Improving the Visualization of Electron-Microscopy Data Through Optical Flow Interpolation

Lucian Carata*
Faculty of Automatic Control and Computer Engineering
"Gh. Asachi" Technical University of Iasi, Romania

Dan Shao†
Institute of Computer Graphics and Algorithms
Vienna University of Technology

Markus Hadwiger‡
Division of Mathematical and Computer Sciences and Engineering
King Abdullah University of Science and Technology

Eduard Groeller§
Institute of Computer Graphics and Algorithms
Vienna University of Technology

Abstract

Technical developments in neurobiology have reached a point where the acquisition of high resolution images representing individual neurons and synapses becomes possible. For this, the brain tissue samples are sliced using a diamond knife and imaged with electron-microscopy (EM). However, the technique achieves a low resolution in the cutting direction, due to limitations of the mechanical process, making a direct visualization of a dataset difficult. We aim to increase the depth resolution of the volume by adding new image slices interpolated from the existing ones, without requiring modifications to the EM image-capturing method. As classical interpolation methods do not provide satisfactory results on this type of data, the current paper proposes a re-framing of the problem in terms of motion volumes, considering the depth axis as a temporal axis. An optical flow method is adapted to estimate the motion vectors of pixels in the EM images, and this information is used to compute and insert multiple new images at certain depths in the volume. We evaluate the visualization results in comparison with interpolation methods currently used on EM data, transforming the highly anisotropic original dataset into a dataset with a larger depth resolution. The interpolation based on optical flow better reveals neurite structures with realistic undistorted shapes, and helps to easier map neuronal connections.

CR Categories: I.4.10 [Image Processing And Computer Vision]: Image Representation —Volumetric;

I.3.3 [Computer Graphics]: Picture/Image Generation—Display algorithms

Keywords: optical flow, interpolation, volume visualization

1 Introduction

Recent technological developments have increased the interest in the large-scale reconstruction of the neuron-connectivity map [Helmstaedter et al. 2008], in order to capture and understand the structures and neural interactions within the brain. However, given the increased complexity and size of raw data, the structure-identification process is not automatic even when using state-of-art software such as NeuroTrace [Jeong et al. 2010]. Most of the current research in this field proposes semi-automatic methods that aim for reduced interaction with human operators. In this context, visualizing the volumetric data becomes an important requirement, enabling specialists to explore and closely investigate the existing datasets in a 3D environment, in order to adjust or verify the results of the non-supervised processing stages.

Among the available imaging technologies, only electron microscopy (EM) provides the required resolution to resolve densely connected neurons and synapses. Current EM technologies are able to attain resolutions of 3-5 nanometers per pixel in the cutting plane (x-y plane).

Like many other bioscience image capturing systems, EM records image information of the tissue in a slice-by-slice manner. Volumetric datasets are obtained by collecting the EM images of all slices in depth order. Because the slicing process is mechanical, using a diamond knife, the minimum achievable thickness for slices is 20 nm, considering the best performance of the operator. As a result, the resolution achieved in the depth direction is much lower compared to the high resolution of the x-y plane. Figure 1 shows an illustration of the resolutions for a representative EM volume.

These anisotropy characteristics of the dataset, caused by the data acquisition process, are unavoidable. They bring difficulties and obstacles in the subsequent volume analysis and visualization tasks. For instance, the structures of neurons are extensively distorted when being displayed in the ray-casting process. As mentioned, this poses a serious problem to the experts that aim to resolve neuron-to-neuron connection ambiguities or that just want to have a realistic view of the network and of inner cell structures.

The goal of our paper is to perform volume interpolation and use the generated images to increase the resolution in the z direction, while maintaining the structures of interest and introducing less visual artifacts when compared to existing approaches.

Contributions

Our work is centered around two main ideas, both aiming at obtaining better visualizations for EM data, while also being applicable to more general volumetric datasets. We evaluate the feasibility of applying a motion-detection algorithm for the anisotropic spatial volume, and then of using the computed displacement vectors in order to generate multiple interpolated images. This poses a significant advantage in comparison with classical interpolation techniques. It avoids the shadowing effect that appears as a result of mixing data

*e-mail: lucian.carata@cg.tuwien.ac.at
†e-mail: dshao@cg.tuwien.at
‡e-mail: markus.hadwiger@kaust.edu.sa
§e-mail: groeller@cg.tuwien.ac.at

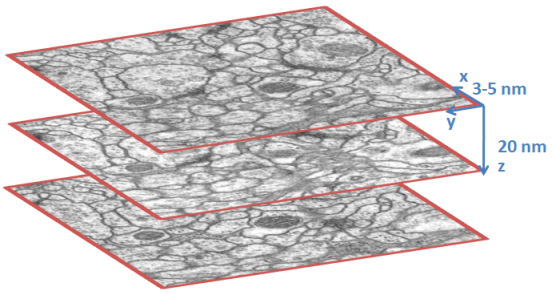

Figure 1: Illustration of resolutions of the EM volume. The resolution of each pixel is $3-5$ nanometers in x and y direction, and a minimum of 20 nanometers in z direction.

from different structures when computing pixel intensities in the new images.

Our approach preserves the topology of the interesting structures, giving a smooth transition between the slices, based on the correct motion detection. This also introduces less artifacts and makes it easier to choose a transfer function for 3D volume-rendering.

The second important improvement refers to the design of our processing pipeline, that is able to integrate additional structural information when it is available. For example, it allows us to mask certain parts of the volume using segmentation data, revealing only some of the neurites and their connections. This is very important for microscopic data, which is extremely dense. The structures are not surrounded by empty space, and furthermore are very difficult to separate based on pixel intensities. This typically causes the structures in the volume to occlude each other, making it hard to obtain good visualizations.

2 Related Work

The use of electron microscopy in imaging neural tissue is not new. However, most of the early work has focused on the manual identification of structures (axons, synapses). Such an approach requires both highly skilled personnel and a huge amount of time. Just to take an example, the connectome (neuron connectivity map) for C. elegans, a worm with only 302 nerve cells, took about 12 years to build [White et al. 1986]. It was therefore clear that in order to aim for the full-brain structural-analysis of more complex organisms, automation methods must be found. With recent increases in computation power, available storage and advances in image processing software, a number of new possibilities are starting to be explored. Scenarios such as neurite segmentation, axon reconstruction or complex neuronal-tissue volume-visualisations are currently being researched. This is done in parallel with the improvements to the image acquisition techniques.

The anisotropy of EM data is recognised as a challenging topic, and some papers have tried to suggest possible solutions. One example [Veeraraghavan et al. 2010] focuses on obtaining images from six different angles for the same tissue slice and increasing the volume's resolution by using sparse tomographic reconstruction. The main limitation of this method is the reduced data acquisition speed, that is caused by the repeated imaging of the same sample. For large volumes of data, this is prohibitive.

In another approach [Jeong et al. 2009], volume visualizations are improved by superimposing 3D reconstructions of particular axons, identified and matched in successive slices by active contour tracking. Also, the paper proposes an on-the-fly linear interpolation technique for reducing the anisotropy of the dataset. However, the linear interpolation usually provides poor results. We are aiming to improve the interpolation quality for better structure identification in the successive ray-casting rendering of the volume. A single pre-processing step is employed to reduce the anisotropy.

Numerous interpolation techniques have been proposed so far, and the vast majority can be classified as scene-based or object-based. Because object-based techniques make use of prior structure information extracted from the datasets (which is not available and expensive to compute), we have focused our attention on scene-based solutions [Grevera and Udupa 1998]. However, finding one that is suitable for EM data is not an easy task, given the typical low signal to noise ratio of the images. Optical flow interpolation is one of the most promising solutions, as it is able to consider the direction of structure movement, even though no information about the existing structures is known. It also avoids mixing data from different structures while computing the output. [Ehrhardt et al. 2006] describes a similar approach, while exploring only a basic implementation of optical flow, that does not consider a multiscale analysis. While this is not an inconvenience for simple datasets, it becomes too unreliable for complex or noisy volume data, such as those obtained through EM scanning.

3 System Pipeline

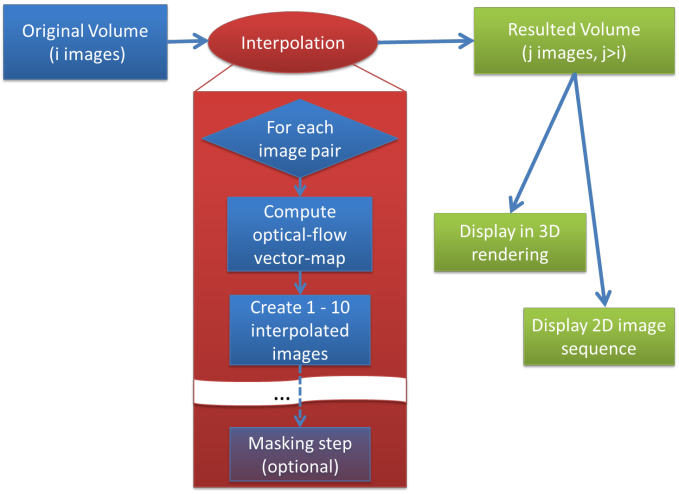

Figure 2: Workflow: EM optical-flow interpolation and visualization

A general workflow of our system pipeline is shown in Figure 2. Let us assume that the original volumetric dataset has a depth equal to i (the volume consists of i slices). After applying our interpolation process, we enlarge the depth to j ($j > i$). The extra $j-i$ slices are generated using the optical flow estimations. Generally, there are two options of demonstrating our interpolated results:

1. Directly display the sequence of interpolated 2D image slices.
2. Display the 3D interpolated volume by ray casting.

The first method has the disadvantage of requiring the expert to build a mental model of the structures present in successive slices in order to have an overall perspective of the dataset. Although there are software packages that support this task, such as TrakEM2 [Cardona et al. 2010], the process is still very tedious. This justifies

our focus on improving volume visualization techniques, allowing an intuitive observation and manipulation of EM data.

However, obtaining good 2D interpolation results does not imply that the ray casting rendering of the same data will give a good volume visualization. The images of brain tissue are very dense and interpolation commonly introduces artifacts that can be observed in the 3D views, considerably reducing the quality of the visualization. This was an additional constraint that was considered when evaluating the optical flow based method.

Furthermore, neuron structures are highly connected and clustered, which makes it nearly impossible to design a transfer function good enough to separate them. When the data is available, we use segmentation masks (the optional masking step mentioned in Figure 2) to reduce the number of features of interest present in the volume so that the end result can be analyzed more clearly in the 3D rendering.

The rest of this section is structured as follows: Section 3.1 describes the main steps of our interpolation process. It starts with the theoretical justification of using motion-detection algorithms on 3D volumetric datasets, interpreting them as 3D motion volumes by considering the z axis as a temporal axis. Then we explain how we apply optical flow to estimate the motion vectors of pixels based on their local gradient with respect to the temporal direction. Motion vectors allow us to calculate the x-y coordinate of each pixel when given a specific temporal position. Section 3.2 describes the method of using segmentation masks to make the volume sparse, for visualizing the interpolated volume in a 3D ray-casting view.

3.1 The Volume Interpolation Process

3.1.1 Interpreting Volumetric Datasets as Temporal Motion Volumes

The choice of using optical flow as an interpolation method can be theoretically justified by reframing the original problem in terms of pixel motion. Therefore, a three dimensional volumetric dataset (2D image + 1D depth) can be seen as a 3D dynamic motion volume (2D image + 1D time), if we consider the z axis to be the temporal axis t. In motion volume representation, 2D cross sections of neuron structures in the x-y plane move their positions and deform their shapes along the temporal axis t.

Assume that we are analyzing a volume with i slices, and that each slice has the resolution of $p \times q$ pixels. Then the volume is understood as a set of $p \times q$ pixels moving along the temporal axis. The goal is to compute the motion of each pixel, and according to this motion, to estimate pixel positions in missing slices.

3.1.2 Computing Optical Flow

Sequences of ordered images allow the estimation of motion as either instantaneous image velocities or discrete image displacements [Beauchemin and Barron 1995]. The optical-flow method intends to compute the motion velocity between two neighbouring images which are taken at time t and $t+\Delta t$, at every pixel position. A pixel from the image with index t at 2D-coordinate (x,y) with intensity $I(x,y,t)$ will move by Δx, Δy, Δt to the next image taken at time $t+\Delta t$. The following constraint equation can be given:

$$I(x,y,t) = I(x+\Delta x, y+\Delta y, t+\Delta t) \tag{1}$$

As the tissue slices are thin, with an interval of 20 nm between them, we can assume the neuron structures do not have big movements between neighboring slices. The above constraint equation can be developed through a Taylor series to get:

$$I(x+\Delta x, y+\Delta y, t+\Delta t) = I(x,y,t) + \frac{\Delta I}{\Delta x}\Delta x + \frac{\Delta I}{\Delta y}\Delta y + \frac{\Delta I}{\Delta t}\Delta t \tag{2}$$

From this equation it follows:

$$\begin{aligned} &\frac{\Delta I}{\Delta x}\Delta x + \frac{\Delta I}{\Delta y}\Delta y + \frac{\Delta I}{\Delta t}\Delta t = 0 \\ &\frac{\Delta I}{\Delta x}\frac{\Delta x}{\Delta t} + \frac{\Delta I}{\Delta y}\frac{\Delta y}{\Delta t} + \frac{\Delta I}{\Delta t}\frac{\Delta t}{\Delta t} = 0 \end{aligned} \tag{3}$$

which leads to the following well-known result:

$$\frac{\Delta I}{\Delta x}V_x + \frac{\Delta I}{\Delta y}V_y + \frac{\Delta I}{\Delta t} = 0 \tag{4}$$

where V_x,V_y are the unknown x and y components of the velocity or optical flow of $P[x,y,t]$ (The pixel located at coordinates $[x,y,t]$), and $\frac{\Delta I}{\Delta x}, \frac{\Delta I}{\Delta y}, \frac{\Delta I}{\Delta t}$, are the derivatives of the pixel intensity in the direction of x, y and t. This is one equation with two unknowns, so

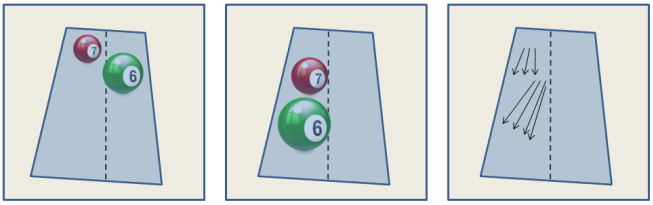

Figure 3: Optical flow presents motions, (a) Time t_1 (b) Time t_2 (c) Optical flow

it is not possible to directly find an unique solution. We make our second assumption that nearby points in the image plane move in a similar manner, which also means the velocity has a smooth distribution. Combining the two constraints, we can calculate the optical flow vectors, as the solution of an energy minimization problem. Our actual implementation (and thus the chosen energy function, that penalizes deviations from the proposed model) is based on the approach taken by [Brox et al. 2004], and for brevity we will omit the full derivation of the results. Figure 3 shows an example of an optical-flow vector-map calculated from two images taken at time t_1 and t_2. In these two images, two balls are changing positions and expand shapes as they approach the camera.

3.1.3 Multi-scale Optical Flow Implementation

The formulation given so far accurately computes the optical flow vectors only if the motion of every given pixel between two slices is constrained within a fixed-size rectangular window around the initial coordinates (defined by $x \pm \Delta x$ and $y \pm \Delta y$). Because the motion speed is not previously known or fixed, it is necessary to implement the equations in a scale-invariant way. This is typically done using a structure that computes the displacements for various scalings of the same image, effectively varying the window size: first, at small scales, global displacements such as translations are detected, while local shape distortions are taken into account at larger scales.

The process starts with building a Gaussian pyramid for each image slice. Let I^0 be the base level of this image pyramid, containing the highest resolution image (the original image). Given the base level,

we apply a Gaussian convolution to blur the image and reduce its size by a certain ratio. We can build the pyramid bottom up until we reach the top which is a single pixel image. However, the number of levels (height) of the pyramid can be varied, as a tradeoff between precision and computation speed.

Let the considered pyramid height be N (a fixed algorithm parameter). We try to compute the optical flow between image a and image b. First we build a pyramid for each image, and obtain two image sets:

1) $\{I_a^1, I_a^2, ..., I_a^i, ...I_a^N\}$, I_a^i is the i^{th} level of the pyramid obtained from image a.

2) $\{I_b^1, I_b^2, ..., I_b^i, ..., I_b^N\}$, I_b^i is the i^{th} level of the pyramid obtained from image b.

For a specific level i, we have a pair of images of equal size from both pyramids. The final optical flow map is computed in two steps:

The first step is to compute optical flow vectors for each pair of images from each level of the pyramid. In our example above, we have N pairs of images with varying sizes. We compute N optical-flow vector-maps for the corresponding image sizes. This particular model, chosen for determining the preliminary optical flow vectors for level i in the pyramid, has the added benefit of being independent in relation to the computations of the other levels, allowing for future optimizations (parallelization). The second step is to iteratively merge all the vector maps top down by successively cumulating them until the base level is reached. Two optical-flow vector-maps are cumulated by expanding the upper level map to the size of the lower level map and by performing simple vector addition. Figure 4 illustrates the process of multi-scale optical flow computation.

The main advantage of using a multi-scale structure is that it correctly computes the collective motion of larger regions. This is usually the case, as pixels in an image region covering the same neuron structure should have similar motions. These vectors can be computed in the top levels of the pyramid and are then accumulated during the top-down vector-merging process. Using a multi-scale pyramid also makes the algorithm more robust to noise and artifacts which cover individual pixels or small regions.

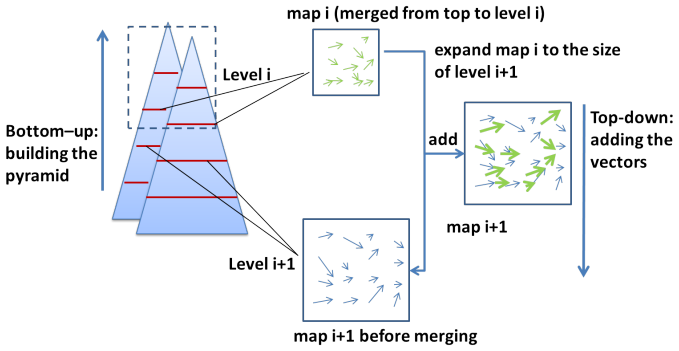

Figure 4: Process of multi-Scale optical flows computation

3.1.4 Creating Interpolated Slices Based on the Optical-Flow Vector-Maps

So far, we have determined the optical-flow vector-maps for each image with respect to its neighbors on the time axis. Figure 5a displays the starting image $I(t)$ and the vector map computed from $I(t)$

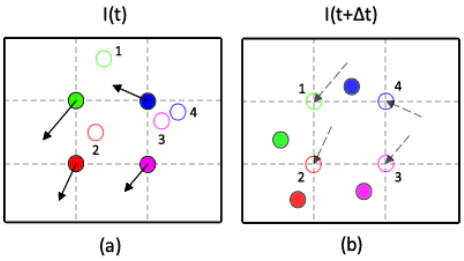

Figure 5: Pixel relations between an existing image and the interpolated image. (a) Existing image $I(t)$ (b) Interpolated image $I(t+\Delta t)$. Solid-filled circles represent the pixel intensities in the existing slice. Hollow circles represent the virtual values that will end up on the discrete pixel grid after applying the computed optical-flow motion-vectors to the original data. These are the values that we need to determine for generating the interpolated image $I(t+\Delta t)$

with respect to $I(t+1)$. In order to generate the image $I(t+\Delta t)$, we need to compute the intensity values of the "virtual" points marked with hollow circles and numbers, that will end up on the discrete pixel grid after applying the motion vectors to the starting image (Figure 5b). These values can be obtained by applying a bilinear interpolation to the surrounding pixel values from the initial image (those marked with solid-filled circles in Figure 5a). The values are "virtual" because in $I(t)$ they might represent points that are placed between two existing pixels. This situation appears since the determined optical-flow vectors are real (and usually do not indicate motions across an integer number of pixels but contain sub-pixel fractions).

The computed optical flow vectors define the displacements between $I(t)$ and $I(t+1)$. In order to determine the motion from $I(t)$ to $I(t+\Delta t)$ we assume a linear variation of the size of the vectors between the original slices. We therefore adjust the size of the motion vectors based on the actual value of Δt, in order to control the positioning of the generated image between $I(t)$ and $I(t+1)$. This provides the base for inserting multiple interpolated images between each pair of slices from the original dataset.

3.2 Sparse Representation

As mentioned previously, a second difficult problem that has to be solved in order to enable a good visualization of neural EM datasets is that each slice contains a large number of structures (neurites, mitochondria - see Figure 6), which densely cover the volume. All the structures of the same type have similar physical properties (translated into similar grey values in the EM datasets). It is very difficult to choose a small relevant fraction for volume visualization by means of a certain transfer function. Therefore, the structures are normally occluding each other, making visual analysis difficult or impossible.

Our proposal for tackling this problem is to apply a pre-processing masking step for the data, in order to make the volume sparse. The mask is a binary image (see Figure 6), containing the value 0 in areas that need to be discarded and 1 for areas of interest. Upon convolution with the EM images, the mask selects only a subset of the structures of interest for volume visualization. Other mask values can also be considered if one wants to apply different weights to some of the features in order to make them more easily selectable with a transfer function.

Ideally, the mask should be dynamically generated from segmentation data - after an expert has indicated which structures should be visualized. His choice could then be modified without re-generating the interpolated images, in order to analyze different areas in the volume. When changing the structures that must be displayed, it is sufficient to apply a fast convolution step with the new mask. While this whole process introduces the additional requirement of having data segmentation available, there is no need to perform it in real time, while visualizing the volume. As a consequence, interactive visualization can be achieved.

Currently, automatic or semi-automatic segmentation techniques for neural EM data are emerging [Turaga et al. 2009], and we expect that our approach could be applied in real-case scenarios. However, when obtaining segmentation data is not an option, the masks could be generated based on algorithms that are also less computationally expensive (for example, edge detection).

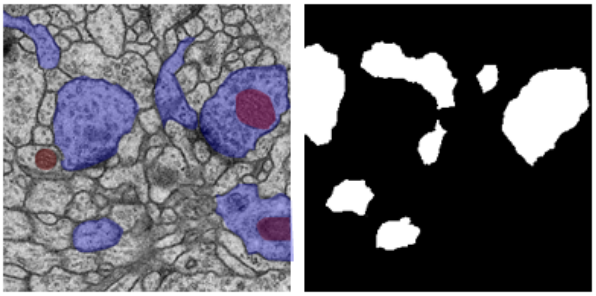

Figure 6: Typical EM image and a corresponding binary mask, that is based on segmentation results. In the image on the left, some of the structures have been highlighted: in blue - a small fraction of the neurites; in red - mitochondria, inner cell structures. The mask selects which neurites will be shown in the volume rendering.

4 Experimental Results

The proposed processing pipeline has been designed for the purpose of improving the visualization of electron microscopy (EM) volumes, taking into account all the particularities of this type of data. By focusing on the end result rather than on that of particular sub-stages, we are taking into account all the variables affecting the outcome. This allows us to better evaluate appropriate interpolation algorithms, in order to help reduce the anisotropy of the volume while also preserving the structures of interest within.

The following section presents the experimental results for two volumetric datasets, `lobster` and `drosophilaTEM` [Cardona et al. 2010], as they are rendered in VolumeShop [Bruckner and Gröller 2005]. `Lobster` is a well known CT (isotropic) dataset, which we have used as a base for evaluating the interpolation quality. Because EM datasets are inherently anisotropic, there is no ground truth to precisely determine how the actual isotropic result should look like. Therefore, for obtaining a quantitative evaluation of the optical flow algorithm, we have used the `lobster` data. First, the volume was made anisotropic artificially, by eliminating half of the image slices. Then, we have applied our interpolation technique and compared each generated image with the corresponding removed slice.

The second dataset, `drosophilaTEM`, contains genuine EM data, anisotropic in the z direction. The results for this dataset were obtained by using exactly the same algorithms and parameter values as the runs for the `lobster` dataset. The additional masking step is used from the pipeline, in order to make the volume sparse and enable a better focus on the interesting regions.

4.1 Experiment setup

If no interpolation techniques are applied, the resulting volume will usually be seen highly distorted along the depth direction, making existing structures and connections particularly hard to identify (see Figure 7). Typically for EM, the slice spacing is about ten times higher than pixel spacing, making it impossible to obtain good visualization results using a direct anisotropic volume rendering, such as the one proposed by [Li et al. 2009]. The mentioned method has been designed only for small differences between slice spacing and pixel spacing.

In order to correct this problem, images must be added to the volume to fill in the missing data between slices. Acceptable visualization results can be obtained by inserting a small number of interpolated images (anywhere between 1 and 10), in order to better visualize structural details along the z direction.

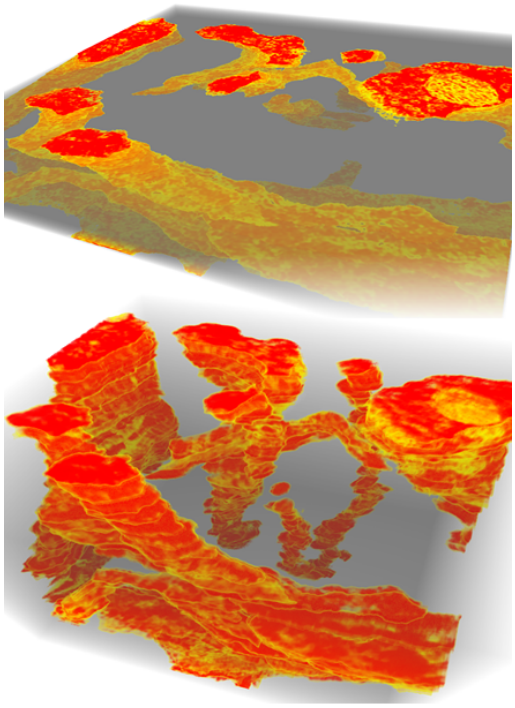

Figure 7: The volume before and after the interpolation, with segmentation masks applied. The top image represents a direct rendering of the anisotropic volume. The bottom image is a rendering of the same dataset, after 8 interpolated images have been added between each pair of images from the original data.

In our first experiment, we have added a single interpolated slice between successive images, in both datasets. For evaluation, we have generated linear interpolation results as well as multi-scale optical-flow results (our proposed method). For the `lobster` dataset, the interpolation was performed after making the volume anisotropic, removing half of the original images. In order to make a visual comparison of the results possible, we have selected a transfer function for visualizing each original dataset, and kept it consistent for all the experiments.

A second experiment was performed to evaluate the ability of the proposed method to reduce the anisotropy of the volume. In this case, we gradually increased the number of inserted images, from 1 to 8. The same number of intermediate images was generated using linear interpolation for comparison. However, the artifacts introduced by the linear interpolation for more than 2 intermediate images made the neural structures extremely hard to visualize.

4.2 Results

4.2.1 Experiment 1

Figure 8 presents a reference comparison between the optical-flow interpolation result and the original (isotropic) `lobster` dataset. The results show that our method preserves the ability to visually identify all the important structures in the volume, while introducing no major artifacts. The number of levels in the multi-scale pyramid that is built for computing the optical flow was set to 4, although higher values were also tried. The conclusion was that the exponential increase in time for running the algorithm computing more scale levels does not justify the minor improvements in the final result. This is also supported by the quantitative evaluation of interpolation results given in Table 1, where the difference in root mean square error (RMS) between optical flow using 4 pyramid levels and the one using 15 levels is less than 0.11%. The RMS error was computed by performing a pixel-wise comparison between the original lobster slices and the interpolated slices from the same position within the volume.

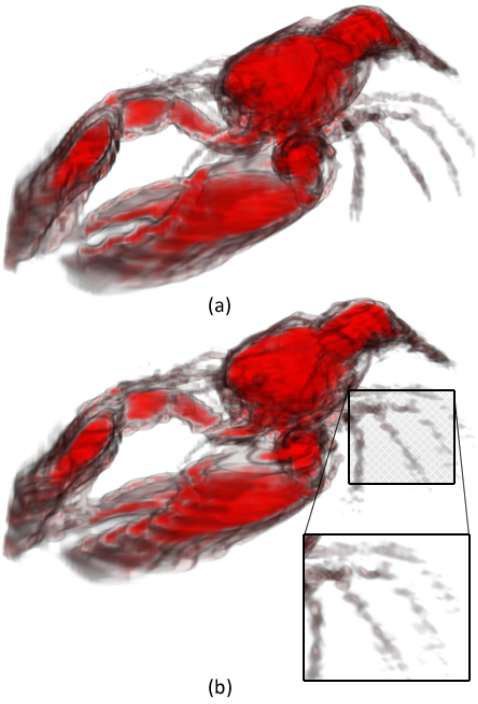

Figure 8: (a) Original `lobster` dataset; (b) Optical-flow interpolation (Experiment 1). The volume contains only half of the original data; the other slices are being interpolated. Note the discontinuities in the legs region.

The images also show an area where the results could be improved: thin structures that have a fast changing rate between consecutive slices, in the direction of anisotropy (for example, the legs section of the lobster). This can be explained considering that the proposed optical-flow algorithm deals with fast changing structures by using the multi-scale approach. Unfortunately, very thin structures are disappearing in the upper levels of the generated image pyramid, reducing the quality of the results. However, this is not a severe limitation considering our final goal of rendering EM data volumes: the high EM resolution assures that most of the regions of interest have sufficiently large cross sections so that optimal interpolation results can be computed.

For comparison, the linear interpolation result is displayed in Figure 9. A big problem with linear interpolation in volume rendering is that for obtaining the end-result, it may combine data from totally different structures. This makes it very difficult for the human operator to do the selection of features by means of a certain transfer function. Additionally, fringing artifacts are also introduced in the resulting visualization. Optical flow manages to overcome these problems by combining data only between corresponding structures. Effectively, the interpolation is done "in the direction" of structure motion.

Because for the `lobster` dataset we have the ground truth data available, we were also able to perform some quantitative measurements regarding the precision of the proposed interpolation method. The results can be seen in Tables 1 and 2. When compared to the linear interpolation, the method based on optical flow constantly gives lower RMS errors.

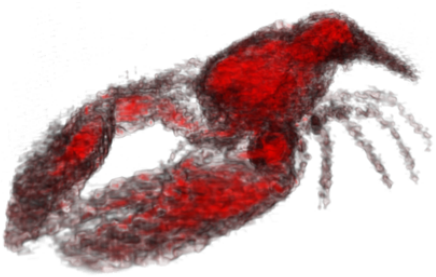

Figure 9: The linear interpolation result for the `lobster` dataset

	RMS	Normalized RMS	Std Dev
Optical Flow, 4 lev	21.8	8.5%	2.8
Optical Flow, 15 lev	21.5	8.4%	2.9
Linear Interpolation	23.4	9.2%	1.5

Table 1: Quantitive interpolation evaluation using Root Mean Square (RMS) error when adding 1 interpolated slice to the anisotropic `lobster` dataset

A side by side evaluation of the two interpolation techniques can also be seen in Figure 10, for the masked `drosophilaTEM` dataset. We observe that the optical-flow algorithm performs very good, allowing clear structure identification and introducing very few artifacts. In contrast, the linear interpolation combines the inner structures, making them difficult to visualize (as shown in the detail views).

4.2.2 Experiment 2

For the second experiment, both linear interpolation (with the insertion of equally spaced interpolated slices) and optical-flow results

	RMS	Normalized RMS	Std Dev
Two interpolated slices			
Optical Flow	26.2	10.3%	3.2
Linear Interpolation	31.7	12.4%	2.7
Five interpolated slices			
Optical Flow	36.3	14.2%	3.2
Linear Interpolation	43.0	16.8%	3.6

Table 2: Quantitive interpolation evaluation using Root Mean Square (RMS) error when adding 2 and 5 interpolated slices to the anisotropic `lobster` dataset

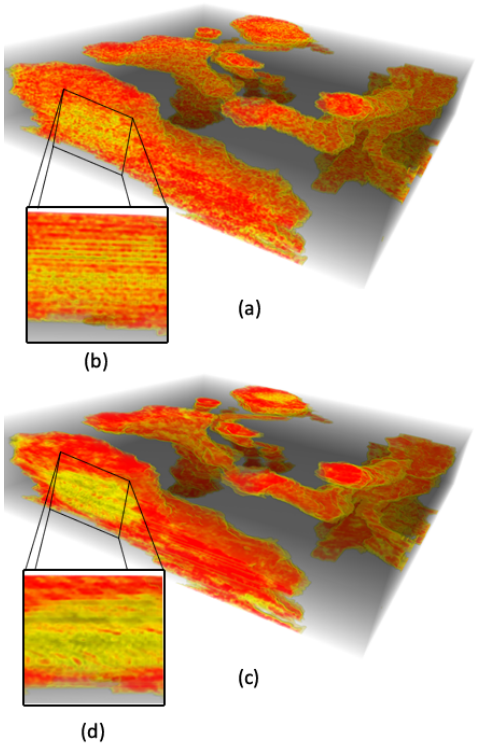

Figure 10: (a) Linear interpolation volume rendering for the `drosophilaTEM` dataset after the masking step. Interesting structures are selected using a transfer function: axons - red; mitochondria - yellow; (b) detail showing visible artifacts. (c) Optical flow interpolation result using the same transfer function. (d) detail from optical-flow interpolation

were computed. However, the linear interpolation output becomes too distorted, so the visualization is really poor. We will skip presenting this result, and instead focus on the volume renderings that use optical flow as the interpolation technique. As can be observed in Figure 11, for the `drosophilaTEM` dataset, there is little degradation in quality when increasing the number of slices, with the considerable advantage of making the volume isotropic. The vertical structures (axons) and their interconnections can now be visualized as shapes that have realistic proportions.

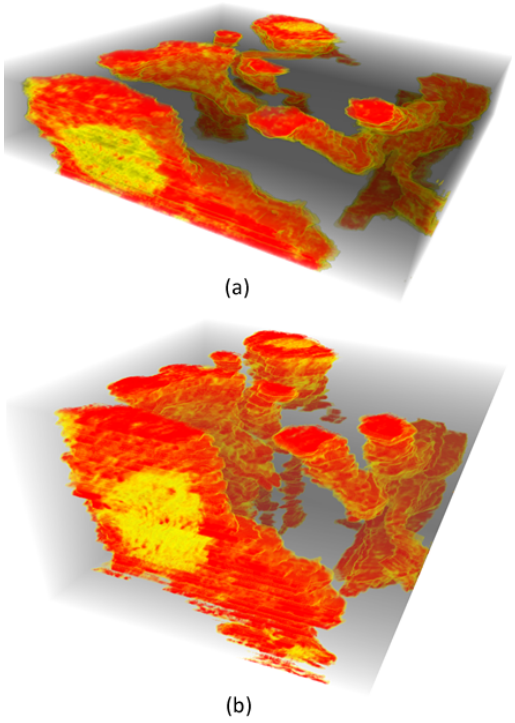

Figure 11: Two visualizations of the volume after optical-flow interpolation when (a) 2 and (b) 8 images are generated for each pair of images from the original data. The anisotropy of the volume is significantly reduced, with little impact on the quality of visualization.

4.3 Performance Considerations

One of the main reasons for continuing to use linear or trilinear interpolations for volume visualizations is that they can be computed very fast with the hardware available today. The implementation of the proposed optical-flow interpolation was done in MATLAB and has focused more on the actual processing pipeline and the quality of the output. Currently, obtaining the interpolation results for a dataset of size $256 \times 256 \times 32$ takes about 20 minutes, scaling linearly with the number of image pixels and volume depth. However, we have not neglected the performance constraints when developing our processing pipeline and choosing the algorithms. The multi-scale optical flow algorithm can be relatively easily parallelized and implemented on the GPU. This remains as a part of future work, but optimal implementations can provide significant running time improvements.

5 Conclusion

In general, interpolation is required whenever the acquired data is not at the desired level of discretization. Bioscience imaging systems usually acquire the data in slice-by-slice order, which yields different sampling rates in x, y, and z directions. Looking into medical imaging systems, sparse sampling is commonly performed in clinical examination (for example, in MRI), although isotropic cross-sectional images are obtainable by using the most advanced facilities. Volume interpolation plays an important role in these tasks, and is usually done as a pre-processing step. This paper evaluates the multi-scale optical flow algorithm as a possible interpola-

tion method. The motion estimations are then used to create new images, increasing the depth resolution of the original data. Our implementation focuses on obtaining good visualizations for EM images of brain tissue.

However, for applying this method to different datasets and for improving our current results, we will have to investigate solutions for dealing with thin structures that have a fast changing rate between consecutive slices. This as well as obtaining efficient implementations remains as future work.

6 Acknowledgements

The work presented in this paper has been partially funded by the WWTF in the scope of the SCALE-VS project (ICT08-040)

References

BEAUCHEMIN, S. S., AND BARRON, J. L. 1995. The computation of optical flow. *ACM Comput. Surv. 27*, 3, 433–466.

BROX, T., BRUHN, A., PAPENBERG, N., AND WEICKERT, J. 2004. High accuracy optical flow estimation based on a theory for warping. In *Computer Vision - ECCV 2004*, T. Pajdla and J. Matas, Eds., vol. 3024 of *Lecture Notes in Computer Science*. Springer Berlin / Heidelberg, 25–36.

BRUCKNER, S., AND GRÖLLER, M. E. 2005. Volumeshop: An interactive system for direct volume illustration. In *Proceedings of IEEE Visualization 2005*, 671–678.

BURT, P. J., EDWARD, AND ADELSON, E. H. 1983. The laplacian pyramid as a compact image code. *IEEE Transactions on Communications 31*, 532–540.

CARDONA, A., SAALFELD, S., PREIBISCH, S., SCHMID, B., CHENG, A., PULOKAS, J., TOMANCAK, P., AND HARTENSTEIN, V. 2010. An integrated micro- and macroarchitectural analysis of the drosophila brain by computer-assisted serial section electron microscopy. *PLoS Biol 8*, 10 (10), e1000502.

EHRHARDT, J., SRING, D., AND HANDELS, H. 2006. *Interpolation of Temporal Image Sequences by Optical Flow Based Registration*. Informatik aktuell. Springer Berlin Heidelberg.

GREVERA, G., AND UDUPA, J. 1998. An objective comparison of 3-d image interpolation methods. *IEEE Transactions on Medical Imaging 17*, 642 – 652.

HELMSTAEDTER, M., BRIGGMAN, K., AND DENK, W. 2008. 3D structural imaging of the brain with photons and electrons. In *Current Opinion in Neurobiology*, 18(6):633–641.

JEONG, W.-K., BEYER, J., HADWIGER, M., VAZQUEZ, A., PFISTER, H., AND WHITAKER, R. T. 2009. Scalable and interactive segmentation and visualization of neural processes in EM datasets. *IEEE Transactions on Visualization and Computer Graphics 15*, 1505–1514.

JEONG, W.-K., BEYER, J., HADWIGER, M., BLUE, R., LAW, C., VAZQUEZ, A., REID, C., LICHTMAN, J., AND PFISTER, H. 2010. Ssecrett and neurotrace: Interactive visualization and analysis tools for large-scale neuroscience datasets. *IEEE Computer Graphics and Applications 30* (05/2010), 58–70.

LI, G., XIE, M., PENG, G., AND WU, B. 2009. Research of rendering anisotropic volume data directly based on the shear-warp algorithm. In *ESIAT (2)'09*, 427–430.

TURAGA, S. C., BRIGGMAN, K. L., HELMSTAEDTER, M., DENK, W., AND SEUNG, H. S. 2009. Maximin affinity learning of image segmentation. *The Computing Research Repository abs/0911.5372*.

VEERARAGHAVAN, A., GENKIN, A. V., VITALADEVUNI, S., SCHEFFER, L., XU, S., HESS, H., FETTER, R., CANTONI, M., KNOTT, G., AND CHKLOVSKII, D. 2010. Increasing depth resolution of electron microscopy of neural circuits using sparse tomographic reconstruction. *IEEE Computer Society Conference on Computer Vision and Pattern Recognition 1*, 1767–1774.

WHITE, J. G., SOUTHGATE, E., THOMSON, J. N., AND BRENNER, S. 1986. The Structure of the Nervous System of the Nematode Caenorhabditis elegans. *Royal Society of London Philosophical Transactions Series B 314* (Nov.), 1–340.