BrainGazer – Visual Queries for Neurobiology Research

Stefan Bruckner, Veronika Šoltészová, M. Eduard Gröller, *Member, IEEE*,
Jiří Hladůvka, Katja Bühler, Jai Y. Yu, and Barry J. Dickson

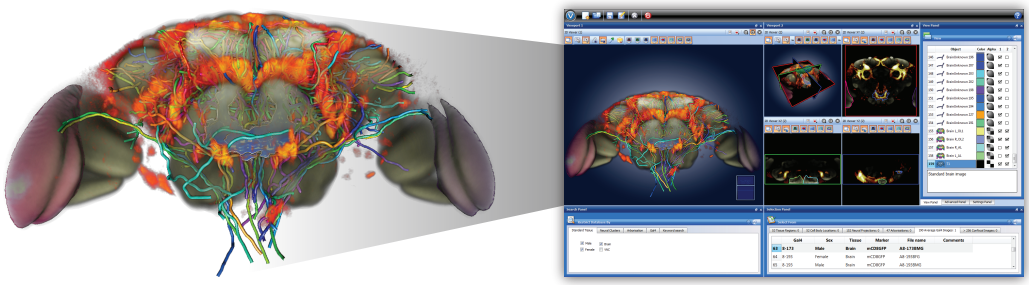


Fig. 1: Neural projections in the brain of the fruit fly visualized using the *BrainGazer* system.

Abstract— Neurobiology investigates how anatomical and physiological relationships in the nervous system mediate behavior. Molecular genetic techniques, applied to species such as the common fruit fly *Drosophila melanogaster*, have proven to be an important tool in this research. Large databases of transgenic specimens are being built and need to be analyzed to establish models of neural information processing. In this paper we present an approach for the exploration and analysis of neural circuits based on such a database. We have designed and implemented *BrainGazer*, a system which integrates visualization techniques for volume data acquired through confocal microscopy as well as annotated anatomical structures with an intuitive approach for accessing the available information. We focus on the ability to visually query the data based on semantic as well as spatial relationships. Additionally, we present visualization techniques for the concurrent depiction of neurobiological volume data and geometric objects which aim to reduce visual clutter. The described system is the result of an ongoing interdisciplinary collaboration between neurobiologists and visualization researchers.

Index Terms—Biomedical visualization, neurobiology, visual queries, volume visualization.

1 Introduction

A major goal in neuroscience is to define the cellular architecture of the brain. Mapping out the fine anatomy of complex neuronal circuits is an essential first step in investigating the neural mechanisms of information processing. This problem is particularly tractable in insects, in which brain structure and function can be studied at the level of single identifiable neurons. Moreover, in the fruit fly *Drosophila melanogaster*, a rich repertoire of molecular genetic tools is available with which the distinct neuronal types can be defined, labeled, and manipulated [36]. Because of the high degree of stereotypy in insect nervous systems, these genetic tools make it feasible to construct digital brain atlases with cellular resolution [33]. Such atlases are an invaluable reference in efforts to compile a comprehensive set of anatomical and functional data, and in formulating hypotheses on the operation of specific neuronal circuits.

One approach in generating a digital atlas of this kind is by acquiring confocal microscope images of a large number of individual brains. In each specimen, one or more distinct neuronal types are highlighted using appropriate molecular genetic techniques. Additionally, a general staining is applied to reveal the overall structure of the brain, providing a reference for non-rigid registration to a standard template. After registration, the specific neuronal types in each specimen are segmented, annotated, and compiled into a database linked to the physical structure of the brain. The complexity and sheer amount of these data necessitate effective visualization and interaction techniques embedded in an extensible framework. We detail *BrainGazer*, a novel visualization system for the study of neural circuits that has resulted from an interdisciplinary collaboration. In particular, in addition to visualization, we focus on intuitively querying the underlying database based on semantic as well as spatial criteria.

The remainder of the paper is structured as follows: Related work is discussed in Section 2. Section 3 outlines the data acquisition workflow and gives a conceptual overview of our system. In Section 4 we detail the visualization techniques employed. Our novel approach for semantic and spatial visual queries is presented in Section 5. Section 6 provides details on the implementation of our system. Results are discussed in Section 7. The paper is concluded in Section 8.

- *Stefan Bruckner, Veronika Šoltészová, and M. Eduard Gröller are with the Institute of Computer Graphics and Algorithms, Vienna University of Technology, Austria, E-mail: {bruckner,vero,groeller}@cg.tuwien.ac.at.*
- *Jiří Hladůvka and Katja Bühler are with the VRVis Research Center, Vienna, Austria, E-mail: {jiri,katja}@vrvis.at.*
- *Jai Y. Yu and Barry J. Dickson are with the Research Institute of Molecular Pathology, Vienna, Austria, E-mail: {Jai.Yu,dickson}@imp.ac.at.*

Manuscript received 31 March 2009; accepted 27 July 2009; posted online 11 October 2009; mailed on 5 October 2009.
For information on obtaining reprints of this article, please send email to: tvcg@computer.org .

2 Related Work

Excellent starting points to get insight into the world of neuroscientists, their data, the huge data collections, and related problems are given by Koslow and Subramaniam [26] as well as Chicurel [11]. Data acquired to study brain structure captures information on the brain on different scales (e.g., molecular, cellular, circuitry, system, behavior), with different focus (e.g., anatomy, metabolism, function) and is multi-modal (text, graphics, 2D and 3D images, audio, video). The establishment of spatial relationships between initially unrelated images and information is a fundamental step towards the exploitation of available data [6]. These relationships provide the basis for the visual representation of a data collection and the generation of further knowledge. Jenett et al. [22] describe techniques and workflow for

quantitative assessment, comparison, and presentation of 3D confocal microscopy images of Drosophila brains and gene expression patterns within these brains. An automatic method to analyze and visualize large collections of 3D microscopy images has been proposed by de Leeuw et al. [13].

Brain atlases are a common way to spatially organize neuroanatomical data. The atlas serves as reference frame for comparing and integrating image data from different biological experiments. Maye et al. [33] give an introduction and survey on the integration and visualization of neural structures in brain atlases. A classical image-based neuroanatomical atlas of *Drosophila melanogaster* is the FlyBrain atlas [16], spatially relating a collection of 2D drawings, microscopic 2D images and text. The web interface provides visual navigation through the data by clicking on labeled structures in images. Brain Explorer [28], an interface to the Allen Brain Atlas, allows the visualization of mouse brain gene expression data in 3D. An example for a 3D atlas of the developing Drosophila brain has been described by Pereanu and Hartenstein [37]. Segmentation, geometric reconstruction, annotation, and rendering of the neural structures was performed using Amira [2]. The Neuroterrain 3D mouse brain atlas [4] consists of segmented 3D structures represented as geometry and references a large collection of normalized 3D confocal images. An interface to interact with the data has been described for neither of these atlases. NeuARt II [9] provides a general 2D visual interface to 3D neuroanatomical atlases including interactive visual browsing by stereotactic coordinate navigation. The CoCoMac-3D Viewer developed by Bezgin et al. [5] implements a visual interface to two databases containing morphology and connectivity data of the macaque brain for analysis and quantification of connectivity data. It also allows graphical manipulation of entire structures.

Most existing interfaces to neuroanatomical databases provide only very limited tools for visual analysis, although there exist powerful general methods for the exploration of multidimensional and spatial data. Surveys of concepts for visual analysis of databases and visual data mining have been published by Derthick et al. [14] and Keim [24]. The most prominent techniques are interactive filtering by dynamic queries [1] and brushing and linking for the exploration of multidimensional data [31]. Special focus on visual analytics of spatial databases discussing multidimensional access methods is subject of the survey by Gaede and Günther [18]. Examples for visual navigation through spatial data can be mainly found in geographical information systems (e.g., Google maps or the public health surveillance system proposed by Maciejewski et al. [30]).

An example for interfaces to neuroanatomical image collections and databases realizing more elaborate visual query functionalities is the European Computerized Human Brain Database (ECHBD) [17]. It connects a conventional database with an infrastructure for direct queries on raster data. Visual queries on image contents can be directly realized by interactive definition of a volume of interest in a 3D reference image. Direct search by drawing regions of interest in a 2D image to query injection and label sites on a set of related studies has been realized by Press et al. [38] as interface to the XANAT database. Ontology-based high-level queries in a database of bee brain images based on pre-generated 3D representations of atlas information have been recently proposed by Kuß et al. [27]. The interactive definition of volumes of interest directly on the 3D data for queries on pre-computed fiber tracts of a Diffusion Tensor Imaging (DTI) data set has been proposed by Sherbondy et al. [42]. Several approaches for the 3D visualization of neurons based on microscopy data have been presented [32, 23, 3, 40]. Rendering of pure geometric representations of large neural networks has been addressed recently by de Heras Ciechomski et al. [12].

Nevertheless, the visual presentation of 3D neuroanatomical image data in query interfaces to large data collections is currently mainly realized as geometric representations of atlas information in combination with or alternatively to axis aligned 2D sections of the image data. The system presented in this paper combines state-of-the-art 3D visualization techniques for neurobiological data with a novel visual query interface, thereby integrating semantic and spatial information.

3 System Overview

The nervous system is composed of individual neurons, which process and transmit information in the form of electrochemical signals. They are the basic structural and functional units of an organism's nervous system and are therefore of primary interest when studying brain function. Different types of specialized neurons exist and knowledge about their arrangement, connectivity, and physiology allows neuroscientists to derive models of cognitive processes. In an interdisciplinary collaboration between neurobiologists and visualization researchers, we investigate neural circuits in the fruit fly *Drosophila melanogaster*. Conserved genes and pathways between flies and other organisms, together with the availability of sophisticated molecular genetic tools make Drosophila a widely used model system for elucidating the mechanisms that affect complex traits such as behavior. This section gives an overview on the basic methodology we use for these studies and the visualization system which has been developed.

3.1 Data Acquisition

We use the Gal4/UAS system [7] to label and manipulate specific neurons in the fly brain and ventral nerve cord. The brain and nerve cord are separately dissected. Specific neurons are stained with a green fluorescent protein (GFP). Additionally, separate neuropil staining is used to facilitate registration – it highlights regions of high synaptic density which provide a stable morphological reference. After preparation and staining, the tissues are scanned using a Zeiss LSM 510 laser scanning confocal microscope with a 25X objective. Data sets of 165 slices at a 1 µm interval and an image resolution of 768×768 pixels are generated.

The neuropil staining is then used to perform non-rigid registration [39] of the scans to a corresponding template for either brain or ventral nerve cord, similar to the approach described by Jenett et al. [22]. The template was generated by averaging a representative set of scans registered against a reference scan. The registration process itself is automatic, but results are manually verified and additional image processing operations may be applied to reduce noise. Only scans considered to be registered with sufficient accuracy are used for the database.

Each neuron is characterized by three types of features: a *cell body*, the *neural projection*, which is an elongated structure that spreads over large areas, and *arborizations*, which contain synapses where communication with other neurons occurs. Neurons are classified based on the morphology or shape of these features. Neurons that share similar cell bodies, patterns of projections, and arborizations, as well as expression of the same Gal4 drivers, are tentatively considered to belong to the same type. Types of neurons having these anatomical properties may perform similar functions.

Standardized volumes are created by generating averages for each Gal4 line which allow evaluation of the biological variability of the corresponding expression patterns. Amira [2] is used to segment cell body locations and arborizations from these average volumes. The resulting objects are therefore representations of the typical locations and shapes of these structures. They are examined together with the corresponding average volumes and individual confocal scans in order to assess their constancy between multiple specimens. Neural projections are traced from individual images using the skeletonizer plugin for Amira [41]. Surface geometry is generated for cell body locations and arborizations, while neural projections are stored as skeleton graphs. References to these files, the original confocal volumes, the average Gal4 volumes, the templates, and template regions (surface geometry based on a template volume representing particular parts of the anatomy such as the antennal lobes) are stored in a relational database. The central entities within this database are *neural clusters* which group cell body locations, neural projections, and arborizations. These neural clusters correspond to particular neuronal types.

3.2 Visualization and Interaction

One of the goals in developing the *BrainGazer* system was to facilitate the study of neural mechanisms in the mating behavior of Drosophila

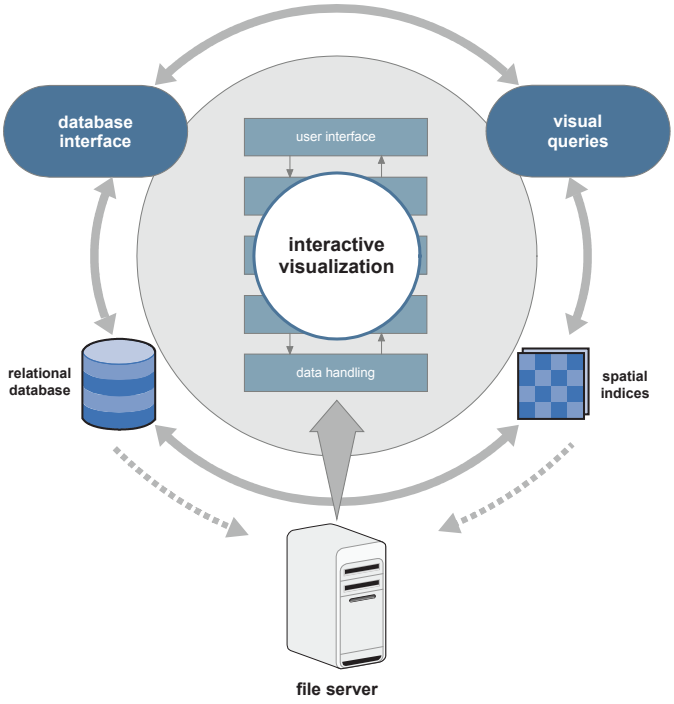


Fig. 2: Conceptual overview of the *BrainGazer* system.

using the acquired data. The research challenge is to reveal how chemical and auditory cues are detected and processed in the fly's brain, how these signals are interpreted in the context of internal physiological states and past experience, and how this information is used to make decisions that are fundamental to the animal's reproductive success [15]. By visualizing individual neurons on a common reference template, potential connections between these neurons based on the spatial colocalisation of their arbor densities can be identified. This information is used to generate network diagrams which allow us to formulate specific hypotheses of circuit function. For example, we have used this principle to identify the neuronal types that constitute a putative pathway for sensing and processing pheromone signals and triggering courtship behavior. In order to facilitate this type of examination, it is important to provide efficient means for interactively accessing the generated database. *BrainGazer* provides two distinct paths to select data for display and analysis (see Figure 2):

Database interface. A traditional table-view database interface allows users to filter and select items based on combinations of different criteria, such as gender or neuronal type. A result view is updated immediately when query parameters change. The user can then select the desired data items and load them into the application. Additionally, it is possible to perform a full-text search of the database to quickly access specific data sets.

Visual queries. While a traditional database interface is useful for quickly accessing a known subset of the data, it is also important to be able to visually search the whole set of available data based on spatial relationships. The visual query interface is displayed directly in the visualization window and provides instant access to contextual information and related structures for selected items and regions of interest.

All data sets are stored on a central file server and transferred on-demand. The relational database storing references to these files is also accessed over the network. To facilitate visual queries, a set of spatial indices is maintained and updated whenever changes to the data occur.

The application itself comprises a rich set of standard tools for 2D/3D navigation (rotation, zoom, pan, slicing), rendering (orthographic and perspective projection, clipping planes, cropping boxes, transfer functions, windowing), multiple linking of 3D and 2D views, multi-screen support, and image and video capture. Working sessions together and all current settings can be saved to disk and later restored with automatic transfer of all loaded data sets. As these features are common in similar systems, we restrict our further discussion to novel aspects of our approach. A typical screenshot of *BrainGazer* is shown in Figure 1.

4 Visualization

One of the challenges in developing *BrainGazer* was the concurrent visualization of many different anatomical structures while minimizing visual clutter. As semi-transparent volume data is depicted together with geometric objects, care has to be taken to avoid occlusion while preserving the ability to identify spatial relationships. In this section, we describe the visualization techniques we employ for this purpose.

Template regions, cell body locations, and arborizations are available as triangle meshes. We render them using standard per-pixel Phong illumination. Neural projections are given as skeleton graphs with optional diameter information. As the diameter values can be unreliable and misleading, the projections are preferably viewed with a constant diameter. However, we provide the option of using the available diameter values as well. The skeleton graph is extruded to cylinders and rendered as polygonal geometry which also enables simple and fast rendering of object outlines in 2D slice views (in the future we also plan to investigate more advanced techniques to improve the visual quality, such as convolution surfaces [35] or self-orienting surfaces [34]). The remaining data sets, templates, average Gal4 volumes, and confocal scans are volume data. The users of *BrainGazer* want to visualize them together with the geometric objects.

4.1 Volume Rendering

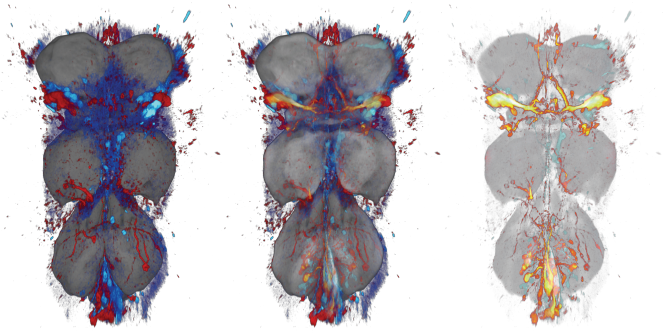
Volume data acquired by confocal microscopy is frequently visualized using Maximum Intensity Projection (MIP) as the stained tissues have the highest data values. When concurrently depicting several different scans, however, the disadvantage of MIP is that spatial relationships are lost. Using Direct Volume Rendering (DVR), on the other hand, suffers from occlusion. This is particularly problematic as it is frequently necessary to visualize several confocal scans together with a template volume which provides anatomical context. The template should not occlude features highlighted in the other data sets but is important for spatial orientation. Thus, in *BrainGazer* we chose to employ a variant of Maximum Intensity Difference Accumulation (MIDA) [8]. As MIDA represents a unifying extension of both DVR and MIP, it is well suited for our problem. We extended the method to enable the concurrent rendering of multi-channel data sets [10].

MIDA uses a generalization of the over operator where the previously accumulated color and opacity are modulated by an additional factor. The accumulated opacity A_i and color C_i at the i-th sample position P_i along a viewing ray traversed in front-to-back order are computed as:

$$\begin{aligned} A_i &= \hat{B}_i A_{i-1} + (1 - \hat{B}_i A_{i-1})\hat{A}_i \\ C_i &= \hat{B}_i C_{i-1} + (1 - \hat{B}_i A_{i-1})\hat{A}_i \hat{C}_i \end{aligned} \tag{1}$$

where $\hat{A}_i$ and $\hat{C}_i$ are the opacity and color, respectively, of the sample and $\hat{B}_i$ is the modulation factor. In the original method, which focused on single-channel data sets, $\hat{B}_i$ was defined based on the absolute difference between the current maximum along the ray and the data value at a sample point. This approach gives increased visual prominence to local maxima. The resulting images share many of the characteristics with MIP, but feature additional spatial cues due to accumulation.

In the following, we present a simple extension of MIDA to multi-channel data. We assume a multi-channel data set consisting of N continuous scalar-valued volumetric functions $f_1(P),\ldots,f_N(P)$ of normalized data values in the range $[0,1]$. Each channel has an associated color function $c_1(P),\ldots,c_N(P)$ and opacity function $\alpha_1(P),\ldots,\alpha_N(P)$.



(a) (b) (c)

Fig. 3: The ventral nerve cord of a fly rendered using (a) DVR, (b) MIDA, and (c) MIP. The template (gray) is depicted together with two average Gal4 data sets (red and blue).

Like Kniss et al. [25], at the i-th sample position P_i along a ray, we sum the opacities and average the colors for the overall opacity $\hat{A}_i$ and color $\hat{C}_i$ of the sample (opacities larger than one are subsequently clamped):

$$\hat{A}_i = \sum_{j=1}^{N} \alpha_j(P_i) \quad \hat{C}_i = \frac{\sum_{j=1}^{N} \alpha_j(P_i) c_j(P_i)}{\sum_{j=1}^{N} \alpha_j(P_i)} \tag{2}$$

We want to enhance regions where the maximum along the ray changes for any channel. Specifically, when the maximum changes from a low to a high value, the corresponding sample should have more influence on the final image compared to the case where the difference is only small. We use δ_j to classify this change at every sample location P_i:

$$\delta_j(P_i) = \begin{cases} f_j(P_i) - \max_{k=1}^{i-1} f_j(P_k) & \text{if } f_j(P_i) > \max_{k=1}^{i-1} f_j(P_k) \\ 0 & \text{otherwise} \end{cases} \tag{3}$$

Whenever a new maximum for channel j is encountered while traversing the ray, δ_j is nonzero. These are the cases where we want to override occlusion relationships. For this purpose, the modulation factor $\hat{B}_i$ from Equation 1 is defined as:

$$\hat{B}_i = 1 - \max_{j=1}^{N} \delta_j(P_i) \frac{\alpha_j(P_i)}{\alpha_{\max}(P_i)} \tag{4}$$

In Equation 4 the maximum of $\delta_j(P_i)$ weighted by the ratio between each channel's opacity $\alpha_j(P_i)$ and the maximum opacity $\alpha_{\max}(P_i)$ of all channels is computed. The additional weighting ensures that invisible samples have no influence on the final image. If the maximum opacity is zero, i.e., no channel is visible at the current sample location, we set $\hat{B}_i$ to one.

The advantage of this approach is that it enables a clear depiction of stained data sets but does not require complex transfer functions to resolve occlusion problems. In our system, transfer function specification is typically performed by defining a linear opacity mapping using standard window/level controls and choosing a pre-defined color map.

Using this volume rendering technique, a high-intensity stained structure immersed in the relatively homogeneous template, for example, will be distinctly visible while still featuring subtle transparency as an additional occlusion cue. Channels with no distinct maxima will appear DVR-like while stained data will exhibit visual characteristics very similar to MIP. In contrast to two-level volume rendering [20] or the approach of Straka et al. [43], no pre-classification of structures of interest is required.

Figure 3 shows the template volume of the ventral nerve cord (in gray tones) together with two stained average Gal4 volumes (depicted in shades of red and blue) rendered using (a) DVR, (b) MIDA, and (c) MIP. The same color and opacity transfer functions are used and all three techniques combine the individual channels using Equation 2. While considerable parts of the stained tissue are occluded in DVR, MIDA and MIP clearly depict the stained neurons. MIDA, however, provides more anatomical context and spatial cues. We allow the user to smoothly transition between these three methods [8].

4.2 Geometry Enhancement

In the targeted application, geometric objects corresponding to segmented anatomical structures are displayed immersed in volumetric data. While the volume data is important as it provides the spatial context, it is undesirable that it fully occludes the geometry. Opacity could be adjusted to prevent occlusion, but it is cumbersome to tune transfer functions individually. Inspired by the MIDA approach to volume rendering, we employ a similar concept to enable the user to see-through the volume even if it would completely occlude intersecting objects. Based on the technique presented by Luft et al. [29], we apply an unsharp masking operation to the depth buffer established during rendering of the geometry. Their spatial importance function ΔD is defined as the difference between the low-pass filtered version of the depth buffer and the original depth buffer. This simple approach gives information about spatially important edges, e.g., areas containing large depth differences.

In our approach, we use ΔD to modulate the accumulated opacity and color along a viewing ray based on the absolute value of ΔD. The result is then blended with the geometry's color contribution. Additionally, as proposed by Luft et al. [29], we can apply depth-enhancement by darkening and brightening the geometry color based on ΔD with no additional cost. The effect of this simple approach is that regions which feature depth discontinuities shine through the volume rendering most. Thus, while giving the user the ability to identify objects immersed in the volume, this methods still indicates occlusion relationships.

An example is shown in Figure 4 – the template brain tissue is depicted together with several neural clusters which are mostly occluded in Figure 4 (a) where no geometry enhancement is applied. In Figure 4 (b) the transfer function was adjusted to make the geometry more visible. Figure 4 (c) clearly depicts the geometry as well as the volume data while still indicating occlusion relationships using our see-through approach.

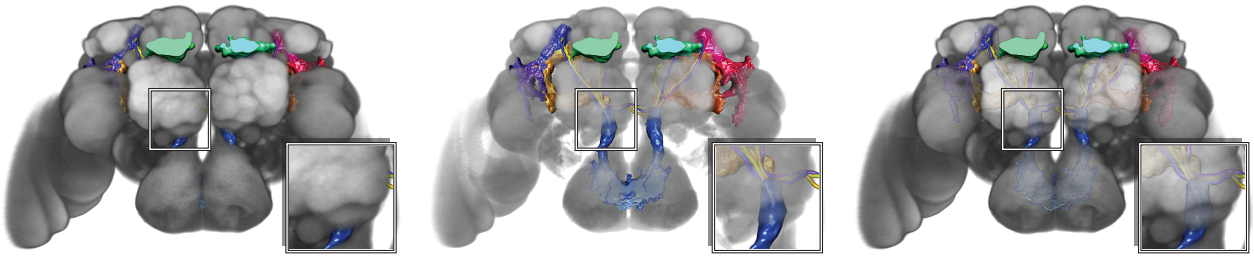
5 Visual Queries

While a traditional database browsing approach is useful for analyzing specific known structures, neurobiological research frequently requires access to the data based on spatial relationships. For example, the biologist may wish to identify neurons or other structures in the vicinity, in order to classify specific objects and to begin to reconstruct neural circuits. A specific case arises as new data is added to the database: the biologist wants to compare it to existing structures in order to decide whether it belongs to a known neuronal type. As there may be substantial variations in individual shapes, it is necessary to investigate all nearby objects to achieve a classification.

BrainGazer provides three basic types of visual queries: Semantic queries give access to related structures using information stored in the database. Object queries are based on the distance between whole objects. Path queries are the most flexible method. Through an intuitive freehand drawing interface, the user can search for proximal structures. These types of queries can be arbitrarily combined. Object and path queries can be used to amend or verify recorded semantic information stored in the database. The user can interact with these different query types through contextual hypertext labels which are displayed in-window.

5.1 Semantic Queries

Semantic queries allow the user to quickly access contextual information and data for an object of interest. They are initiated by simply selecting an object in the visualization window through a mouse click.



(a) (b) (c)

Fig. 4: Neural clusters in the fly brain depicted together with the template volume using (a) no see-through enhancement, (b) no see-through enhancement with an adjusted transfer function, and (c) see-through mode using the same transfer function as in the leftmost image.

If multiple objects overlap in depth, subsequent clicks at the same position allow cycling through them. As soon as a new object has been picked, a contextual hypertext label appears on-screen and provides the information stored in the database such as the name of the structure and comments. References to other related objects are displayed as hyperlinks which can be used to access the associated structure. This includes geometric objects, e.g., other cell body locations, neural projections, or arborizations of the same neuronal type, as well as volumetric data such as the scan the object has been segmented from. This setup can be used to navigate through the data. For instance, an arborization may be part of one or several neuronal types. When selecting the arborization, the label shows all neuronal types linked to the arborization together with the cell body locations, neural projections, and other arborizations as hyperlinks. Selecting another arborization in one of these neuronal types will provide access to further structures considered connected to this arborization. Hovering over a hyperlink also highlights the corresponding objects if they already have been loaded. If the objects are not already visible, activating the link by clicking it will initiate a load operation. This simple approach allows the user to quickly navigate known neural circuits using a familiar interface.

5.2 Object Queries

In addition to providing access to semantic information already present in the database, our system allows users quick access to spatial proximity information in order to aid identification of new relationships.

For this purpose, we create a table which stores the minimum distances of an object to all other objects in a pre-processing step. We use signed distance volumes generated for all objects in the database. The minimal surface distance between two objects i and j is computed by sampling the distance volume of j for every voxel along the surface of i and vice versa. If the minimum is negative, we continue to compute the volume of intersection between the two objects and record it as a negative value. Table entries for each object are then sorted according to ascending distance values. This table is loaded into memory at startup and allows quick access to proximity information whenever an object is selected.

The result of an object query is displayed in conjunction with the semantic information in a contextual label when a picking operation occurs. The label will display hyperlinks for all objects within a certain range from the selected structure. A slider widget integrated with the label allows interactive filtering of the query results based on distance. When moving the slider, the label immediately updates. For each object type, the number of objects within the specified distance range is displayed followed by a list of their names (as retrieved from the database). Each name represents a hyperlink which can be used to load and highlight the object. These links are additionally color-coded to quickly identify the object's degree of spatial proximity.

5.3 Path Queries

Path queries are based on an intuitive freehand drawing interface: the user sketches an arbitrary path on top of the visualization and gets immediate feedback about nearby objects. The result of the query can then be loaded into view for further inspection. Sketches were chosen over more conventual selection tools such as rectangular or circular regions as they allow a more accurate characterization of the region of interest in the context of complex neural anatomy.

5.3.1 Index Generation

To facilitate fully interactive visual queries, we generate a spatial index which allows us to quickly retrieve the objects in the vicinity of a specific location. In a pre-processing step we create a lookup volume and a distance table. At runtime, the lookup volume is kept in memory while the distance table may be accessed out-of-core. The distance table grows with the number of objects in the database while the size of the lookup volume remains constant. This is important for scalability as significant growth in the number of annotated objects is expected. For each voxel, the lookup volume stores an offset into the distance table and the number N of proximal objects found for the voxel position P. Each entry in the distance table corresponds to such a position in the volume and contains a list of N <distance, identifier> pairs. The pairs in the list are sorted according to their ascending distance from P. Negative distances indicate that the point P is located inside of the respective object. During pre-processing, the distances are determined using signed distance fields stored for each object. As we are not interested in objects located far from a queried point, all distances above a certain threshold are ignored and not stored. In practice, a maximum distance of 40 voxels has proven to be useful and is used in our current implementation.

Using these data structures, during interaction objects close to any voxel can be found by simply reading the offset and count from the corresponding location in the lookup volume and then retrieving the respective set of <distance, identifier> pairs from the distance table. In order to enable efficient caching for subsequent accesses to the distance table, it is advantageous to choose a locality-preserving storage scheme. Many out-of-core approaches employ space-filling curves for this purpose. In our current implementation, the entries of the distance table are arranged based on the three-dimensional Hilbert curve [19] which has been shown to have good locality-preserving properties [21]. Figure 5 illustrates lookup volume and distance table for the two-dimensional case.

Our concept also allows easy merging of distance tables and lookup volumes for disjoint sets of objects which is practically needed when new objects are inserted into the database. We merge the entries of the distance tables with a union operation and resort them. The offsets and counts in the lookup volumes can simply be added.

5.3.2 Query Processing

Using the described data structures, we can efficiently determine objects in the vicinity of a voxel. For performing path queries, it is therefore necessary to identify a corresponding 3D object-space position for each 2D point along the path. Whenever a new point is added to the path, we read the depth buffer at the corresponding 2D location and use the inverse viewing transformation to transform it into

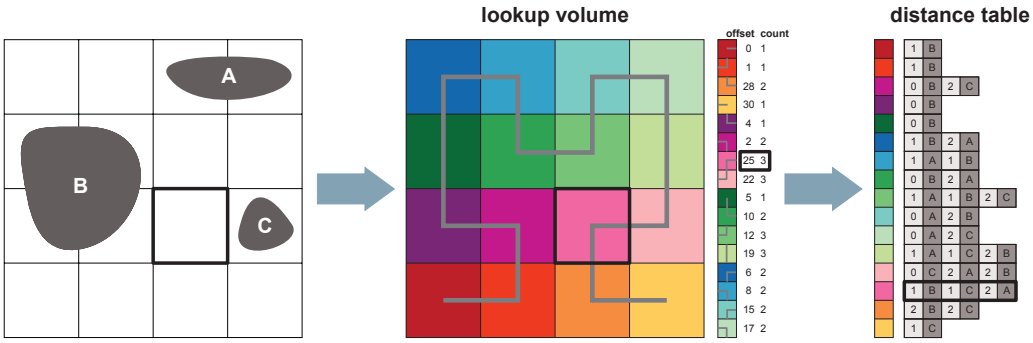


Fig. 5: Lookup volume and distance table for a simple two-dimensional scene containing three objects – the city block distance metric is used and distances above 2 are ignored. The Hilbert curve used to arrange the distance table is overlayed in light gray. An example lookup is indicated with black outlines.

object space. For geometry rendering, this results in the position on the surface of the object closest to the viewer. For volume rendering, however, several samples along a ray may contribute to a pixel. For each viewing ray, we therefore choose to write the depth of the sample which contributes most to the final pixel color. Particularly in conjunction with the volume rendering technique described in Section 4.1 this approach has proven useful – as stained tissues are presented visually more prominent, the respective depth will give access to objects in the vicinity of high-intensity volumetric structures. Using the depth buffer in this way also ensures that there is always a good correspondence between the selected query locations and the actual visualization when operations such as cropping have been applied or a slicing plane is displayed in the 3D visualization.

As path queries are used to find structures which are not visible, all currently displayed objects are ignored. During the query, we maintain a sorted list of <distance, identifier> pairs for all objects encountered along the path. When a new point is added, we retrieve its <distance, identifier> pairs from the distance table and merge them into this list. If an object has already been encountered along the path, the lower of the two distances is stored. Similar lists are kept separately for each object type. This information is then used to present the query results to the user.

5.3.3 User Interaction

A path query can be initiated by the user by simply clicking on any position in the window and painting the desired path while keeping the mouse button pressed. A hypertext label pops up on the side of the window and is constantly updated with current information on the number and type of objects found. As soon as the user releases the mouse button, the contextual label moves to the center of the screen prompting the user to inspect the results. Activating a link by clicking it loads and highlights the corresponding objects. Query results can be discarded by right-clicking the label. The query results are displayed in the same way as for object queries using an integrated slider widget for interactive filtering.

During the query, the specified path is overlayed with proximity clouds which provide an instant visual indication of close objects without having to load the geometry first. For this purpose, for every point of the path we draw a circle for each detected nearby object into an offscreen buffer. The radius of each circle corresponds to the recorded distance and each pixel inside the circle is set to this distance value – entries for objects which intersect the point are drawn using a default radius. Pixel values are combined using minimum blending. The result is a buffer which stores the closest distance to any object at each pixel. These values are then mapped to colors and opacities and displayed semi-transparently as shown in Figure 6 (b). After the query, when the user hovers over any of the hyperlinks the corresponding proximity overlay is shown.

6 Implementation

The presented system was implemented in C++ using OpenGL and GLSL. For the user interface, the Qt toolkit was used. The architecture is based on a flexible plug-in mechanism which allows independent modification or even replacement of system components. This modular concept has proven to be very useful as it allows rapid prototyping of new functionality. For instance, the traditional database browsing components were implemented and deployed first. The visual query module uses the same interface which greatly simplified integration and testing. As the system was developed within the scope of an ongoing project, we expect to add new features such as integration with additional databases using the same procedure. The application is designed to run on commodity PCs equipped with Shader Model 3.0 capable graphics hardware. The system is used on a number of different computers ranging from laptops to high-end visualization workstations.

7 Results and Discussion

Currently, the database contains several thousand individual confocal scans, several hundred average Gal4 volumes, as well as hundreds of geometric objects (cell body locations, neural projections, and arborizations) with new data items being added on a regular basis. The lookup volume is generated at a resolution of $384 \times 384 \times 82$ voxels. For each template (either brain or ventral nerve cord) the current distance table for a cutoff distance of 40 voxels requires approximately 200 MB of storage. The time required for adding a new object is approximately 5 minutes including distance field generation, computation of distance and object tables, and subsequent merging of these tables. These operations are performed in an offline batch process.

In order to gain estimates about the scalability of our approach, we performed experiments with higher cutoff distances which result in a larger number of entries per point. For a maximum distance of 256 voxels, the distance table requires 2 GB of storage thus approximately simulating a growth of one order of magnitude in the number of segmented structures. For the small distance table, the average times for lookup and retrieval of all entries for a single location from the hard disk are below 1 ms even without the use of an explicit caching mechanism. In the case of the large table, the time increases to approximately 3 ms indicating that the approach is prepared to handle a substantial increase in the number of objects.

A typical use-case of our visual query approach is illustrated in Figure 6. Since it is difficult to depict an interactive process using still images and because our user interface is designed for on-screen viewing, we refer to the accompanying video for a sample of an interaction session. Initially, in Figure 6 (a), the brain template is shown together with an average Gal4 volume which has been selected using the database browser. A path query is then specified in Figure 6 (b). The best match of the query – an arborization – is loaded and selected. In Figure 6 (c), the contextual label gives access to semantic query results: the arborization's neural cluster which contains one cell body lo-

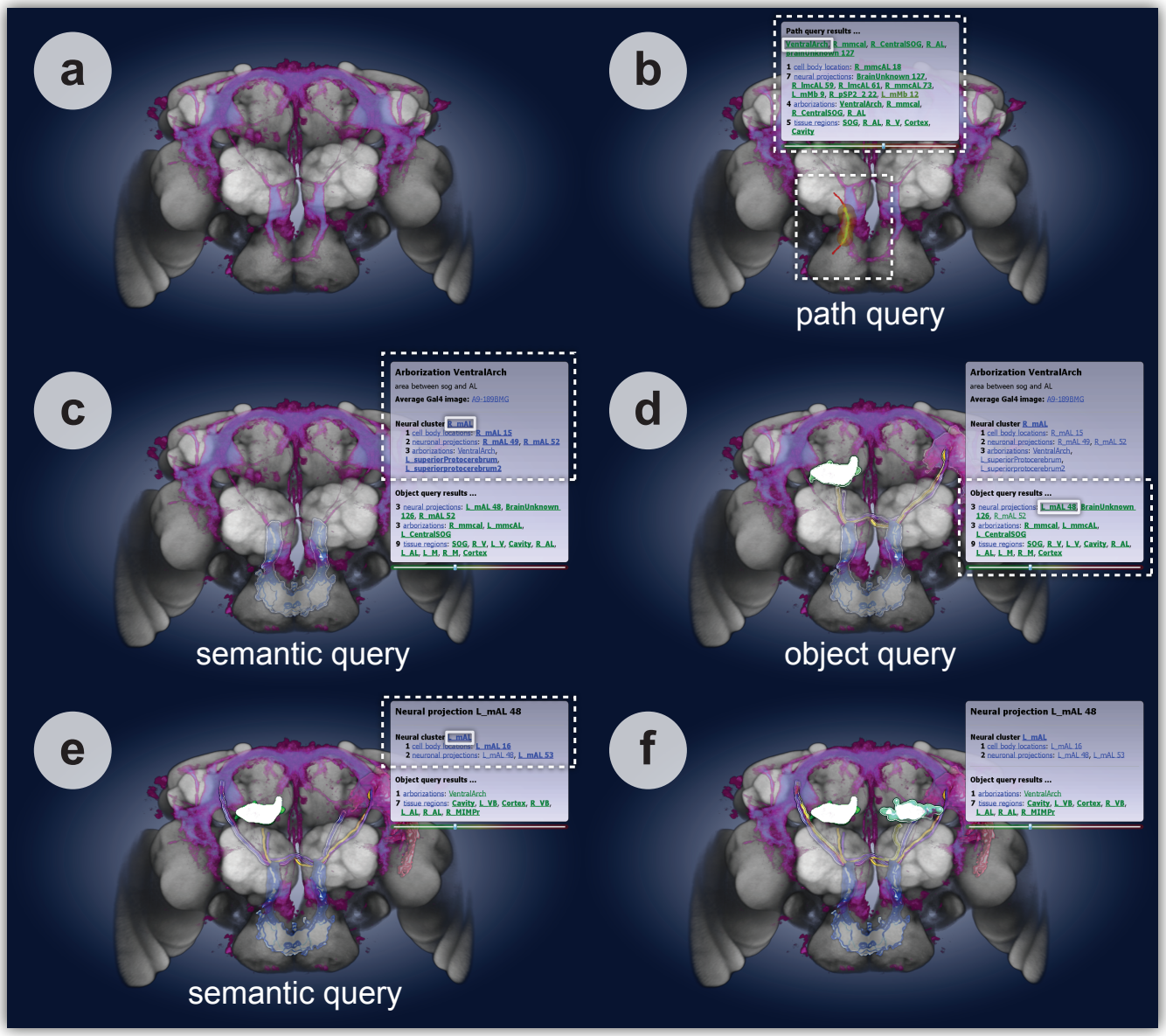


Fig. 6: A simple interaction session using visual queries. (a) Initial state. (b) Path query and selection of an arborization. (c) Selection of the arborization's neuronal type. (d) Object query for nearby neural projection. (e) Selection of the projection's neuronal type. (d) Final state.

cation, two neural projections, and two further arborizations, is loaded. In Figure 6 (d), the object query information is used to load an additional intersecting neural projection. Finally, in Figure 6 (e) the neural projection's associated cluster is loaded using another semantic query resulting in Figure 6 (f). This simple example demonstrates how our approach – using a combination of semantic queries, object queries, and path queries – allows intuitive navigation through complex data.

As *BrainGazer* was developed in an interdisciplinary effort together with domain experts, the described techniques benefitted from constant input by neurobiologists. The possibility of being able to quickly access semantically related or proximal objects was an important goal. We received very encouraging feedback on how the availability of such a system will ease future research. In particular, the concept of presenting query results directly in the visualization window using hypertext labels – as opposed to displaying them in a separate user-interface widget – was appreciated, as it allows the user to remain focused on the visualization. After initial demonstrations of our visual query approach several changes were made based on user comments. For instance, we integrated the distance slider with the contextual label to allow interactive filtering with immediate feedback. We also color-coded the object names in the query results to give a better indication of an object's placement within the query range. Another request was to leave the contextual label visible until explicitly discarded – initially, the label disappeared as soon as an object had been selected for loading. The new behavior allows users to inspect all likely matches sequentially before coming to a conclusion. As a neuronal type, for example, may contain several neural projections which – with slight variation – follow the same path, it is important to view them all when judging connectedness.

We are currently using the techniques presented in this paper to assemble a cellular atlas of the network of neurons that express the *fruitless* gene (*fru*), which have been functionally linked to male courtship behavior [15]. The *BrainGazer* system has proven invaluable in the digital reconstruction of this network, which now comprises over 90 distinct neuronal types. We are now working towards expanding this database to encompass an even broader range of neurons, while also

further developing the database and visualization software. Our aim is to release these tools to the neuroscience research community in the near future, in the expectation that they will similarly facilitate the anatomical exploration of other neuronal circuits in the fly. Although this system has been developed for analysis of the Drosophila nervous system, the computational methods are equally applicable to any species that exhibits a high degree of stereotypy in the cellular architecture of its nervous system, including most other prominent model organisms in neurobiology research.

8 Conclusion

In this paper we presented a system for the interactive visualization, exploration, and analysis of neural circuits based on a neurobiological atlas. We discussed visualization techniques for the effective depiction of multi-channel confocal microscopy volume data in conjunction with segmented anatomical structures. An intuitive visual query approach for navigating through the available data based on semantic as well as spatial relationships was presented. The system was designed and implemented in collaboration with domain experts and is currently in use to assist their research.

In the future we want to extend the scope of this project. Our goal is to build a complete online atlas of neural anatomy. We plan to further develop the *BrainGazer* system so that it fully integrates with this atlas and make it freely available to researchers in the field. In particular, we aim to enable the interactive definition, modification, and annotation of semantic relationships between anatomical structures in the atlas based on visual queries. We envision such a system to facilitate large scale collaborative research in neuroscience.

Furthermore, we hope that making the system available to a larger user base will enable us to study and improve the effectiveness of the presented visualization and interaction techniques. One viable strategy could be the automatic gathering of anonymized usage logs together with evaluation forms directly in the application. Information derived from this data could be employed to optimize the workflow and to identify areas of future research.

Acknowledgments

We thank Wolfgang Lugmayr for help in implementing the *BrainGazer* database. The work presented in this publication was jointly supported by the Austrian Science Fund (FWF) grant no. P18322, the City of Vienna through the VRVis Vienna Spot of Excellence "Visual Computing Vienna", and the Research Institute of Molecular Pathology (IMP) funded by Boehringer Ingelheim.

References

[1] C. Ahlberg, C. Williamson, and B. Shneiderman. Dynamic queries for information exploration: An implementation and evaluation. In *Proceedings of ACM CHI*, pages 619–626, 1992.

[2] Amira. `http://www.amira.com`.

[3] R. S. Avila, L. M. Sobierajski, and A. E. Kaufman. Visualizing nerve cells. *IEEE Computer Graphics and Applications*, 14(5):11–13, 1994.

[4] L. Bertrand and J. Nissanov. The neuroterrain 3D mouse brain atlas. *Frontiers in Neuroinformatics*, 2, 2008.

[5] G. Bezgin, A. Reid, D. Schubert, and R. Kötter. Matching spatial with ontological brain regions using java tools for visualization, database access, and integrated data analysis. *Neuroinformatics*, 7(1):7–22, 2009.

[6] J. Bjaalie. Localization in the brain: New solutions emerging. *Nature Reviews Neuroscience*, 3:322–325, 2002.

[7] A. H. Brand and N. Perrimon. Targeted gene expression as a means of altering cell fates and generating dominant phenotypes. *Development*, 118(2):401–415, 1993.

[8] S. Bruckner and M. E. Gröller. Instant volume visualization using maximum intensity difference accumulation. *Computer Graphics Forum*, 28(3):775–782, 2009.

[9] G. A. Burns, W.-C. Cheng, R. H. Thompson, and L. W. Swanson. The NeuARt II system: A viewing tool for neuroanatomical data based on published neuroanatomical atlases. *BMC Bioinformatics*, 7:531–549, 2006.

[10] W. Cai and G. Sakas. Data intermixing and multi-volume rendering. *Computer Graphics Forum*, 18(3):359–368, 1999.

[11] M. Chicurel. Databasing the brain. *Nature*, 406:822–825, 2000.

[12] P. de Heras Ciechomski, R. Mange, and A. Peternier. Two-phased real-time rendering of large neuron databases. In *Proceedings of International Conference on Innovations in Information Technology 2008*, pages 712–716, 2008.

[13] W. de Leeuw, P. J. Verschure, and R. van Liere. Visualization and analysis of large data collections: A case study applied to confocal microscopy data. *IEEE Transactions on Visualization and Computer Graphics*, 12(5):1251–1258, 2006.

[14] M. Derthick, J. Kolojejchick, and S. F. Roth. An interactive visual query environment for exploring data. In *Proceedings of ACM UIST*, pages 189–198, 1997.

[15] B. J. Dickson. Wired for sex: the neurobiology of drosophila mating decisions. *Science*, 322(5903):904–909, 2008.

[16] Flybrain. `http://flybrain.neurobio.arizona.edu`.

[17] J. Fredriksson. Design of an internet accessible visual human brain database system. In *Proceedings of IEEE International Conference on Multimedia Computing and Systems*, volume 1, pages 469–474, 1999.

[18] V. Gaede and O. Günther. Multidimensional access methods. *ACM Computing Surveys*, 30(2):170–231, 1998.

[19] W. Gilbert. A cube-filling hilbert curve. *The Mathematical Intelligencer*, 6(3):78, 1984.

[20] H. Hauser, L. Mroz, G.-I. Bischi, and M. E. Gröller. Two-level volume rendering. *IEEE Transactions on Visualization and Computer Graphics*, 7(3):242–252, 2001.

[21] H. V. Jagadish. Linear clustering of objects with multiple attributes. *ACM SIGMOD*, 19(2):332–342, 1990.

[22] A. Jenett, J. E. Schindelin, and M. Heisenberg. The virtual insect brain protocol: Creating and comparing standardized neuroanatomy. *BMC Bioinformatics*, 7(1):544–555, 2006.

[23] A. E. Kaufman, R. Yagel, R. Bakalash, and I. Spector. Volume visualization in cell biology. In *Proceedings of IEEE Visualization*, pages 160–167, 1990.

[24] D. Keim. Information visualization and visual data mining. *IEEE Transactions on Visualization and Computer Graphics*, 7(1):100–107, 2002.

[25] J. Kniss, S. Premoze, M. Ikits, A. Lefohn, C. Hansen, and E. Praun. Gaussian transfer functions for multi-field volume visualization. In *Proceedings of IEEE Visualization*, pages 497–504, 2003.

[26] S. H. Koslow and S. Subramaniam, editors. *Databasing the Brain: From Data to Knowledge (Neuroinformatics)*. Wiley, 2002.

[27] A. Kuß, S. Prohaska, B. Meyer, J. Rybak, and H.-C. Hege. Ontology-based visualization of hierarchical neuroanatomical structures. In *Proceedings of Visual Computing for Biomedicine*, pages 177–184, 2008.

[28] C. Lau, L. Ng, C. Thompson, S. Pathak, L. Kuan, A. Jones, and M. Hawrylycz. Exploration and visualization of gene expression with neuroanatomy in the adult mouse brain. *BMC Bioinformatics*, 9(1):153–163, 2008.

[29] T. Luft, C. Colditz, and O. Deussen. Image enhancement by unsharp masking the depth buffer. *ACM Transactions on Graphics*, 25(3):1206–1213, 2006.

[30] R. Maciejewski, S. Rudolph, R. Hafen, A. Abusalah, M. Yakout, M. Ouzzani, W. Cleveland, S. Grannis, M. Wade, and D. Ebert. Understanding syndromic hotspots - a visual analytics approach. In *Proceedings of IEEE Symposium on Visual Analytics Science and Technology*, pages 35–42, 2008.

[31] A. Martin and M. Ward. High dimensional brushing for interactive exploration of multivariate data. In *Proceedings of IEEE Visualization*, pages 271–278, 1995.

[32] N. L. Max. Computer rendering of lobster neurons. In *Proceedings of ACM SIGGRAPH*, pages 241–245, 1976.

[33] A. Maye, T. H. Wenckebach, and H.-C. Hege. Visualization, reconstruction, and integration of neuronal structures in digital brain atlases. *International Journal of Neuroscience*, 116(4):431–459, 2006.

[34] Z. Melek, D. Mayerich, C. Yuksel, and J. Keyser. Visualization of fibrous and thread-like data. *IEEE Transactions on Visualization and Computer Graphics*, 12(5):1165–1172, 2006.

[35] S. Oeltze and B. Preim. Visualization of vasculature with convolution surfaces: method, validation and evaluation. *IEEE Transactions on Medical Imaging*, 24(4):540–548, 2005.

[36] S. R. Olsen and R. I. Wilson. Cracking neural circuits in a tiny brain: new approaches for understanding the neural circuitry of drosophila. *Trends in Neurosciences*, 31(10):512–520, 2008.

[37] W. Pereanu and V. Hartenstein. Neural lineages of the drosophila brain: A three-dimensional digital atlas of the pattern of lineage location and projection at the late larval stage. *The Journal of Neuroscience*, 26(20):5534–5553, 2006.

[38] W. A. Press, B. A. Olshausen, and D. C. V. Essen. A graphical anatomical database of neural connectivity. *Philosophical Transactions of the Royal Society*, 356:1147–1157, 2001.

[39] T. Rohlfing and J. C.R. Maurer. Nonrigid image registration in shared-memory multiprocessor environments with application to brains, breasts, and bees. *IEEE Transactions on Information Technology in Biomedicine*, 7(1):16–25, 2003.

[40] G. Sakas, M. G. Vicker, and P. J. Plath. Visualization of laser confocal microscopy datasets. In *Proceedings of IEEE Visualization*, pages 375–379, 1996.

[41] S. Schmitt, J. F. Evers, C. Duch, M. Scholz, and K. Obermayer. New methods for the computer-assisted 3-D reconstruction of neurons from confocal image stacks. *NeuroImage*, 23(4):1283–1298, 2004.

[42] A. Sherbondy, D. Akers, R. Mackenzie, R. Dougherty, and B. Wandell. Exploring connectivity of the brain's white matter with dynamic queries. *IEEE Transactions on Visualization and Computer Graphics*, 11(4):419–430, 2005.

[43] M. Straka, M. Cervenansky, A. L. Cruz, A. Köchl, M. Šrámek, M. E. Gröller, and D. Fleischmann. The VesselGlyph: Focus & context visualization in CT-angiography. In *Proceedings of IEEE Visualization*, pages 385–392, 2004.