EuroVis 2018
R. S. Laramee, G. E. Marai, and M. Sedlmair
(Guest Editors)

Volume 37 (2018), Number 3
STAR – State of The Art Report

Project Report: Direct Volume Rendering Techniques for Cardiovascular Visualization

Christian Brändle & Niko Leopold

Visualization of Medical Data 2, 2017W

Abstract

Categories and Subject Descriptors (according to ACM CCS): I.3.3 [Computer Graphics]: Picture/Image Generation—; I.3.7 [Computer Graphics]: Three-dimensional Graphics and Realism—; I.3.5 [Computer Graphics]: Computational Geometry and Object Modeling—;

Keywords: Cardiovascular visualization, direct volume rendering, raytracing, transfer function design, intensity projection, curviplanar reformation, geometric modeling, vascular system modeling, convolution surfaces, implicit modeling, vessel reconstruction method, level sets, quad dominant meshes

1. Motivation

In the STAR report we have discussed various techniques related to the visualization of cardiovascular structures. We have covered both model-based and model-free approaches. Model-based mesh generation follows some more restrictive assumptions about the vascular structure whereas model-free methods such as direct volume rendering tries to capture as much original information as possible. Model-based techniques are often used for treatment planning and basic anatomical overview, whereas model-free methods are required for precise diagnostics.

We decided that for the project implementation we wanted to focus on direct volume rendering. The goal was to effectively visualize thin vessel structures and if possible even their narrowings due to stenoses etc. At first we considered to implement the automatic transfer function specification for visual emphasis of coronary artery plaque by Glasser et al. [GOH*10]. However we realized that due to time constraints this would not be possible as it involves a large number of different pipelines stages. We thus decided to implement instant volume visualization using maximum intensity difference accumulation by Bruckner and Gröller [BG09].

2. DICOM Data

To visualize cardiovascular data we first had to search approprite data that exhibit the information we want to visualize.

In the field of medical visualization DICOM is the defacto standard for capturing, annotating and deploying data. Including meta information and high dynamic range images DICOM is our preferred way to load medical data into our system.

2.1. Data Acquisition

To our disappointment we had to find out that it is rather difficult to find appropriate vascular data of the heart.

2.2. Data Import

One of the key advantages of DICOM data over simple image stacks is the fact that pixel intensities and radiometric intensities do not need to be the same in a given image stack. So DICOM provides a mapping to use several images together within a homogen radiometric background.

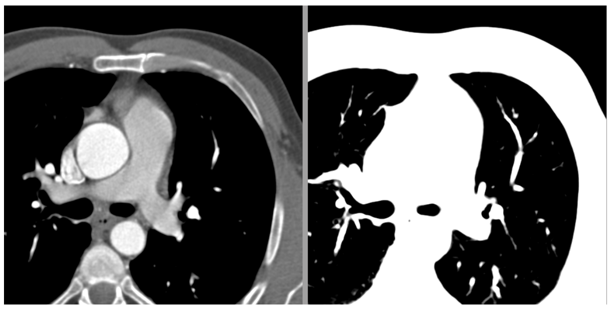

Figure 1: *Stacked images with inhomogenous radiometric space. DIOCM sample data* CARDIX *[gim18a] aquired from sample data section of the GIMAIS project [gim18b] and processed with* dicom2jpeg *from imbera project [ime18].*

Simple exports of JPEG data out of DICOM without respecting this relation spills out images with are not homogenous across the stack as seen for instance in figure 1.

2.3. Data Conversion

As it was not possible to find a suitable DICOM reader for our *CARDIX* dataset [gim18a] for VTK we used an intermediate step via MITK and convert DICOM data with the help of MITK main application to VTK 3D Image (VIK) data readable by VTK. In this way we could overcome the problems with codecs and intensity differences across the image stack.

3. System

As our original goal was to use an existing application and enhance it with certain abilities we researched the web to find open source applications that are cross platform and run under Windows and Linux environments. As this is a quite restrictive approach we actually found a lot more applications and frameworks but exclude them for the reasons above.

Our search continued then with frameworks as we saw that there was no application that fulfilled our needs and was capable for rapid prototyping our system. With frameworks we unfortunately also didn't have more luck as the way the frameworks limit us in regards of data import and data visualization was as well not suitable to develop our chosen algoritm rapidly.

In the end we utilize an OpenGL based visualion project to implement our algorithm there. Additional libraries to load DICOM data were searched but also there we experience several problems with codecs for instance.

3.1. 3DSlicer

„3D Slicer is an open source software platform for medical image informatics, image processing, and three-dimensional visualization [3DS18].“

The power of 3DSlicer lies in the handy user interface and the amount of resources how to deal with the program. Also youtube is a good source and this is how we lernt to do some basic tasks with 3DSlicer. It can also import several kinds of DICOM data, which is a prerequisite for our medical visualization application.

Although very nice at user level we do not discover a developer guide that assists us well through the internals of 3DSlicer. As our goal was not to use the application but rather to implement renderers for example we hadn't the impression that this would be possible in reasonable amount of time.

3.1.1. *MAGIX* Dataset in 3DSlicer

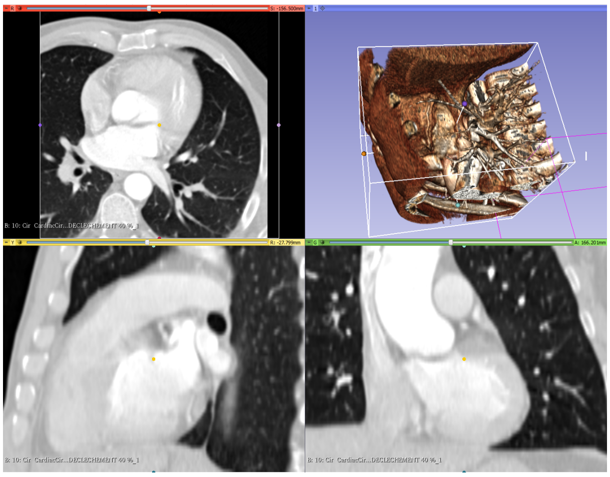

Figure 2: MAGIX *[gim18a] dataset visualized with* Slicer3D.

3.2. MITK

> The Medical Imaging Interaction Toolkit (MITK) is a free open-source software system for development of interactive medical image processing software. MITK combines the Insight Toolkit (ITK) and the Visualization Toolkit (VTK) with an application framework [MIT18].

In comparison to 3DSlicer MITK offers a better developer documentation and a description of the internal structure of MITK. Further it supplies guides how to create plugins or template based-projects that help to develop own applications with the MITK framework. Build instructions and a description of the render concept concludes our decision to take a closer look at this open source project. Also DICOM loaders are integrated and we managed to visualize one of our demo datasets in MITK. A downside was the stability of MITK as it happened several times that the application crashed after selecting DICOM data for volumetric visualization.

As we are interested in writing an own renderer and as MITK uses VTK to visualize data, we refer here to the VTK section that describes our experience with VTK in more detai.

3.2.1. *CARDIX* Dataset in MITK

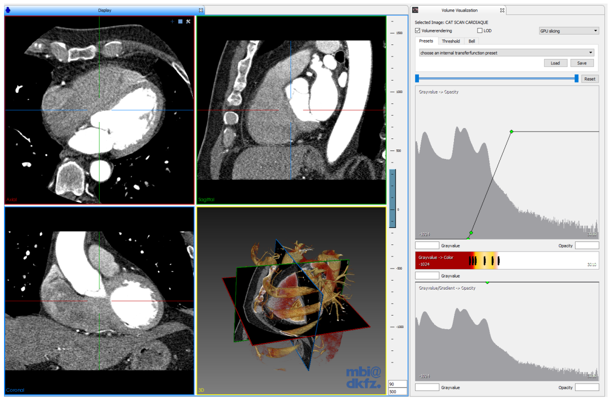

Figure 3: CARDIX *[gim18a] dataset visualized with* MITK.

3.3. VTK

> The Visualization Toolkit (VTK) is an open-source, freely available software system for 3D computer graphics, image processing, and visualization. It consists of a C++ class library and several interpreted interface layers including Tcl/Tk, Java, and Python. VTK supports a wide variety of visualization algorithms including scalar, vector, tensor, texture, and volumetric methods, as well as advanced modeling techniques such as implicit modeling, polygon reduction, mesh smoothing, cutting, contouring, and Delaunay triangulation. VTK has an extensive information visualization framework and a suite of 3D interaction widgets [vtk18].

As VTK was the central part of actual all medical visualization toolkits researched we wanted to use this framework for our application. Several parts of the VTK chain were researched, as there are:

3.3.1. Loading Data with VTK Readers

VTK also supports to load DICOM data via specific VTK Readers. As our data is formed of DICOM files we take a closer look at that feature. Unfortunately due to problems with the codec included in our DICOM data we were not able to load the data as we want to. As we haven't found a solution to fix this we come up with a pre-conversion step of DICOM data with MITK, as there it is possible to save DICOM data into a VTK 3D image.

3.3.2. Processing Data with VTK Filters

The VTK way of processing data is through VTK Filters. The philosophy is that filters transform or combine data within a chain or network of coupled filters that finally provide an output to one or more VTK Mappers. To implement our direct volume rendering approach the only easy way would have been to dynamically adapt our DICOM data according to the settings of MIDA and the current view direction and further use a built-in volume renderer like RayCast Mapper to visualize the result. As we would have to implement somehow a reverse version of MIDA and as the data transformation would have been a seperate step in the processing chain we decided not to use VTK Filters at all.

3.3.3. Render Data with VTK Mappers

In VTK renderer are called Mappers as they provide the ability to map a given dataset to a given device. While it is possible to override specific Mappers it is not straight forward to interfere with the renderer at OpenGL-level.

For this reason VTK introduced VTK shaders, that are based on structured OpenGL shaders. While this give access to GLSL shaders the structure of the shaders have to follow the VTK guidelines. That include string replacement of VTK specific shading commands within the OpenGL shader.

As we haven't found appropriate information on what for string replacements exist, how they should be used and how they interact with the rest of the VTK application we decided not to use an OpenGL shader based on the VTK framework.

3.3.4. VTK Example Application

3.4. Imbera DICOM Reader

„Imebra SDK is an open source C++ library that handles DICOM network messages and DICOM files [ime18].“

While Imbera seems to be the library we searched for to easily read DICOM data on multiple platforms in practice it fails because of the codec used in our *CARDIX* dataset. We don't know if this is due to the open source version used as we haven't tried the commercial version instead and will leave this as an open point.

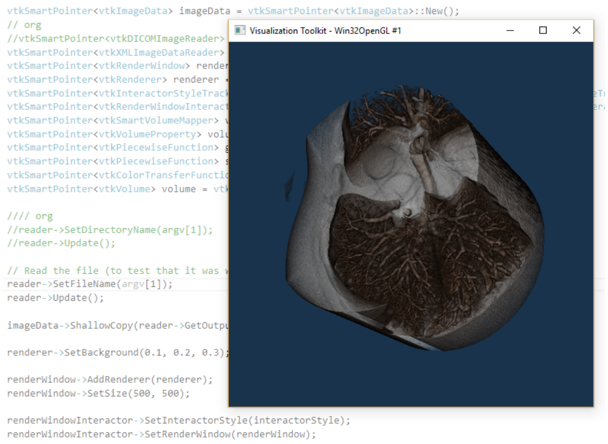

Figure 4: CARDIX *[gim18a] dataset rendered from VTI file with VTK Raycast Mapper.*

4. OpenGL Application

At first we considered using the VTK (Visualization ToolKit) framework as well as the MITK (Medical Imaging Interaction Toolkit) framework. However we soon realized that it is hard to implement a custom volume renderer in these frameworks, and thus decided to fall back to our own implementation. For this we reused an old Qt framework from the vis1 course that had simple GPU direct volume rendering implemented with simple alpha compositing, but with no proper interaction, no transfer functions and no shading.

The goal was to extend this framework with

- More compositing methods: Maximum Intensity Projection (MIP), Minimum Intensity Projection (MINIP), Average Intensity Projection (AIP) as well as Maximum Intensity Difference Accumulation (MIDA)
- Interactive controls
- Gradient-based shading
- Customizable transfer functions

5. MIDA - Maximum Intensity Difference Accumulation

MIDA is a compositing technique for Direct Volume Rendering (DVR) and was first published from Bruckner et al. [BG09]. It combines the advantage of Maximum Intensity Projection (MIP), where important structure shine through all data, with the advantage from traditinal DVR, where depth cues stem from color and opacity accumulation.

The advantages and disadvantages are more precisely:

- Maximum Intensity Projection (MIP)
  + MIP does not require a transfer function .
  - The spatial context in MIP visualization is lost.

- Direct Volume Rendering (DVR)
  + DVR accumulates color and opacity, it uses mostly gradient-based shading

- DVR needs appropriate transfer function, otherwise crucial information won't be visualized well and important features will disapear in fog or will be occluded by unimporant data.

5.1. MIDA - Accumulation

MIDA is rendered with Front-to-Back Traversal. It focus the interest on a ray where the maximum are and values change from low to high, where the interest is represented as δ_i. It overrides the occlusion relationship when a new maximum is encountered, where the weights are defined by $beta_i$.

$$\delta_i = \begin{cases} f_{P_i} - f_{max_i}, & \text{if } f_{P_i} > f_{max_i}. \\ 0, & \text{otherwise.} \end{cases} \tag{1}$$

$$\beta_i = 1 - \delta_i \tag{2}$$

$$c_i = \beta_i c_{i-1} + (1 - \beta_i \alpha_{i-1})\alpha(f_{P_i})c(f_{P_i}) \tag{3}$$

$$\alpha_i = \beta_i \alpha_{i-1} + (1 - \beta_i \alpha_{i-1})\alpha(f_{P_i}) \tag{4}$$

Color c_i and opacity α_i are computed the same way as for normal DVR, except that there is the additional weighting with β_i, see equation (4).

5.2. MIDA - Interpolation

MIDA can interpolate between it's two extremes, namely DVR and MIP in a continuous fashion. To achieve this a interpolation variable γ is defined that ranges from -1 to 1, i.e. $\gamma = [-1, 1]$. As interpoaltion is different in the direction of DVR compared to the direction of MIP we treat each case accordingly.

5.2.1. MIDA to DVR

Interpolation is done in 3D space as described before. It is based on the modulation of previously accumulated color and opacity.

5.2.2. MIDA to MIP

In contrast interpolation here is done in image space between color and opacity values of MIDA and color and opacity values of MIP.

5.3. MIDA - Shading & Classification

The magnitude of gradient to interpolate between shaded and unshaded color. This is due to the fact that shading expose artifacts if it is based on gradient directions with low magnitude like gradients from noise.

The used transfer function is limited to brightness and contrast adjustment using the common window/level approach.

6. Conclusion

7. Outlook

References

[3DS18] 3d slicer, Jan 2018. 2

[BG09] BRUCKNER S., GRÖLLER M. E.: Instant volume visualization using maximum intensity difference accumulation. In *Computer Graphics Forum* (2009), vol. 28, Wiley Online Library, pp. 775–782. 1, 3

[gim18a] Gimias sample data, Jan 2018. 1, 2, 3

[gim18b] Graphical interface for medical image analysis and simulation, Jan 2018. 1

[GOH*10] GLASSER S., OELTZE S., HENNEMUTH A., KUBISCH C., MAHNKEN A., WILHELMSEN S., PREIM B.: Automatic transfer function specification for visual emphasis of coronary artery plaque. In *Computer Graphics Forum* (2010), vol. 29, Wiley Online Library, pp. 191–201. 1

[ime18] Imbera - open source c dicom library /windows linux ios os-x android, Jan 2018. 1, 3

[MIT18] The medical imaging interaction toolkit (mitk), Jan 2018. 2

[vtk18] The visualization toolkit (vtk), Jan 2018. 2