CEIG - Spanish Computer Graphics Conference (2018)
I. García-Fernández and C. Ureña (Editors)

Design and implementation of a volumetric merging tool

I. Vidaurre-Gallart, A. Sújar, M. García

URJC Universidad Rey Juan Carlos, España
GMRV Grupo de Modelado y Realidad Virtual

Abstract

The study of the human brain is one of the biggest research challenges nowadays. Thanks to the impressive improvement in technology, great advances are being achieved in the study of the microanatomical structures of the neurons, which are the main nervous system processing cells. The study of those structures requires to capture high-resolution images. In such cases, due to the resolution required, processing of complete neurons is unfeasible. The purpose of this project is to provide the neuroscientists with a tool that allows them to visualize and reconstruct complete neurons from multiple high-resolution sections. To this end, we developed a tool capable of loading different volumetric data sets and merging them into a new volume. A rendering engine was designed to support several volumetric rendering modes. The graphical user interface allows the user to interact with the volumes in intuitive way and facilitates volume stitching task. Furthermore, the implementation makes use of GPU to improve the application.

CCS Concepts

•***Computing methodologies*** → *Volumetric models;*

1. Introducción

Tradicionalmente en la informática gráfica, los objetos 3D se modelan mediante mallas formadas por facetas poligonales 2D que representan el contorno del objeto. Esta aproximación ofrece ventajas como que se evalúa la iluminación únicamente en la superficie, sin tener en cuenta la interacción de la luz en el interior del objeto. Frente a esta técnica se encuentra el renderizado volumétrico que hace referencia a un conjunto de técnicas para generar imágenes a partir de conjuntos de datos tridimensionales. Habitualmente los objetos volumétricos se representan como una pila de imágenes 2D (*stack* de imágenes). Si las imágenes 2D que forman el *stack* de imágenes son observadas por separado, no se tiene una visión de conjunto necesaria para comprender los datos, por lo que la motivación principal del renderizado volumétrico es presentar estos datos de manera comprensible para el usuario.

Cuando se desea analizar datos con alto nivel de detalle, es común extraer sólo porciones de los datos y analizarlos por separado como por ejemplo cuando las imágenes se obtienen mediante microscopía confocal con gran resolución. La motivación de este trabajo es visualizar y reconstruir este tipo de datos para recuperar una visión global.

Las contribuciones de este trabajo son una herramienta de fusión de volúmenes y una librería de renderizado volumétrico. Esta librería soporta lo siguientes modos de renderizado basados en *ray casting*: renderizado basado en el cálculo analítico de una isosuperficie, renderizado basado en función de transferencia y renderizado basado en función de transferencia con iluminación. Para enfatizar las diferentes características y mejorar la visualización también soporta la funcionalidad de planos de recorte y se ha diseñado el cauce para permitir el renderizado de objetos opacos, de esta manera, también se ha incluido el modo de visualización de rodajas del volumen (renderizado basado en *slices*).

2. Antecedentes

Principalmente, las técnicas de renderizado volumétrico se pueden dividir en métodos directos o indirectos; mientras que los métodos indirectos (ej.: *marching cubes* [LC87]) intentan extraer explícitamente superficies geométricas a partir de los datos, los métodos directos muestran los datos del voxel al evaluar un modelo óptico que describe cómo el volumen emite, refleja, dispersa y absorbe la luz [Max95].

Todos los algoritmos de renderizado volumétrico directos evalúan la integral de renderizado volumétrico de una manera o de otra. El modelo óptico de emisión-absorción nos lleva a la siguiente integral:

$$C = \int_{s_0}^{D} C(s) e^{-\int_s^D k(t)dt} ds \qquad (1)$$

siendo las propiedades ópticas k (coeficiente de absorción) y C

DOI: 10.2312/ceig.20181154

(color emisivo) y la integración desde el punto de entrada del volumen $s = s_0$ al punto de salida $s = D$. Esta descripción asume que el volumen y el mapeado a las propiedades ópticas son continuos. En la práctica, los datos volumétricos son discretos y la evaluación de la integral se aproxima numéricamente mediante un esquema de composición.

En la literatura encontramos que los métodos directos de renderizado volumétrico se pueden agrupar en cuatro grandes categorías [Rao05]: *ray casting* [HW91], basados en *slices* [CCF94], *splatting* [Wes90] y *shear-warp* [DCH88]. Entre ellos, los más comúnmente utilizados son el método de *ray casting* volumétrico que genera al menos un rayo por cada pixel que recorre el volumen y recoge los datos volumétricos a intervalos regulares y las técnicas basadas en *slices* en las que se proyectan las imágenes 2D que forman el volumen en el plano de imagen de acuerdo a un esquema de composición.

En el ámbito de la visualización y análisis de imágenes científicas, es necesario recurrir a mecanismos de fusión para diferentes fines, como vemos en la revisión del estado del arte de [JD14]. Concretamente, en la reconstrucción de trazados de neuronas se encuentra software como NeuroLucida 360 [MBF] y Vaa3D [PRL*10] que automatiza o semi-automatiza como parte de este proceso de reconstrucción, el proceso de fusionado de dendritas (*stitching*). [BHAA*13] presenta un trabajo completo de adquisición y reconstrucción. En la etapa de adquisición de las secciones de datos en el microscopio se guardan metadatos que permiten la reconstrucción automática. Para realizar la fusión usan o bien un filtro bilinear implementado en GPU o filtros de mayor orden implementados en el *shader* de fragmentos. En [PST09] presentan un método de reconstrucción automática basado en la transformada de Fourier que consigue la mejor configuración para el conjunto de imágenes 3D tras la comparación de diferentes combinaciones.

3. Tratamiento de volúmenes

3.1. Renderizado volumétrico

Esta herramienta incluye un motor de renderizado completo que soporta varios modos de renderizado. Tanto el modo de renderizado volumétrico basado en función de transferencia como el renderizado basado en isosuperficie propuestos en este trabajo implementan el algoritmo de *ray casting*. En esta implementación, los datos escalares se guardan en *slices* y éstas a su vez se almacenan en una única textura volumétrica que se mapea en un cubo con coordenadas unitarias. También es necesario calcular el punto de entrada del volumen, es decir, la primera intersección entre el rayo y la geometría que engloba al conjunto de datos volumétricos, y el punto de salida; a partir de estos dos componentes se obtiene la dirección del rayo. Para obtener estos puntos, la implementación que se propone sigue la base del algoritmo propuesto por [KW03] realizando dos pasadas de render. En primer lugar, se renderizan únicamente las caras delanteras del cubo y se guardan las posiciones en un *buffer* y en una segunda pasada se obtienen las caras traseras. De esta manera tenemos acceso en los *shaders* al punto de entrada y de salida y podemos recorrer el volumen siguiendo la ecuación de la recta.

En el renderizado volumétrico directo, el rayo muestrea a intervalos regulares y obtiene en cada posición el dato escalar guardado en la textura volumétrica. La función de transferencia mapea un valor de color y opacidad para cada intensidad, finalmente el valor de color y opacidad del pixel se obtiene mediante un esquema de composición *front-to-back*. El volumen también se puede representar mediante isosuperficies, o superficies con valor de intensidad constante. Si se contemplan las celdas como unidades del volumen, la isosuperficie de una celda se puede modelar con una función implícita y la intersección entre un rayo y la isosuperficie se puede calcular de manera analítica [LC96].

Con el objetivo de mejorar la visualización de datos complejos, es útil realizar recortes en el volumen que permitan mostrar la estructura interna del volumen, mostrando partes que de otra manera permanecerían ocultas. Para la realización de planos de recorte que permitan ver secciones del volumen, se sustituyen los puntos de entrada o de salida según la posición del plano de recorte. Para el cálculo de los puntos de intersección entre el rayo y el plano de recorte, se evalúan los puntos de entrada y de salida en la ecuación implícita del plano (ec.2).

$$Ax + By + Cz + D = 0 \tag{2}$$

Alternativamente, se puede visualizar un conjunto de datos 3D a partir de planos 2D (rodajas) que intersequen el volumen. Para ello, se mapea la textura volumétrica en planos que se renderizan de manera tradicional.

3.2. Etapas del proceso de renderizado

Para añadir el soporte de renderizado de objetos geométricos opacos es necesario añadir 2 etapas adicionales antes de las dos pasadas del renderizado volumétrico clásico, además se modifica esta última para tener en cuenta las oclusiones. Por tanto, las etapas de renderizado quedarían de la siguiente manera:

- **Primera pasada:** Se renderiza la geometría opaca en un *buffer*. De esta manera, se almacena la profundidad y el color de estos fragmentos.
- **Segunda pasada:** Se copia profundidad y color de los objetos opacos en el *buffer* que se utilizará en la cuarta pasada. Así dichos valores pueden utilizarse tanto de entrada como de salida.

Las siguientes dos etapas corresponden a las dos pasadas que se utilizan en el algoritmo de *ray casting* para renderizado volumétrico básico.

- **Tercera pasada:** Se calculan los puntos de entrada para realizar el *ray casting* y se almacenan en un *buffer* de profundidad.
- **Cuarta pasada:** Recibe como entrada la profundidad del sólido y la profundidad de las caras delanteras. Esta etapa produce como salida una nueva profundidad y un nuevo color que es la fusión de la imagen volumétrica y del sólido.

El *shader* de la última pasada, se ha modificado para que compruebe qué fragmentos pintar y qué fragmentos descartar, según la posición del opaco respecto al volumen. Si la profundidad de la geometría es menor a la profundidad de las caras delanteras, el fragmento del volumen se descarta pues está ocluido por la geometría. Si la profundidad de la geometría es menor a la profundidad

de las caras traseras, hay que sustituir el punto de salida del rayo del *ray casting* por el punto de intersección con el objeto opaco.

3.3. Algoritmo de fusión

Cuando se trabaja a alto nivel de detalle y los datos adquiridos son pequeños, la fusión de volúmenes otorga una visión de conjunto de los datos. En el contexto de este trabajo se entiende como fusión obtener una nueva imagen volumétrica a partir de la fusión de los datos volumétricos de dos volúmenes previamente cargados en la herramienta. Para ello, es necesario crear un nuevo volumen que contenga a los dos volúmenes en cuestión y acceder a las coordenadas de cada textura volumétrica, ya que a partir de estos datos se creará la nueva textura.

El valor de intensidad de los píxeles de la nueva imagen se calcula a partir de su posición:

1. Si se encuentra dentro de un volumen, pero no del otro, el valor de densidad será el valor de densidad de ese volumen.
2. Si se encuentra dentro de los dos volúmenes, se decide guardar el valor de densidad máximo de las dos densidades. De esta manera se consigue una transición suave entre los dos volúmenes en los límites que coinciden y una fusión de densidades correspondiente al resultado deseado.
3. Si se encuentra fuera de los dos volúmenes, se guarda un valor de densidad 0.

El proceso de fusión se realiza en un *shader* de cómputo para poder paralelizar la escritura de los píxeles de la nueva imagen. Finalmente, esta imagen volumétrica se trata de la misma manera que un único volumen.

4. Herramienta de fusión de volúmenes

La herramienta se compone de una interfaz (figura 1) que se encargará de renderizar los volúmenes, que renderiza como ejemplo un volumen creado a partir de la fusión de otros dos. La herramienta soporta la carga de archivos en formato .pvm y .tiff. Permite la selección entre los diferentes modos de renderizado (figura 2), la edición de la función de transferencia, añadir rodajas al volumen, editar las posiciones de los planos de recorte, así como la traslación y la rotación de los volúmenes para conseguir la posición deseada en la fusión. Cuando se elige el modo de dos volúmenes, se puede crear una nueva imagen volumétrica a partir de los mismos con la resolución deseada.

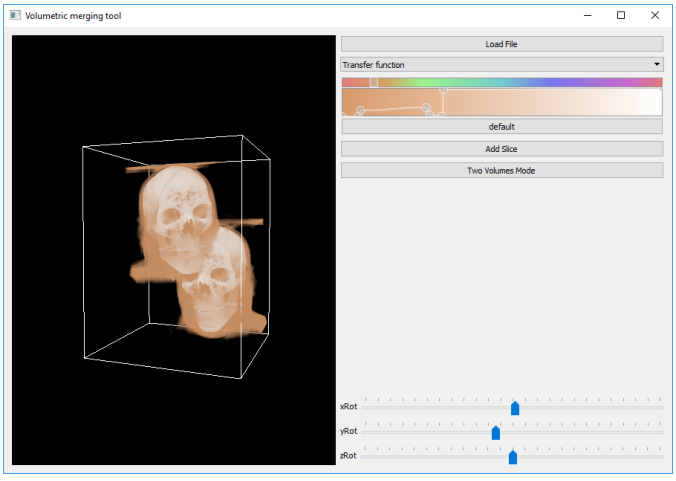


Figure 1: *Herramienta de fusión*

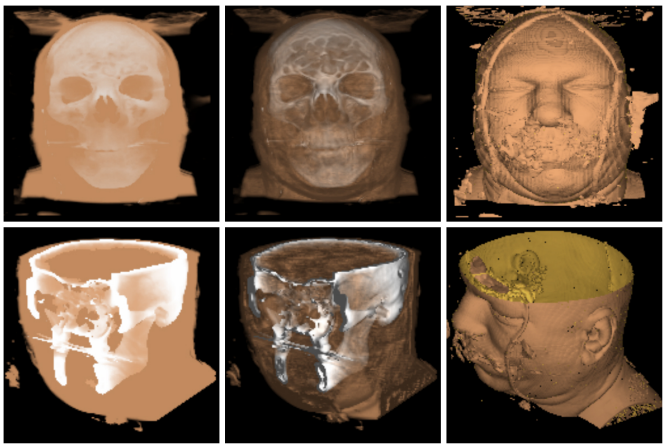
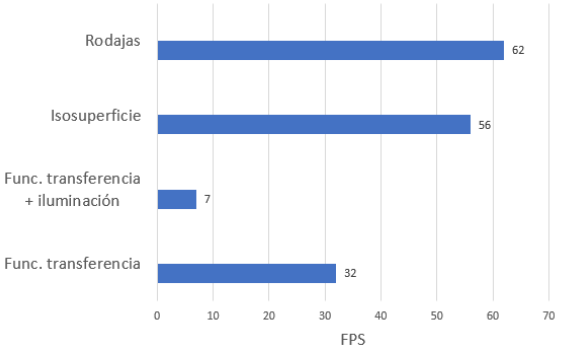


Figure 2: *Diferentes modos de renderizado volumétrico soportados por la herramienta*

5. Pruebas de rendimiento

El equipo con el que se realizan las pruebas utiliza el procesador Intel Core i7-7700 y tarjeta gráfica NVIDIA GeForce GTX 1050 Ti. Se ha utilizado el mismo modelo *vismale* que la figura 2 con una resolución de 256x256x256. En la figura 3a se puede ver la comparativa entre los diferentes modos de visualización: los modos de visualización basada en rodajas y basado en isosuperficies ofrecen una tasa de refresco alta (los resultados son alrededor de 60 *fps*) en comparación con los modos basados en función de transferencia, esto es debido a que no evalúan la integral de renderizado de manera directa. El modo de renderizado basado en isosuperficies es el modo de renderizado volumétrico más relevante matemáticamente que se presenta en este proyecto al calcular directamente el punto de intersección entre el rayo y la isosuperficie de manera analítica. El proceso iterativo mediante el cual se calcula el color en los modos basados en función de transferencia es un proceso costoso, ya que en cada pixel se calcula el color acumulado a lo largo de todo el volumen mediante un esquema de composición. El renderizado basado en función de transferencia con iluminación es especialmente costoso debido a que el cálculo de gradientes se realiza al momento en lugar de usar gradientes pre-calculados. El rendimiento varía considerablemente en función de la complejidad del modelo. En la figura 3b se pueden ver los resultados de la variación de complejidad. Todas las pruebas se están realizando con un incremento de avance en el rayo dt muy pequeño (dt=0.001). Conforme se decrementa este paso mejora la tasa de renderizado en una relación inversamente proporcional.



(a) *Rendimiento en fps según modo de renderizado utilizando el modelo vismale girando sobre sí mismo con diferentes funciones de transferencia*

(b) *Rendimiento en fps según resolución de modelo*

Figure 3: *Resultados pruebas rendimiento en fps*

6. Conclusiones y trabajo futuro

El resultado final de este trabajo es una herramienta de fusión de volúmenes que integra un motor de renderizado volumétrico que soporta diferentes modos de renderizado y una serie de funcionalidades que completan el objetivo final de esta herramienta que es fusionar dos volúmenes. Se ha conseguido un producto final funcional y con un gran potencial para ser utilizado.

Esta herramienta se crea para solucionar una problemática existente en el análisis de datos neurocientíficos a alto nivel de detalle: la creación de una visión de conjunto que se pierde al adquirir datos en pequeñas secciones. Esta herramienta permite interactuar con los datos en tiempo real para poder fusionarlos de manera manual, permitiendo versatilidad para resolver este problema. En la figura 4 se puede ver un ejemplo de esta aplicación con secciones de dendritas.

Sin embargo, sería necesario llevar a cabo un proceso completo de pruebas de usuario para conocer la utilidad real de la herramienta y mejorarla de acuerdo a los resultados de estas pruebas.

En el futuro, sería interesante mejorar el rendimiento de la herramienta mediante algunas nuevas implementaciones. Se podría realizar una optimización con una estructura de datos para evitar las zonas que no contienen información y así poder cargar conjuntos de datos relativamente grandes. Además, tal y como está diseñada la aplicación sería fácilmente escalable para cargar y fusionar más de dos volúmenes al mismo tiempo y para soportar la carga y guardado en más formatos, lo que haría más flexible su adaptación a otras disciplinas. Por último, podría ser de interés añadir diferentes algoritmos de fusión de intensidades (media, diferencia, etc.) permitiendo al usuario la selección del mismo y conseguir visualizaciones de diferentes características.

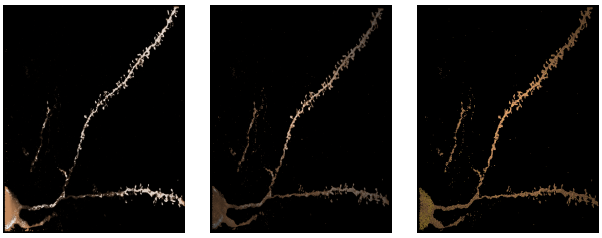


Figure 4: *Resultado de fusión de multitud de secciones de dendritas usando diferentes modos de renderizado*

References

[BHAA*13] BEYER J., HADWIGER M., AL-AWAMI A., JEONG W. K., KASTHURI N., LICHTMAN J. W., PFISTER H.: Exploring the connectome: Petascale volume visualization of microscopy data streams. *IEEE Computer Graphics and Applications* (2013). 2

[CCF94] CABRAL B., CAM N., FORAN J.: Accelerated volume rendering and tomographic reconstruction using texture mapping hardware. *Proceedings of the 1994 Symposium on Volume Visualization* (1994). 2

[DCH88] DREBIN R., CARPENTER L., HANRAHAN P.: Volume rendering. *ACM SIGGRAPH Computer Graphics* (1988). 2

[HW91] HALL P. M., WATT A. H.: Rapid Volume Rendering Using a Boundary-Fill Guided Ray Cast Algorithm. In *Scientific Visualization of Physical Phenomena* (Tokyo, 1991), Patrikalakis N. M., (Ed.), Springer Japan, pp. 235–249. 2

[JD14] JAMES A. P., DASARATHY B. V.: Medical image fusion: A survey of the state of the art. *Information Fusion* (2014). 2

[KW03] KRÜGER J., WESTERMANN R.: Acceleration Techniques for GPU-based Volume Rendering. *IEEE Visualization, 2003* (2003), 287–292. 2

[LC87] LORENSEN W. E., CLINE H. E.: Marching cubes: A high resolution 3D surface construction algorithm. *ACM SIGGRAPH Computer Graphics* (1987). 1

[LC96] LIN C.-C., CHING Y.-T.: An efficient volume- rendering algorithm with an analytic approach. In *The Visual Computer*. Springer-Verlag, 1996, pp. 515–526. 2

[Max95] MAX N.: Optical Models for Direct Volume Rendering. *IEEE Transactions on Visualization and Computer Graphics 1*, 2 (1995), 99–108. 1

[MBF] MBF BIOSCIENCE: NeuroLucida 360. 2

[PRL*10] PENG H., RUAN Z., LONG F., SIMPSON J. H., MYERS E. W.: V3D enables real-time 3D visualization and quantitative analysis of large-scale biological image data sets. *Nat Biotechnol* (2010). 2

[PST09] PREIBISCH S., SAALFELD S., TOMANCAK P.: Globally optimal stitching of tiled 3D microscopic image acquisitions. *Bioinformatics* (2009), 1463–1465. 2

[Rao05] RAO G.: Advanced Graphics Programming Using OpenGL. 2

[Wes90] WESTOVER L.: Footprint evaluation for volume rendering. *ACM SIGGRAPH Computer Graphics* (1990). 2