Computerized Medical Imaging and Graphics xxx (2009) xxx–xxx

Contents lists available at ScienceDirect

Computerized Medical Imaging and Graphics

journal homepage: www.elsevier.com/locate/compmedimag

Non-convex polyhedral volume of interest selection

Raphael Fuchs[a,*], Volkmar Welker[b], Joachim Hornegger[c]

[a] Computer Graphics Lab, ETH Zürich, Switzerland
[b] Institute of Mathematics, Philipps-University Marburg, Germany
[c] Institute of Pattern Recognition, Friedrich-Alexander-University Erlangen, Germany

ARTICLE INFO

Article history:
Received 4 March 2009
Received in revised form 6 July 2009
Accepted 14 July 2009

Keywords:
Human–computer interaction
Volume of interest (VOI)
Non-convex sub-volume selection
User interaction
Polyhedra
Clipping planes
Fuzzy selection
3D polygons
Volume editing

ABSTRACT

We introduce a novel approach to specify and edit volumes of interest (VOI for short) interactively. Enhancing the capabilities of standard systems we provide tools to edit the VOI by defining a not necessarily convex polyhedral bounding object. We suggest to use low-level editing interactions for moving, inserting and deleting vertices, edges and faces of the polyhedron. The low-level operations can be used as building blocks for more complex higher order operations fitting the application demands. Flexible initialization allows the user to select within a few clicks convex VOI that in the classical clipping plane model need the specification of a large number of cutting planes. In our model it is similarly simple to select non-convex VOI. Boolean combinations allow to select non-connected VOI of arbitrary complexity. The polyhedral VOI selection technique enables the user to define VOI with complex boundary structure interactively, in an easy to comprehend and predictable manner.

© 2009 Elsevier Ltd. All rights reserved.

1. Introduction

Defining a volume of interest (VOI for short) is basically as difficult as defining any other object in three dimensions. In medical and scientific volume visualization applications the user wants to specify his degree of interest for each data sample in the volume. For modeling three-dimensional objects a wide range of computer aided design (CAD) techniques are available. Therefore, it can be appropriate to use the full power of CAD tools when designing a very intricate VOI geometry. However, these tools require the user to have undergone a specific training. In addition, their user interface is too sophisticated to be used in the medical environment.

In this paper we aim at a general approach for VOI specification that is still suitable for real-time interaction with volume data. We propose the use of a polygonal mesh that defines a polyhedral region. Typical interactions such as removing small artifacts from reconstruction, getting an overview of the data and clipping obscuring regions from the view for presentation can be done very easily by manipulating a polygonal mesh.

Understanding volumetric data is hampered by occlusion and reconstruction artifacts, posing problems especially for data of high structural complexity. An important task is to remove obstructing structures from the volume, e.g. by assigning high transparency values to them. When it is not possible to use automatic segmentation or interactive transfer function definition then interactive VOI selection offers a solution that is independent from the data values in the volume. Especially in medical data we find structures of high semantic correlation and neighboring in space that do not exhibit similar density values. In this case changing the transfer function may not help in understanding the data. Automatic segmentation may be impossible when the VOI is of very complex structure. Injuries are an example. Here different types of affected tissue and bone belong to one VOI. In cases like this the portion of the body that needs treatment is often well defined in terms of its spatial properties, but it is difficult or even impossible to find value ranges in the dataset that describe this VOI. Also, the more pathological the tissue in the VOI, the less we can expect automated techniques to be applied successfully.

Interactive VOI selection and transfer function specification can complement each other: The VOI selection helps to select data features based on different spatial characteristics of the feature, while transfer functions help to inspect the volume by discriminating between attribute values of the data. Another situation where general interactive VOI selections are important is searching and examining small structures (like teeth) in volumetric scans [1], when the number of relevant voxels is very small and automatic techniques have large error rates. Especially in surgery there has to be a trade-off between the conflicting requirements of complexity,

* Corresponding author.
E-mail addresses: raphael@inf.ethz.ch (R. Fuchs), welker@mathematik.uni-marburg.de (V. Welker), joachim.hornegger@informatik.uni-erlangen.de (J. Hornegger).

0895-6111/$ – see front matter © 2009 Elsevier Ltd. All rights reserved.
doi:10.1016/j.compmedimag.2009.07.002

Please cite this article in press as: Fuchs R, et al. Non-convex polyhedral volume of interest selection. Comput Med Imaging Graph (2009), doi:10.1016/j.compmedimag.2009.07.002

accuracy and generality of the model of interaction on the one side and the speed of interaction with the model on the other side [2]. In the medical environment interaction has to allow switching between medical tools and using input devices like joysticks or trackballs. A recent study [3] concludes that the currently employed trackball may not be the optimal device for navigation within large CT angiography data sets. Therefore, any new approach that eases region selection on medical equipment will be beneficial.

The contributions of the presented paper are:

- A general approach for creating non-convex volumetric selections in three dimensions by editing vertices, edges and faces of a polyhedron.
- High-level operators for simplified user-interaction.
- An approach for simple specification of volumetric selections of arbitrary genus by combining multiple selections.
- Finally, a detailed discussion of algorithmic problems that arise during interaction, including a proven algorithm to deal with holes when deleting vertices of the mesh.

This paper is organized in the two parts *method description* and *algorithm with proof*. The first part (Sections 3 and 4) gives an illustrative description of the suggested method. The second part (Sections 5 and 6) discusses the most relevant algorithmic problems and proves the correctness of the presented algorithm.

2. Related work

Weiskopf et al. [4,5] discuss efficient implementation of volume clipping for direct volume visualization applications. Their approach allows fast evaluation of very general clipping geometries based on polygonal meshes. Weiskopf et al. illustrate their approach with different clipping geometries such as cubes and spheres of varying sizes, but do not discuss interactive manipulation of the shape of the clipping geometry. Museth and Lombeyda [6] apply volume clipping as suggested by Weiskopf to unstructured grids. They present a rendering algorithm that allows interactive constructive solid geometry (CSG) style clipping. Here the user can interactively define a clipping geometry by incremental combination of predefined clipping objects with Boolean operations. The approach is inspired by the work of Chen and Tucker [7] on constructive volume geometry.

Pham et al. [8] give an overview of automatic and half-automatic approaches to the computer aided inspection of anatomic structures. Automatic techniques often use combinations of finding seed points (anatomical landmarks) [9], region growing based on additional criteria [10], or pattern matching to include background knowledge [11].

Wong et al. [12] present 'intelligent scissors'. This contour selection tool for surfaces in volumetric models allows to cut off the volume of interest. The user first draws a closed contour on the volume surface as the boundary. Then, a cutting surface is computed based on minimizing a cost function that locally tries to match the isosurface of the volumetric data. Sibbing and Kobbelt [13] suggest to segment the volume with isosurfaces and to define the VOI by selecting these regions. Similar approaches use watershed catchment basins [14] or regions where voxels belong together if they 'slide' to the same local minimum [15]. Volume painting [16] is another data dependent approach that is inspired by artistic techniques. The user adds 'paint' and the first non-transparent voxel in viewing direction is added to the selection. Wang and Kaufmann [17] presented volume sculpting: here the user can model three-dimensional objects starting with the bounding box and refine them with tools inspired by sculpting (i.e. carving knives, saw, drill, etc.). Nakao et al. [18] suggest the use of a tetrahedral grid to support selection of VOI with arbitrary boundary. They introduce predefined clipping geometries (spheres, cubes and so on) with user interactivity in the form of position and diameter controls.

Using polyhedra for volume selection is not a new idea. Darrah et al. [19] propose a method to construct convex polyhedral selections by selecting points in 3D and using their convex hull as the VOI. Dietrich et al. [20] suggest a combination of using a convex polyhedron and voxel based sculpting. Using polyhedra for interactive volume selection has been proposed multiple times, but to our knowledge, their key benefit has not been exploited so far, namely that polyhedra are able to describe non-convex shapes.

A recent (not yet published) patent [21] discusses volume selection by adding and moving vertices of a polyhedron for volume selection. Because of this restriction some natural shape transformations can only be performed through a complex sequence of vertex operations. Only after the integration of edge operations a full set of natural transformations is available. In addition, the patent [21] does not mention solutions to geometrical problems arising from vertex deletion.

3. Non-convex VOI selection

For an introductory illustration of the suggested volume selection approach we refer to Fig. 1.

(a) We assume that no prior information about the VOI is available in the loaded data set. Initially the clipping polyhedron is given as the bounding box of the current volume. Thus nothing is clipped. After inspection of the data the user can initialize the VOI using a 2D selection by defining a contour around a feature of interest.
(b) The user can modify the VOI by editing the geometry of a polyhedron using operations that move, delete and create new vertices, edges and faces. There are movement operators that work on different primitives: in (1) the user has selected the vertex in the upper right front and moved it backwards in the direction of the red arrow and in (2) the user has moved a face along its default direction (e.g., its normal). (3) When vertices become useless or hindering, it is possible to remove them. Increasing the complexity of clipping objects is possible by introducing new vertices to the object (4). The new vertices do not change the currently clipped volume. Only the polygon containing the new vertex is triangulated. Cutting allows to do rough editing operations either using clipping planes (5) or using a 2D scissor tool (6).
(c) Without constraints the VOI can become too complex for intuitive interaction. Therefore we have to set certain restrictions on an individual polyhedron:

1. The faces of the polyhedron are convex and planar.
2. The polyhedron does not self-intersect.
3. The polyhedron is homeomorphic to a ball (i.e., it has no holes).

Due to these usability considerations it is not possible to create a single selection with holes or multiple non-connected components. Selections of this kind become possible by combination of multiple polyhedra. Adding the combination step we get a flexible yet intuitive interaction model.

3.1. Initialization

Very often the user has a clear idea of the two-dimensional region on the screen that is of interest. Therefore we allow to build the selection from a two-dimensional polygon p which is drawn interactively on the viewing plane. A three-dimensional selection

Please cite this article in press as: Fuchs R, et al. Non-convex polyhedral volume of interest selection. Comput Med Imaging Graph (2009), doi:10.1016/j.compmedimag.2009.07.002

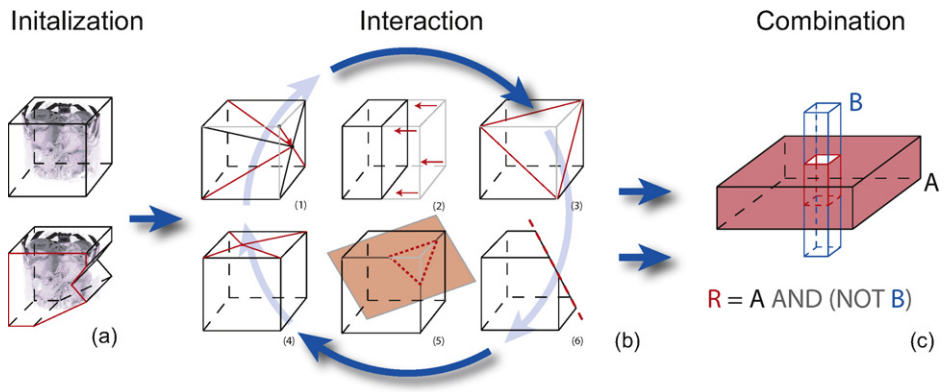

Fig. 1. Volume selection is based on three stages. (a) The selection is initialized as the bounding box of the volume. Alternatively, the user draws a polygon onto the image plane, which is extruded to 3D. (b) Through iterated application of editing operators that insert, move, delete or cut elements and regions of the polyhedron the user can refine the selection. The set of presented operators is complete in the sense that they are sufficient to construct any conceivable polyhedron. (c) Selections with holes and fuzzy membership values can be achieved by fuzzy logic combination of multiple selections. Here a selection with a hole is the result (R) the expression 'A AND (NOT B)'.

polygon is built by extruding the polygon in viewing direction. Fig. 3 gives an illustration. More formally, if v is the viewing projection and D the spatial domain, the resulting polyhedron is $v^{-1}(p) \cap D$.

3.2. Interaction

In this section we define interactions that change the shape of a polyhedron. The low-level modifications are sufficient to construct any conceivable VOI, i.e., it is possible to construct any closed polyhedron using just the low-level interactions. This may take a large number of editing steps though. High-level interactions enable the user to perform large scale modifications. The high-level interactions are more application centered and the operators we present are just examples for the wide range of possibilities on that level. Table 1 gives an overview of the suggested interactions.

Instead of using the presented interactions we could use the Euler operators. These are more general than necessary for defining closed polyhedral volumes which do not contain holes or dangling faces and are far more difficult to interact with. Therefore we suggest to use a simpler set of basic operations which is fully sufficient for VOI specification.

The *split* interaction and its complimentary *merge* interaction are simple operations that split and merge faces and edges of the polyhedron.

The *move* interaction (Fig. 2) changes the position of a vertex. If vertex v is to be moved, the adjacent faces are split until all the faces surrounding v are triangles. Then the position of the vertex is updated and for the surrounding vertices an intersection test is performed. If the movement of the vertex tests results in self-intersections, the vertex position is set back and the move operation fails. For interaction we offer two types of controls. The first moves a vertex along a straight line using a slider widget. We estimate a normal at a vertex and the user can then move the vertex along the straight line defined by the normal and the position of the vertex. The initial manipulation direction is computed using a weighted average of the normals of the surrounding surfaces. Let v be the selected vertex, f_i, $1 \le i \le n$, the faces surrounding v with normal n_i and α_i, $1 \le i \le n$, the opening angles of the faces at the vertex. We use an angle-weighted average normal $n_v = \sum_{i=0}^{n} \left(\sum_{j=1}^{n} \alpha_j\right)^{-1} \cdot \alpha_i \cdot n_i$. If the user wants to move the vertex in another direction than the suggested one, she can change this direction by manipulating a trackball surrounding the vertex. The manipulation direction is then changed correspondingly. Another option to specify a new position for a vertex is to drag it within a plane. This is often more natural if the interaction is done using a mouse or trackball. The user selects the normal n for the plane to be used for dragging (e.g., a clipping plane or the viewing plane), then the vertex can be dragged on the plane given by the position of the vertex and n.

The *insert vertex* interaction and its inverse the *remove vertex* operation are crucial for changing the structure of the polyhedron. By creating and moving a new vertex the user can add local detail to the VOI. By removing a vertex the user can reduce the complexity of the VOI. The possibility to remove vertices in a way that avoids self-intersections is crucial for large scale manipulation of a

Table 1
The interactions change the shape and behavior of the polyhedron. Low-level interactions change a single element of the polyhedron and are in principle sufficient to specify any possible polyhedron. The high-level interactions are combinations of the low-level operations, and can be used to build arbitrarily complex commands matching the application needs. This way the user can perform large changes as well as subtle manipulations of the VOI with few interactions.

Interactions			
	Target	Arguments	Description
Low-level interactions			
Move	Vertex or edge or face	Vector	The position field is updated, non-planar faces are split
Insert vertex	Face	Vertex	New vertex is created, face is split into triangles
Delete vertex	Vertex		Ears are cut until three neighbors remain, then tetrahedron is deleted
Merge	Face a,b	Face	Two adjacent faces are merged into one
Split	Edge	Vertex	Edge is split at vertex
Split	Face	Vertex a,b	Face is split by connecting two vertices
High-level interactions			
Init		Polygon, view direction	A new polyhedron is built using the polygon as front and back face
Delete boundary	Polyhedron	Closed loop of edges	All vertices inside loop are deleted
Cut	Polyhedron	Line, viewing parameters	Plane equation is computed from viewing parameters, clip operation is called
Clip	Polyhedron	Plane equation	Intersected edges are split, intersected faces are split, vertices inside cut boundary are deleted

Please cite this article in press as: Fuchs R, et al. Non-convex polyhedral volume of interest selection. Comput Med Imaging Graph (2009), doi:10.1016/j.compmedimag.2009.07.002

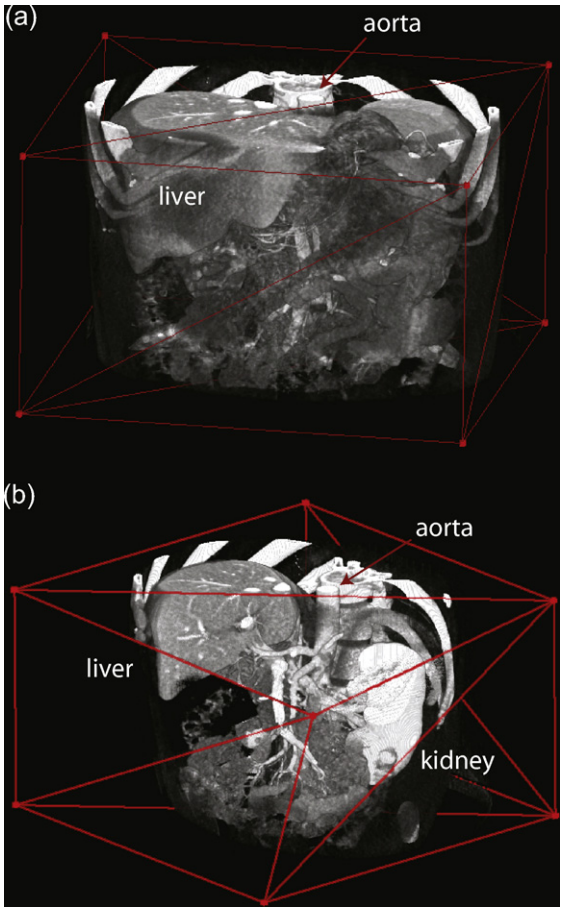

Fig. 2. We have moved one vertex to the interior of the volume. (a) The transfer function is set such that the liver has good contrast but the aorta is occluded. (b) The VOI selection allows to inspect the interior of the liver and we can clearly see the course of the aorta and the interior of the liver.

given polyhedron. It allows to build (high-level) interactions which remove large parts of the volume (Fig. 3).

Clipping: A convex polyhedron can be defined as the intersection of a set of half-spaces. Therefore a convex polyhedron can be specified by positioning clip planes. In this manner it is possible to remove parts of the volume quickly. Also, using the data structure that stores the polyhedron, the number of active clipping planes is no longer limited. This way it is possible to define a rough convex clipping object around the volume of interest. Non-convex details can be added using vertex operations. For specification of cuts we use the clipping plane or a view aligned plane that is defined by a two-dimensional line in view space and the view direction (see Fig. 4). When clipping the polyhedron all edges are tested for intersections with the clipping plane and the edges are split at the corresponding locations. Then the adjacent faces are split along the newly inserted vertices. Finally the vertices in the half-space which is to be clipped away are deleted.

3.3. Combination

The restrictions we have set for a single polyhedron can be compensated by combining multiple polyhedra. Each polyhedron can be associated with its specific degree of interest value $\mathrm{DOI}_P \in [0, 1]$. This way each polyhedron defines a degree of interest for sample point $x \in \mathbb{R}^3$:

$$\mathrm{DOI}(P, c) := \begin{cases} \mathrm{DOI}_P & x \in P \\ 0 & \text{otherwise} \end{cases}$$

A Boolean expression B structures how the specified polyhedra $P_{i,j}$ are combined. For intuitive interaction we restrict the user to work with expressions in disjunctive normal form:

$$B := \bigvee_{i=1\ldots n} \left(\bigwedge_{j=1\ldots m_i} (P_{i,j}) \right)$$

See, for example Fig. 1(c), where the selection is a combination of two polyhedra combined by the Boolean expression *A AND (NOT B)*.

The restriction to disjunctive normal form allows to specify the combination by placing the polyhedra in a simple tree structure: the tree has two levels and the polyhedra can be positioned inside the expression at the leaves interactively.

The DOI value for a sample point x specified by a clause in the form $C := P_0 \wedge \cdots \wedge P_m$ is evaluated as the minimum degree of interest value of the selections inside the clause:

$$\mathrm{DOI}(C, x) := \min(\mathrm{DOI}(P_0, x), \ldots, \mathrm{DOI}(P_m, x))$$

The combined membership value for the Boolean expression $B = \bigvee_{i=1\ldots n} C_i$ is simply the maximum over all clauses: $\mathrm{DOI}(B, x) := max(\mathrm{DOI}(C_i, x))$. The definition using minimum and maximum to compute the logic AND and OR operations is consistent with the standard extensions of Boolean values to fuzzy sets [22]. In Fig. 1 you can see an example where a selection of genus 1 (i.e. with one hole) is created by combining two polyhedra using the expression *A AND (NOT B)*.

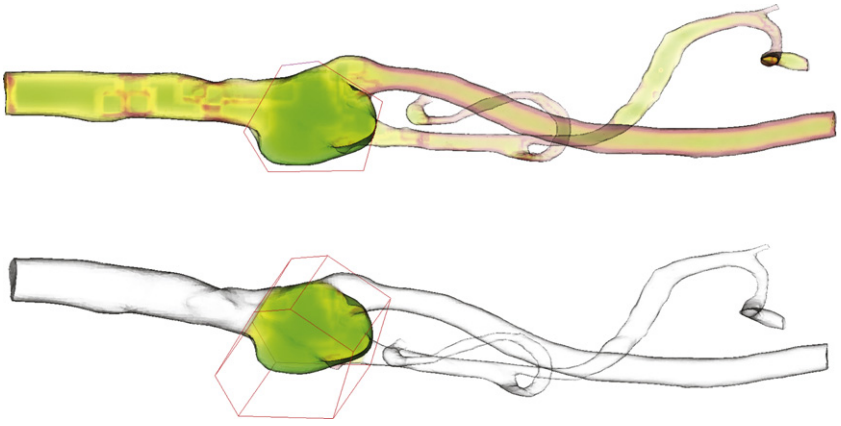

Fig. 3. The VOI is initialized by drawing a 2D polygon onto the image plane. The three-dimensional polygonal VOI is built by extruding the polygon in viewing direction.

Please cite this article in press as: Fuchs R, et al. Non-convex polyhedral volume of interest selection. Comput Med Imaging Graph (2009), doi:10.1016/j.compmedimag.2009.07.002

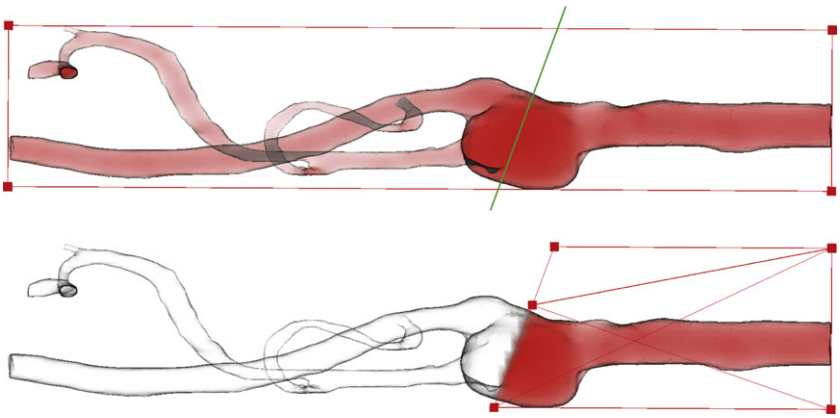

Fig. 4. The polyhedral boundary allows for simple use of clipping planes. The plane changes the shape of the polyhedron. This way it is simple to perform rough editing of the VOI.

4. Comparison to related techniques and an application example

In this section we relate the suggested approach to other VOI selection techniques and give an exemplary application. We have already mentioned in the introduction that volume selection techniques based on the data values in the volume are not always sufficient to define the VOI. Therefore the presented approach can be considered as complementary to techniques based on the data values such as (automated) segmentation [23], transfer function specification [24], automated VOI selection [25], volume painting [16], pattern recognition [11] or iso-surface based approaches [13].

Two standard approaches are covered by the presented approach:

1. Volume selection using an arbitrary number of clip planes.
2. Techniques based on combining polygonal selections from multiple slices [26]. This kind of selection can be done with the presented approach by adding vertices in one slice and moving them to the appropriate positions.

Constructive techniques have been used for volume data editing and selection [6,5,18]. When comparing the presented approach to CSG related techniques which combine primitives using Boolean operations we can see that the presented approach adds an intermediate step of editing before combination. Also the polygon based initialization tool allows the user to select non-convex regions from the beginning.

Two classes of techniques are not directly covered by the presented approach. The first class are volume sculpting techniques. Here the user can manipulate the selection by adding and removing voxels of the selection using metaphorical tools such as a saw, paint brush or chisel. This can be beneficial for editing small details of the selection in an intuitive way. Equivalently, in the presented approach small details can be edited by inserting a vertex locally and positioning it to include small features or remove local artifacts, since changing a vertex position affects the VOI locally according to the size of the surrounding faces.

The second class are free-form shape manipulation techniques, such as the work of Welch and Witkin [27] where a free-form shape design approach for triangulated surfaces allows to perform topological operations on a surface. This surface could be used as a bounding object of the VOI, but to our knowledge this kind of technique has not been applied to interactive VOI selection so far. Barr [28] discusses deformations as a tool for shape manipulation. These (and many subsequent works) are applicable for editing a selection based on a polyhedral boundary. The advantage of the presented method is that it is very predictable for the user while shape editing tools very often rely on surface smoothing and blend operations to create results. For these operations it is difficult to eliminate large portions of the shape which is a common task in VOI specification.

Segmentation is the method of choice if it comes to inspection of organs and other structures that have distinctive value distributions. If there is no background information on the VOI available, then current algorithms like model based systems are not applicable. Interactive VOI selection may actually help to apply segmentation. Indeed, it can be used as a preprocessing step which removes noise from the dataset. Vice versa for many automated region-selection methods (e.g., seed based region growing) it is still necessary to remove artifacts or wrongly selected parts outside the volume of interest. For example, methods based on diffusion have problems due to overflooding in fuzzy regions which can be dealt with interactively. In the remainder of this section we give an example how specification of non-convex regions can help to analyze scientific data and discuss why this is not possible using standard techniques. The main feature in the geometry of the discussed dataset is an aneurysm, i.e., an extension of the blood vessel caused by weakening of the vessel wall. Without intervention an aneurysm will expand until the arterial wall can no longer bear the tension. Aneurysms can also cause thrombosis, due to the clotting of blood inside the expanded aneurysm. There are three known factors contributing to the progressive expansion of the aneurysm. Firstly, the Laplace law states that wall tension is proportional to the pressure times the radius of the vessel. With increasing size of the aneurysm, wall tension increases as well, contributing to further increase in diameter. The second factor is an additional weakening of the wall due to ischemia (insufficient blood supply).

A third factor of current interest is the influence of turbulence. In Fig. 5 we are interested in the origin of flow separation inside a simulation of an aneurysm. In (a) we can see that flow separation occurs close to the two outlets at the wall of the artery. An enlarged view focuses on this region with vorticity mapped to color. The region where the streamlines first separate lies in the occluded interior region of the aneurysm. In (b) we have moved one face of a convex selection similar to the one in Fig. 3 into the interior. Slowly moving the face it is possible to get an understanding of the situation inside the flow and crop the occluding streamlines intuitively and quickly. In (c) we can see the region inside the aneurysm where the separation originally occurs.

To create a meaningful visualization of this situation using a convex selection tool (e.g., clipping planes) is difficult if not impossible. The necessary changes of geometry also had to crop precisely the occluding streamlines, which may be possible but a lot more difficult with shape deformation tools [28] since here the resulting VOI

Please cite this article in press as: Fuchs R, et al. Non-convex polyhedral volume of interest selection. Comput Med Imaging Graph (2009), doi:10.1016/j.compmedimag.2009.07.002

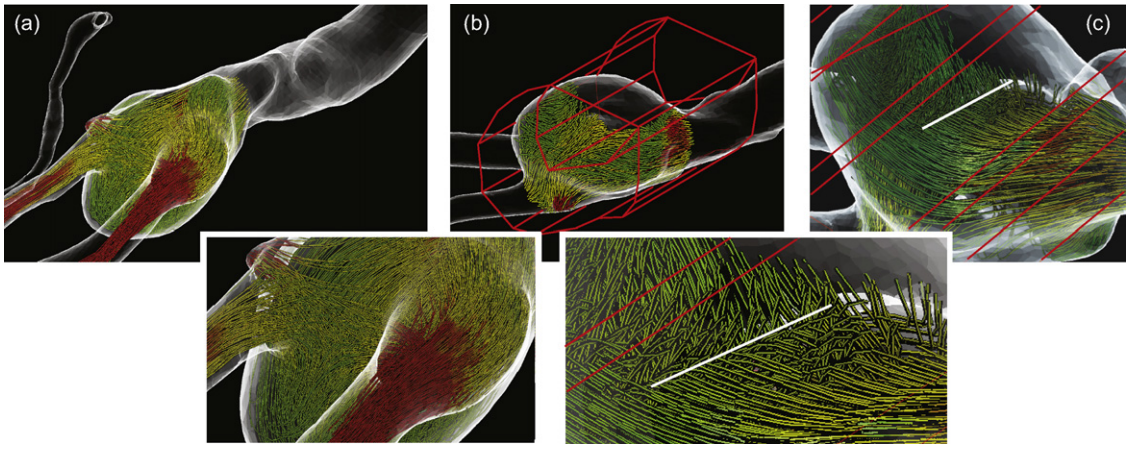

Fig. 5. An example of a non-convex selection in scientific data. (a) We can see a separation of flow at the boundary of the aneurysm. (b) A non-convex selection allows to locate the flow separation in the interior of the aneurysm (highlighted white). (c) The occluding streamlines are cropped. We have highlighted the polyhedron for print.

is a lot more difficult to predict. Also we had to remove large parts of the volume, which is difficult to steer precisely using volume sculpting approaches. Based on the course of the flow separation inside the volume we can hypothesize that the separation inside the aneurysm is the cause for the deceleration of blood flow in the present case. This can be an explanation for the occurrence of clotting. Following the simulation results, clotting could be made possible by the turbulent behavior decelerating the flow at the far end of the aneurysm.

5. Algorithmic details

In this section we give a detailed description of the algorithms behind the editing operations of the polyhedron available to the user. The core difference to standard mesh manipulation are the additional properties we have to maintain for the polyhedron. That is, it has to remain free of self-intersections and watertight all the time.

First we recall some definitions:

- A set of points S is called convex if for $a, b \in S$ the line segment $\overline{ab} := \{\lambda a + (1-\lambda)b | 0 \leq \lambda \leq 1\}$ is a subset of S.
- A polygon is a finite set of line segments where each end point is shared by exactly two line segments and no two line segments intersect in their interior. We call these line segments edges and the end points vertices.
- A polyhedron is a finite set P of planar polygons (i.e., its edges lie in a plane) such that (a) each edge of a polygon is shared by exactly one other polygon, (b) no two polygons intersect in their interior, no subset of P has properties (a) and (b).
- As usual for a subset $X \subseteq \mathbb{R}^3$ we denote by $Conv(X)$ the convex hull of X.

All operations described in this paper are combinations of the low-level operations. Of these moving vertices, inserting vertices to faces and edges and merging/splitting of faces are straightforward to implement, and we work on the assumption that these operations are available.

The *intersection* operator checks whether a triangular face intersects another face in its interior. This operator is necessary to guarantee the intersection restriction for polyhedra, also discussed in the previous section.

The *remove tetrahedron* operator implements the simplest case of deleting a vertex, i.e. when the vertex has only three neighbors. The only obstruction to the deletion is the existence of another face in the interior of the tetrahedron. This can be checked using the intersection operator and then the removal can be done in constant time.

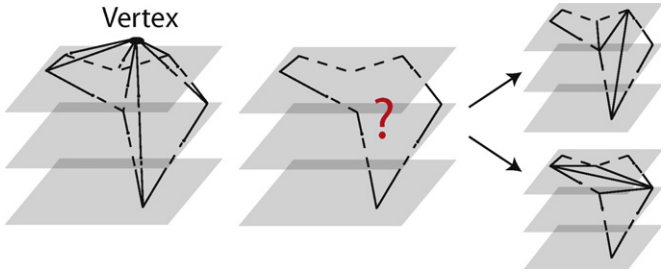

Fig. 6. After deleting a vertex it is necessary to find a triangulation for the remaining hole.

Delete Vertex is one of the most important operations for changing the geometry of the polyhedron. It is also one of the most subtle operations since the deletion leaves a polygonal hole that has to be re-triangulated. Re-triangulating three-dimensional polygons can be a difficult problem [29]. Consider Fig. 6—it illustrates the situation that arises when deleting a vertex. When deleting a vertex v from a polyhedron the edges not incident to v of the faces containing v form a three-dimensional polygon. We call this polygon the boundary polygon of v. We know that the question whether or not a triangulation of a three-dimensional polygon exists is NP-complete in general.

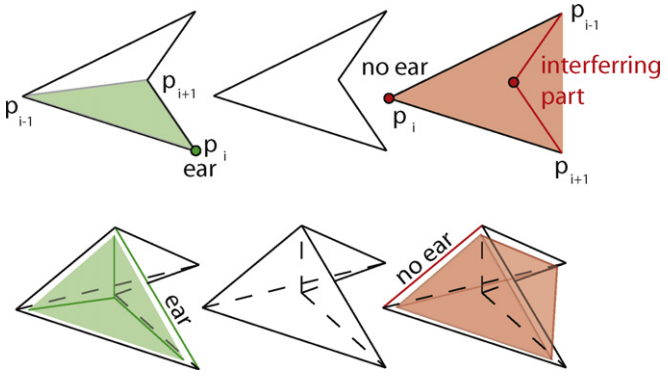

Fig. 7. Definition of an ear: (top) In 2D a vertex p_i is defined as an ear if the triangle defined by the two adjacent vertices p_{i-1} and p_{i+1} does not contain vertices of the polygon. (bottom) The green edge (left) is an ear, since the tetrahedron defined by its adjacent triangles is empty. The red edge (right) is not an ear, since the tetrahedron contains edges and a vertex of the polyhedron (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of the article.).

Please cite this article in press as: Fuchs R, et al. Non-convex polyhedral volume of interest selection. Comput Med Imaging Graph (2009), doi:10.1016/j.compmedimag.2009.07.002

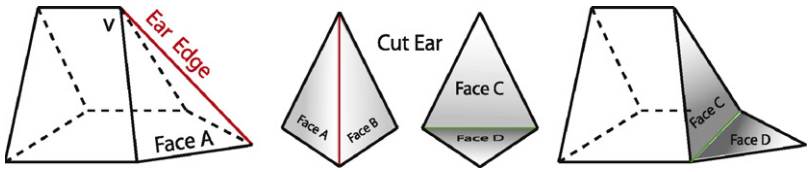

Fig. 8. An ear is cut. As long as more than three surrounding edges remain, the triangulation of the boundary can be done in a progressive manner, cutting away one ear after another. This can be done by merging the face adjacent to the ear and splitting the resulting face afterwards as shown in the figure.

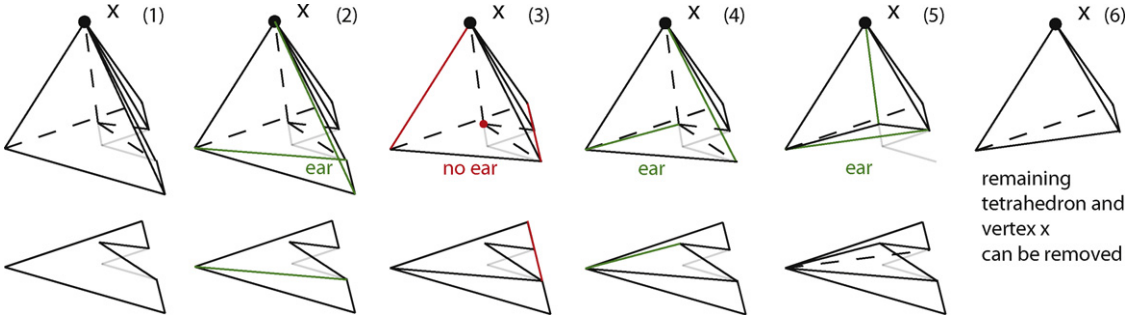

Fig. 9. Illustration of the ear-cutting approach to triangulate three-dimensional polygons. In each step one edge starting from x is tested for the ear property. If it is an ear it can be clipped away and the number of edges leaving from x is reduced by one. When only three edges are left, the remaining tetrahedron can be removed keeping the polyhedron watertight all the time.

Barequet et al. [29] show:

Theorem 1. *A three-dimensional polygon P is triangulable if there exists a spherical projection π on the plane such that $\pi(P)$ has no self-intersections.*

The algorithm that follows from the proof of Theorem 1 has complexity $O(n^6)$, where n is the number of vertices in P. Subsequently we will use Theorem 1 for theoretical reasoning, since the triangulation derived from this algorithm may create self-intersecting polyhedra. We discuss an alternative algorithm at the end of this section. Nevertheless, we deduce from Theorem 1 that in our situation the boundary polygons are always triangulable:

Lemma 1. *The boundary polygon of a vertex v is triangulable.*

Proof. Let e be the longest edge connected to v in Euclidean metric. We know that perspective projection of the polygon into the sphere with center v and radius $\geq |e|$ cannot lead to self-intersections, since otherwise the faces around v would have intersected before. Then we can select a point p on the sphere and project into the tangent plane to get a simple planar projection of the hole polygon. Thus by Theorem 1 the result follows. For later use we denote this projection as $\pi(P)$. □

In the following we discuss the ear-clipping algorithm for triangulating three-dimensional boundary polygons.

Cut ear is an operation that will be called when performing the delete vertex operation. It is basically an edge flip with a preceding ear-test. First we have to define what is an ear of a three-dimensional polygon: say, v is the vertex that is to be deleted, P its boundary polygon with vertices $p_1, \ldots, p_n$ in circular order. We call $e := \overline{vp_i}$ an ear if the tetrahedron $Conv(v, p_i, p_{i-1}, p_{i+1})$ does not contain other vertices of the polyhedron.

This definition is illustrated in Fig. 7. The upper row shows the two dimensional case. The green vertex p_i is an ear since the green triangle is empty. The red vertex is not an ear since another vertex is contained in the red triangle. The lower row of Fig. 7 describes the analogous situation in three dimensions. Here a vertex can be an ear if the tetrahedron spanned by the adjacent faces is empty. A single ear cut operation is illustrated in Fig. 8. The edge highlighted red is an ear. After flipping the edge the two faces A and B are replaced by new faces C and D. The ear has been replaced by the green edge such that the number of edges starting a vertex v is reduced by one.

An graphical example of the ear-cutting approach is given in Fig. 9. In Fig. 9 (1) we can see a vertex x and in the lower row the three-dimensional hole that would remain when deleting x and the surrounding faces. In steps (2), (3) and (5) we can see how the hole is triangulated while ears are clipped away from the polyhedron. In Fig. 9 (3) a vertex is selected that is not an ear and is therefore not cut away. The figure also illustrates the beneficial property of the ear-cutting approach to leave the polyhedron intact in each step. In Fig. 10 we state the ear-cutting algorithm for deleting vertices. The ear-cutting algorithm runs in $O(n * m)$ where n is the number of vertices in the polyhedron and m the number of vertices in the boundary polygon.

Now it remains to be shown that the ear-cutting algorithm will find a triangulation for a given polyhedron after deleting a vertex:

Lemma 2. *Let P be the boundary polygon of a vertex v. Then the ear-cutting algorithm constructs a triangulation.*

Proof. We show that the three-dimensional boundary polygon has ears. Then it will follow by induction that it is triangulable (i.e., if we can cut away an ear for each size of the polygon, then we will arrive at a triangulation after $n-2$ cutting operations). Given the projection $\pi(P)$ from the proof of Lemma 1 we know that this two-dimensional polygon has an ear. Therefore no other vertex can have

```
INPUT: vertex v, polyhedron
OUTPUT: new polyhedron

while(hasEar(v))
{
    cutEar(v);
}
removeTetrahedron(v);
```

Fig. 10. The ear-cutting algorithm for boundary polygons. The removeTetrahedron() method removes a vertex from the polyhedron that has three adjacent vertices. See Fig. 9 for an illustration.

Please cite this article in press as: Fuchs R, et al. Non-convex polyhedral volume of interest selection. Comput Med Imaging Graph (2009), doi:10.1016/j.compmedimag.2009.07.002

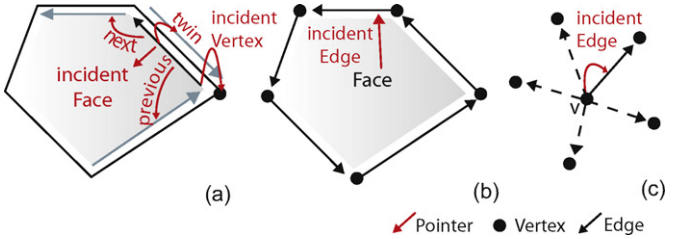

Fig. 11. The half-edge data structure: (a) edges, (b) faces and (c) vertices.

been inside the volume through which the ear was projected. This means that the tetrahedron spanned by the two adjacent faces must have been empty. Therefore the three-dimensional polygon had an ear at the same position. Note, however that we can evaluate the three-dimensional ear condition *without knowing the projection* π thus cutting away a single ear has complexity $O(m)$, where m is the number of vertices in the polyhedron. □

6. Implementation

The half-edge data structure [30] is used for polyhedron representation. The polyhedral VOI is represented by objects representing vertices, directed edges and faces of the polyhedron. This allows fast manipulations of the geometry. An edge consists of pointers to the vertex where the edge starts, the next edge around a face, the previous edge, the twin edge pointing in the opposite direction and a pointer to the incident face. The face data structure contains a pointer to one of the edges surrounding the face, and a vertex contains a pointer to one edge starting a that vertex. This is illustrated in Fig. 11. (a) Each polygonal face is surrounded in counterclockwise order by a doubly linked list of directed edges connected by pointers to the next and previous edge. Each edge also stores a pointer to its starting vertex, its twin along the adjacent face and a pointer to the face it belongs to. (b) Faces simply contain a pointer to one of the surrounding edges. (c) Vertices contain a pointer to one of the outward edges.

The editing operations are based on a few basic operations. We illustrated the dependencies of these operations in Fig. 12. The implementation of the low-level interactions is based simple manipulations the connectivity information in the half-edge representation, in Fig. 13 we give an implementation of the split operation for faces. Other operations can be implement similarly based on the split operator. The user specified polyhedra consists of few triangles from a computational perspective. Their inclusion into the volume ray-casting algorithm is straightforward: we compute ray starting positions with the polyhedron and start sampling at the nearest intersection point.

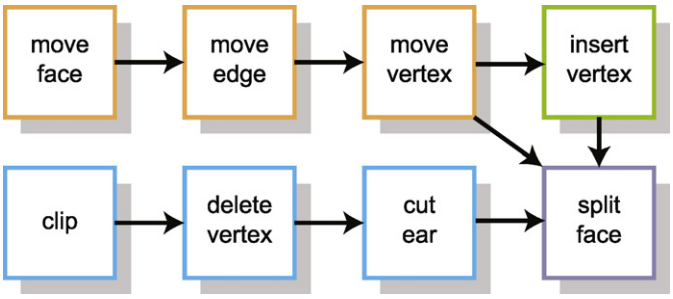

Fig. 12. Dependencies between operations. (orange) The move operations are based on the move vertex operation. (blue) The deletion operations are based on the cut ear operation. (green) The vertex insert operation splits the face the vertex is inserted to (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of the article.).

```
INPUT: face f to split, two vertices of f
OUTPUT: a new face

Edge *Face::split(Vertex *a, Vertex *b)
{
    Edge *newA, *newB = new Edge(a);
    Face *newFace = new Face(newA);
    Edge *fromA = a->getEdgeAtFace(this);
    Edge *fromB = b->getEdgeAtFace(this);
    Edge *toA = fromA->getPrev();
    Edge *toB = fromB->getPrev();

    //set edge
    newA->set(newB, newFace, fromB, toA);
    newB->set(newA, this, fromA, toB);
    this->setOuter(neuB);
    toA->setNext(newA);
    toB->setNext(newB);
    fromA->setPrev(newB);
    fromB->setPrev(newA);

    //set incident face A
    Edge *curEdge = newA;
    do {
        curEdge->setFace(newFace);
        curEdge=curEdge->getNext();
    }while(curEdge!=newA);

    //set incident face B
    curEdge = newB;
    do {
        curEdge->setFace(this);
        curEdge = curEdge->getNext();
    }while(curEdge!=newB);

    return newB;
}
```

Fig. 13. The split operation. The function 'getEdgeAtFace' returns the edge starting at vertex a incident to the face f. The 'set' function of an edge has the arguments (Edge *twin, Face *incident, Edge *next, Edge *prev). The 'setOuter' function sets the incident edge of a face.

7. Conclusion and future work

A polygonal mesh poses little restrictions on the shape of the VOI and is not limited to convex geometries. The possibility to define complex clipping geometries is beneficial especially when very intricate structures, like tumors in medical imaging or streamlines of a flow-volume in scientific visualization need inspection. Today, a common approach is the use of multiple clipping planes, i.e. a convex region. This is not a natural restriction.

We have presented an easy to understand interactive VOI selection method that needs only moderate computational resources. It fits well into the continuum between very restrictive clipping methods such as predefined clipping objects or clipping planes and general CAD techniques. Our approach produces a VOI that matches the geometry of the underlying structure more closely compared to clipping planes, but less closely than other approaches such as region-growing or drawing manual outlines on each slice. So it is middle-of-the-road in terms of "quality" of the segmentation. In terms of interaction time, it can requires quite a bit of tweaking, but less time than to produce a high-quality segmentation. In Figs. 2 and 5 we show non-convex selections created by the movement of a single vertex or edge respectively, enhancing the amount visible information.

The method includes a vertex removal operator which makes the approach closed in the sense that any conceivable polyhedron can be constructed based on the presented operations. This is not true

Please cite this article in press as: Fuchs R, et al. Non-convex polyhedral volume of interest selection. Comput Med Imaging Graph (2009), doi:10.1016/j.compmedimag.2009.07.002

for previous mesh manipulation procedures and is possible due to the presented ear-cutting algorithm. The algorithm is proven to be correct and has good worst-case performance. The compatibility of the suggested technique with alternative techniques such as region growing as another feature of the presented approach. Since we are not limited to using a voxel based representation as in sculpting or convex geometries as in previous polyhedra based approaches we can directly edit results from other (semi-) automatic VOI selection algorithms.

In future work we would like to explore how and when specialized mesh editing methods can increase the effectiveness of our approach. When defining volumes of interest a user with medical background may like to have field specific editing metaphors available: drilling into the polyhedron or stretching it in a physics-related manner for example. Physicists might be interested in changing the behavior of the polyhedron according to the data values present in the data set (e.g., moving vertices along streamlines). This can be considered as another benefit of the presented technique, namely that it is general and can be tuned to fit to the specific analysis task at hand by defining new high-level interactions.

Acknowledgements

The authors wish to thank Dr. Michael Prümmer for his support. The aneurysm data set is courtesy of the VRVIS research center, Vienna.

References

[1] Bro-Nielsen M, Larsen P, Kreiborg S. Virtual teeth: a 3D method for editing and visualizing small structures in CT scans. In: Proceedings of computer assisted radiology (CAR'96). 1996. p. 921–4.
[2] Pflesser B, Petersik A, Tiede U, Hohne K, Leuwer R. Volume cutting for virtual petrous bone surgery. Computer Aided Surgery 2002;7(June (2)):74–83.
[3] Sherbondy AJ, Holmlund D, Rubin GD, Schraedley PK, Winograd K, Napel S. Alternative input devices for efficient navigation of large CT angiography data sets. Radiology 2005;234(February (2)):391–8.
[4] Weiskopf D, Engel K, Ertl T. Volume clipping via per-fragment operations in texture-based volume visualization. In: Proceedings of IEEE visualization. IEEE Computer Society; 2002. p. 93–100.
[5] Weiskopf D, Engel K, Ertl T. Interactive clipping techniques for texture-based volume visualization and volume shading. IEEE Transactions on Visualization and Computer Graphics 2003;9(July (3)):298–312.
[6] Museth K, Lombeyda S. Tetsplat real-time rendering and volume clipping of large unstructured tetrahedral meshes. In: Proceedings IEEE visualization 2004. 2004. p. 433–40.
[7] Chen M, Tucker JV. Constructive volume geometry. Computer Graphics Forum 2000;19(4):281–93.
[8] Pham DL, Xu C, Prince JL. Current methods in medical image segmentation. Annual Review of Biomedical Engineering 2000;2(August):315–37.
[9] Frantz S, Rohr K, Stiehl HS. Multi-step procedures for the localization of 2D and 3d point landmarks and automatic ROI size selection. In: Proceedings of the 5th European Conference on computer vision EVVC, vol. 1. 1998. p. 687–703.
[10] Ashton EA, Parker KJ, Berg MJ, Chen CW. A novel volumetric feature extraction technique, with applications to MR images. In: ICIP '95: Proceedings of the 1995 International Conference on image processing, vol. 3. 1995. p. 564–7.
[11] Barber DC. Automatic generation of regions of interest for radionuclide renograms. In: Proceedings of medical image understanding and analysis. 2003.
[12] Wong KChun-Ho, Siu Yu-Hang, Heng Pheng-Ann, Sun Hanqiu. Interactive volume cutting. In: Proceedings of graphics interface. 1998. p. 99–106.
[13] Sibbing D, Kobbelt L. Fast interactive region of interest selection for volume visualization. In: Proceedings of Bildverarbeitung für die Medizin. 2007. p. 338–42.
[14] Vincent L, Soille P. Watersheds in digital spaces: an efficient algorithm based on immersion simulations. IEEE Transactions on Pattern Analysis and Machine Intelligence 1991;13(June (6)):583–98.
[15] Armstrong CJ, Price BL, Barrett WA. Interactive segmentation of image volumes with live surface. Computer Graphics 2007;31(April (2)):212–29.
[16] Bruckner S, Grimm S, Kanitsar A, Meister E, Gröller. Illustrative context-preserving volume rendering. In: Proceedings of the 7th Joint IEEE VGTC—EUROGRAPHICS Symposium on visualization (VisSym 2005). 2005. p. 69–76.
[17] Wang SW, Kaufman AE. Volume sculpting. In: SI3D '95: Proceedings of the 1995 symposium on interactive 3D graphics. 1995. p. 151–6.
[18] Nakao M, Watanabe T, Kuroda T, Yoshihara H. Interactive 3D region extraction of volume data using deformable boundary object. Studies in Health Technology and Informatics 2005;111:349–52.
[19] Darrah M, Kime A, van Scoy F. A tool for editing 3D objects: implemented using a haptics device. In: Proceedings of PHANTOM user group workshop. 2002.
[20] Dietrich CA, Nedel LP, Olabarriaga SD, Comba JLD, Zanchet DJ, da Silva AMM, et al. Real-time interactive visualization and manipulation of the volumetric data using GPU-based methods. In: SPIE medical imaging, vol. 5367. 2004. p. 181–92.
[21] Hornegger J, Welker V. Rechnergestütztes Verfahren für einen Teil eines Volumens. International publication 1. Patent-No: WO 2004/027641; April 2004.
[22] Klement E, Mesiar R, Pap E. Triangular norms, volume 8 of trends in logic. Kluwer Academic Publishers; 2000.
[23] Setarehdan K, Singh S, Suri J. Advanced algorithmic approaches to medical image segmentation. UK: Springer–Verlag; 2002.
[24] Kniss J, Kindlmann G, Hansen C. Interactive volume rendering using multi-dimensional transfer functions and direct manipulation widgets. In: Proceedings IEEE visualization 2001. 2001. p. 255–62.
[25] Boykov YY, Jolly MP. Interactive graph cuts for optimal boundary & region segmentation of objects in n–d images. In: Proceedings of ICCV, vol. 1. 2001. p. 105–12.
[26] Barequet G, Sharir M. Piecewise-linear interpolation between polygonal slices. Computer Vision and Image Understanding 1996;63(March (2)):251–72.
[27] Welch W, Witkin A. Free-form shape design using triangulated surfaces. In: SIGGRAPH '94: Proceedings of the 21st annual conference on computer graphics and interactive techniques. 1994. p. 247–56.
[28] Barr Alan H. Global and local deformations of solid primitives. SIGGRAPH Computer Graphics 1984;18(July (3)):21–30.
[29] Barequet G, Dickerson M, Eppstein D. On triangulating three-dimensional polygons. Computational Geometry 1998;10(3):155–70.
[30] de Berg M, van Kreveld M, Overmars M, Schwarzkopf O. Computational geometry: algorithms and applications, 2nd ed. Springer; 2000.

Raphael Fuchs graduated in computer science (2006) at Philipps-University Marburg (Germany) under the supervision of Prof. Volkmar Welker and received his Ph.D. degree in computer science (2008) from the Technical University Vienna (Austria). He is now with the Computer Graphics Lab (CGL) at the ETH Zurich (Switzerland). His field of research is scientific visualization.

Volkmar Welker graduated in mathematics (1988) and received his Ph.D. degree in mathematics (1990) from Friedrich-Alexander-University Erlangen-Nuremberg (Germany). After spending time as a researcher in Erlangen he was a visiting scholar at Massachusetts Institute of Technology (MIT) in the academic year 1992/93. From 1993 until 1998 he held positions in the department of Mathematics at University Essen (Germany). In 1995 he was awarded the Heinz-Meier-Leibnitz price of the German Science Council (DFG) and in 1996 he complete his habilitation. After a year as a Heisenberg scholar at TU-Berlin (Germany) in 1998/99 he moved to Philipps-University Marburg (Germany) as an associate professor of Mathematics. He was promoted to full professor in 2002. His field of research is algebraic and geometric combinatorics and its applications. He has worked on application in medical imaging and electrical engineering..

Joachim Hornegger graduated in theoretical computer science/mathematics (1992) and received his Ph.D. degree in Applied computer science (1996) at the Friedrich-Alexander-University (FAU) of Erlangen-Nuremberg. His Ph.D. thesis was on statistical learning, recognition and pose estimation of 3D objects. Joachim was a visiting scholar and lecturer at Stanford University (CA, USA) in the academic year 1997/98, and a visiting professor at Stanford's Radiological Science Lab (RSL) in winter 2007/2008. In 1998 he joined Siemens Medical Solutions Inc. where he was working on 3D angiography. In parallel to his responsibilities in industry he was a lecturer at the Universities of Erlangen (1998-1999), Eichstaett-Ingolstadt (2000), and Mannheim (2000-2003). In 2003 Joachim became professor of Medical Imaging Processing at the FAU and since 2005 he is a chaired professor heading the Chair of Pattern Recognition. His main research topics are currently pattern recognition methods in medicine and sports.

Please cite this article in press as: Fuchs R, et al. Non-convex polyhedral volume of interest selection. Comput Med Imaging Graph (2009), doi:10.1016/j.compmedimag.2009.07.002