3D Reconstruction and Visualization of Liver and Vascular Networks from CT data using VTK and IGSTK

João Fradinho Oliveira
C3i/Instituto Politécnico de Portalegre
Portalegre, Portugal
jfoliveira@estgp.pt

José Luis Moyano-Cuevas José Blas Pagador
Bioengineering and Health Technology Unit
Jesús Usón Minimally Invasive Surgery Center
Cáceres, Spain
{jmoyano, jbpagador}@cmmijesususon.com

Hugo Capote
Hospital Dr. José Maria Grande
Portalegre, Portugal
hugocapote@gmail.com

Francisco Miguel Sánchez-Margallo
Jesús Usón Minimally Invasive Surgery Center
Cáceres, Spain
msanchez@cmmijesususon.com

Abstract

Spatial reasoning of vascular structures in organs such as the liver is an imperative task performed pre-operatively in resection planning when minimising risks of bleeding in a procedure and intra-operatively during surgery. Accurate automatic 3D reconstruction of surfaces from computerized tomography (CT) contours is complex or impossible without user intervention. Often the gap between scan slices is large enough to make contour correspondence between adjacent slices hard to establish and branching difficult to determine. Freely available open source libraries such as the image guided surgery toolkit and the visualization toolkit (IGSTK and VTK respectively) provide building blocks that enable one to speed up the development time whilst allowing one to focus on new algorithms that might help the user. In this paper we present a new automatic solution for visualization/spatial reasoning of vascular networks within the liver that uses two separate 3D reconstruction approaches respectively. In order to make the system automatic, instead of creating contour correspondences where often crucial data is missing between slices we create a layered approach, where the surface of the liver is represented as one or more layered closed surfaces, and vascular networks where correspondence is more complex are represented as stacks of extruded individual contour blocks. Since the geometric primitive used in either reconstruction is the triangle, other algorithms such as collision detection in resection planning can be used.

Keywords

3D reconstruction, IGSTK, VTK, computerized tomography

1. INTRODUCTION

Liver imaging has multiple applications such as measurements of liver volume [Sánchez-Margallo11]; diagnosis and quantification of tumours and other diseases [Goryawala12] ; surgical planning prior to hepatic resection or surgical navigation systems [Maeda09]. All these applications need to previously solve the problem of the segmentation of the liver and the problems associated with reconstruction from CT contour polygons broadly set as contour correspondence, branching and tiling [Meyers94].

Segmentation

Segmentation of the liver is a complex task in the CT image because of the noise, the variations of grey level inside the liver, the morphological alterations and similarity of grey levels with other neighbouring structures. There are different types of segmentation depending on the user participation in the process: manual, semiautomatic and automatic. In manual segmentation, the user draws the contour of the anatomical structure in all slices of the study. These contours must be performed by an experienced radiologist because sometimes the limits between the liver and neighboring structures is complicated. Semiautomatic and automatic segmentation are desirable since the contour annotation time is greatly reduced in a system that enables editing of any error in automatic results. The interest in the segmentation methods is reflected on the Segmentation of the Liver Competition 2007 (SLIVER07 [Sliver14]). This competition started as part of the workshop **3D Segmentation in the Clinic: A Grand Challenge (Miccai 2007),** and aims to compare different algorithms to segment the liver from clinical 3D

computed tomography (CT) scans [Heimann09] , [Sliver14].

Contour correspondences and branching

A polygon contour on one slice can represent a section of an organ that is physically connected to another section/polygon contour on a different slice. The contour could be the bottom or top part of an organ which is normally implicit if there is no branching. Vascular networks have considerable branching, thus the problem of finding contour correspondence is harder.

Tiling

How vertices from one contour connect to vertices of a contour in a different slice to form triangles/tiles.

The following properties are desirable in a reconstruction system:

- the representation spatially represents the correct geometric level of detail for the task.
- fast to render.
- portability, the models/representation can be used in other systems.
- minimises the required user intervention in establishing contour correspondence, branching, correcting polygon tiling.
- resilient (errors do not compromise the whole model).

In this paper we present two different reconstruction solutions. A surface based reconstruction method for the reconstruction of the liver and simple genus objects of interest, and a block like structure for the reconstruction of vascular structures. For the surface reconstruction we take advantage that there is mostly an implicit 1 to 1 correspondence between all contours of the liver, thus not requiring user intervention establishing the links, for the vascular structures where branching is predominant we stack vertically individual extruded contours of each slice thus not requiring user intervention for establishing correspondence or tiling corrections. Three tools were developed in the scope of this work: a contour annotation tool (a), an intra-operative visualization/Navigation tool (b) and a simple resection planning tool (c). Re-using VTK (a, c) and IGSTK (b) functionality for different components in these tools was desirable for future projects but also presented some less well documented challenges that are also presented in this paper.

Our contour annotation tool uses VTK functionality to load, display DICOM data and specific widgets for spline editing. Our annotation tool exports both kinds of reconstructions into IGSTK's mesh and ply format. We use IGSTK functionality in a separate Navigation tool to amongst other tasks: load and visualize patient DICOM data, control the rendering opacity of loaded objects, and communicate with surgical tool position trackers in an intra-operative or training setting. Since the geometrical primitive of either exported reconstructions is the triangle, we also show an example of collision detection in the Resection planning tool that also uses VTK functionality.

In section 2 we review related work on 3D reconstruction from polygon contours, in section 3 we present our system, in section 4 we present results and discuss future work and conclusions in section 5.

2. RELATED WORK

Several 3D reconstruction methods have been published. We focus on methods that assume that the manual or automatic segmentation of polygon contours on the CT data has already been made.

Amenta et al. [Amenta00] use a 3D Voronoi diagram to create a surface that has the same topology of the underlying point sample surface. Hoppe et al. create surfaces that are at a zero distance from the tangent planes of the surface [Hoppe92]. Whilst these reconstruction methods work with unorganized points, points derived from CT polygon contours present interesting modeling problems in that not enough slices exist to always capture where an organ starts or where exactly an object physically branches. Meyers decomposes the problems of reconstruction from CT polygon contours into three problems [Meyers94]: a) finding the correspondence between contours of different slices; b) the triangle tiling problem, the stitching of points between different slices and c) contour branching problem. Christiansen et. al measure the area of overlap of any polygon pair from adjacent slices and establishes a correspondence with a threshold [Christiansen78], Meyers also presents a method for automatically finding correspondences between contours, however the level of user correction required is not clear.

Regarding the problem of tiling, early work by [Keppel75] uses graph theory to find tiling arrangements that maximize the volume of the surfaces, Fuchs searches a directed graph, where the edge weights are surface area, finding the shortest path minimizes surface area [Fuchs77]. Christiansen et. al presents an elegant method for tiling that minimises the diagonal length of the connecting triangles with a simple heuristic for branching, branch pairs are grouped into one polygon, and a new vertex half-way the closest points between the branches is inserted [Christiansen78].

A comprehensive review of branching algorithms can be found in [Bajaj95]. Itk-snap [Itk14] is a tool that reads medical imaging files (DICOM) and allows one to annotate polygon contours and create a 3D model. Organs and vascular networks are built in the same way, with a vertical stack of individual extruded polygon contours (blocks). Simply extruding polygon contours eliminates the need to establish polygon correspondence, which with vascular networks could be quite time consuming. It is not clear however whether the organs and vascular

networks can be visualized at the same time with the added visual clutter of geometry internal to these blocks, it is also not clear whether the top and bottom of each block is triangulated with a centroid strategy which in some polygons might not be internal to a polygon, or if a more elaborate triangulation was used. We adopt a similar stack strategy for reconstructing vascular structures using ear clipping triangulation and reconstruct the surface of the liver rather than use blocks to avoid internal visual clutter of internal block structures that would prevent viewing the internal vascular objects.

IGSTK [Igstk14] builds on VTK, and provides useful functionality for creating intra-operative applications such as classes that communicate with known ultra sound devices (e.g. igstk::UltrasoundProbeObjectRepresentation) and has a class that represents vascular structures (igstk::VascularNetworkObjectRepresentation) as a network of tubes, where vtkPolyLines define the centerline of tubes rather than building them from polygon contours. Our system uses VTK functionality directly for DICOM file reading and viewing, and uses a useful vtkwidget for manipulating the polygon contours. Our system outputs geometry that can later be read in IGSTK. Other, volume based methods such as the Marching cubes exist, but are beyond the scope of this paper. Such methods create iso-surfaces in various cells, whereas surface based approaches enable the definition of iso-contours/polygon contour marking in each slice and then create the iso-surface from these contours [Bajaj95].

3. SYSTEM ARCHITECTURE

Our contour annotation tool, that exports a triangulated surface of an organ (a) or block models of vascular networks (b) is composed of four components:

- contour creation/editing (a, b)
- contour correspondence and branching (only optional in a)
- tiling algorithm (a, b)
- contour triangulation (a, b)

In other words the reconstruction of surfaces that do not have branching requires first contour annotation by the user (section 3.1) , the algorithm then processes each slice and checks for polygons, if there is one, it checks to see if there are polygons present in the next or previous slice. If no polygons are present in adjacent slices the algorithm triangulates the hole from above or below (section 3.4) depending on whether it is the top or bottom slice of the organ. After the contour triangulation checks, the polygon is tiled with the polygon in the next slice using the minimal spanning diagonal criteria (section 3.3).

For the reconstruction of vascular structures, after polygon contour annotation (section 3.1), each slice is checked for polygons, if there is one, this contour polygon is simply copy and pasted automatically to the next slice and tiled together (section 3.3). Each contour polygon is then always triangulated from below (section 3.4) and its extruded counter part is triangulated from above (section 3.4). Branching of surfaces is supported but requires user intervention to establish polygon correspodences (section 3.2).

3.1 Contour annotation

For both vascular models and organ models one first needs to manually define the contour polygons on each slice, or read a previous automatic or manual segmentation created by a different system. Our file layout supports multiple contours per slice and can record any type of branching correspondences. As mentioned previously our contour annotation tool uses VTK to load and view DICOM CT scans (vtkDICOMImageReader and vtkImageViewer2 respectively). VTK provides a slider bar (vtkSliderWidget) that enables one to use the mouse to change the slice being viewed. We use a spline based widget (vtkContourWidget) to annotate the polygon contours (Figure 1). In our system, each slice can store multiple contour widgets, and the control points of each can be saved finished or unfinished to file for future editing operations.

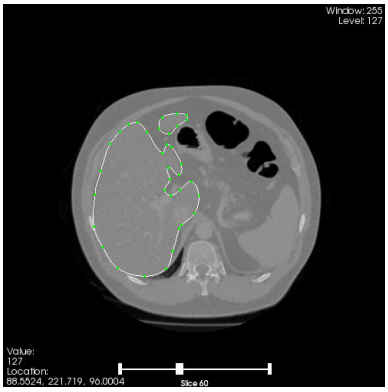

Figure 1. Polygon contour annotation.

To speed up annotation, an option to copy and paste all the contours from another slice was added. VTK allows for different interaction styles to be defined to interact with different components, left click drag over the DICOM image changes the brightness of the image, this is particularly useful for the radiologist when segmenting the image/defining polygon contours. Right click drag over the image zooms the image. A left click inside the image creates a polygon control point, right click inside the image displays the VTK coordinates of the point click in the lower left of the window.

When extending the vtkslider widget we noticed that the slice number label on the slider did not correspond to the actual slice being viewed. To fix this we disabled the label and set it to invisible, we then used the slider image value to create the correct slice number displayed as text below the slider.

One problem we found with vtkImageViewer2 was that the z coordinate calculation during picking would give the same z value for different slices. We solved this problem by computing the z value according to the

current slice number with respect to the z origin of the first slice (reader->GetDataOrigin(), *zorig*), the orientation of the data (reader->GetImageOrientationPatient(), and the z spacing between slices (reader->GetPixelSpacing(), *deltaz*).

We note that the first slice in this vtkImageViewer2 widget corresponds to the last slice in IGSTK's DICOM viewer. In addition, with vtkImageViewer2 the positive Y points up, whereas in IGSTK the positive Y points down. IGSTK adopts a LPS reference frame centered on the patient, left is the left of the patient (Figure 2). Since we wish to overlay the 3D models and DICOM data in IGSTK, the output vertex coordinates need to be converted to the target reference frame. The following equation computes z for each slice.

```
int nx=n-1;                          //n is the total number of slices
for(int i=0; i<n; i++)
{          zcurrent=zorig+(  ((float) (nx-i))*deltaz  ); }
```

It is tempting to simply convert Y, by subtract Y from the the vertical image size in pixels , however the coordinates of the points in the VTK widget are projected coordinates, hence we need to subract Y from the maximum Y display value instead. This can be retrieved with:

```
double *bounds = imageViewer->GetImageActor()->GetDisplayBounds();
```

Some DICOM files have a position offset stored, so the final x, and y conversion is:

```
float *offset=reader->GetImagePositionPatient();
p[0]=(p[0])+offset[0];
p[1]=bounds[1]-p[1]+offset[1];
```

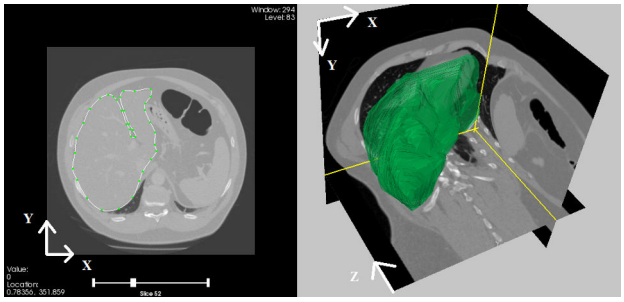

Figure 2. left) VTK coordinate system;right) IGSTK LPS coordinate system.

Finally both types of our reconstructed models can have a low and high resolution version. The low resolution versions simply extract the control points from the contour widget:

```
vtkPolyData *polyData(0);
polyData = vtkPolyData::New();
crep->GetNodePolyData(polyData);
```

Whereas the high resolution versions extracts additional points that make the curves more smooth:

```
polyData = crep->GetContourRepresentationAsPolyData();
```

Before any tiling or polygon triangulation can take place, it is important to establish a consistent vertex ordering in all the contours. The user is likely to make polygon contours with arbitrary orientations, hence we enforce a counter clockwise ordering by firing a ray perpendicular to the first segment (V2-V1) and count the number of intersections (alike [Oliveira02] but in 2D); if there is an odd number of hits it was counter clockwise, if there are zero or an even number of hits it is clockwise and the vertex order needs to be reversed.

3.2 Contour correspondence and branching

Contours can be quite far apart horizontally from one slice to the next, making it hard [Bajaj95] or impossible for automatic 3D reconstruction solutions to know when is a contour supposed to stitch to another, or to simply end and start a new structure. Additionally the proximity of contours can make it equally difficult to establish how a structure actually branches.

For the vascular network reconstruction the user does not need to establish contour correspondence, as each individual extruded contour block is systematically stacked over each other.

For the surface reconstruction of the liver we point out that the genus of the object is quite simple, and that the outer surface typically only branches once at one of the lower tips of the organ. Since we wish to see the vascular networks inside the surface of the liver, the alpha blending actually allows for this extra organ tip to be treated as a separate closed surface. Much like Metaballs[Shen86] that are a collection of overlapping spheres representing the shape of an object, treating the organ as two separate objects has one considerable benefit in that a 1 to 1 correspondence can be infered, instead of requiring the user to define all correspondences in the model (the user is likely to define the contours corresponding to each branch in arbitrary order, making the correspondence not automatic anymore, in addition with branching. simply checking whether the next slice has a polygon contour or not is not enough to establish whether the polygon is the top or bottom of a structure as shorter branch structures terminate before the main branch and this needs to be signalled by the user). To avoid considerable user intervention establishing the contour correspondence, the user merely needs to copy and paste the first polygon contour of the branch into the slice with the last section before the branch thus creating two overlapping objects. Each organ keeps a separate polygon contour definition without the other contours. Nevertheless we implemented branching, which affects slightly how the contour polygons are tiled and is explained in the next section. The user can establish correpondence between two polygon contours by first right-click a position near a polygon and pressing the 'c' key, the user can then change the slice being viewed, select a different contour in the same way and press 'v' to create a post-connection and a pre-connection between contours according to depth. To speed up the annotation of contour correspondences, when the contours of a slice are copy and pasted to the next, the 1-1 contour

correspondences between identical contours are automatically created. Multiple pre-connections and post-connections can be accumulated for a given polygon, when the user right clicks near a polygon, typically the following is printed in the console:

3-0 **[4-0]** 5-0, 5-1

which means that the first polygon contour of slice 4 was selected and this polygon has a correspondence/pre-connection with the first polygon of slice 3, in addition the polygon has a branch post-connection with the first and second polygon contours of slice 5.

3.3 Tiling

Several algorithms exist for tiling [Meyers94]. For the vascular structures we simply duplicate the entire contour polygon into the next adjacent slice, and as a 1 to 1 correspondence exists between the vertices the tiling configuration illustrated in Figure 3-a) was used.

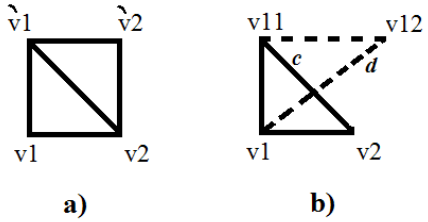

Figure 3. a) Tiling of vascular network polygons/ polygon extrusion; b) Tiling in surface based reconstruction using the minimal length diagonal.

For the surface reconstruction we implemented Christiansen et al. minimal spanning diagonal method for tiling [Christiansen78] as it also offered an elegant solution for branching. This algorithm essentially finds two vertices that are in proximity to each other (v11 and v1 in Figure 3-b form a bridge between the different slices) it then measures the length *c* between v11 and v2 and the length *d* between v1 and v12, the shortest length indicates the triangle that is created, in this case the triangle (v11, v1, v2) is inserted instead of the triangle (v11, v1, v12). The bridge then becomes v11 and v2, and the length from v12 to v2 is compared with the length from v11 to the vertex to the right of v2 and so forth.

For branching, essentially the branched contour polygons are inserted into one ordered polygon that goes through a new vertex. This new vertex is halfway in-depth beetween the two slices, and is positioned half-way between the closest vertex pair between the branched polygons. The new polygon definition goes through the closest pairs and midpoint twice as it bridges the adjacent polygon and returns (see Figure 7).

3.4 Contour Triangulation

In the surface reconstruction, the first vertical contour and the last vertical contour of a structure need to be triangulated otherwise holes will result in the model. A simple algorithm for triangulating the planar contour, would be to compute a centroid vertex and create triangles with adjacent contour vertices and the centroid (Figure 4, right). However more complex polygons (Figure 5, right) representing the start of an organ are possible, and the centroid computation might yield a vertex that actually is not inside the contour or can create intersections with other triangles in the contour. We implemented David Eberly´s *Triangulation by Ear Clipping* [Eberly08] to address these issues.

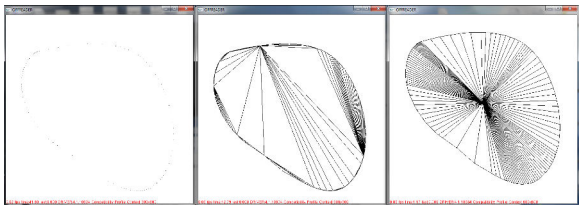

Figure 4. Contour triangulation: left) vertices; middle) Triangulation by Ear Clipping; right) centroid triangulation.

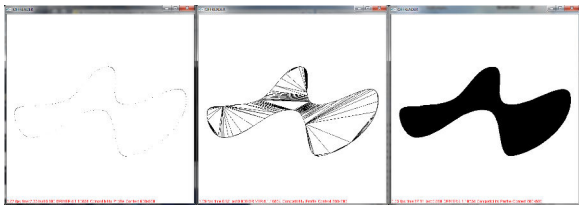

Figure 5. Contour triangulation by Ear Clipping: left) vertices middle) wireframe right) flat shading.

This algorithm works by first computing a list of ear vertices, an ear is a triangle formed by three consecutive vertices of a polygon that does not contain any other vertices inside. He calls this the triangle containment test. As each ear vertex is removed, a triangle is added, and the contour polygon shrinks. A simple polygon of n vertices always has n-2 triangles. After removing an ear vertex, the status of vertices adjacent to the ear vertex needs to be updated, vertices that were not ears could become an ear and need to be added to the ear list, and vertices that were ears might stop being an ear and need to be removed from the ear list. Eberly outlines an implementation that is O(n^2).

In the vascular network reconstruction, each contour polygon that is extruded and tiled, has to be triangulated from the top and the bottom creating a collection of wedge like structures.

4. RESULTS

Numerical results are summarized in Table 1. The R column indicates which reconstruction technique was used. S and s indicate models reconstructed with the surface based approach using high resolution vertices of the contours and just the contour points respectively. B and b indicate vascular networks that were built with the block like approach, likewise B used the high resolution vertices and b used just the control points. The Object denoted object was a small object of interest created with high resolution vertices for tumour visualization. With rendering performance in mind, it can be seen that the full

resolution liver surface together with the full resolution vascular networks could present some performance issues in some tasks (61048+275360= 336408 triangles[liver1 total]). Fortunately the lower resolution versions (4980+76+11574=16630[liver1], Figure 8) and (3248+550+7000=10798[liver2], Figure 9) respectively enable fast rendering and spatially convey all the bifurcations required for risk assessment in a planned intervention. In particular, the alpha blending set in IGSTK to 0.4 enables the additional surface structure in the smaller liver object to be almost unnoticeable whilst freeing the user from the polygon correspondence task all together. We argue that some polygon tilling will simply always create rendering artefacts unless for example a compromise is made in the detail (Figure 6 indicates corresponding problematic tiling locations). No user adjustment attempts were made with the models of Figure 8 and 9 where three out 111 slice connections had some difficulties in Figure 8. These difficulties did not occur with the models shown in Figure 9. Again the alpha blending lessens any visual clutter allowing one to focus on the structures that really matter, however the slice by slice tiling approach proved to have the resilience quality we sought.

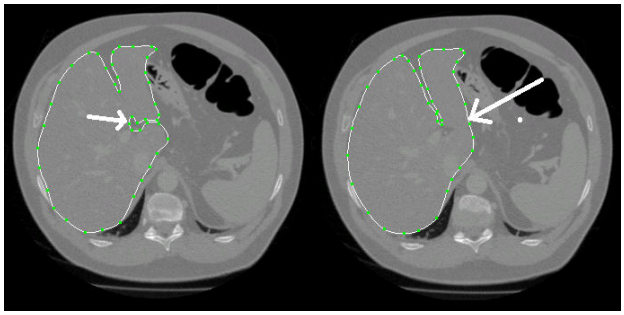

Figure 6. Polygon tiling challenge.

Contour annotation proved to be the longest task in the 3D reconstruction, the copy and paste of contours were only introduced at a later stage and were not used or timed. Manual annotation of 118 contours from 155 CT slices took around 1 hour and 20 minutes to annotate. The system generated a triangulated model (Figure 7) with 61048 triangles and 30526 vertices in under 1 second (Table 1) on a Sony Vaio laptop (Pentium i7-2640M 2.80 GHz with 6GB of RAM). The Liver2 dataset with 74 CT slices, took around 1h30 mins to annotate 59 contours, whilst the liver sub-parts (liver1_b and liver2_b) took around 10-15 minutes to annotate. Both vascular structures took close to 2 hours and 2.5 hours respectively to annotate. We note that the segmentation is inherently a non-linear task as the radiologist needs to browse image patterns in different slices for a given type of object.

Results of the surface reconstruction of the liver over-layed with the surface of the tip of the liver along with the reconstructed vascular network block structures can be seen in Figure 8 and Figure 9. We show an application that uses the reconstructed ply models in Figure 10. For this application we used VTK glyph and implicit plane classes for basic collision detection/resection planning.

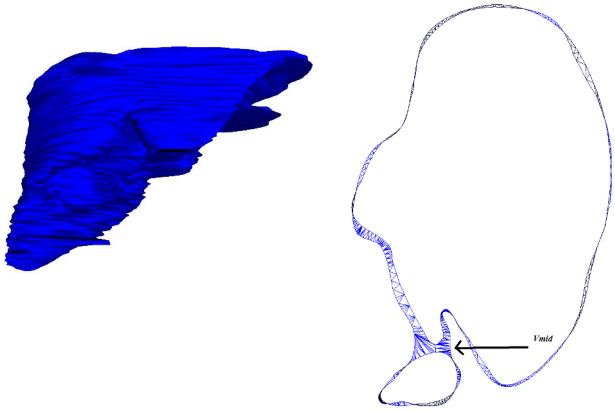

Figure 7. 3D surface reconstructed liver from a pig with 1 branching occurrence. Left) overall model; right) 2 slices with branching and newly inserted vertex Vmid.

Object	R	#loop	#tris	#verts	Time (s)
Liver w/branch	S	118	61048	30526	<1
	s	118	4966	2485	<1
liver1	s	113	4980	2492	<1
	S	113	61964	30984	<1
liver1_b	s	6	76	40	<1
veins1	b	590	11574	7420	<1
	B	590	275360	142338	3
object	S	7	1574	789	<1
liver2	s	59	3248	1626	<1
	S	59	32954	16479	<1
liver2_b	s	19	550	277	<1
veins2	b	300	7000	4388	<1
	B	300	147896	76572	2

Table 1: 3D (R)econstructed models. s/S surfaces (liver, tumor); b/B vascular network; s/b – just uses the control points; S/B – hi-res;

5. CONCLUSIONS AND FUTURE WORK

We presented two reconstruction algorithms for the surface reconstruction of the liver and the vascular structures respectively. We found that the models using only the control points of the polygons are enough to spatially represent the structures for decision making, and allow for fast rendering. Decomposing any branching from a surface into a separate overlapping object produced negligible visual clutter (Figure 8 and 9) while greatly reducing the need for user intervention in contour correspondence and adjusting any tiling mistake in the branching. We believe that the minimal length diagonal can be further improved if the algorithm builds tiles from both sides of the bridge connection, considering the minimal length of either side might allow the tiling to progress correctly until the other side of a problem area. Semi-automatic segmentation is an exciting area of

research where contour selection growing algorithms alike Photoshop's magic wand are possible and the use of texture information could perhaps further help segmentation, the authors would like to explore this area and help reduce the contour annotation time. This work aims to lessen the cognitive load of the surgeon during image-guided surgery. Many laparoscopic surgeries are performed just with the live video and ultrasound imaging. Our intra-operative setup uses our IGSTK customized navigation application to view the reconstructed models and communicate/visualize with the surgical instrument position trackers whilst the live-video from the endoscopic camera is broadcast in a separate viewing window. The system is currently ongoing clinical trials using pigs, but if successful we aim to use the system with humans. We are planning to make the code available open source.

6. ACKNOWLEDGEMENTS

This project is funded by Programa de Cooperación Transfronteriza España Portugal (POCTEP) and Fondo Europeo de Desarrollo Regional (FEDER) Reference code: 0401_RITECA_II_4_E.

7. REFERENCES

[Amenta00] Amenta, N., Choi, S., Dey, T. K., Leekha, N., A Simple Algorithm for Homeomorphic Surface Reconstruction, In proc. of the sixteenth annual symp. on Computational geometry, 2000, 213-222.

[Bajaj95] Bajaj, C. L., Coyle, E. J., Lin, K-N, Arbitrary Topology Shape Reconstruction from Planar Cross Sections, Computer Science Tech. Reports. Paper 1205. 1995. <http://docs.lib.purdue.edu/cstech/1205/>

[Christiansen78] Christiansen, H. N., Sederberg, T. W., Conversion of complex contour line definitions into polygonal element mosaics, Computer Graphics XIII,2(August, 1978), 187-192.

[Eberly08] Eberly, D., Triangulation by Ear Clipping, 2008. <http://www.geometrictools.com>,

[Fuchs77] Fuchs, H., Kedem, Z. M., Uselton, S. P., Optimal Surface Reconstruction from PlanarContours, Communications of the ACM, XX-10(1977),693-702.

[Goryawala12] Goryawala, M., Guillen, M. R.,Cabrerizo, M., Barreto, A., Gulec, S., Barot, T.C., Suthar, R.R., Bhatt, R.N., Mcgoron, A., and Adjouadi, M, A 3-D Liver Segmentation Method with Parallel Computing for Selective Internal Radiation Therapy. IEEE Trans. Inf. Technology in Biomedicine, 16 (1), 2012, 62-69.

[Heimann09] Heimann, T., et al., Comparison and Evaluation of Methods for Liver Segmentation from CT datasets. IEEE Transactions on Medical Imaging, volume 28, number 8, 2009, 1251-1265.

[Hoppe92] Hoppe, H., DeRose, T., Duchamp, T., McDonald, J., Stuetzle, W., Surface reconstruction from unorganized points. SIGGRAPH'92, 71-78.

[Igstk14] IGSTK, The Image-Guided Surgery Toolkit, 2014, <http://www.igstk.org/>

[Itk14] itk-SNAP, 2014, <http://www.itksnap.org>

[Keppel75] Keppel, E., Approximating complex surfaces by triangulation of contour lines, IBM J. Res. Develop. 19 (Jan. 1975), 2-11.

[Maeda09] Maeda, T., Hong, J., Konishi, K et al., Tumor ablation therapy of liver cancers with an open magnetic resonance imaging-based navigation system. Surg Endosc 23(5), 2009, 1048-1053.

[Meyers94] Meyers, D., Reconstruction of Surfaces From Planar Contours, PhD Thesis,UniversityofWashington

[Oliveira02] Oliveira, J. F., Steed, A., Determining orientation of Laser scanned surfaces, in proceedings of 1st Iberoamerican Symposium in Computer Graphics, SIACG2002,Guimarães, Portugal, 281-288.

[Sánchez-Margallo11] Sánchez-Margallo, F. M., Moyano-Cuevas, J. L., Latorre, R., Maestre, J., Correa, L., Pagador, J. B., Sánchez-Peralta, L. F., Sánchez-Margallo, J. A., Usón-Gargallo, J., Anatomical changes due to pneumoperitoneum analyzed by MRI: an experimental study in pigs, Surg. Radiol Anat, 33(5), 2011, 389-96.

[Shen86] Shen, J., Thalmann, D., Interactive shape design using metaballs and splines, SIGGRAPH86, 151-160.

[Sliver14] Sliver, 2014, <http://www.sliver07.org/>

8. APPENDIX A – IGSTK INSTALLATION

In order to compile the IGSTK/VTK libraries with minimal problems on Windows 7, we found that the following steps should be taken:

- Use *git* to download and checkout VTK and IGSTK libraries, instead of any url distribution.
- Turn-off the shared library option in CMake2.8 when compiling every library.

The following libraries need to be installed: IGSTK 5.2, VTK 5.10.0, ITK 4.4.1, QT 4.8.5, FLTK 1.0.11

9. APPENDIX B – APPLICATION

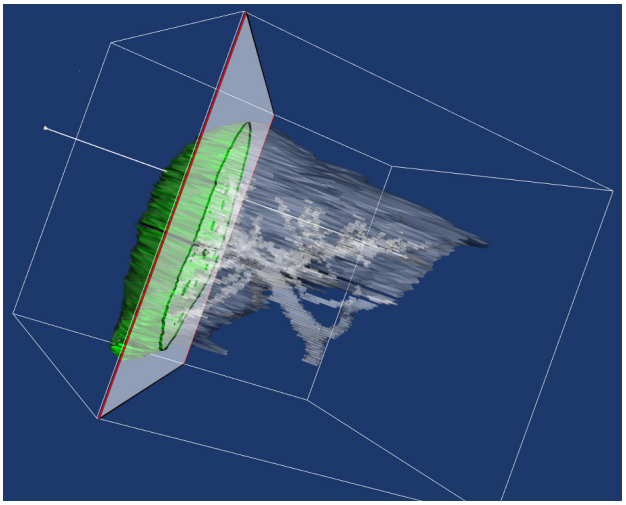

Figure 10. Resection Planning tool with liver1.

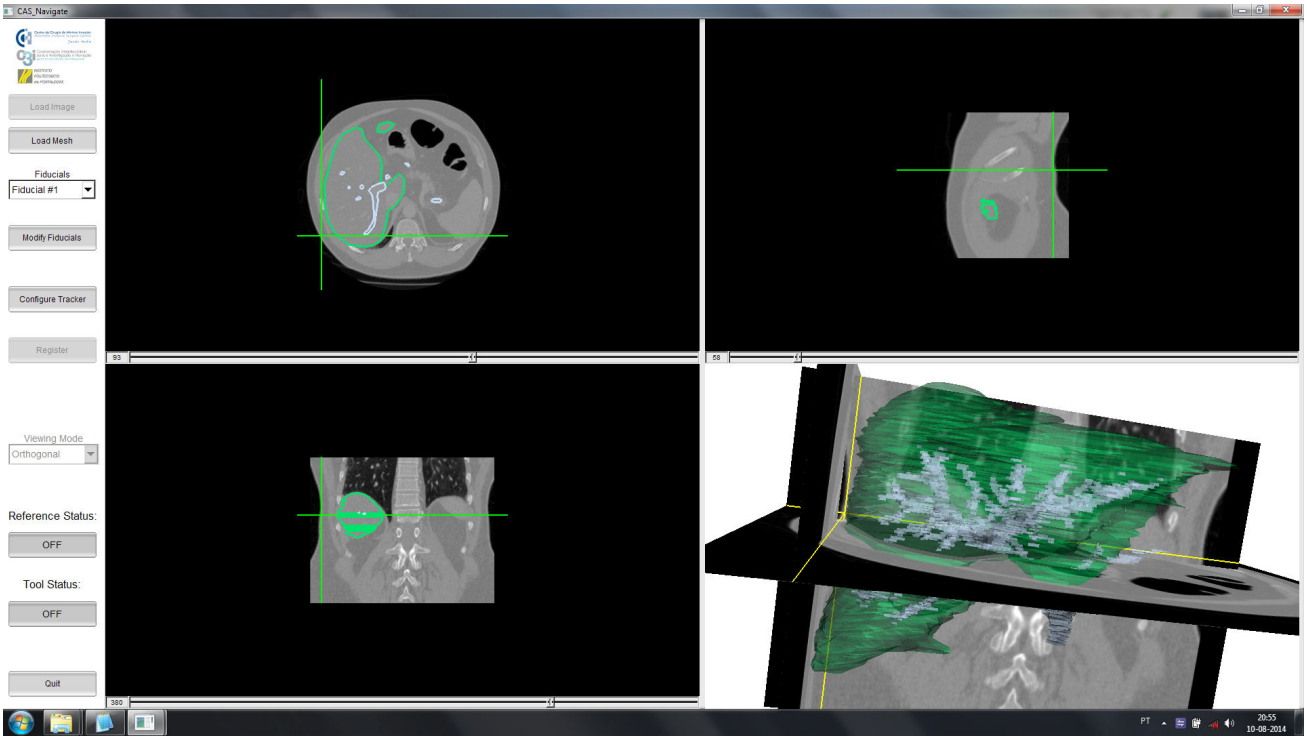

Figure 8. IGSTK visualization of liver1 result: liver1 (4980) + liver1_b (76) + veins1(11574) = 16630 triangles.

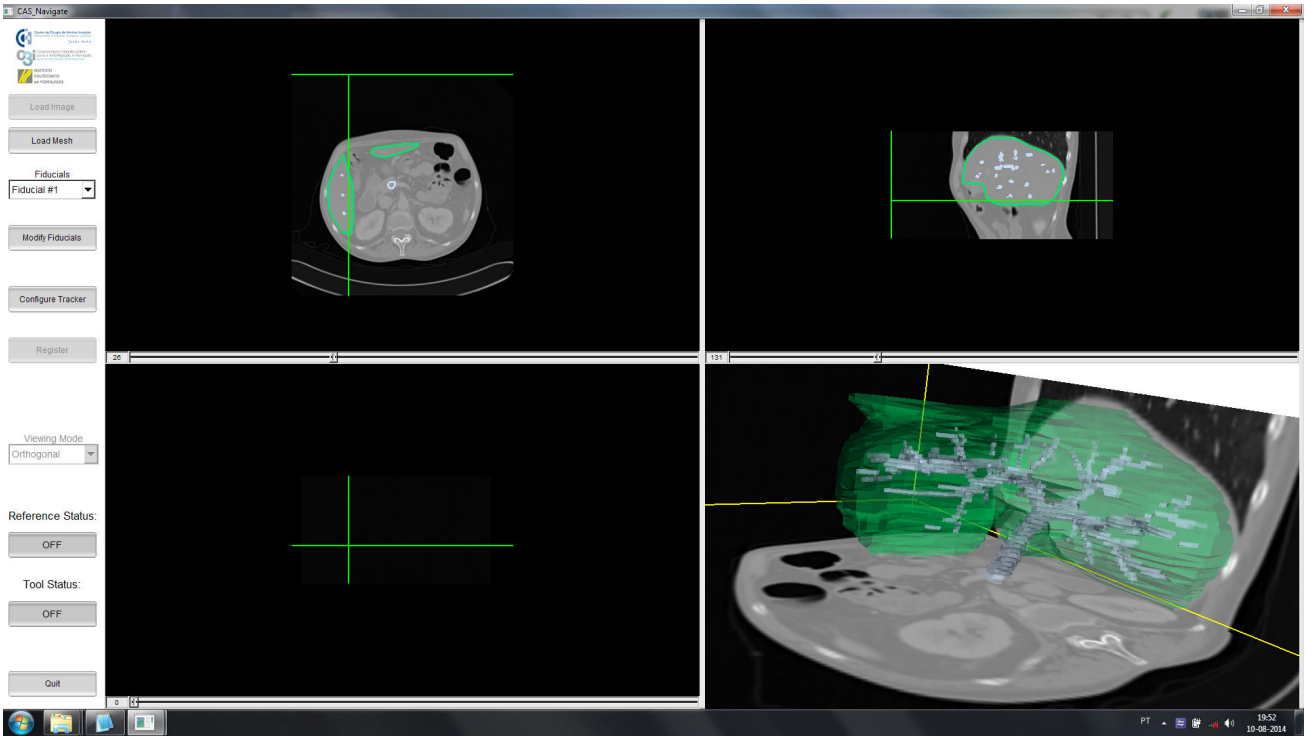

Figure 9. IGSTK visualization of liver2 result: liver2 (3248) + liver2_b (550) + veins2(7000) = 10798 triangles.