A Framework for the Visualization of Brain Structures

Sebastian Thelen
s_thelen@informatik.uni-kl.de
University of Kaiserslautern, Germany

Torsten Bierz
bierz@informatik.uni-kl.de
International Research Training Group
University of Kaiserslautern, Germany

Britta Müller
bmuelle@rhrk.uni-kl.de
University of Kaiserslautern, Germany

Hans Hagen
hagen@informatik.uni-kl.de
University of Kaiserslautern, Germany

Achim Ebert
ebert@informatik.uni-kl.de
University of Kaiserslautern, Germany

Eckhard Friauf
eckhard.friauf@biologie.uni-kl.de
University of Kaiserslautern, Germany

Jörg Meyer
jmeyer@uci.edu
University of California, Irvine, USA

Abstract: Nowadays, biologists investigate different causes for deafness. One reason is a damage in a particular region of the auditory brain stem. These differences were discovered by investigating brain slices of different laboratory mice. However, these slices are only a two dimensional representation of the whole brain part. The arising question was how these differences of structure affect the three dimensional representation of this region. Therefore, an interdisciplinary framework was developed, which allows even unknown users to investigate and compare these regions.

1 Introduction

The *Superior Olivary Complex* (SOC) is a part of the auditory brain stem playing an important role in hearing. Consisting of several *cores*, each is responsible for a particular aspect of sound processing. The discovering of genetic suppression of a certain calcium channel in mouse brains results, among other things, in deafness by birth. Cross-sections of the SOC revealed anatomic differences in a core called *lateral superior olivary* (LSO). The LSOs of mice with the genetic suppression (knock-out mice) differ in form and size, respectively number and density of cells from the LSOs of healthy mice. Further studies will be supported by a framework, that can visualize the cores of the SOC (especially the LSO) and determine characteristic parameters, such as volumes and surface areas of the cores.

The following section provides an overview of related work on the field of brain visualization. Then, the reconstruction of cores, which is based on contour information taken from digital gray images of the brain slices is presented. Afterwards, methods of volume- and

surface estimation for the analysis of the cores are described. Finally, the framework itself is presented. The paper ends with a conclusion and perspective on future work.

2 Related work

Nowadays, numerous different applications deal with the visualization and analysis of cerebral data sets, with various purposes.
The *Allen Brain Explorer* [Al07], a product of the *Allen Institute for Brain Science*, provides a detailed cellular-resolution, genome-wide map of gene expression in the mouse brain. In 2006, the explorer was able to visualize the expression of approximately 20,000 genes in a three dimensional model of the brain.
AnatQuest [An07] is based on the *Visual Human Project* [NLM07], a collection of digital images of complete human male and female cadavers in MRI, CT and anatomical modes. The generated images in AnatQuest are used to create three dimensional models of anatomical objects within these slices. The *Insight Toolkit* (ITK) [ITK07] also supports the Visual Human Project and employs leading-edge segmentation and registration algorithms in two, three, and more dimensions.
Nearly all applications rely on a method to reconstruct their models. Currently, this is done based on contour information. Besides the approach applied in this paper and first presented in [EPO91], various other authors dedicated their work to this topic. A detailed summary of previous work can be found in [MS92].
In order to achieve good results in the reconstruction process, it is often necessary to align the data in a preceding step. For cross-sectional data sets, pin holes in the preparations can simplify this task [SM01]. Unfortunately this method was not applicable in the current case because the thin brain slices were too fragile.

3 Core reconstruction

A typical data set consists of 6 to 10 gray scale images of anatomical cuts of the SOC, sliced at approximately 30 μm intervals. All images were segmented by manually marking the contours of the cores to be reconstructed (e.g., the LSO) with unique colors. The steps of the reconstruction process are described subsequently.

3.1 Contour extraction

First, the contours of the cores need to be extracted from the gray scale images. In a mathematical sense, a contour is an ordered sequence of two dimensional points. Therefore, it is not possible to simply traverse the image row by row and check for contour pixel coordinates. Otherwise, their order will be ignored. *Contour Tracing Algorithms*, like the one of *Pavlidis* (see [Pav82]), provide a solution. Pavlidis' algorithm extracts all contour

pixels in counterclockwise direction by always searching for the next rightmost contour pixel in moving direction. Figure 1 illustrates how the contour pixels of the gray object are identified one after another in correct order. Solid red arrows indicate directions in which a pixel is searched for. Directions with dashed red arrows do not have to be checked anymore, because a suited pixel was already found before.

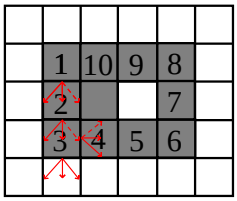

Figure 1: Contour extraction with Pavlidis' algorithm

3.2 Alignment

The next steps in the reconstruction process are based on the extracted contours. However, a misalignment of the slices due to manual placement onto glass object carriers before taking the photos prevents them from a direct use. Slices can be displaced, twisted and even distorted. The former two result from an imprecise placement, whereas the latter one is due to the varying pressure of the brush, used to place the preparations. Therefore, contours must be aligned in order to compensate the transformations. By performing a *principal component analysis* (PCA) it is possible to treat translations and rotations between successive contours (see [Lac06]). Identifying the principal components of a two dimensional contour yields to two perpendicular vectors in the barycenter of each contour (i.e. the eigenvectors of their covariance matrix). Contours are aligned by making the

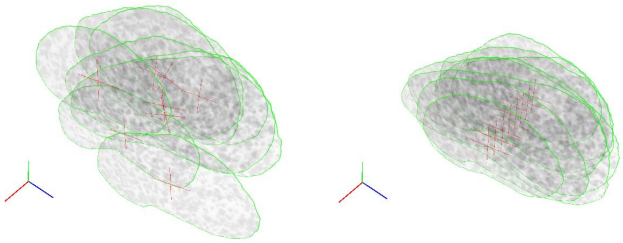

Figure 2: Alignment via PCA

crosses in their barycenters coincident through mutual shifting and twisting (see Figure 2). Provided that contours are quite similar and not too much transformed, this technique produces useful results in most situations.

Figure 3: Loss of structure

However, it should not be concealed that in some cases the structure of a core can be destroyed. Figure 3 illustrates the problem. The original structure of a core is as shown on the left side with contour centers mutually displaced. Through the alignment the centers are made coincident and the native structure of the core gets lost. The reason is the impossible determination whether the displacement of barycenters is due to the core's natural structure or to the manual positioning onto object carriers. Finding an appropriate alternative will certainly be a topic for future work.

3.3 Establishing a correspondence between points of consecutive contours

The aligned contours can be used to identify corresponding points in successive contours. The applied method was first presented by Ekoule et. al. [EPO91]. A correspondence can be expressed by a function $Z : \mathcal{P} \rightarrow \mathcal{Q}$ from the set of points of the first contour into the set of points of the following one. The mapping of Z should be "natural", as illustrated in Figure 4. Here the point p is mapped to $Z(p)$ which seems quite unnatural because with regard to the geometry, q would be better suited. Ekoule et. al. determined corresponding

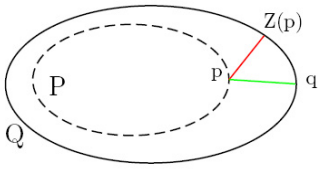

Figure 4: Natural mapping

points by the use of a *local distance criterion* for *convex* contours. If the pair of points $(p_i, Z(p_i))$ is selected, $Z(p_{i+1})$ is searched for among the successors of $Z(p_i)$ within a local neighborhood, so that the distance between p_{i+1} and $Z(p_{i+1})$ becomes minimal.
Nonconvex contours are treated by the projection onto their convex hulls and determining the correspondence between the projected contours. The projection procedure must take

into account the local deformation of the contour. For this purpose, contours are subdivided recursively into concavities which are projected onto the convex hull of the superior concavity.
The framework is able to identify *single branching* correspondences. For each core is exactly one contour in every slice of the data set. *Multiple branching* situations arise e.g. when a core splits up and are discussed by Ekoule et. al. in [EPO91].

3.4 Forming triangle meshes

After identifying correspondences between points of consecutive contours, the triangle meshes of the core models are constructed. The order of pixels within a contour and the mapping of $Z : \mathcal{P} \rightarrow \mathcal{Q}$ already define triangle edges. However, assuming without loss of generality $|\mathcal{P}| \leq |\mathcal{Q}|$ holds (otherwise they are switched), so there can be points in $\mathcal{Q}$ that are not linked to points in $\mathcal{P}$. These must be taken into account when constructing the triangles. The method used is described by Linsen in [Lin97]. Unlinked points are assigned to points in $\mathcal{P}$ and possible gaps are closed. It is verified that the orientation of points is consistent for all triangles of the model. As an example of the hole reconstruction process, the wire frame model of a Lateral Superior Olivary is shown in Figure 5.

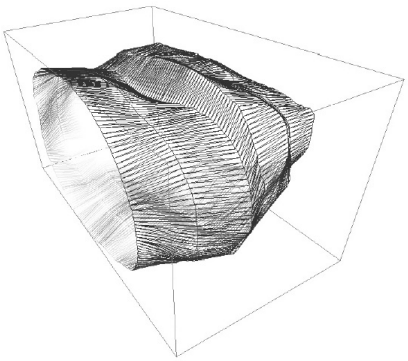

Figure 5: Wire frame model of the LSO

4 Analysis

As stated in the introduction, the LSOs of knock-out mice and normal mice differ from each other. Several parameters characterize these differences. The framework supports the comparison of the two mouse groups by providing an automated estimation of the volume- and surface values for the reconstructed cores.

4.1 Volume estimation

The volume of a core can be calculated by figuring out the integral $\int_Z A(z)dz$, where $A(z) =$ *contour area at depth* z (see [RH90]). This integral is approximated by

$$\sum_{i=0}^{n-2}(z_{i+1} - z_i)\left[\frac{A(z_i) + A(z_{i+1})}{2}\right],$$

a discrete sum of trapezoidal areas beneath the graph of A. The parameter n describes of the number of contours, z_i their depth values and $A(z_i)$ the contour area at position z_i. A comparison shows a high degree of similarity between the results of automatic estimations and the ones of manual calculations. This underlines the correctness of the approach.
The results are independent from the method of alignment (as far as only translations and rotations are treated). This is guaranteed by *Cavalieri's Principle*, which claims that "*if, in two solids of equal altitude, the sections made by planes parallel to and at the same distance from their respective bases are always equal, then the volumes of the two solids are equal*" (see [Wei03]). Hence, possible variations in the method of alignment will not affect volume estimation.

4.2 Surface estimation

The surface estimation of a core is straight forward compared to its volume estimation by summing up the triangle areas of a core's model. The method of alignment strongly affects the results of this estimation, so possible alterations in future versions will lead to different results.

5 Implementation

The framework itself is designed for Windows. C++ and Qt4 were used for the implementation of the base code and the *graphical user interface* (GUI). In the following section the basic architecture of the framework is described and extracts of the GUI are presented.

5.1 Basic architecture

The framework consists of four different components which implement the methods described in the previous sections. They behave like a pipeline, for each component processes the output of the component run before. Their names and functionalities are:

ContourMaker This component extracts contours from image files in the *Portable Network Graphics* format (PNG) by executing an implementation of Pavlidis' algorithm. All contour pixels within a slice are identified and saved in *xls*-files.

SliceConstructor The *xls*-files of the previous stage are processed by the SliceConstructor, which performs a PCA on the contour points and identifies translation vectors and rotation angles for the following alignment. All alignment information is stored in *xls-files* for further use.

VolumeConstructor This component is responsible for the reconstruction of the core models. Points in consecutive contours can be linked as already mentioned before and create triangle meshes out of these points. The result is saved in form of *Obj/Wavefront*-files, which can be displayed by various viewers.

Viewer3D Finally a three dimensional representation of the SOC and its cores must be created and displayed. The component offers the user the possibility to examine the data set in a virtual environment and to interact with it by adjusting the perspective as required (through translation, rotation and zooming).

All components are controlled by *configuration files* which are loaded and contain information about the input files to process and the output files to create.

5.2 Graphical User Interface

In the following section, various screen shots of the framework are presented, in order to provide a better view of the application. Figure 6 shows the main window. It mainly

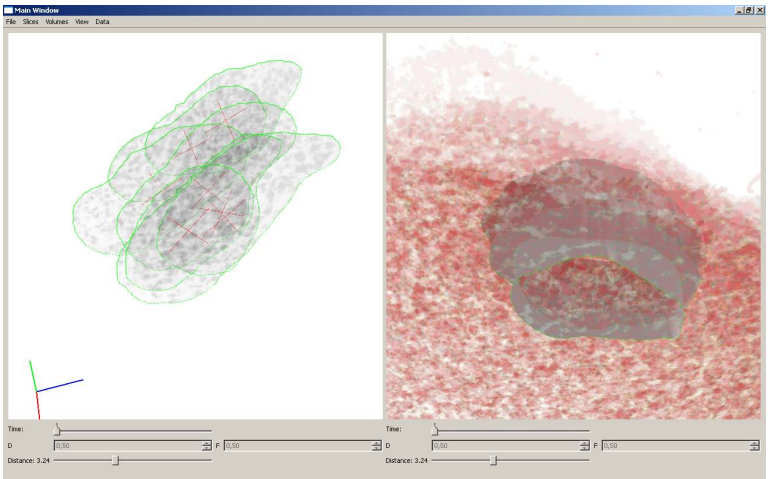

Figure 6: MainWindow

consists of two graphic windows (instances of Viewer3D) that is able to display different data sets simultaneously. Thus it is possible to directly compare the two mouse groups. Beneath the windows, several control elements allow the user to take influence on the representation of the data (distance between slices, current time step, parameters for artificial end pieces). This is further supported by elements of the main menu.
The original gray scale images in the right graphic window have been colored by applying a transfer function. With the dialog shown in Figure 7 the user can determine how gray scale values are mapped to RGB-colors. Up to three disjoint intervals can exemplarily be declared in the set of gray scale values over which RGB-colors are interpolated linearly. By applying a transfer function, it is possible to emphasize certain features of the data set (e.g., dark regions with lots of cells).

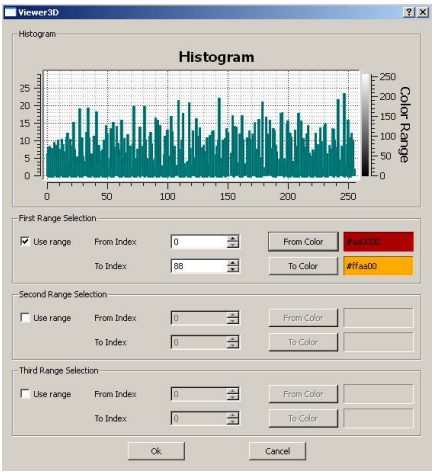

Figure 7: Transfer function dialog

Finally Figure 8 shows how the results of the core analysis can be compared by the use of histograms. A histogram always shows the analysis results of the cores in the left *and*

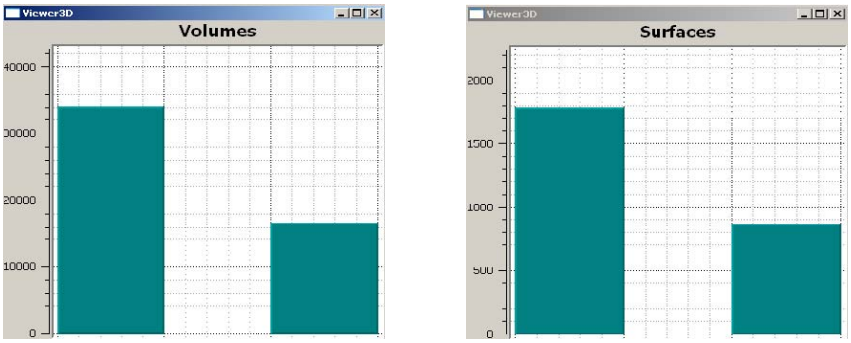

Figure 8: Analysis diagrams

right graphic window, which simplifies recognizing regularities and differences in the data sets (e.g., the LSO volume of a normal mouse is bigger than that of a knock-out mouse).

6 Conclusion and Future Work

In this paper the development of a inter disciplinary framework for the visualization of brain structures was described. The framework can visualize a particular region of the auditory brain stem of a mouse. Data sets consist of cross-sections containing contour information about the cores to be reconstructed. The reconstruction process was discussed in detail and difficulties with the alignment were pointed out. Different opportunities to analyze the structures were described.
Considering the specific needs of the involved biologists, the framework is used in the validation of a deafness model by providing ways to visualize and quantify differences between two mouse groups.
Future work will focus on enhancing the alignment of contours and increasing the framework's flexibility by providing ways to handle branching structures. Another extension is the automatic estimation and calculation of the amount of cells and cell cores, because biologist still try to count their occurrences by hand.

References

[Al07] Allen Brain Atlas - Neuroscience Gateway. http://www.brainatlas.org/aba/, 2007.

[An07] AnatQuest - Anatomic Images Online. http://anatquest.nlm.nih.gov/, 2007.

[CS78] H. Christiansen and T. Sederberg. Conversion of complex contour line definitions into polygonal element mosaics. In *Computer Graphics*, volume 12, pages 187–192. ACM, 1978.

[EPO91] A. Ekoule, F. Peyrin, and C. Odet. A triangulation algorithm from arbitrary shaped multiple planar contours. In *Transactions on Graphics*, volume 10, pages 182–199. ACM, Juli 1991.

[Fri07] E. Friauf. Vom Gehör zum Gehirn: die Neurobiologie des Hörens. http://www.uni-kl.de/wcms/agf_hoeren.html, 2007.

[ITK07] Insight Toolkit (ITK). http://www.itk.org/index.htm, 2007.

[Kab99] W. Kaballo. *Einführung in die Analysis III*. Spektrum, 1999.

[Lac06] F. Lacour. Large-scale biomedical image registration using the principal component analysis. Project thesis, University of Kaiserslautern, 2006.

[Lin97] L. Linsen. Schnitt und Vereinigung von Kontrollnetzen. Diploma thesis, University of Karlsruhe, 1997.

[MS92] D. Meyers and S. Skinner. Surfaces from Contours. In *Transactions on Graphics*, volume 11, pages 228–258. ACM, Juli 1992.

[NLM07] The Visible Human Project. http://www.nlm.nih.gov/research/visible/visible_human.html, 2007.

[Pav82] T. Pavlidis. *Algorithms for Graphics and Image Processing*. Computer Science Press, Maryland, 1982.

[RH90] G. Rosen and J. Harry. Brain volume estimation from serial section measurements: A comparison of methodologies. In *Journal of Neuroscience Methods*, volume 35, pages 115–124, 1990.

[SM01] D. Shulga and J. Meyer. Aligning large-scale medical and biological data sets: Exploring a monkey brain. In *Visualization, Imaging and Image Processing (VIIP 2001)*, volume 3, pages 434–439. The International Association of Science and Technology for Development (IASTED), ACTA Press, September 2001.

[The07] S. Thelen. Entwicklung eines Frameworks zur Visualisierung, Exploration und Komparation biologischer Datensätze. Diploma thesis, University of Kaiserslautern, 2007.

[Wei03] E.W. Weisstein. Cavalieri's Principle. From MathWorld –A Wolfram Web Resource. http://mathworld.wolfram.com/CavalierisPrinciple.html, 2003.