Advanced Volume Painting with Game Controllers

Veronika Šoltészová*
Vienna University of Technology

Maurice Termeer†
Vienna University of Technology

M. Eduard Gröller‡
Vienna University of Technology

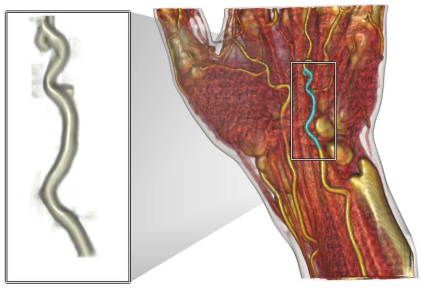


Figure 1: Segmentation of a blood vessel produced with our painting tool.

Abstract

Volume painting is an interactive segmentation technique for volumetric datasets. While volume painting facilitates quickly creating segmentations, e.g., for illustration purposes, segmenting features precisely is often problematic. The volume painting system we present addresses two common problems. First, we introduce several data-dependent painting mechanisms to make precisely segmenting features of interest easy. Second, we use game controllers such as a joystick and a gamepad to create a simple user interface to our system while still providing full control over the large amount of parameters of the painting mechanism. We demonstrate our work giving several examples and compare the effectiveness of the various brushes. We prove that our system is intuitive to use with preliminary user testing. The results indicate that our volume painting framework is an effective, interactive segmentation tool.

CR Categories: I.3.1 [Computer Graphics]: Hardware Architecture—Input devices I.3.4 [Computer Graphics]: Graphics Utilities—Paint systems

Keywords: Volume painting, game controller.

1 Introduction

Volume painting is a manual segmentation technique for volumetric datasets using an interface similar to painting. Features can be segmented by brushing over them in a 2D rendering of the volumetric dataset. The user has adequate control of the properties of the brush, the viewpoint, and the transfer function employed for rendering the dataset. Volume painting can be an intuitive and efficient way to segment features in a dataset. The accuracy of the produced segmentations is often limited, making volume painting more suitable for creating illustrations than for most other industrial applications.

In this work we present a volume painting system that improves on common approaches in two ways. First, we explore advanced brushes that use properties of the volume data in a neighborhood of the brush. By using edge-detection filters, weighted averaging and thresholding we are able to produce more accurate segmentations while requiring less interaction. Figure 1 indicates the effectiveness of our system, where a vessel was approximately segmented by brushing over it. Opposed to existing approaches, features surrounding the vessel are not included in the segmentation.

Second, we provide an intuitive user interface by using game controllers. The large variety of controls provides the user with full control over the brush properties and the viewpoint from a single device. Traditional systems controlled via a keyboard and a mouse require the user to remember a lot of shortcuts. Most interaction devices used in virtual environments provide a high accuracy, but are often expensive. We instead focused on consumer-level devices, as these are more affordable to the average user and are more widely used outside laboratory conditions. Their ergonomic design offers simple and intuitive steering in three-dimensional environments.

In this paper, we describe our volume painting system using the following structure. In Section 2, we discuss related work. Section 3 is dedicated to volume painting. It includes a technical introduction to volume painting and discusses brushes based on Gaussian, edge-detection and bilateral filters. In Section 4 we introduce our approach of interaction by game controllers and explain improvements of the user interface. In Section 5 we present the results achieved using our tool. In Section 6 we discuss the results of the preliminary user testing. We conclude the paper and outline future work in Section 7.

*vero@cg.tuwien.ac.at
†maurice@cg.tuwien.ac.at
‡groeller@cg.tuwien.ac.at

2 Related work

In the communities of computer graphics, user interfaces and virtual reality, researchers have developed different setups of 3D interaction in the context of painting and editing.
Hanrahan and Haeberli [Hanrahan and Haeberli 1990] first described an interactive 3D painting using a 2D mouse interface. They introduced a direct approach to WYSIWYG (What You See Is What You Get) painting. The user controls the brush with a tablet and directly applies different pigments onto 3D shapes. Their pigments present material properties such as surface roughness, specularity, and thickness. Based on the idea of direct 3D painting, Gregory et al. [Gregory et al. 2000] presented an integrated system for 3D model editing and painting of 3D polygonal meshes with a haptic interface. They proposed the interactive 3D painting tool *InTouch*, which enables painting directly onto the surface of the model.
Kim et al. [Kim et al. 2003] introduced haptic editing of decorations. They used a volume fill algorithm to identify texturized 3D triangles to be painted within a brush. They determine the color of a texel by a function similar as proposed by Hanrahan and Haeberli [Hanrahan and Haeberli 1990] and by Gregory et al. [Gregory et al. 2000]. The brush specification is a function of parameters such as the brush size, brush color, background-color and distance of the brush volume during triangle rasterization.
Wang and Kaufman [Wang and Kaufman 1995] studied interactive sculpting of 3D objects. They developed a real-time volume sculpting tool where the user forms the three-dimensional models by moving voxels. They created realistic objects from natural materials like marble and wood.
Bruckner and Gröller [Bruckner and Gröller 2005] used volume painting as part of the open source volume visualization framework VolumeShop, an interactive system for volume illustration. They developed a painting tool, where the user can easily add or remove objects from a selection set.
Shirai et al. [Shirai et al. 2007] and Wischgoll et al. [Wischgoll et al. 2004] prove that game controllers are suitable for navigation and interaction tasks. Shirai et al. described the drawing software Papier-Poupee-Painter (PPP) which should stimulate children's imagination. They emphasize the use of consumer-level hardware, like standard video game controllers. The painting tool was a WiiRemote dressed as a doll (fr. poupée) and simulated brush strokes. Wischgoll et al. presented a visualization and exploration system for cardiovascular trees. To improve control and navigation, they employed different standard game controllers for changing the viewpoint, controlling the zoom factor, panning and rotation, as well as navigation through the coronary model.
Earlier work indicates that game controllers can cause a considerable improvement in navigational tasks and steering. We therefore designed an intuitive user interface for our volume painting system based on game controllers.

3 Volume painting

Volume painting is a technique used for, e.g., illustration purposes or as a simple segmentation method. Volume painting aims to interactively segment the object around the center of the brush $\vec{p}$. Similar to traditional painting on a real canvas, multiple voxels are selected by one brush stroke. While painting, the user marks the underlying voxels and segments the object.
Our implementation of volume painting works as follows. We consider a click at position $\vec{p_0}$ on the screen as one brush stroke. We cast a ray from $\vec{p_0}$ into the volume and determine the first non-transparent intersection voxel at the position $\vec{p}$ in the volume space. Our compositing scheme is based on the over-operator as introduced by Porter and Duff [Porter and Duff 1984]. We composite the brush function $b : \mathbb{Z}^3 \rightarrow \mathbb{R}$ with the data volume v_d. The brush function b is related to the Gaussian function[Weisstein 2009]. We define a similar function $\widehat{G}_{\lambda,\sigma,\vec{p}}(\vec{x})$ which assigns a real scalar value to a given vector $\vec{x}$ referring to a position in the data volume v_d (see Equation 1). Additional parameters λ and σ are positive real constants and $||\vec{x}||$ denotes the norm of $\vec{x}$. Parameters λ and σ are user-defined and we interpret them as brush properties: σ as size and λ as density.

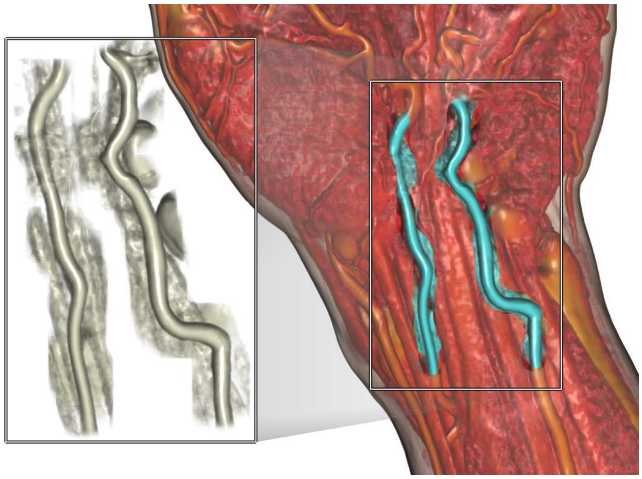


Figure 2: Results of painting with a Gaussian brush: Only vessels were intended to be painted.

$$\widehat{G}_{\lambda,\sigma,\vec{p}}(\vec{x}) = \lambda \exp(-\frac{||\vec{x}-\vec{p}||^2}{2\sigma^2}), \quad \lambda,\sigma \in \mathbb{R}^+. \tag{1}$$

We are interested in evaluations of $\widehat{G}$ only in a neighborhood around the position $\vec{p}$, i.e., within the bounding box of the brush. This assumption leads us to the definition of the brush function b as in Equation 2. This function depends on parameters σ, λ, $\vec{p}$, but for simplicity we keep the short notation b. In the remainder of this work, we refer to the 3D subspace where the brush function b is non-zero within the bounding box of the brush. Under active brush area we understand all voxels within the bounding box of the brush.

$$b(\vec{x}) = \begin{cases} \widehat{G}_{\lambda,\sigma,\vec{p}}(\vec{x}), & \text{if } max(\vec{x}-\vec{p}) \leq \sigma \\ 0, & \text{otherwise} \end{cases} \tag{2}$$

The composition of b and v_d modifies the resulting selection volume v_s. The resulting v_s is a fuzzy set of voxels with a real value range $[0.0, 1.0]$ where 0.0 means "not selected" and 1.0 means "fully selected" [Bruckner and Gröller 2005]. We refer to the value of a voxel in the selection volume also as to the "fuzziness".
An artist uses several types of brushes while painting on canvas. They differ in shape, size, and density of hair. Kim et al. [Kim et al. 2003] use some of these as properties for their brushes. Similarly, we experimented with different types of brushes. Composition methods of b and v_d are specific for different types of brushes described further in this section.

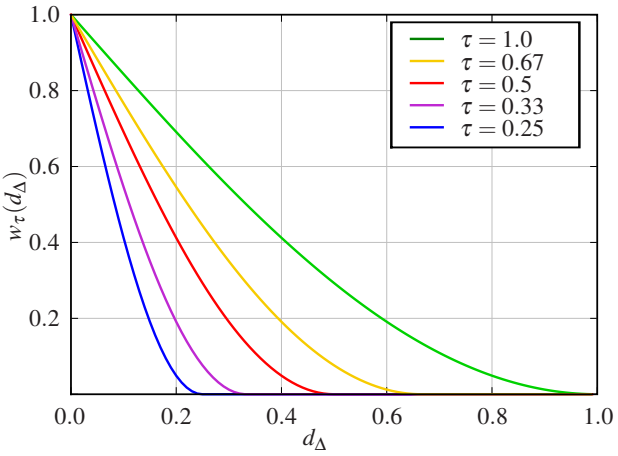


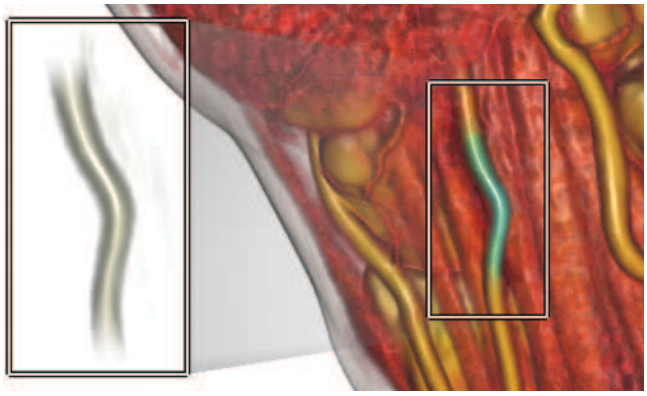
Figure 3: Weighting function $w_\tau(d_\Delta)$ for various values of τ.

3.1 Gaussian brush

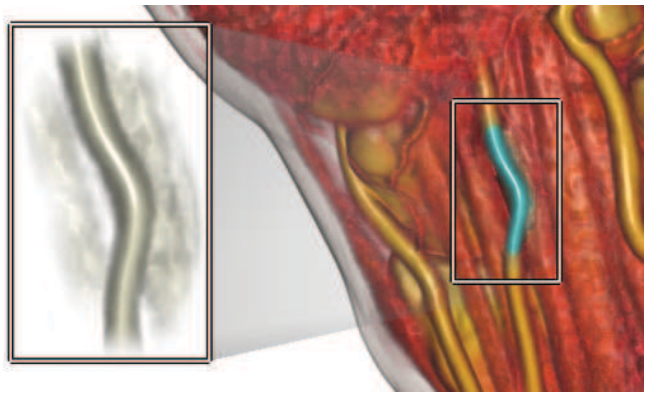
The simplest composition method of function b and the volume v_d is multiplication as implemented in the work of Bruckner and Gröller [Bruckner and Gröller 2005]. The product of the densities of elementwise multiplication of v_d and b is added to the selection volume v_s. Repeatedly brushing the same region leads to an accumulation of the multiplication product. Thus regions with higher accumulated values have higher fuzziness. The user sees the rendered current selection set so she can interactively modify it. For the volume rendering of the selection volume we use a transfer function which gives high opacity to fully selected voxels and makes non-selected voxels transparent. We call this brush the Gaussian brush, as the multiplication of the brush function b with volume v_d leads to a contribution with a Gaussian distribution. This brush does not take the density values of the voxels in its neighborhood into account. Figure 2 illustrates the typical problem with painting with a Gaussian brush. As the brush is independent of the data, it is difficult to select features with sharp edges in the data, such as the vessels in Figure 2. To solve this problem, we designed edge-filtering brushes which we discuss in Section 3.2. By painting with a Gaussian brush, all voxels located inside the bounding box of the brush are included into a selection. In Figure 2 the targeted objects are two vessels. We observe a large portion of surrounding tissue visible in the selection. To solve this problem, we designed bilateral brushes which we discuss in Section 3.3.

3.2 Edge-filtering brushes

With a Gaussian brush it is difficult to select features with fine structures. We solve this problem by the design of a brush that enhances edges through the use of an edge-detection filter.
In this work we distinguish edges and borders. We define edges as a local property of the volumetric function to be present in regions with a high gradient magnitude. Edges express the local properties of the volumetric function. Borders convey boundaries between objects. As borders show the intensity and color changes, we can often expect that edges identify borders. Edge-detection is closely related to discrete first or second derivative approximations as edges are areas within the volume where the volumetric function shows large variations in density values. Convolution with a kernel of a gradient operator is a widely used method for approximating derivatives. In our approach, we use the Sobel and Laplace kernels [Sonka et al. 1993] as gradient operators. We use the



(a)



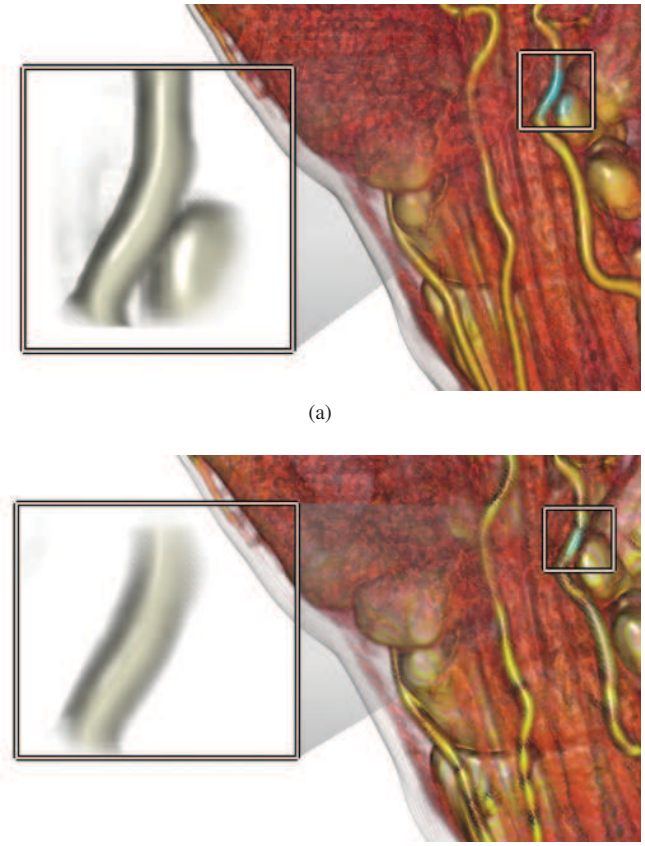
(b)

Figure 4: Results for different weighting functions: (a) $w_{0.25}(d_\Delta)$ and (b) $w_{0.4}(d_\Delta)$.

3D edge-filtering operator, i.e., Sobel or Laplace operator, as 3D discrete functions $o : \mathbb{Z}^3 \rightarrow \mathbb{R}$ similar to the operators' 2D versions [Sonka et al. 1993]. We convolve the data volume v_d in the active paint area with the operator o: $v_e = v_d * o$. As opposed to simple elementwise multiplication used for the Gaussian brush, we multiply the brush function b with the edge-enhanced data volume v_e.

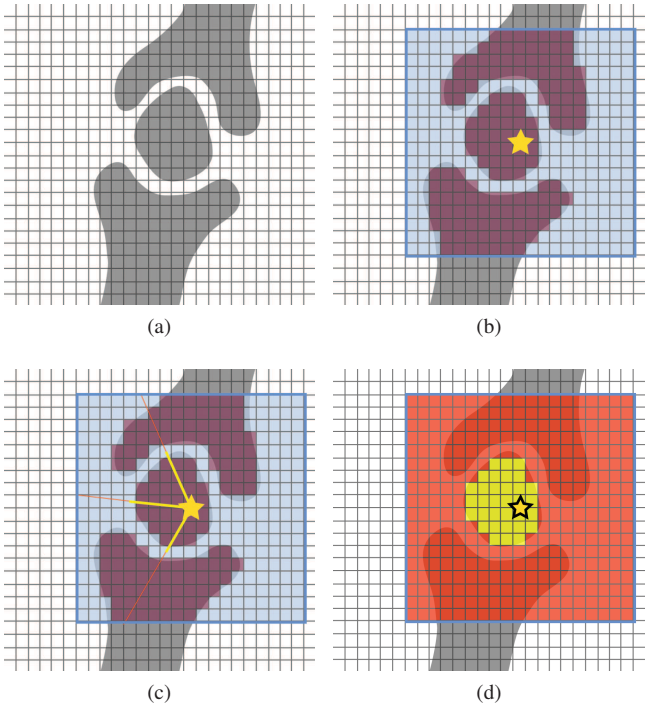
3.3 Bilateral brushes

As can be seen in the zoom-in in Figure 2, all voxels in the active brush area contribute to the selection. Thus, voxels which do not belong to the same object appear in the selection. This is because the object properties are not influencing the weighting factor. Our goal is to suppress the selection of the objects which are not targeted. We introduce a method inspired by bilateral brushes as in Guarnieri et al. [Guarnieri et al. 2006].
By bilateral brush for volume painting, we understand a brush whose paintings depend on two mutually independent factors. In this section, we discuss bilateral brushes which depend on the distance from the brush position (Section 3.3.1) as the Gaussian brush and additionally on an object property such as density (Section 3.3.2).



(a) (b)

Figure 5: Results of painting with (a) difference-weighted bilateral brush and (b) two-pass difference-weighted bilateral brush.



(a) (b) (c) (d)

Figure 6: Simplified binary masking of voxels for a cross section of a data volume. (a) An example of a two-dimensional cross section where grayscale values represent the precomputed differences $w_{0.25}(d_\Delta)$. (b) All voxels inside the brush bounding box with $w_{0.25}(d_\Delta) < t$ are flagged with one (purple). Parameter t signifies a defined threshold value. (c) Shooting rays from the center of the brush. (d). All voxels which are part of the selected object of interest are yellow, others are red.

3.3.1 Difference-weighted bilateral brush

When the user starts painting, she aims to segment the object intersecting the point p in the data volume which we previously interpreted as the center of the brush. Thus, we assume the voxel density in p is the property of the object of interest. Sampling the density value only at the center of the brush is very noise sensitive. Instead, we compute a mean voxel density d_μ of a small number of samples around the center of the brush. Voxel densities of our data range in $[0,1]$ and so does d_μ. We assume that an object of interest has voxel densities within a certain range and that the mean density d_μ belongs to this range. Furthermore, we assume that voxel densities d for which the difference $d_\Delta = |d - d_\mu| < \tau$ holds are also part of the object of interest. Parameter τ specifies the threshold and ranges in $[0,1]$.

For difference-weighted bilateral brushes, we adapt the brush function used for the Gaussian brush. We multiply it by a weighting function $w_\tau(d_\Delta)$ that is continuous and monotonically decreasing within its definition range $[0,1]$ and with a value range $[0,1]$. Furthermore, we require for the minimal difference $w_\tau(d_\Delta = 0) = 1$ and for differences greater than the threshold $w_\tau(d_\Delta \geq \tau) = 0$. Equation 3 provides a weighting function for differences $d_\Delta \in [0,1]$. Figure 3.1 illustrates the graph of the function $w_\tau(d_\Delta)$ for different values of τ.

$$w_\tau(d_\Delta) = \begin{cases} 1 - \sin(\frac{\pi d_\Delta}{2\tau}), & \text{if } d_\Delta < \tau \\ 0, & \text{if } d_\Delta \geq \tau \end{cases} \quad (3)$$

Our results indicate that $w_{0.25}(d_\Delta)$ generally leads to good results. In comparison to the function $w_{0.4}(d_\Delta)$ which is zero for differences $d_\Delta > 0.4$, the result for $w_{0.25}(d_\Delta)$ contains less background voxels as it decreases more rapidly and is zero for $d_\Delta > 0.25$ (see Figure 4). However, a threshold $\tau < 0.25$ causes that the painting process slows down too much. The value $\tau = 0.25$ is a good trade-off for our difference-weighted bilateral brush.

3.3.2 Two-pass difference-weighted bilateral brush

The difference-weighted brush causes objects with similar density values located close to each other to all be included in the selection, even if some of them are not intended to be. We solve this problem with the introduction of a two-pass difference-weighted bilateral brush. Figure 5 demonstrates the difference between the single-pass and two-pass difference-weighted bilateral brush.

The computation is done in two passes. First, we precompute weights $w_\tau(d_\Delta)$ for every voxel according to Equation 3. During the first pass, we also assign a flag, i.e., a mask bit, to each voxel. Voxels with weights $w_\tau(d_\Delta)$ below a certain threshold $t \in [0,1]$ are assigned a zero-flag. All other voxels are assigned a one-flag. The examples presented here were produced using a value 0.6 for t. Figure 6 demonstrates the steps of the marking algorithm on the structure given in Figure 6a. In the next step voxels with $w_{0.25}(d_\Delta)$ below the threshold t are marked purple as shown in Figure 6b. The square represents the bounding box of the brush. The blue parts of the brush active area represent zero flagged

(a) (b)

Figure 7: Game controllers: Logitech™ Joystick 3D Force Pro (a) (image taken from www.logitech.com) and Logitech™ Gamepad WingMan Rumblepad (b) (image taken from www.amazon.de).

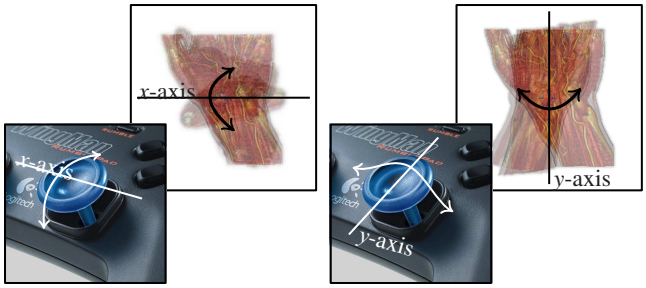
voxels. Further, rays are then cast from the center of the brush as illustrated in Figure 6c. We investigated the following criterion for each voxel: if the connecting line between the voxel v and the center of the brush p intersects a voxel with a zero-flag, we assign it a zero selection value, and terminate the ray. In Figure 6c, rays are yellow, until such intersection is found. The final state where all voxels in the cross-section have a flag is illustrated in Figure 6d. Convex structures can be selected entirely with a single brush operation anywhere within the structure. For concave structures, only voxels are selected for which all voxels along a straight to p belong entirely to the same structure. In practice this difference is hardly noticeable, as concave structures can easily be segmented using multiple brush-strokes.

4 Game controllers for volume painting

Game controllers are widely employed to improve the user interfaces to computer graphics applications. A good example is the gaming industry, which has encouraged the development of several different types of controllers. Most of these offer high precision in a wide variety of applications. This led us to explore the applicability of these controllers to volume painting. We have built a user interface for our volume painting system using both a joystick and a gamepad. Although the controllers are different in shape, the degrees of freedom they offer are similar which is shown in Figure 7. In Sections 4.1 and 4.2 we give a short description of the game controllers we explored. In Section 4.3 we explain in detail how we used their features for our application, especially for volume painting.

4.1 Joystick

The first device we investigated is a Force 3D Pro joystick from Logitech which is shown in Figure 7a. The important controls for interaction with our application are a handheld stick, buttons, a slider, and a directional pad. The latter is also called D-pad, coolie hat or a view switch. Modern joysticks provide three degrees of freedom. The first two degrees of freedom correspond to the left-right and front-back motion of their handheld stick and the third degree of freedom corresponds to twisting the stick.
In the game industry, joysticks are very popular for flight simulators, where the three axes of movement relate to the aircraft's pitch, roll and yaw. The employment of a joystick is not limited to video games. It is also widely used in the machine industry, for example for elevators and cranes.



(a) (b)

Figure 8: Rotation around the x-axis (a) and the y-axis (b) with an analog stick.

4.2 Gamepad

The second device we explored is a WingMan Rumblepad gamepad from Logitech. A gamepad is currently the most frequently used game controller for game consoles. Figure 7b shows a description of our gamepad and its parts. Its interface includes nine buttons, a throttle, a D-pad and two analog sticks. An analog stick is a short pivoted stick which a user controls with her thumbs. Analog sticks cannot be twisted as opposed to joysticks, which creates the biggest difference between our two interfaces.

4.3 Interaction Details

Our volume painting system can be completely controlled with a game controller using the DirectInput API [Microsoft 2006]. Our application provides interaction mechanisms to manipulate the viewing parameters, including rotation, panning and scaling. This is essential in most of the three-dimensional visualization systems. The location and size of the brush can also be manipulated using the controller. Continuous parameters including rotation, panning, scaling, and location of the brush are controlled by pivoting a stick. To facilitate a high amount of precision, the speed at which the respective parameter is adjusted is controlled by a slider. In the following paragraphs the interaction mechanisms are explained in detail.

Rotation Both devices, the joystick and gamepad, have either an analog stick or a pivoted stick that offers two or three degrees of freedom (DOF, see Figure 7). Our joystick has a pivoted stick with three DOF and our gamepad has two analog sticks with two DOF. The movement to rotation mapping is simpler for sticks with three DOF. We are able to map the movements onto a 3D vector $\vec{v_m}$. We interpret the three components of $\vec{v_m}$ as angles of rotation around respective coordinate axes. This is advantageous because the user achieves a rotation around three different axes with a single hand movement.
With only 2 DOF, e.g., analog sticks of our gamepad, we are limited only to two axes of rotation per analog stick. Assuming that users operate each analog stick with one hand, we assigned two rotation axes to the right hand and one rotation axis to the left hand. Front-back and left-right rotations correspond to rotations around the x-axis and the y-axis respectively which is shown in Figure 8. The left hand serves the rotation around the z-axis.

Panning For panning, the direction, in which the user pushes a

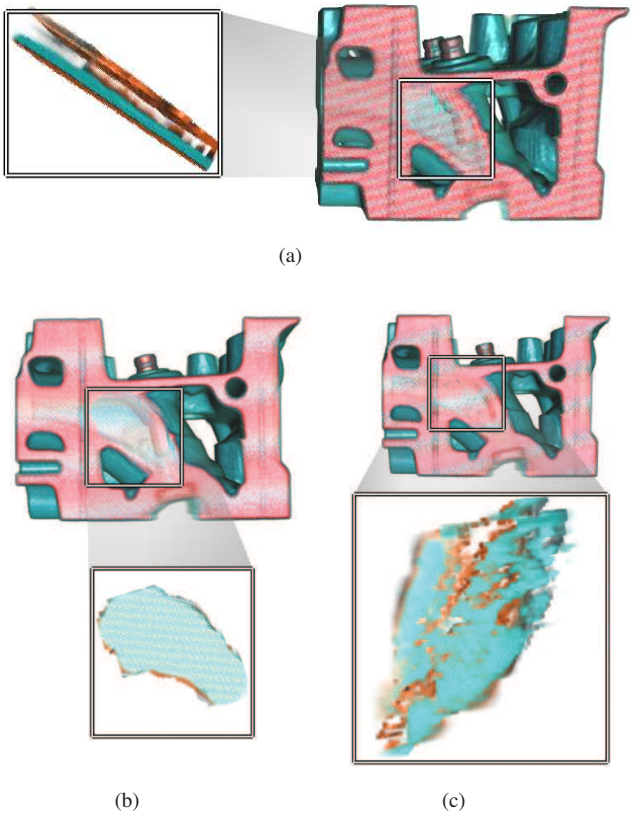


Figure 9: Segmentations defined with volume painting and edge-filtering brushes: (a) the side-view on the selection set defined by a Laplace brush, (b) the frontal-view on the selection set defined by a Laplace brush, (c) the frontal-view on the selection set defined by a Sobel brush.

stick of the joystick or an analog stick, determines the translation vector in screen space.

Scaling We only provide support for uniform scaling. Similarly as for rotation and for panning, the scaling factor is influenced by the deviation of the stick on the device from its neutral position.

Volume painting We extended the basic features of steering and included the volume painting module for interaction with game controllers. When painting is performed using a mouse, the motion is primarily supplied through the wrists and lower forearm. This may provide less precision than using a combination of the wrists and fingers, as is for example the case with writing text. Additionally, our volume painting system requires a significant number of controls: set-up of the orientation, position and size of the volumetric data, and brush parameters such as size, density and type. Our goal is to provide a tool which makes the navigation simple, and precise and whose controls are easy to remember.
Our framework offers two modi: painting and erasing. In the painting mode, the user creates a segmentation with the active brush. The brush is represented by a cursor whose position is controlled by the game controller in a similar fashion to panning. As long as the painting mode is activated, brush strokes are triggered in constant time intervals. Based on the cursor position we compute the brush position as discussed in Section 3. The user gets a direct feedback about how the current selection looks like. She can decide to save it or erase parts of it. The erasing mode is very similar to the painting modus. While painting, the brush function is multiplied with the data volume and added to the selection. Contrarily, during erasing the brush function is subtracted from the selection.

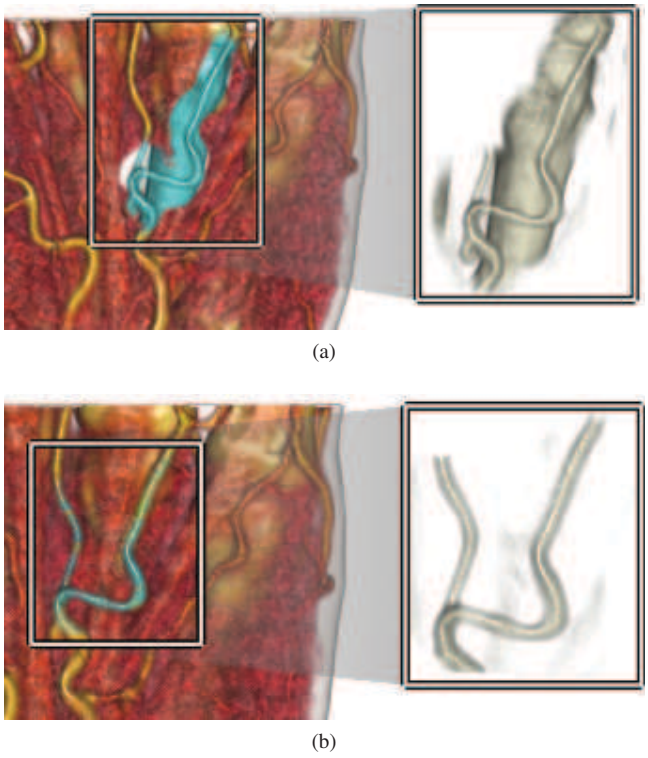


Figure 10: Comparison of selections sets defined by: (a) the Gaussian brush and (b) by the two-pass difference-weighted bilateral brush.

The user can access all functionality we discussed from her game controller. Selecting brushes and changing all their parameters such as density and radius is possible using the game controller. Our set of brushes integrated it into a volume rendering framework and the game controller interface is a compact tool for quick manual segmentation.

5 Results

In order to evaluate our brushes we used each of them to segment blood vessels in a CT scan of a human hand. These vessels are fine structures with well defined edges, but their complex geometry often makes them hard to segment. Segmenting these vessels allowed us to evaluate the advantages and disadvantages of each brush, as well as which brush is best suitable for segmenting fine structures. In Figure 2 we give an example where the Gaussian brush is impractical for selecting thin and delicate objects. We recommend to use this brush only for rough segmentations. Compared with edge-filtering brushes, the segmentations produced with a Gaussian brush include large portions of neighboring features. On the other, Sobel edge-filtering brushes are noise sensitive. Many axis-aligned edges are false positives depending on the used Sobel operator. In Figure 9c, we observe edges which appeared due to the noise-sensitivity of Sobel operators. In Figures 9a and 9b, it can be

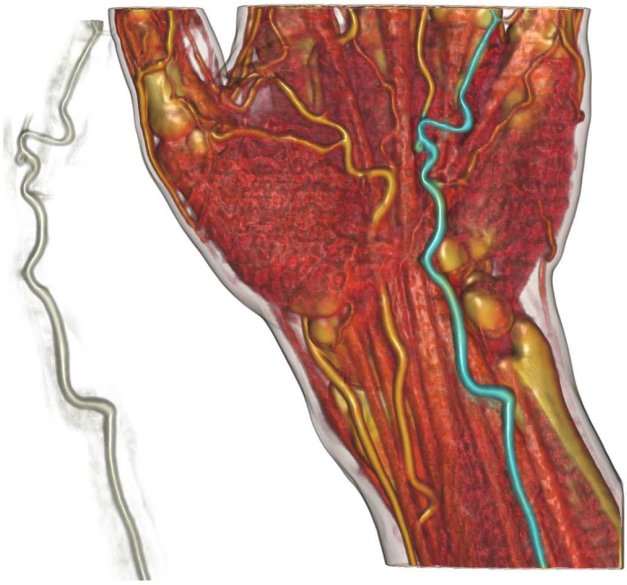


Figure 11: A screen capture of a segmentation of a blood vessel produced by a user who had almost no previous experience with game controllers.

seen that the Laplace operator produces less false positives. However, it did not separate two objects which are both in the brush active area. In this case, we suggest a bilateral brush.

In out experience bilateral brushes better separate targeted objects from the background. The number of misclassified voxels is significantly smaller than for the Gaussian brush which is shown in Figure 10a. Figure 10b illustrates the painted selection set defined by the two-pass difference-weighted bilateral brush which gave the best segmentations.

While testing the system we achieved interactive frame rates ranging from 17.5Hz to 22.5Hz. The worst performance was achieved while working with the most computationally costly two-pass difference-weighted bilateral brush. Our testing workstation was equipped with an Intel® 6600 Core Duo™ CPU 2.4GHz, 2.0GB RAM and a NVIDIA® 8800 GTX GPU. We used datasets with $244 \times 124 \times 257$ voxels and brushes with $50 \times 50 \times 50$ voxels. Our painting system is an extension of the VolumeShop framework [Bruckner and Gröller 2005] thus the interface logic and performance are very similar.

6 Preliminary user testing

To evaluate our user interface using game controllers, we conducted an informal evaluation experiment with two expert and three non-expert users. While the number of users is too small for a proper evaluation study, we have obtained preliminary results that help us identify the key strengths and weaknesses of our system. The results of our tool are summarized below.

Each user was asked to accomplish three tasks. First users had to brush over a blood vessel which is enhanced in Figure 11. They had to precisely steer the path of the cursor along this vessel ten times. We measured the time in seconds they needed for each attempt with a gamepad and with a mouse. The learning curve (see Figure 12a) indicates that the time needed to accomplish the task with a gamepad significantly decreases with attempts. A user who is familiar with game controllers could accomplish the task within

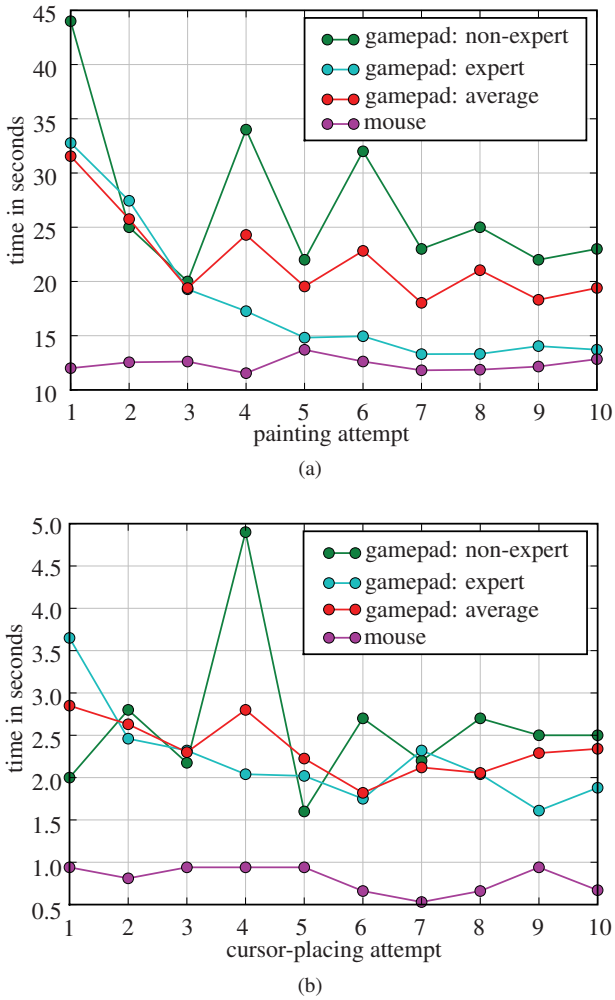


Figure 12: Learning curves for (a) brushing over a blood vessel and for (b) cursor placing with a gamepad and a mouse. The "non-expert" curves indicate results for the user with almost no previous experience with game controllers. The "expert" curves represent results for the user with the most experience with game controllers. The "average" curve shows average over results of all testers. The "mouse" curve gives comparable times when a mouse was used to perform the corresponding task instead of a gamepad.

$t_m = 10.32s$ with a mouse and within $t_g = 13.30s$ with a gamepad, which is 128.9% of t_m. Both times correspond to the best times an expert user achieved with a corresponding input device. We statistically evaluated last five attempts of the expert user. The mean time $13.86s$ to accomplish the task with a gamepad was 112.03% of time needed to accomplish the same task with a mouse.The second task was to move the cursor from a given position A to a given position B as quickly as possible. In Figure 12b we give a learning curve which indicates the time measured in seconds to accomplish this task and comparable values to accomplish the same task with a mouse. An expert user accomplished this task with a gamepad in $2.2s$ and with a mouse in $0.74s$ which are the mean values over her ten attempts. The third task was to perform the most accurate segmentation of a given blood vessel. After a short training, the users achieved precise segmentations with the volume painting tool and the gamepad. Figure 11 demonstrates that even a user with almost no previous experience with game controllors can produce an accurate segmentation using our painting tool.

During the informal evaluation and discussions with users, they gave us constructive feedback concerning the usability of our tool. We were recording in writing their suggestions, problems and comments. Their suggestions concerned above all the usage of the controls which enabled us to improve our tool. They appreciated that our painting tool works without a keyboard or without a complicated graphical user interface. Our testing showed that controlling the interaction with the tip of a finger is easier than interacting by employing the whole wrist. Thus, volume painting feels more natural with an analog stick of a gamepad than with a joystick. Some users would also prefer having a twistable analog stick on the gamepad because twisting corresponds to rotations with respect to the z-axis in an intuitive way. Also a touch sensor on the analog stick would be advantageous because the users would like to avoid keeping the tracking activated by pressing a button. The results of our informal testing convinced us that game controllers are a viable substitution of a mouse with many advantages and suitable for volume painting purposes.

7 Conclusions and future work

We described a new approach for volume painting, using advanced brushes and game controllers. We believe that game controllers offer an intuitive environment for interacting with visualization software and especially for volume painting. In combination with an appropriate brush, we achieved good results with an intuitive user interface.

A Gaussian brush is suitable only for quick and approximate segmentations for illustration purposes, where high precision is not necessary. For precise segmentation, we propose edge-enhancing brushes and bilateral brushes. For the extraction of fine surfaces, e.g., painting a thin plate, an edge-enhancing brush is more suitable. To focus on an object of interest, bilateral brushes are appropriate. We combined user interaction and image processing methods to show that volume painting is able to produce precise segmentations.

We explored game controllers as an interface to volume painting. Our preliminary user testing indicates that this can be an interesting and intuitive interface. The results suggest that the system as a whole is a good segmentation tool.

The use of game controllers brings a new perspective of how to interact with visualization software. We emphasize, that it is easier to launch different actions without remembering a shortcut for each of them. The use of analog sticks or of a pivoted stick provides at least as many degrees of freedom as a mouse. Additionally, we exploited the second thumb joystick or the twist of the pivoted stick to achieve an additional degree of freedom which a mouse does not offer.

Future improvements to our system may include a support for a WiiRemote controller. A WiiRemote, as used in the work by Shirai et al. [Shirai et al. 2007], has drawn a lot of attention lately. It may be worthwhile to explore as an interface to volume painting, as it offers more possibilities for movement tracking. Moreover, the game controllers could be employed not only for volume painting, but for sculpting-based segmentation as well. Finally, improving the interaction and experimenting with different game controllers could provide new dimensions to the interaction with visualization software and interactive segmentation techniques.

8 Acknowledgments

The authors wish to thank the anonymous reviewers for their comments, to the testers for their patience, and to Ivan. This work was supported by the COMRADE project funded by Philips Healthcare, Best, The Netherlands.

References

BRUCKNER, S., AND GRÖLLER, E. 2005. Volumeshop: An interactive system for direct volume illustration. In *Proceedings IEEE Visualization Conference 2005 (Vis '05)*, 671–678.

GREGORY, A., EHMANN, S., AND LIN, M. 2000. *InTouch*: Interactive multiresolution and 3D painting with a haptic interface. In *Proceedings IEEE Virtual Reality Conference 2000 (VR '00)*, 45–52.

GUARNIERI, G., MARSI, S., AND RAMPONI, G. 2006. Fast bilateral filtering for edge-preserving smoothing. In *Electronics Letters No. 7*, vol. 42, 396–397.

HANRAHAN, P., AND HAEBERLI, P. 1990. Direct WYSIWYG painting and texturing on 3D shapes. In *Proceedings of the SIGGRAPH Conference*, vol. 24, 215–223.

KIM, L., SUKHATME, G., AND DESBRUN, M. 2003. Haptic editing for decoration and material properties. In *Proceedings of the ASME 11th Annual Symposium on Haptic Interfaces for Virtual Environment and Teleoperator Systems*, 213–213.

MICROSOFT, 2006. Microsoft DirectX DirectInput API. `http://msdn.microsoft.com`.

PORTER, T., AND DUFF, T. 1984. Compositing digital images. In *ACM SIGGRAPH Computer Graphics*, vol. 18 (3), 253–259.

SHIRAI, A., GRESLIN, E., AND RICHIR, S. 2007. Wiimedia: motion analysis methods and applications using a consumer video game controller. In *Proceedings of the 2007 ACM SIGGRAPH symposium on Video games*, 133–140.

SONKA, M., HLAVAC, V., AND BOYLE, R. 1993. *Image Processing, Analysis and Machine Vision*, first ed. Chapman and Hall.

WANG, S., AND KAUFMAN, A. 1995. Volume sculpting. In *Proceedings of the symposium on Interactive 3D graphics*, 151–157.

WEISSTEIN, E., 2009. Gaussian function. From MathWorld – A Wolfram Web Resource. (Accessed March, 2009). `http://mathworld.wolfram.com/GaussianFunction.html`.

WISCHGOLL, T., MORITZ, E., AND MEYER, J. 2004. Force-feedback-enhanced navigation for interactive visualization of coronary vessels. In *Proceedings IEEE Visualization Conference 2004 (Vis '04)*, 32–32.