel is selected, it will appear with the specified color mask in the slice and volume view.

Finally, after selecting all the seed points, the algorithm starts to segment volume objects by pressing the "R" key on the keyboard (Figure 3). When the algorithm

finishes, the segmentation will appear in the volume and slice view as shown in Figure 3. It is also possible to change colors after segmenting regions.

6.3 Exporting Geometry

After voxels have been segmented, it is possible to create a geometry and export it. Again, the selected mask is used to specify the segmented voxels that will be converted and exported. The export algorithm is started by hitting "E" on the keyboard. A save file dialog opens where the user specifies the file name and the location where he wants to store exported result. To store the geometry, the binary StereoLithography Interface Specification (STL) format is used. The result is displayed in the three-dimensional view and in the file list the geometry will be added.

6.4 Vessel Segmentation

The program is also able to do vessel segmentation. But this needs some more user interaction.

The first step, after the volume data has already been loaded, is to apply a vessel filter (which is based on an Hessian filter in combination with a Hessian to vesselness filter provided by ITK) which filters out tubular information of the volume. This filter is initiated by hitting "V" on the keyboard (Figure 4(a)). The loaded volume will be changed (not the file itself!). The changed volume will be visible in the slice view, as well as by applying a proper transfer function, see Figure 4(b), in the three-dimensional view.

The second step is to measure the values values for the further segmentation. By moving the mouse cursor over the color band of the transfer function, a value will show the density of the current X-position. This can be used to measure a lower threshold as well as an upper threshold, see Figure 4(c).

Finally the segmentation algorithm is started using the context menu in the visualization control's file list. By right clicking on the volume filename it is possible to select "Voxel Range Selector", see Figure 4(d). This will open up two input dialogs where the lower and the upper threshold can be specified. Then the algorithm is started.

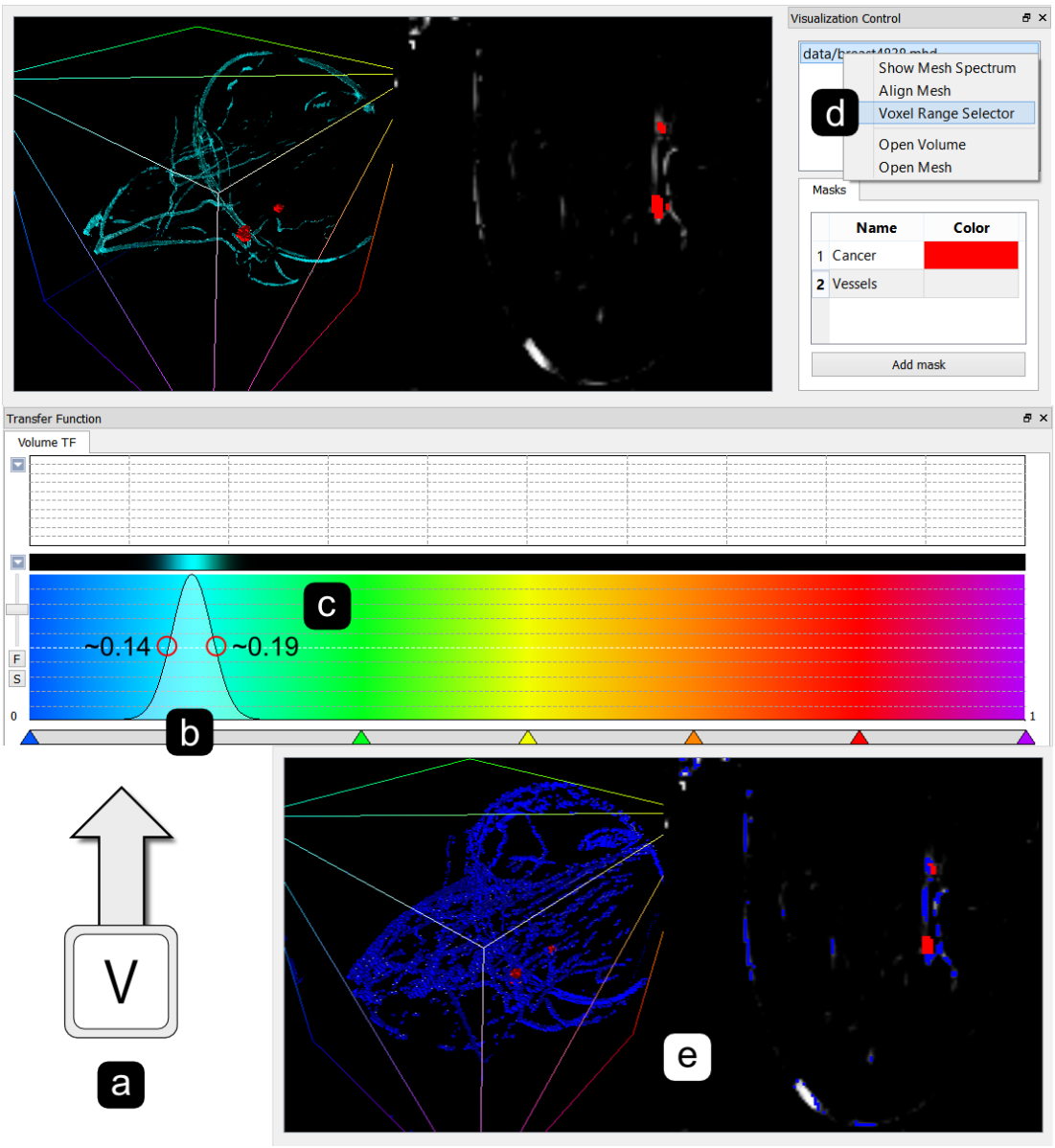

Figure 4: An overview of the basic parts to segment vessels.

After the algorithm finished, the segmented voxels are rendered in the slice- as well as in the three-dimensional view, see Figure 4(e). Like described in Section 6.2, the selected mask is used for segmenting the vessels. In Figure 4(e) two segmented masks are shown. The red objects are cancer parts, blue is used for rendering the segmented vessels.

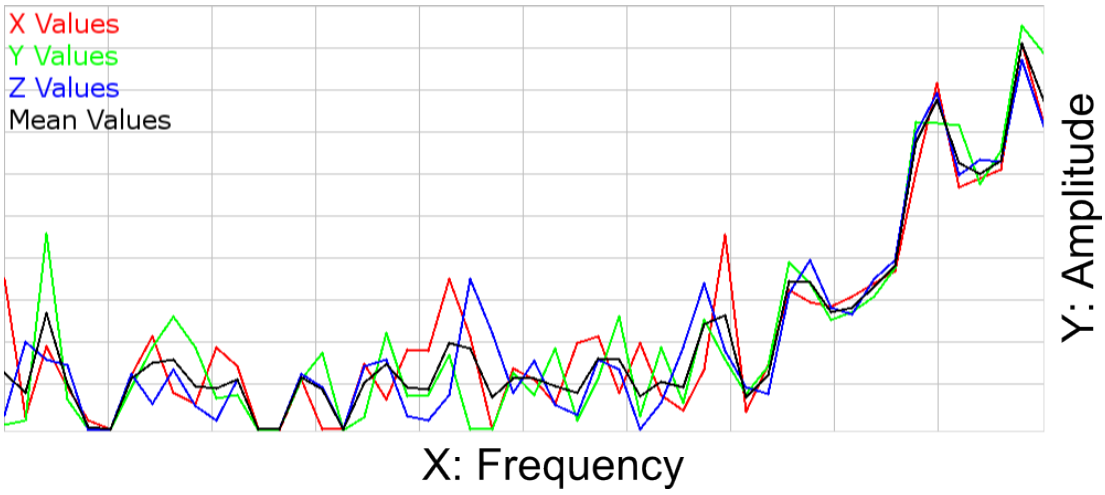

Figure 6: The frequency spectrum window. A sample frequency spectrum of a star-like mesh was computed and is shown here.

6.5 Spectral Analysis

To do spectral analysis a mesh needs to be loaded in the program. This can either be done by loading a geometry file (Section 6.1) or by segmenting objects of the volume (Section 6.2) and exporting the segmented objects (Section 6.3).

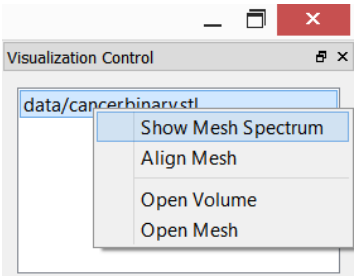

Figure 5: The context menu changes when a mesh is selected.

When a geometry mesh is loaded, the spectral analysis is started by clicking "Show Mesh Spectrum" in the context menu of the file list, see Figure 5. The context menu is opened by clicking on the desired list entry of the geometry mesh with the right mouse button.

When the algorithm finishes, a new window with the result pops up. Figure 6 shows such a sample frequency spectrum. In this example the absolute values of the amplitudes of the spectra are rendered. The amplitudes are aligned and zoomed to provide a better view of the result. The X axis represents the different frequencies, the Y axis is the value of the corresponding amplitude of the frequency. The results show the spectrum for the X (red), Y (green), Z (blue) and the mean value of all axis (black).

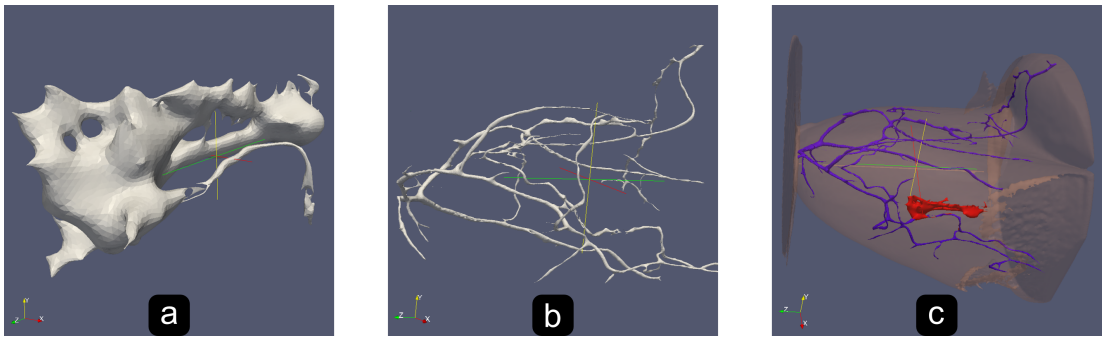

Figure 7: Results of segmentation. (a) a segmented cancer, (b) segmented vessels using the vessel filter, (c) shows a combination of (a), (b) and a segmented breast shape.

7 Results

This section shows some results that are created with our developed program. Figure 7 shows results of segmented objects, like a cancer (Figure 7(a)), segmented vessels (Figure 7(b)). Figure 7(c) shows the combination of the segmented objects (a) and (b) with a segmented breast surface rendered semi-transparent.

First, in order to validate our implementation and discover means for visualizing frequency spectra, we generated several phantom data sets (Figure 8(a) and Figure 8(b)). The phantom data sets, a sphere, Figure 8(a), and a star-like structure, Figure 8(b), were chosen because of the smoothness of a sphere and the irregularity of the star-like structure. The spectra of the corresponding objects are aligned to the same amplitude ranges and already show a big difference. Whereas the spectrum of the sphere, Figure 8(a), is in a same range over the whole frequency values, the spectrum of the star-like structure, Figure 8(b), shows an high amount of high frequencies.

Secondly, we applied our method to the medical data provided by domain experts. As we did not get sufficient data sets, a more elaborate analysis is still to be done. Figure 8(c) shows the segmented cancer with the corresponding spectrum. The vessels and the corresponding spectrum is shown in Figure 8(d).

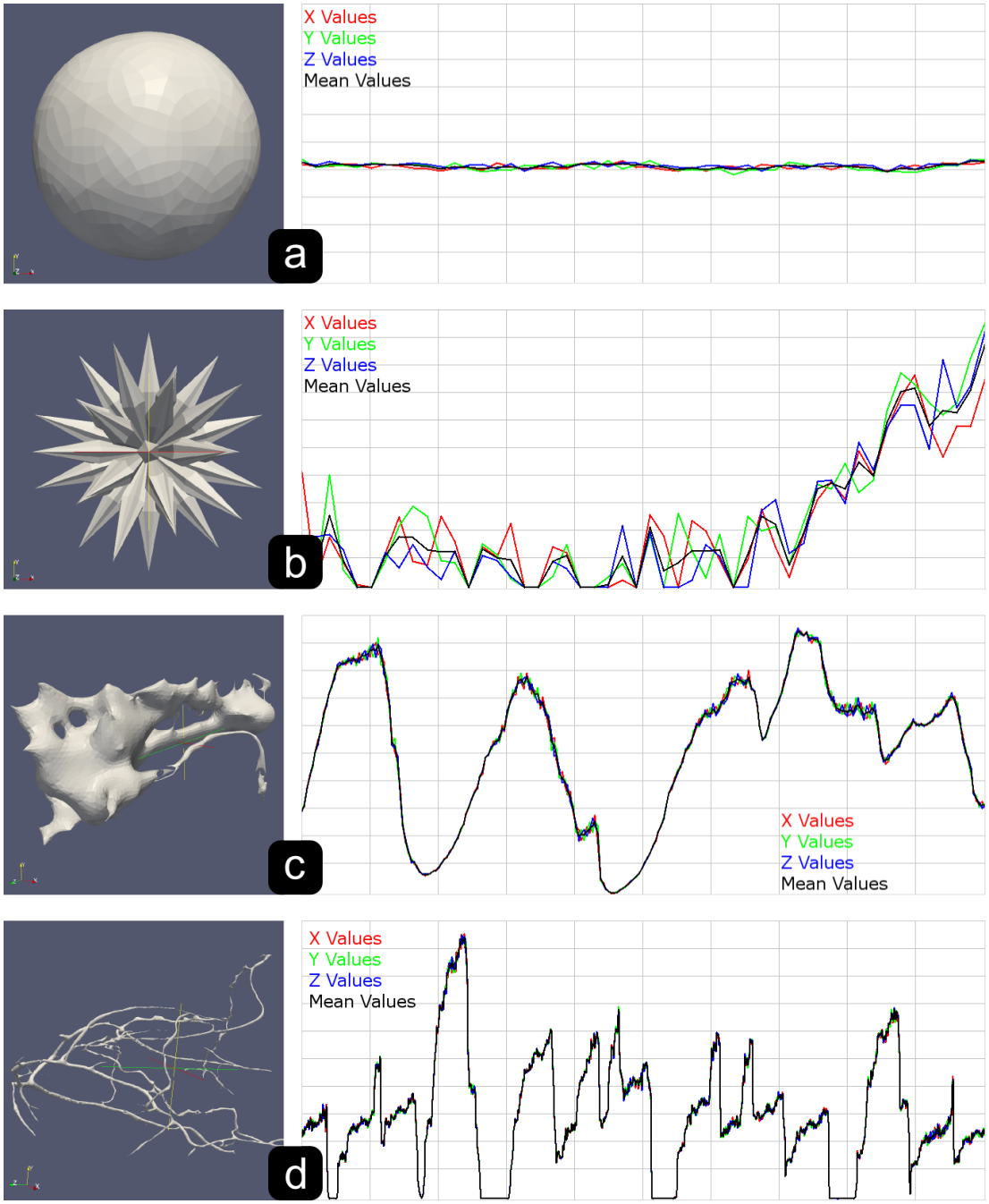

Figure 8: The frequency spectrum window. A sample frequency spectrum of a star-like mesh was computed and is shown here.

8 Continuation and Outlook

We implemented several tasks in order to reach our main goals. Nevertheless, our program is still under construction and there a many avenues for future improvements.

First of all the most important improvements that will be implemented are the visualizations and calculations of the frequency spectrum of geometry meshes. It should be possible to change between different views of the spectrum, e.g., between real and absolute values. Furthermore, the views need to be extended to show more information, like the amplitude values (on the grid), or the displayed frequencies (on the grid). A nice feature is to show the frequency information on the mesh itself: by mapping the frequency information with help from a transfer function back to a vertex. Additionally, we want to provide means of computing a quantitative value from a frequency spectrum.

Improving and extending the GUI: To start algorithms and filters by using key input solely is not optimal, although medical doctors usually appreciate shortcuts. Additionally, we will provide a menu for the program, as well as suitable context menus should replace the initiations and make the program easier to use.

Moreover, advanced segmentation techniques can be applied and implemented. As the focus of our project is visualization, we will tackle this point if we have sufficient time available. First of all, we want to provide a comprehensible visualization of different tumor shapes in order to visually distinct between benign and malignant ones.

References

[1] H.-Y. Wu, T. Luo, L. Wang, X.-L. Wang, and H. Zha, "3D shape retrieval by using manifold harmonics analysis with an augmentedly local feature representation," *Proceedings of the 8th International Conference on Virtual Reality Continuum and its Applications in Industry - VRCAI '09*, p. 311, 2009.

[2] T. Nguyen and R. Rangayyan, "Shape Analysis of Breast Masses in Mammograms via the Fractal Dimension.," *Proceedings of the Conference of the IEEE Engineering in Medicine and Biology Society.*, vol. 3, pp. 3210–3213, 2005.

[3] D. Nain, M. Styner, M. Niethammer, J. J. Levitt, M. E. Shenton, G. Gerig, A. Bobick, and A. Tannenbaum, "Statistical Shape Analysis of Brain Structures Using Spherical Wavelets.," *Proceedings / IEEE International Symposium on Biomedical Imaging: from nano to macro. IEEE International Symposium on Biomedical Imaging*, vol. 4, pp. 209–212, apr 2007.

[4] P. Yu, X. Han, and F. Ségonne, "Cortical surface shape analysis based on spherical wavelet transformation," *Proceedings of the Conference on Computer Vision and Pattern Recognition*, pp. 0–7, 2006.

[5] J. Antoine and D. Barache, "Multiscale shape analysis using the continuous wavelet transform," *Proceedings of the International Conference on Image Processing*, vol. 3, no. 1, pp. 291–294, 1996.

[6] B. Vallet and B. Lévy, "Spectral Geometry Processing with Manifold Harmonics," *Computer Graphics Forum*, vol. 27, no. 2, pp. 251–260, 2008.

[7] T. Lewiner, C. Marques, and J. Paixão, "Stereo music visualization through manifold harmonics," *The Visual Computer*, vol. 27, no. 10, pp. 905–916, 2011.

[8] M. Berger, L. Gustavo Nonato, V. Pascucci, and C. T. Silva, "Fiedler trees for multiscale surface analysis," *Computers & Graphics*, vol. 34, no. 3, pp. 272–281, 2010.

[9] R. da S. Torres, A. Falcão, and L. da F. Costa, "A graph-based approach for multiscale shape analysis," *Pattern Recognition*, vol. 37, no. 6, pp. 1163–1174, 2004.

[10] W. Schroeder, K. Martin, and B. Lorenzen, "The Visualization Toolkit-An Object-Oriented Approach to 3D Graphics," 1998.

[11] T. S. Yoo, M. J. Ackerman, W. E. Lorensen, W. Schroeder, V. Chalana, S. Aylward, D. Metaxas, and R. Whitaker, "Engineering and Algorithm Design for an Image Processing API: A Technical Report on ITK - the Insight Toolkit," 2002.

[12] I. Wolf, M. Vetter, I. Wegner, T. Bttger, M. Nolden, M. Schbinger, M. Hastenteufel, T. Kunert, and H.-P. Meinzer, "The Medical Imaging Interaction Toolkit," *Medical Image Analysis*, vol. 9, pp. 594–604, 2005.

[13] F. M. Gomes and D. C. Sorensen, "ARPACK++: A C++ implementation of ARPACK eigenvalue package," 1997.

[14] S. Toledo, "TAUCS: A Library of Sparse Linear Solvers," 2003.